Package: fitdistcp (via r-universe)

September 22, 2026

Type Package

Title Distribution Fitting with Calibrating Priors for Commonly Used Distributions

Version 0.2.3

Maintainer Stephen Jewson <stephen.jewson@gmail.com>

Description Generates predictive distributions based on calibrating priors for various commonly used statistical models, including models with predictors. Routines for densities, probabilities, quantiles, random deviates and the parameter posterior are provided. The predictions are generated from the Bayesian prediction integral, with priors chosen to give good reliability (also known as calibration). For homogeneous models, the prior is set to the right Haar prior, giving predictions which are exactly reliable. As a result, in repeated testing, the frequencies of out-of-sample outcomes and the probabilities from the predictions agree. For other models, the prior is chosen to give good reliability. Where possible, the Bayesian prediction integral is solved exactly. Where exact solutions are not possible, the Bayesian prediction integral is solved using the Datta-Mukerjee-Ghosh-Sweeting (DMGS) asymptotic expansion. Optionally, the prediction integral can also be solved using posterior samples generated using Paul Northrop's ratio of uniforms sampling package ('rust'). Results are also generated based on maximum likelihood, for comparison purposes. Various model selection diagnostics and testing routines are included. Based on ``Reducing reliability bias in assessments of extreme weather risk using calibrating priors'', Jewson, S., Sweeting, T. and Jewson, L. (2024); <doi:10.5194/ascmo-11-1-2025>.

Imports stats, mev, extraDistr, gnorm, fdrtool, pracma, rust, actuar, fExtremes

Depends R (>= 3.5.0)

License MIT + file LICENSE

BugReports https://github.com/stephenjewson/fitdistcp/issues

URL https://www.fitdistcp.info

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Roxygen list(markdown = TRUE)

Suggests knitr, rmarkdown

Repository https://stephenjewson.r-universe.dev

Date/Publication 2026-07-24 15:19:22 UTC

RemoteUrl https://github.com/stephenjewson/fitdistcp

RemoteRef HEAD

RemoteSha 2423a0fe1294b3d0e77f8e445587da3ea7612c72

Contents

adhoc_dmgs_cpmethod . . . 24
analytic_cpmethod . . . 24
bayesian_dq_4terms_v1 . . . 25
calc_revert2ml . . . 25
cauchy_cp . . . 26
cauchy_f1f . . . 33
cauchy_f1fa . . . 33
cauchy_f2f . . . 34
cauchy_f2fa . . . 34
cauchy_fd . . . 35
cauchy_fdd . . . 35
cauchy_ldd . . . 36
cauchy_ldda . . . 36
cauchy_lddd . . . 37
cauchy_lddda . . . 37
cauchy_lmn . . . 38
cauchy_lmnp . . . 38
cauchy_logf . . . 39
cauchy_logfdd . . . 40
cauchy_logfddd . . . 40
cauchy_loglik . . . 41
cauchy_logscores . . . 41
cauchy_mu1f . . . 42
cauchy_mu2f . . . 42
cauchy_p1_cp . . . 43
cauchy_p1_f1f . . . 51
cauchy_p1_f1fa . . . 51
cauchy_p1_f1fw . . . 52
cauchy_p1_f2f . . . 52
cauchy_p1_f2fa . . . 53

cauchy_p1_f2fw . . . 54
cauchy_p1_fd . . . 54
cauchy_p1_fdd . . . 55
cauchy_p1_ldd . . . 55
cauchy_p1_ldda . . . 56
cauchy_p1_lddd . . . 57
cauchy_p1_lddda . . . 57
cauchy_p1_lmn . . . 58
cauchy_p1_lmnp . . . 59
cauchy_p1_logf . . . 59
cauchy_p1_logfdd . . . 60
cauchy_p1_logfddd . . . 61
cauchy_p1_loglik . . . 61
cauchy_p1_logscores . . . 62
cauchy_p1_means . . . 62
cauchy_p1_mu1f . . . 63
cauchy_p1_mu2f . . . 64
cauchy_p1_p1f . . . 64
cauchy_p1_p2f . . . 65
cauchy_p1_predictordata . . . 66
cauchy_p1_waic . . . 66
cauchy_p1f . . . 67
cauchy_p2f . . . 68
cauchy_waic . . . 68
crhpflat_dmgs_cpmethod . . . 69
d010exp_example_data_v1 . . . 69
d011pareto_k2_example_data_v1 . . . 70
d020halfnorm_example_data_v1 . . . 70
d025unif_example_data_v1 . . . 70
d030norm_example_data_v1 . . . 70
d031norm_dmgs_example_data_v1 . . . 70
d032gnorm_k3_example_data_v1 . . . 71
d035lnorm_example_data_v1 . . . 71
d036lnorm_dmgs_example_data_v1 . . . 71
d040logis_example_data_v1 . . . 71
d041lst_k3_example_data_v1 . . . 71
d042cauchy_example_data_v1 . . . 72
d050gumbel_example_data_v1 . . . 72
d051frechet_k1_example_data_v1 . . . 72
d052weibull_example_data_v1 . . . 72
d053gev_k3_example_data_v1 . . . 72
d055exp_p1_example_data_v1_t . . . 73
d055exp_p1_example_data_v1_x . . . 73
d056pareto_p1k2_example_data_v1_t . . . 73
d056pareto_p1k2_example_data_v1_x . . . 73
d060norm_p1_example_data_v1_t . . . 73
d060norm_p1_example_data_v1_x . . . 74
d061lnorm_p1_example_data_v1_t . . . 74

d061lnorm_p1_example_data_v1_x . . . 74
d062logis_p1_example_data_v1_t . . . 74
d062logis_p1_example_data_v1_x . . . 74
d063lst_p1k3_example_data_v1_t . . . 75
d063lst_p1k3_example_data_v1_x . . . 75
d064cauchy_p1_example_data_v1_t . . . 75
d064cauchy_p1_example_data_v1_x . . . 75
d070gumbel_p1_example_data_v1_t . . . 75
d070gumbel_p1_example_data_v1_x . . . 76
d071frechet_p2k1_example_data_v1_t . . . 76
d071frechet_p2k1_example_data_v1_x . . . 76
d072weibull_p1_example_data_v1_t . . . 76
d072weibull_p1_example_data_v1_x . . . 76
d073weibull_p2_example_data_v1_t . . . 77
d073weibull_p2_example_data_v1_x . . . 77
d074gev_p1k3_example_data_v1_t . . . 77
d074gev_p1k3_example_data_v1_x . . . 77
d080norm_p12_example_data_v1_t1 . . . 77
d080norm_p12_example_data_v1_t2 . . . 78
d080norm_p12_example_data_v1_x . . . 78
d081lst_p12k3_example_data_v1_t1 . . . 78
d081lst_p12k3_example_data_v1_t2 . . . 78
d081lst_p12k3_example_data_v1_x . . . 78
d082weibull_p12_example_data_v1_t1 . . . 79
d082weibull_p12_example_data_v1_t2 . . . 79
d082weibull_p12_example_data_v1_x . . . 79
d100gamma_example_data_v1 . . . 79
d101invgamma_example_data_v1 . . . 79
d102invgauss_example_data_v1 . . . 80
d105burr_example_data_v1 . . . 80
d110gev_example_data_v1 . . . 80
d120gpd_k1_example_data_v1 . . . 80
d150gev_p1_example_data_v1_t . . . 80
d150gev_p1_example_data_v1_x . . . 81
d151gev_p12_example_data_v1_t . . . 81
d151gev_p12_example_data_v1_x . . . 81
d152gev_p123_example_data_v1_t . . . 81
d152gev_p123_example_data_v1_x . . . 81
dcauchy_p1 . . . 82
dcauchy_p1sub . . . 82
dcauchysub . . . 83
deriv_copyfdd . . . 83
deriv_copyld2 . . . 84
deriv_copyldd . . . 84
deriv_copylddd . . . 85
dexp_p1 . . . 85
dexp_p1sub . . . 86
dexpsub . . . 86

dfrechet_p2k1 . . . 87
dfrechet_p2k1sub . . . 87
dfrechetsub . . . 88
dgammasub . . . 88
dgev_k3sub . . . 89
dgev_p1 . . . 89
dgev_p12 . . . 90
dgev_p123 . . . 90
dgev_p123sub . . . 91
dgev_p12sub . . . 92
dgev_p1k3 . . . 93
dgev_p1k3sub . . . 93
dgev_p1n . . . 94
dgev_p1nsub . . . 94
dgev_p1sub . . . 95
dgevsub . . . 95
dgnorm_k3sub . . . 96
dgpdsub . . . 97
dgumbel_p1 . . . 97
dgumbel_p1sub . . . 98
dgumbelsub . . . 98
dhalfnormsub . . . 99
dinvgammasub . . . 99
dinvgausssub . . . 100
dlnorm_dmgssub . . . 100
dlnorm_p1 . . . 101
dlnorm_p1sub . . . 101
dlnormsub . . . 102
dlogis_p1 . . . 102
dlogis_p1sub . . . 103
dlogis2sub . . . 103
dlst_k3sub . . . 104
dlst_p1k3 . . . 104
dlst_p1k3sub . . . 105
dmgs . . . 105
dnorm_dmgssub . . . 106
dnorm_p1 . . . 107
dnorm_p1_formula . . . 107
dnorm_p12 . . . 108
dnorm_p12dmgs . . . 108
dnorm_p1sub . . . 109
dnormsub . . . 110
dpareto_k2_sub . . . 110
dpareto_p1k2 . . . 111
dpareto_p1k2sub . . . 111
dunif_formula . . . 112
dweibull_p2 . . . 112
dweibull_p2sub . . . 113

dweibullsub . . . 113
exp_cp . . . 114
exp_f1fa . . . 120
exp_f2fa . . . 120
exp_fd . . . 121
exp_fdd . . . 121
exp_ldda . . . 122
exp_lddda . . . 123
exp_logf . . . 123
exp_logfdd . . . 124
exp_logfddd . . . 124
exp_logscores . . . 125
exp_p1_cp . . . 125
exp_p1_f1fa . . . 132
exp_p1_f1fw . . . 133
exp_p1_f2fa . . . 134
exp_p1_f2fw . . . 134
exp_p1_fd . . . 135
exp_p1_fdd . . . 135
exp_p1_ldda . . . 136
exp_p1_lddda . . . 136
exp_p1_logf . . . 137
exp_p1_logfdd . . . 137
exp_p1_logfddd . . . 138
exp_p1_loglik . . . 138
exp_p1_logscores . . . 139
exp_p1_means . . . 139
exp_p1_mu1fa . . . 140
exp_p1_mu2fa . . . 140
exp_p1_p1fa . . . 141
exp_p1_p2fa . . . 141
exp_p1_pd . . . 142
exp_p1_pdd . . . 142
exp_p1_predictordata . . . 143
exp_p1_waic . . . 143
exp_p1fa . . . 144
exp_p2fa . . . 144
exp_pd . . . 145
exp_pdd . . . 145
exp_waic . . . 146
findnt . . . 147
fixgevrange . . . 147
fixgpdrange . . . 148
frechet_k1_cp . . . 148
frechet_k1_f1fa . . . 154
frechet_k1_f2fa . . . 155
frechet_k1_fd . . . 156
frechet_k1_fdd . . . 156

frechet_k1_ldda . . . 157
frechet_k1_lddda . . . 157
frechet_k1_logf . . . 158
frechet_k1_logfdd . . . 158
frechet_k1_logfddd . . . 159
frechet_k1_mu1fa . . . 159
frechet_k1_mu2fa . . . 160
frechet_k1_p1fa . . . 160
frechet_k1_p2fa . . . 161
frechet_k1_pd . . . 161
frechet_k1_pdd . . . 162
frechet_k1_waic . . . 162
frechet_loglik . . . 163
frechet_logscores . . . 163
frechet_means . . . 164
frechet_p2k1_cp . . . 164
frechet_p2k1_f1fa . . . 172
frechet_p2k1_f1fw . . . 172
frechet_p2k1_f2fa . . . 173
frechet_p2k1_f2fw . . . 173
frechet_p2k1_fd . . . 174
frechet_p2k1_fdd . . . 174
frechet_p2k1_ldda . . . 175
frechet_p2k1_lddda . . . 176
frechet_p2k1_logf . . . 176
frechet_p2k1_logfdd . . . 177
frechet_p2k1_logfddd . . . 177
frechet_p2k1_loglik . . . 178
frechet_p2k1_logscores . . . 179
frechet_p2k1_means . . . 179
frechet_p2k1_mu1fa . . . 180
frechet_p2k1_mu2fa . . . 181
frechet_p2k1_p1fa . . . 181
frechet_p2k1_p2fa . . . 182
frechet_p2k1_pd . . . 182
frechet_p2k1_pdd . . . 183
frechet_p2k1_predictordata . . . 184
frechet_p2k1_waic . . . 184
gamma_cp . . . 185
gamma_f1f . . . 192
gamma_f1fa . . . 193
gamma_f2f . . . 193
gamma_f2fa . . . 194
gamma_fd . . . 194
gamma_fdd . . . 195
gamma_gg . . . 195
gamma_gmn . . . 196
gamma_ldd . . . 196

gamma_ldda . . . 197
gamma_lddd . . . 198
gamma_lddda . . . 198
gamma_lmn . . . 199
gamma_lmnp . . . 199
gamma_logf . . . 200
gamma_logfdd . . . 201
gamma_logfddd . . . 201
gamma_loglik . . . 202
gamma_logscores . . . 202
gamma_means . . . 203
gamma_mu1f . . . 203
gamma_mu2f . . . 204
gamma_p1f . . . 204
gamma_p2f . . . 205
gamma_waic . . . 206
gev_boot . . . 206
gev_checkmle . . . 207
gev_cp . . . 207
gev_f1fa . . . 216
gev_f2fa . . . 217
gev_fd . . . 217
gev_fdd . . . 218
gev_k12_ppm_minusloglik . . . 218
gev_k3_cp . . . 220
gev_k3_f1fa . . . 226
gev_k3_f2fa . . . 227
gev_k3_fd . . . 228
gev_k3_fdd . . . 228
gev_k3_ldda . . . 229
gev_k3_lddda . . . 229
gev_k3_logf . . . 230
gev_k3_logfdd . . . 230
gev_k3_logfddd . . . 231
gev_k3_loglik . . . 231
gev_k3_means . . . 232
gev_k3_mu1fa . . . 232
gev_k3_mu2fa . . . 233
gev_k3_pd . . . 233
gev_k3_pdd . . . 234
gev_k3_waic . . . 234
gev_ld12a . . . 235
gev_lda . . . 236
gev_ldda . . . 236
gev_lddda . . . 237
gev_logf . . . 237
gev_logfd . . . 238
gev_logfdd . . . 238

gev_logfddd . . . 239
gev_loglik . . . 239
gev_means . . . 240
gev_mu1fa . . . 241
gev_mu2fa . . . 241
gev_p1_checkmle . . . 242
gev_p1_cp . . . 242
gev_p1_logf . . . 251
gev_p1_loglik . . . 252
gev_p1_means . . . 252
gev_p1_predictordata . . . 253
gev_p1_setics . . . 253
gev_p1_waic . . . 254
gev_p12_checkmle . . . 254
gev_p12_cp . . . 255
gev_p12_cp . . . 263
gev_p12_f1fa . . . 272
gev_p12_f1fw . . . 272
gev_p12_f2fa . . . 273
gev_p12_f2fw . . . 274
gev_p12_fd . . . 274
gev_p12_fdd . . . 275
gev_p12_ldda . . . 276
gev_p12_lddda . . . 276
gev_p12_logf . . . 277
gev_p12_logfdd . . . 278
gev_p12_logfddd . . . 278
gev_p12_loglik . . . 279
gev_p12_means . . . 280
gev_p12_mu1fa . . . 280
gev_p12_mu2fa . . . 281
gev_p12_pd . . . 282
gev_p12_pdd . . . 282
gev_p12_predictordata . . . 283
gev_p12_setics . . . 284
gev_p12_waic . . . 284
gev_p123_checkmle . . . 285
gev_p123_cp . . . 286
gev_p123_cp . . . 294
gev_p123_f1fa . . . 303
gev_p123_f1fw . . . 304
gev_p123_f2fa . . . 305
gev_p123_f2fw . . . 305
gev_p123_fd . . . 306
gev_p123_fdd . . . 307
gev_p123_ldda . . . 307
gev_p123_lddda . . . 308
gev_p123_logf . . . 309

gev_p123_logfdd . . . 309
gev_p123_logfddd . . . 310
gev_p123_loglik . . . 311
gev_p123_means . . . 311
gev_p123_mu1fa . . . 312
gev_p123_mu2fa . . . 313
gev_p123_pd . . . 313
gev_p123_pdd . . . 314
gev_p123_predictordata . . . 315
gev_p123_setics . . . 315
gev_p123_waic . . . 316
gev_p12k3_f1fa . . . 317
gev_p12k3_f1fw . . . 317
gev_p12k3_f2fa . . . 318
gev_p12k3_f2fw . . . 319
gev_p12k3_fd . . . 319
gev_p12k3_fdd . . . 320
gev_p12k3_ldda . . . 321
gev_p12k3_lddda . . . 321
gev_p12k3_logfdd . . . 322
gev_p12k3_logfddd . . . 323
gev_p12k3_mu1fa . . . 323
gev_p12k3_mu2fa . . . 324
gev_p12k3_pd . . . 325
gev_p12k3_pdd . . . 325
gev_p1a_f1fa . . . 326
gev_p1a_f1fw . . . 327
gev_p1a_f2fa . . . 327
gev_p1a_f2fw . . . 328
gev_p1a_fd . . . 328
gev_p1a_fdd . . . 329
gev_p1a_ldda . . . 330
gev_p1a_lddda . . . 330
gev_p1a_logfdd . . . 331
gev_p1a_logfddd . . . 331
gev_p1a_mu1fa . . . 332
gev_p1a_mu2fa . . . 333
gev_p1a_pd . . . 333
gev_p1a_pdd . . . 334
gev_p1b_f1fa . . . 334
gev_p1b_f1fw . . . 335
gev_p1b_f2fa . . . 336
gev_p1b_f2fw . . . 336
gev_p1b_fd . . . 337
gev_p1b_fdd . . . 338
gev_p1b_ldda . . . 338
gev_p1b_lddda . . . 339
gev_p1b_logfdd . . . 340

gev_p1b_logfddd . . . 340
gev_p1b_mu1fa . . . 341
gev_p1b_mu2fa . . . 342
gev_p1b_pd . . . 342
gev_p1b_pdd . . . 343
gev_p1c_f1fa . . . 344
gev_p1c_f1fw . . . 344
gev_p1c_f2fa . . . 345
gev_p1c_f2fw . . . 346
gev_p1c_fd . . . 347
gev_p1c_fdd . . . 347
gev_p1c_ldda . . . 348
gev_p1c_lddda . . . 349
gev_p1c_logfdd . . . 349
gev_p1c_logfddd . . . 350
gev_p1c_mu1fa . . . 351
gev_p1c_mu2fa . . . 351
gev_p1c_pd . . . 352
gev_p1c_pdd . . . 353
gev_p1k3_cp . . . 354
gev_p1k3_f1fa . . . 361
gev_p1k3_f1fw . . . 362
gev_p1k3_f2fa . . . 362
gev_p1k3_f2fw . . . 363
gev_p1k3_fd . . . 363
gev_p1k3_fdd . . . 364
gev_p1k3_ldda . . . 365
gev_p1k3_lddda . . . 365
gev_p1k3_logf . . . 366
gev_p1k3_logfdd . . . 366
gev_p1k3_logfddd . . . 367
gev_p1k3_loglik . . . 367
gev_p1k3_means . . . 368
gev_p1k3_mu1fa . . . 369
gev_p1k3_mu2fa . . . 369
gev_p1k3_pd . . . 370
gev_p1k3_pdd . . . 370
gev_p1k3_predictordata . . . 371
gev_p1k3_waic . . . 372
gev_p1n_checkmle . . . 373
gev_p1n_cp . . . 373
gev_p1n_cp . . . 382
gev_p1n_logf . . . 391
gev_p1n_loglik . . . 391
gev_p1n_means . . . 392
gev_p1n_n1_exampledata . . . 393
gev_p1n_n2_exampledata . . . 393
gev_p1n_predictordata . . . 394

gev_p1n_setics . . . 394
gev_p1n_waic . . . 395
gev_pd . . . 395
gev_pdd . . . 396
gev_pwm_params . . . 396
gev_setics . . . 397
gev_waic . . . 397
gnorm_k3_cp . . . 398
gnorm_k3_f1f . . . 405
gnorm_k3_f1fa . . . 406
gnorm_k3_f2f . . . 406
gnorm_k3_f2fa . . . 407
gnorm_k3_fd . . . 407
gnorm_k3_fdd . . . 408
gnorm_k3_ldd . . . 408
gnorm_k3_ldda . . . 409
gnorm_k3_lddd . . . 409
gnorm_k3_lddda . . . 410
gnorm_k3_lmn . . . 410
gnorm_k3_logf . . . 411
gnorm_k3_logfdd . . . 412
gnorm_k3_logfddd . . . 412
gnorm_k3_loglik . . . 413
gnorm_k3_logscores . . . 413
gnorm_k3_mu1f . . . 414
gnorm_k3_mu2f . . . 414
gnorm_k3_p1f . . . 415
gnorm_k3_p2f . . . 415
gnorm_lmnp . . . 416
gnorm_waic . . . 417
gpd_k1_checkmle . . . 418
gpd_k1_cp . . . 418
gpd_k1_f1fa . . . 426
gpd_k1_f2fa . . . 427
gpd_k1_fd . . . 428
gpd_k1_fdd . . . 428
gpd_k1_ldda . . . 429
gpd_k1_lddda . . . 429
gpd_k1_logf . . . 430
gpd_k1_logfdd . . . 430
gpd_k1_logfddd . . . 431
gpd_k1_loglik . . . 431
gpd_k1_means . . . 432
gpd_k1_mu1fa . . . 433
gpd_k1_mu2fa . . . 433
gpd_k1_pd . . . 434
gpd_k1_pdd . . . 434
gpd_k1_setics . . . 435

gpd_k1_waic . . . 435
gpd_k13_f1fa . . . 436
gpd_k13_f2fa . . . 436
gpd_k13_fd . . . 437
gpd_k13_fdd . . . 437
gpd_k13_ldda . . . 438
gpd_k13_lddda . . . 438
gpd_k13_logfdd . . . 439
gpd_k13_logfddd . . . 439
gpd_k13_mu1fa . . . 440
gpd_k13_mu2fa . . . 440
gpd_k13_pd . . . 441
gpd_k13_pdd . . . 441
gumbel_cp . . . 442
gumbel_f1fa . . . 448
gumbel_f2fa . . . 448
gumbel_fd . . . 449
gumbel_fdd . . . 449
gumbel_ldda . . . 450
gumbel_lddda . . . 450
gumbel_logf . . . 451
gumbel_logfdd . . . 451
gumbel_logfddd . . . 452
gumbel_loglik . . . 452
gumbel_logscores . . . 453
gumbel_means . . . 453
gumbel_mu1fa . . . 454
gumbel_mu2fa . . . 455
gumbel_p1_cp . . . 455
gumbel_p1_f1fa . . . 462
gumbel_p1_f1fw . . . 463
gumbel_p1_f2fa . . . 464
gumbel_p1_f2fw . . . 464
gumbel_p1_fd . . . 465
gumbel_p1_fdd . . . 465
gumbel_p1_ldda . . . 466
gumbel_p1_lddda . . . 466
gumbel_p1_logf . . . 467
gumbel_p1_logfdd . . . 467
gumbel_p1_logfddd . . . 468
gumbel_p1_loglik . . . 468
gumbel_p1_logscores . . . 469
gumbel_p1_means . . . 469
gumbel_p1_mu1fa . . . 470
gumbel_p1_mu2fa . . . 471
gumbel_p1_p1fa . . . 471
gumbel_p1_p2fa . . . 472
gumbel_p1_pd . . . 472

gumbel_p1_pdd . . . 473
gumbel_p1_predictordata . . . 473
gumbel_p1_waic . . . 474
gumbel_p1fa . . . 475
gumbel_p2fa . . . 475
gumbel_pd . . . 476
gumbel_pdd . . . 476
gumbel_waic . . . 477
halfnorm_cp . . . 477
halfnorm_f1f . . . 484
halfnorm_f1fa . . . 485
halfnorm_f2f . . . 485
halfnorm_f2fa . . . 486
halfnorm_fd . . . 486
halfnorm_fdd . . . 487
halfnorm_gg . . . 487
halfnorm_gg11 . . . 488
halfnorm_l111 . . . 488
halfnorm_ldd . . . 489
halfnorm_ldda . . . 489
halfnorm_lddd . . . 490
halfnorm_lddda . . . 490
halfnorm_logf . . . 491
halfnorm_logfdd . . . 491
halfnorm_logfddd . . . 492
halfnorm_loglik . . . 493
halfnorm_logscores . . . 493
halfnorm_means . . . 494
halfnorm_mu1f . . . 494
halfnorm_mu2f . . . 495
halfnorm_p1f . . . 495
halfnorm_p2f . . . 496
halfnorm_waic . . . 496
ifvectorthenmatrix . . . 497
invgamma_cp . . . 497
invgamma_f1f . . . 504
invgamma_f1fa . . . 505
invgamma_f2f . . . 505
invgamma_f2fa . . . 506
invgamma_fd . . . 506
invgamma_fdd . . . 507
invgamma_ldd . . . 507
invgamma_ldda . . . 508
invgamma_lddd . . . 508
invgamma_lddda . . . 509
invgamma_lmn . . . 509
invgamma_lmnp . . . 510
invgamma_logf . . . 510

invgamma_logfdd . . . 511
invgamma_logfddd . . . 511
invgamma_loglik . . . 512
invgamma_logscores . . . 512
invgamma_mu1f . . . 513
invgamma_mu2f . . . 514
invgamma_p1f . . . 514
invgamma_p2f . . . 515
invgamma_waic . . . 515
invgauss_cp . . . 516
invgauss_f1f . . . 524
invgauss_f1fa . . . 524
invgauss_f2f . . . 525
invgauss_f2fa . . . 525
invgauss_fd . . . 526
invgauss_fdd . . . 526
invgauss_ldd . . . 527
invgauss_ldda . . . 527
invgauss_lddd . . . 528
invgauss_lddda . . . 528
invgauss_lmn . . . 529
invgauss_lmnp . . . 529
invgauss_logf . . . 530
invgauss_logfdd . . . 531
invgauss_logfddd . . . 531
invgauss_loglik . . . 532
invgauss_logscores . . . 532
invgauss_means . . . 533
invgauss_mu1f . . . 533
invgauss_mu2f . . . 534
invgauss_p1f . . . 534
invgauss_p2f . . . 535
invgauss_waic . . . 536
jpf2p . . . 537
jpf3p . . . 537
jpf4p . . . 538
lnorm_cp . . . 538
lnorm_dmgs_cp . . . 544
lnorm_dmgs_loglik . . . 550
lnorm_dmgs_logscores . . . 551
lnorm_dmgs_means . . . 551
lnorm_dmgs_waic . . . 552
lnorm_f1fa . . . 552
lnorm_f2fa . . . 553
lnorm_fd . . . 553
lnorm_fdd . . . 554
lnorm_ldda . . . 554
lnorm_lddda . . . 555

lnorm_logf . . . 555
lnorm_logfdd . . . 556
lnorm_logfddd . . . 556
lnorm_logscores . . . 557
lnorm_mu1fa . . . 557
lnorm_mu2fa . . . 558
lnorm_p1_cp . . . 558
lnorm_p1_f1fa . . . 565
lnorm_p1_f1fw . . . 566
lnorm_p1_f2fa . . . 566
lnorm_p1_f2fw . . . 567
lnorm_p1_fd . . . 567
lnorm_p1_fdd . . . 568
lnorm_p1_ldda . . . 568
lnorm_p1_lddda . . . 569
lnorm_p1_logf . . . 569
lnorm_p1_logfdd . . . 570
lnorm_p1_logfddd . . . 570
lnorm_p1_loglik . . . 571
lnorm_p1_logscores . . . 571
lnorm_p1_mu1fa . . . 572
lnorm_p1_mu2fa . . . 572
lnorm_p1_p1fa . . . 573
lnorm_p1_p2fa . . . 573
lnorm_p1_pd . . . 574
lnorm_p1_pdd . . . 574
lnorm_p1_predictordata . . . 575
lnorm_p1_waic . . . 575
lnorm_p1fa . . . 576
lnorm_p2fa . . . 576
lnorm_pd . . . 577
lnorm_pdd . . . 577
lnorm_waic . . . 578
logis_cp . . . 578
logis_f1fa . . . 584
logis_f2fa . . . 585
logis_fd . . . 585
logis_fdd . . . 586
logis_ldda . . . 586
logis_lddda . . . 587
logis_logf . . . 587
logis_logfdd . . . 588
logis_logfddd . . . 588
logis_loglik . . . 589
logis_logscores . . . 589
logis_mu1fa . . . 590
logis_mu2fa . . . 590
logis_p1_cp . . . 591

logis_p1_f1fa . . . 598
logis_p1_f1fw . . . 599
logis_p1_f2fa . . . 599
logis_p1_f2fw . . . 600
logis_p1_fd . . . 600
logis_p1_fdd . . . 601
logis_p1_ldda . . . 601
logis_p1_lddda . . . 602
logis_p1_logf . . . 602
logis_p1_logfdd . . . 603
logis_p1_logfddd . . . 603
logis_p1_loglik . . . 604
logis_p1_logscores . . . 604
logis_p1_means . . . 605
logis_p1_mu1fa . . . 605
logis_p1_mu2fa . . . 606
logis_p1_p1fa . . . 606
logis_p1_p2fa . . . 607
logis_p1_pd . . . 607
logis_p1_pdd . . . 608
logis_p1_predictordata . . . 608
logis_p1_waic . . . 609
logis_p1fa . . . 610
logis_p2fa . . . 610
logis_pd . . . 611
logis_pdd . . . 611
logis_waic . . . 612
lst_k3_cp . . . 612
lst_k3_f1f . . . 619
lst_k3_f1fa . . . 620
lst_k3_f2f . . . 621
lst_k3_f2fa . . . 621
lst_k3_fd . . . 622
lst_k3_fdd . . . 622
lst_k3_ldd . . . 623
lst_k3_ldda . . . 623
lst_k3_lddd . . . 624
lst_k3_lddda . . . 624
lst_k3_lmn . . . 625
lst_k3_lmnp . . . 625
lst_k3_logf . . . 626
lst_k3_logfdd . . . 627
lst_k3_logfddd . . . 627
lst_k3_loglik . . . 628
lst_k3_logscores . . . 628
lst_k3_mu1f . . . 629
lst_k3_mu2f . . . 629
lst_k3_p1f . . . 630

lst_k3_p2f . . . 630
lst_k3_waic . . . 631
lst_p1k3_cp . . . 632
lst_p1k3_f1f . . . 640
lst_p1k3_f1fa . . . 641
lst_p1k3_f1fw . . . 642
lst_p1k3_f2f . . . 642
lst_p1k3_f2fa . . . 643
lst_p1k3_f2fw . . . 644
lst_p1k3_fd . . . 644
lst_p1k3_fdd . . . 645
lst_p1k3_ldd . . . 645
lst_p1k3_ldda . . . 646
lst_p1k3_lddd . . . 647
lst_p1k3_lddda . . . 647
lst_p1k3_lmn . . . 648
lst_p1k3_lmnp . . . 649
lst_p1k3_logf . . . 649
lst_p1k3_logfdd . . . 650
lst_p1k3_logfddd . . . 651
lst_p1k3_loglik . . . 651
lst_p1k3_logscores . . . 652
lst_p1k3_mu1f . . . 652
lst_p1k3_mu2f . . . 653
lst_p1k3_p1f . . . 654
lst_p1k3_p2f . . . 654
lst_p1k3_predictordata . . . 655
lst_p1k3_setics . . . 656
lst_p1k3_waic . . . 656
make_cwaic . . . 657
make_maic . . . 658
make_se . . . 658
make_waic . . . 659
makebetat0 . . . 659
makebetatm . . . 660
makemuhat0 . . . 660
makeq . . . 661
maket0 . . . 661
maketresid0 . . . 662
man . . . 662
man1f . . . 672
man2f . . . 673
manboot . . . 673
mancheckmle . . . 673
mandsub . . . 674
manf . . . 674
manldd . . . 681
manlddd . . . 681

manlnn . . . 682
manlnnn . . . 682
manlogf . . . 682
manloglik . . . 683
manlogscores . . . 683
manmeans . . . 683
manpredictor . . . 684
manvector . . . 684
manwaic . . . 684
movexiawayfromzero . . . 685
ms_flat_1tail . . . 685
ms_flat_2tail . . . 687
ms_predictors_1tail . . . 688
ms_predictors_2tail . . . 689
nopdfcdfmsg . . . 690
norm_boot . . . 691
norm_cp . . . 691
norm_dmgs_cp . . . 697
norm_dmgs_loglik . . . 703
norm_dmgs_logscores . . . 703
norm_dmgs_means . . . 704
norm_dmgs_waic . . . 705
norm_f1fa . . . 705
norm_f2fa . . . 706
norm_fd . . . 706
norm_fdd . . . 707
norm_ldda . . . 707
norm_lddda . . . 708
norm_logf . . . 708
norm_logfdd . . . 709
norm_logfddd . . . 709
norm_logscores . . . 710
norm_ml_params . . . 710
norm_mu1fa . . . 711
norm_mu2fa . . . 711
norm_p1_cp . . . 712
norm_p1_f1fa . . . 719
norm_p1_f1fw . . . 719
norm_p1_f2fa . . . 720
norm_p1_f2fw . . . 720
norm_p1_fd . . . 721
norm_p1_fdd . . . 722
norm_p1_ldda . . . 722
norm_p1_lddda . . . 723
norm_p1_logf . . . 724
norm_p1_logfdd . . . 724
norm_p1_logfddd . . . 725
norm_p1_loglik . . . 725

norm_p1_logscores . . . 726
norm_p1_mlparams . . . 726
norm_p1_mu1fa . . . 727
norm_p1_mu2fa . . . 728
norm_p1_p1fa . . . 728
norm_p1_p2fa . . . 729
norm_p1_pd . . . 730
norm_p1_pdd . . . 730
norm_p1_predictordata . . . 731
norm_p1_waic . . . 732
norm_p12_boot . . . 732
norm_p12_checkmle . . . 733
norm_p12_cp . . . 733
norm_p12_cp . . . 741
norm_p12_exampledata . . . 749
norm_p12_f1fa . . . 749
norm_p12_f1fw . . . 750
norm_p12_f2fa . . . 750
norm_p12_f2fw . . . 751
norm_p12_fd . . . 751
norm_p12_fdd . . . 752
norm_p12_ldda . . . 753
norm_p12_lddda . . . 753
norm_p12_logf . . . 754
norm_p12_logfdd . . . 754
norm_p12_logfddd . . . 755
norm_p12_loglik . . . 756
norm_p12_logscores . . . 756
norm_p12_mu1fa . . . 757
norm_p12_mu2fa . . . 757
norm_p12_p1fa . . . 758
norm_p12_p2fa . . . 758
norm_p12_pd . . . 759
norm_p12_pdd . . . 760
norm_p12_predictordata . . . 760
norm_p12_setics . . . 761
norm_p12_waic . . . 762
norm_p1fa . . . 763
norm_p2fa . . . 763
norm_pd . . . 764
norm_pdd . . . 764
norm_unbiasedv_params . . . 765
norm_waic . . . 765
pareto_k2_cp . . . 766
pareto_k2_f1fa . . . 772
pareto_k2_f2fa . . . 772
pareto_k2_fd . . . 773
pareto_k2_fdd . . . 773

pareto_k2_ldda . . . 774
pareto_k2_lddda . . . 775
pareto_k2_logf . . . 775
pareto_k2_logfdd . . . 776
pareto_k2_logfddd . . . 776
pareto_k2_logscores . . . 777
pareto_k2_ml_params . . . 778
pareto_k2_mu1fa . . . 778
pareto_k2_mu2fa . . . 779
pareto_k2_p1fa . . . 779
pareto_k2_p2fa . . . 780
pareto_k2_pd . . . 780
pareto_k2_pdd . . . 781
pareto_k2_waic . . . 782
pareto_p1k2_cp . . . 782
pareto_p1k2_f1fa . . . 790
pareto_p1k2_f1fw . . . 790
pareto_p1k2_f2fa . . . 791
pareto_p1k2_f2fw . . . 791
pareto_p1k2_fd . . . 792
pareto_p1k2_fdd . . . 792
pareto_p1k2_ldda . . . 793
pareto_p1k2_lddda . . . 793
pareto_p1k2_logf . . . 794
pareto_p1k2_logfdd . . . 794
pareto_p1k2_logfddd . . . 795
pareto_p1k2_loglik . . . 795
pareto_p1k2_logscores . . . 796
pareto_p1k2_means . . . 796
pareto_p1k2_mu1fa . . . 797
pareto_p1k2_mu2fa . . . 798
pareto_p1k2_p1fa . . . 798
pareto_p1k2_p2fa . . . 799
pareto_p1k2_pd . . . 799
pareto_p1k2_pdd . . . 800
pareto_p1k2_predictordata . . . 800
pareto_p1k2_waic . . . 801
pcauchy_p1 . . . 802
pexp_p1 . . . 802
pfrechet_p2k1 . . . 803
pgev_p1 . . . 803
pgev_p12 . . . 804
pgev_p123 . . . 804
pgev_p1k3 . . . 805
pgev_p1n . . . 806
pgumbel_p1 . . . 806
plnorm_p1 . . . 807
plogis_p1 . . . 807

plst_p1k3 . . . 808
pnorm_p1 . . . 808
pnorm_p1_formula . . . 809
pnorm_p12 . . . 809
ppareto_p1k2 . . . 810
punif_formula . . . 810
pweibull_p2 . . . 811
qcauchy_p1 . . . 811
qexp_p1 . . . 812
qfrechet_p2k1 . . . 812
qgamma_k1_ppm . . . 813
qgamma_ppm . . . 814
qgev_k12_ppm . . . 815
qgev_mpd_ppm . . . 817
qgev_p1 . . . 818
qgev_p1_ppm . . . 819
qgev_p12 . . . 820
qgev_p123 . . . 821
qgev_p1k3 . . . 821
qgev_p1n . . . 822
qgev_ppm . . . 823
qgpd_k1_ppm . . . 824
qgumbel_p1 . . . 825
qlnorm_p1 . . . 826
qlogis_p1 . . . 827
qlst_p1k3 . . . 827
qnorm_p1 . . . 828
qnorm_p1_formula . . . 828
qnorm_p12 . . . 829
qntt_ppm . . . 829
qpareto_p1k2 . . . 831
qunif_formula . . . 831
qweibull_p2 . . . 832
reltest . . . 832
reltest . . . 836
reltest_makeep . . . 839
reltest_makemaxep . . . 840
reltest_params . . . 841
reltest_predict . . . 841
reltest_simulate . . . 843
reltest2 . . . 844
reltest2 . . . 846
reltest2_cases . . . 849
reltest2_makeep . . . 850
reltest2_plot . . . 850
reltest2_predict . . . 851
reltest2_simulate . . . 852
rgev_minmax . . . 852

rgev_p1_minmax . . . 853
rgev_p12_minmax . . . 854
rgev_p123_minmax . . . 855
rgev_p1n_minmax . . . 856
rgpd_k1_minmax . . . 857
rhp_dmgs_cpmethod . . . 857
rust_pumethod . . . 858
testppm_plot . . . 858
unif_cp . . . 859
unif_cp . . . 865
weibull_cp . . . 871
weibull_cp . . . 877
weibull_f1fa . . . 883
weibull_f2fa . . . 884
weibull_fd . . . 884
weibull_fdd . . . 885
weibull_ldda . . . 885
weibull_lddda . . . 886
weibull_logf . . . 886
weibull_logfdd . . . 887
weibull_logfddd . . . 887
weibull_loglik . . . 888
weibull_logscores . . . 888
weibull_means . . . 889
weibull_mu1fa . . . 889
weibull_mu2fa . . . 890
weibull_p1fa . . . 890
weibull_p2_cp . . . 891
weibull_p2_cp . . . 898
weibull_p2_f1fa . . . 905
weibull_p2_f1fw . . . 906
weibull_p2_f2fa . . . 906
weibull_p2_f2fw . . . 907
weibull_p2_fd . . . 907
weibull_p2_fdd . . . 908
weibull_p2_ldda . . . 908
weibull_p2_lddda . . . 909
weibull_p2_logf . . . 909
weibull_p2_logfdd . . . 910
weibull_p2_logfddd . . . 910
weibull_p2_loglik . . . 911
weibull_p2_logscores . . . 911
weibull_p2_means . . . 912
weibull_p2_mu1fa . . . 912
weibull_p2_mu2fa . . . 913
weibull_p2_p1fa . . . 913
weibull_p2_p2fa . . . 914
weibull_p2_pd . . . 914

weibull_p2_pdd 915
weibull_p2_predictordata 915
weibull_p2_waic 916
weibull_p2fa 917
weibull_pd 917
weibull_pdd 918
weibull_waic 918

Index **920**

`adhoc_dmgs_cpmethod` *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
adhoc_dmgs_cpmethod()
```

Value

String

`analytic_cpmethod` *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
analytic_cpmethod()
```

Value

String

bayesian_dq_4terms_v1 *Evaluate DMGS equation 3.3*

Description

Evaluate DMGS equation 3.3

Usage

```
bayesian_dq_4terms_v1(lddi, lddd, mu1, pidopi1, pidopi2, mu2, dim)
```

Arguments

lddi	inverse of second derivative of observed log-likelihood
lddd	third derivative of observed log-likelihood
mu1	DMGS mu1 vector
pidopi1	first part of the prior term
pidopi2	second part of the prior term
mu2	DMGS mu2 matrix
dim	number of parameters

Value

Vector

calc_revert2ml *determine revert2ml or not*

Description

determine revert2ml or not

Usage

```
calc_revert2ml(v5h, v6h, t3)
```

Arguments

v5h	fifth parameter
v6h	sixth parameter
t3	a vector of predictors for the shape

Value

Logical

cauchy_cp *Cauchy Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qcauchy_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rcauchy_cp(
```

```
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dcauchy_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pcauchy_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tcauchy_cp(n, x, d1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Cauchy distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\pi\sigma}\left(1+\left(\frac{x-\mu}{\sigma}\right)^2\right)^{-1}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),

- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d042cauchy_example_data_v1
p=c(1:9)/10
q=qlogis_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qcauchy_cp)",
 main="Cauchy: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

cauchy_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
cauchy_f1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
cauchy_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

cauchy_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
cauchy_f2f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
cauchy_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

cauchy_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
cauchy_ldd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

cauchy_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
cauchy_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
cauchy_lddd(x, v1, d1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

cauchy_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
cauchy_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

cauchy_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_lmn(x, v1, d1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

cauchy_lmnp *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
cauchy_lmnp(x, v1, d1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

cauchy_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
cauchy_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

cauchy_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

cauchy_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

cauchy_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
cauchy_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

cauchy_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
cauchy_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

cauchy_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
cauchy_mu1f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
cauchy_mu2f(alpha, v1, d1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

cauchy_p1_cp *Cauchy Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
```

```
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rcauchy_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

pcauchy_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
```

```
  fd3 = 0.01,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tcauchy_p1_cp(n, x, t, d1 = 0.01, d2 = 0.01, fd3 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Cauchy distribution with a predictor has probability density function

$$f(x; a, b, \sigma) = \frac{1}{\pi\sigma}\left(1 + \left(\frac{x - \mu(a,b)}{\sigma}\right)^2\right)^{-1}$$

where x is the random variable, $\mu = a + bt$ is the location parameter as a function of parameters a, b, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),
- Weibull with linear predictor on the scale (`weibull_p2`),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
#
# example 1
x=fitdistcp::d064cauchy_p1_example_data_v1_x
tt=fitdistcp::d064cauchy_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qcauchy_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qcauchy_p1_cp)",
 main="Cauchy w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

cauchy_p1_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
cauchy_p1_f1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
cauchy_p1_f1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`cauchy_p1_f1fw` *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
cauchy_p1_f1fw(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`cauchy_p1_f2f` *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
cauchy_p1_f2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

---

cauchy_p1_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
cauchy_p1_f2fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

cauchy_p1_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
cauchy_p1_f2fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

cauchy_p1_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`cauchy_p1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_fdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

Matrix

`cauchy_p1_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
cauchy_p1_ldd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

cauchy_p1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

cauchy_p1_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
cauchy_p1_lddd(x, t, v1, d1, v2, d2, v3, fd3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

cauchy_p1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lddda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`cauchy_p1_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lmn(x, t, v1, d1, v2, d2, v3, fd3, mm, nn)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`v3`	third parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

cauchy_p1_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
cauchy_p1_lmnp(x, t, v1, d1, v2, d2, v3, fd3, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

cauchy_p1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
cauchy_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

---

cauchy_p1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

cauchy_p1_logfdddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
cauchy_p1_logfdddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

cauchy_p1_loglik *Cauchy-with-p1 observed log-likelihood function*

Description

Cauchy-with-p1 observed log-likelihood function

Usage

```
cauchy_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

cauchy_p1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
cauchy_p1_logscores(logscores, x, t, d1, d2, fd3, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

cauchy_p1_means *Cauchy distribution: RHP mean*

Description

Cauchy distribution: RHP mean

Usage

```
cauchy_p1_means(t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

cauchy_p1_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
cauchy_p1_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

cauchy_p1_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
cauchy_p1_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1_p1f *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
cauchy_p1_p1f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

cauchy_p1_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
cauchy_p1_p2f(y, t0, v1, d1, v2, d2, v3, fd3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

cauchy_p1_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
cauchy_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

cauchy_p1_waic *Waic*

Description

Waic

Usage

```
cauchy_p1_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  lddi,
```

```
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

cauchy_p1f *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
cauchy_p1f(y, v1, d1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`cauchy_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
cauchy_p2f(y, v1, d1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`cauchy_waic` *Waic*

Description

Waic

Usage

```
cauchy_waic(waicscores, x, v1hat, d1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

crhpflat_dmgs_cpmethod

Generates a comment about the method

Description

Generates a comment about the method

Usage

```
crhpflat_dmgs_cpmethod()
```

Value

String

---

d010exp_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d011pareto_k2_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d020halfnorm_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d025unif_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d030norm_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d031norm_dmgs_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d032gnorm_k3_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d035lnorm_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d036lnorm_dmgs_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d040logis_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d041lst_k3_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d042cauchy_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d050gumbel_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d051frechet_k1_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d052weibull_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d053gev_k3_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d055exp_p1_example_data_v1_t

This is data to be included in my package

Description

This is data to be included in my package

d055exp_p1_example_data_v1_x

This is data to be included in my package

Description

This is data to be included in my package

d056pareto_p1k2_example_data_v1_t

This is data to be included in my package

Description

This is data to be included in my package

d056pareto_p1k2_example_data_v1_x

This is data to be included in my package

Description

This is data to be included in my package

d060norm_p1_example_data_v1_t

This is data to be included in my package

Description

This is data to be included in my package

d060norm_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d061lnorm_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d061lnorm_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d062logis_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d062logis_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d063lst_p1k3_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d063lst_p1k3_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d064cauchy_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d064cauchy_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d070gumbel_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d070gumbel_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d071frechet_p2k1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d071frechet_p2k1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d072weibull_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d072weibull_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d073weibull_p2_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d073weibull_p2_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d074gev_p1k3_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d074gev_p1k3_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d080norm_p12_example_data_v1_t1
This is data to be included in my package

Description

This is data to be included in my package

d080norm_p12_example_data_v1_t2
This is data to be included in my package

Description

This is data to be included in my package

d080norm_p12_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d081lst_p12k3_example_data_v1_t1
This is data to be included in my package

Description

This is data to be included in my package

d081lst_p12k3_example_data_v1_t2
This is data to be included in my package

Description

This is data to be included in my package

d081lst_p12k3_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d082weibull_p12_example_data_v1_t1

This is data to be included in my package

Description

This is data to be included in my package

d082weibull_p12_example_data_v1_t2

This is data to be included in my package

Description

This is data to be included in my package

d082weibull_p12_example_data_v1_x

This is data to be included in my package

Description

This is data to be included in my package

d100gamma_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d101invgamma_example_data_v1

This is data to be included in my package

Description

This is data to be included in my package

d102invgauss_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d105burr_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d110gev_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d120gpd_k1_example_data_v1
This is data to be included in my package

Description

This is data to be included in my package

d150gev_p1_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d150gev_p1_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d151gev_p12_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d151gev_p12_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

d152gev_p123_example_data_v1_t
This is data to be included in my package

Description

This is data to be included in my package

d152gev_p123_example_data_v1_x
This is data to be included in my package

Description

This is data to be included in my package

dcauchy_p1 *Cauchy-with-p1 density function*

Description

Cauchy-with-p1 density function

Usage

```
dcauchy_p1(x, t0, ymn, slope, scale, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution
log	logical for the density evaluation

Value

Vector

dcauchy_p1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dcauchy_p1sub(x, t, y, t0, d1, d2, fd3, aderivs = TRUE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
d1	the delta used in the numerical derivatives with respect to the parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dcauchysub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dcauchysub(x, y, d1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

deriv_copyfdd *Extract the results from derivatives and put them into f2*

Description

Extract the results from derivatives and put them into f2

Usage

```
deriv_copyfdd(temp1, nx, dim)
```

Arguments

temp1	output from derivative calculations
nx	number of x values
dim	number of parameters

Value

3d array

deriv_copyld2 *Extract the results from derivatives and put them into ldd*

Description

Extract the results from derivatives and put them into ldd

Usage

```
deriv_copyld2(temp1, nx, dim)
```

Arguments

temp1	output from derivative calculations
nx	number of x values
dim	number of parameters

Value

3d array

deriv_copyldd *Extract the results from derivatives and put them into ldd*

Description

Extract the results from derivatives and put them into ldd

Usage

```
deriv_copyldd(temp1, nx, dim)
```

Arguments

temp1	output from derivative calculations
nx	number of x values
dim	number of parameters

Value

Matrix

deriv_copylddd *Extract the results from derivatives and put them into lddd*

Description

Extract the results from derivatives and put them into lddd

Usage

```
deriv_copylddd(temp1, nx, dim)
```

Arguments

temp1	output from derivative calculations
nx	number of x values
dim	number of parameters

Value

3d array

dexp_p1 *Exponential-with-p1 density function*

Description

Exponential-with-p1 density function

Usage

```
dexp_p1(x, t0, ymn, slope, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
log	logical for the density evaluation

Value

Vector

dexp_p1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dexp_p1sub(x, t, y, t0)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dexpsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dexpsub(x, y)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dfrechet_p2k1 *Frechet_k1-with-p2 density function*

Description

Frechet_k1-with-p2 density function

Usage

```
dfrechet_p2k1(x, t0, ymn, slope, lambda, log = FALSE, kloc)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
lambda	the lambda parameter of the distribution
log	logical for the density evaluation
kloc	the known location parameter

Value

Vector

dfrechet_p2k1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dfrechet_p2k1sub(x, t, y, t0, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
kloc	the known location parameter

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dfrechetsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dfrechetsub(x, y, kloc)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
kloc	the known location parameter

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dgammasub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgammasub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dgev_k3sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgev_k3sub(x, y, kshape)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
kshape	the known shape parameter

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dgev_p1 *GEVD-with-p1: Density function*

Description

GEVD-with-p1: Density function

Usage

```
dgev_p1(x, t0, ymn, slope, sigma, xi, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

`dgev_p12` *GEVD-with-p1: Density function*

Description

GEVD-with-p1: Density function

Usage

```
dgev_p12(x, t1, t2, ymn, slope, sigma1, sigma2, xi, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma1`	first coefficient for the sigma parameter of the distribution
`sigma2`	second coefficient for the sigma parameter of the distribution
`xi`	the shape parameter of the distribution
`log`	logical for the density evaluation

Value

Vector

`dgev_p123` *GEVD-with-p1: Density function*

Description

GEVD-with-p1: Density function

Usage

```
dgev_p123(x, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2, log = FALSE)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution
log	logical for the density evaluation

Value

Vector

---

dgev_p123sub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgev_p123sub(x, t1, t2, t3, y, t01, t02, t03, ics, extramodels, debug)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
ics	initial conditions for the maximum likelihood search
extramodels	logical that indicates whether to add three additional prediction models
debug	debug flag

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_p12sub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgev_p12sub(
  x,
  t1,
  t2,
  y,
  t01,
  t02,
  ics,
  minxi,
  maxxi,
  debug,
  extramodels = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
ics	initial conditions for the maximum likelihood search
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
debug	debug flag
extramodels	logical that indicates whether to add three additional prediction models

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgev_p1k3 *GEV-with-known-shape-with-p1 density function*

Description

GEV-with-known-shape-with-p1 density function

Usage

```
dgev_p1k3(x, t0, ymn, slope, sigma, log = FALSE, kshape)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation
kshape	the known shape parameter

Value

Vector

dgev_p1k3sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgev_p1k3sub(x, t, y, t0, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
kshape	the known shape parameter

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dgev_p1n` *GEVD-with-p1: Density function*

Description

GEVD-with-p1: Density function

Usage

```
dgev_p1n(x, t0, params, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf
`log`	logical for the density evaluation

Value

Vector

`dgev_p1nsub` *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgev_p1nsub(x, t, y, t0, ics, minxi, maxxi, extramodels = FALSE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ics`	initial conditions for the maximum likelihood search
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi
`extramodels`	logical that indicates whether to add three additional prediction models

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dgev_p1sub` *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgev_p1sub(x, t, y, t0, ics, minxi, maxxi, extramodels = FALSE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ics`	initial conditions for the maximum likelihood search
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi
`extramodels`	logical that indicates whether to add three additional prediction models

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dgevsub` *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgevsub(x, y, ics, minxi, maxxi)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dgnorm_k3sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgnorm_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kbeta, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgpdsub *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dgpdsub(x, y, ics, kloc = 0, dlogpi = 0, minxi, maxxi, extramodels = FALSE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
kloc	the known location parameter
dlogpi	gradient of the log prior
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
extramodels	logical that indicates whether to add three additional prediction models

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dgumbel_p1 *Gumbel-with-p1 density function*

Description

Gumbel-with-p1 density function

Usage

```
dgumbel_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation

Value

Vector

`dgumbel_p1sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgumbel_p1sub(x, t, y, t0)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dgumbelsub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dgumbelsub(x, y)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dhalfnormsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dhalfnormsub(x, y, fd1 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dinvgammasub *Densities from MLE and cp*

Description

Densities from MLE and cp

Usage

```
dinvgammasub(x, y, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dinvgausssub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dinvgausssub(x, y, prior, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions
`prior`	logical indicating which prior to use
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dlnorm_dmgssub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlnorm_dmgssub(x, y)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dlnorm_p1` *Normal-with-p1 density function*

Description

Normal-with-p1 density function

Usage

```
dlnorm_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma`	the sigma parameter of the distribution
`log`	logical for the density evaluation

Value

Vector

`dlnorm_p1sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlnorm_p1sub(x, t, y, t0, debug = FALSE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`debug`	debug flag

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dlnormsub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlnormsub(x, y)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dlogis_p1` *Logistic-with-p1 density function*

Description

Logistic-with-p1 density function

Usage

```
dlogis_p1(x, t0, ymn, slope, scale, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`scale`	the scale parameter of the distribution
`log`	logical for the density evaluation

Value

Vector

`dlogis_p1sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlogis_p1sub(x, t, y, t0)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

`dlogis2sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlogis2sub(x, y)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlst_k3sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlst_k3sub(x, y, d1 = 0.01, fd2 = 0.01, kdf, aderivs = TRUE)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dlst_p1k3 *LST-with-p1 density function*

Description

LST-with-p1 density function

Usage

```
dlst_p1k3(x, t0, ymn, slope, sigma, log = FALSE, kdf)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation
kdf	the known degrees of freedom parameter

Value

Vector

`dlst_p1k3sub` *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dlst_p1k3sub(x, t, y, t0, d1, d2, fd3, kdf, aderivs = TRUE)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`y`	a vector of values at which to calculate the density and distribution functions
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

`dmgs` *Evaluate DMGS equation 3.3*

Description

Evaluate DMGS equation 3.3

Usage

```
dmgs(lddi, lddd, mu1, pidopi, mu2, dim)
```

Arguments

lddi	inverse of second derivative of observed log-likelihood
lddd	third derivative of observed log-likelihood
mu1	DMGS mu1 vector
pidopi	derivative of log prior
mu2	DMGS mu2 matrix
dim	number of parameters

Value

Vector

---

dnorm_dmgssub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dnorm_dmgssub(x, y)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dnorm_p1 *Normal-with-p1 density function*

Description

Normal-with-p1 density function

Usage

```
dnorm_p1(x, t0, ymn, slope, sigma, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
log	logical for the density evaluation

Value

Vector

dnorm_p1_formula *Linear regression formula, densities*

Description

Linear regression formula, densities

Usage

```
dnorm_p1_formula(y, tresid, tresid0, nx, muhat0, v3hat)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
tresid	predictor residuals
tresid0	predictor residual at the point being predicted
nx	length of training data
muhat0	muhat at the point being predicted
v3hat	third parameter

Value

Vector

`dnorm_p12` *Normal-with-p12: Density function*

Description

Normal-with-p12: Density function

Usage

```
dnorm_p12(x, t01, t02, ymn, slope, sigma1, sigma2, log = FALSE)
```

Arguments

`x`	a vector of training data values
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma1`	first coefficient for the sigma parameter of the distribution
`sigma2`	second coefficient for the sigma parameter of the distribution
`log`	logical for the density evaluation

Value

Vector

`dnorm_p12dmgs` *Densities for 5 predictions*

Description

Densities for 5 predictions

Usage

```
dnorm_p12dmgs(x, t1, t2, y, t01, t02, ics)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
ics	initial conditions for the maximum likelihood search

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dnorm_p1sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dnorm_p1sub(x, t, y, t0)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dnormsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dnormsub(x, y)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dpareto_k2_sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dpareto_k2_sub(x, y, kscale)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions
kscale	the known scale parameter

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dpareto_p1k2 *pareto_k1-with-p2 density function*

Description

pareto_k1-with-p2 density function

Usage

```
dpareto_p1k2(x, t0, ymn, slope, kscale, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
kscale	the known scale parameter
log	logical for the density evaluation

Value

Vector

dpareto_p1k2sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dpareto_p1k2sub(x, t, y, t0, kscale, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
kscale	the known scale parameter
debug	debug flag

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

---

dunif_formula *Predictive PDFs*

Description

Predictive PDFs

Usage

```
dunif_formula(x, y)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions

Value

Two vectors

---

dweibull_p2 *Weibull-with-p1 density function*

Description

Weibull-with-p1 density function

Usage

```
dweibull_p2(x, t0, shape, ymn, slope, log = FALSE)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
shape	the shape parameter of the distribution
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
log	logical for the density evaluation

Value

Vector

dweibull_p2sub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dweibull_p2sub(x, t, y, t0)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

dweibullsub *Densities from MLE and RHP*

Description

Densities from MLE and RHP

Usage

```
dweibullsub(x, y)
```

Arguments

x	a vector of training data values
y	a vector of values at which to calculate the density and distribution functions

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

exp_cp *Exponential Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qexp_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rexp_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE)

dexp_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

pexp_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

texp_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The exponential distribution has exceedance distribution function

$$S(x;\lambda) = \exp(-\lambda x)$$

where $x \geq 0$ is the random variable and $\lambda > 0$ is the rate parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\lambda) \propto \frac{1}{\lambda}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)

- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),

- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d010exp_example_data_v1
p=c(1:9)/10
q=qexp_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qexp_cp)",
 main="Exponential: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

exp_f1fa *The first derivative of the density*

Description

The first derivative of the density

The first derivative of the density

Usage

```
exp_f1fa(x, v1)

exp_f1fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

Vector

exp_f2fa *The second derivative of the density*

Description

The second derivative of the density

The second derivative of the density

Usage

```
exp_f2fa(x, v1)

exp_f2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

`exp_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_fd(x, v1)

exp_fd(x, v1)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter

Value

Vector

Vector

`exp_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_fdd(x, v1)
```

```
exp_fdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

`exp_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

The second derivative of the normalized log-likelihood

Usage

```
exp_ldda(x, v1)
```

```
exp_ldda(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

exp_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

The third derivative of the normalized log-likelihood

Usage

```
exp_lddda(x, v1)
```


```
exp_lddda(x, v1)
```


Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

3d array

exp_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
exp_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

exp_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_logfdd(x, v1)

exp_logfdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

exp_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_logfddd(x, v1)

exp_logfddd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

3d array

`exp_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
exp_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

`exp_p1_cp` *Exponential Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qexp_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rexp_p1_cp(n, x, t, t0 = NA, n0 = NA, rust = FALSE, mlcp = TRUE, debug = FALSE)

dexp_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)
```

```
pexp_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

texp_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The exponential distribution with a predictor has exceedance distribution function

$$S(x; a, b) = \exp(-x\lambda(a, b))$$

where $x \geq 0$ is the random variable and $\lambda(a, b) = e^{-a-bt}$ is the rate parameter, modelled as a function of the parameters a, b and a predictor t.

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

. as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),

- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d055exp_p1_example_data_v1_x
tt=fitdistcp::d055exp_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qexp_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qexp_p1_cp)",
 main="Exponential w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

exp_p1_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
exp_p1_f1fa(x, t0, v1, v2)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter

Value

Vector

`exp_p1_f1fw` *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
exp_p1_f1fw(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
exp_p1_f2fa(x, t0, v1, v2)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
exp_p1_f2fw(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

`exp_p1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_fd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Vector

`exp_p1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_fdd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
exp_p1_ldda(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
exp_p1_lddda(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

3d array

exp_p1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
exp_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

exp_p1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_logfdd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

Matrix

exp_p1_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_logfddd(x, t, v1, v2)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter

Value

3d array

exp_p1_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
exp_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

exp_p1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
exp_p1_logscores(logscores, x, t)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors

Value

Two scalars

exp_p1_means *exp distribution: RHP means*

Description

exp distribution: RHP means

Usage

```
exp_p1_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

`exp_p1_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
exp_p1_mu1fa(alpha, t0, v1, v2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter

Value

Vector

`exp_p1_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
exp_p1_mu2fa(alpha, t0, v1, v2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter

Value

Matrix

exp_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
exp_p1_p1fa(x, t0, v1, v2)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter

Value

Vector

exp_p1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
exp_p1_p2fa(x, t0, v1, v2)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter

Value

Matrix

`exp_p1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_pd(x, t, v1, v2)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter

Value

Vector

`exp_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_p1_pdd(x, t, v1, v2)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter

Value

Matrix

exp_p1_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
exp_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

exp_p1_waic *Waic*

Description

Waic

Usage

```
exp_p1_waic(waicscores, x, t, v1hat, v2hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

`exp_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

The first derivative of the cdf

Usage

```
exp_p1fa(x, v1)

exp_p1fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

Vector

`exp_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

The second derivative of the cdf

Usage

```
exp_p2fa(x, v1)

exp_p2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

---

exp_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_pd(x, v1)

exp_pd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

Vector

---

exp_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
exp_pdd(x, v1)

exp_pdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

`exp_waic` *Waicscores*

Description

Waicscores

Usage

```
exp_waic(waicscores, x, v1hat)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter

Value

Two numeric values.

findnt — *Find the number of predictors in the predictor*

Description

Find the number of predictors in the predictor

Usage

```
findnt(t)
```

Arguments

t	a vector or matrix of predictors

Value

Vector

fixgevrange — *Deal with situations in which the user wants d or p outside the GEV range*

Description

Deal with situations in which the user wants d or p outside the GEV range

Usage

```
fixgevrange(y, v1, v2, v3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

fixgpdrange — *Deal with situations in which the user wants d or p outside the GPD range*

Description

Deal with situations in which the user wants d or p outside the GPD range

Usage

```
fixgpdrange(y, v1, v2, v3)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

frechet_k1_cp — *Frechet Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qfrechet_k1_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kloc = 0,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rfrechet_k1_cp(n, x, kloc = 0, rust = FALSE, mlcp = TRUE, debug = FALSE)

dfrechet_k1_cp(x, y = x, kloc = 0, rust = FALSE, nrust = 1000, debug = FALSE)

pfrechet_k1_cp(x, y = x, kloc = 0, rust = FALSE, nrust = 1000, debug = FALSE)

tfrechet_k1_cp(n, x, kloc = 0, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kloc	the known location parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations

debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Frechet distribution has distribution function

$$F(x;\sigma,\lambda) = \exp\left(-\left(\frac{x-\mu}{\sigma}\right)^{-\lambda}\right)$$

where $x > \mu$ is the random variable, $\sigma > 0, \lambda > 0$ are the parameters and we consider μ to be known (hence the k1 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma,\lambda) \propto \frac{1}{\sigma\lambda}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d051frechet_k1_example_data_v1
p=c(1:9)/10
q=qfrechet_k1_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qfrechet_k1_cp)",
 main="Frechet: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

frechet_k1_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
frechet_k1_f1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

---

frechet_k1_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
frechet_k1_f2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

`frechet_k1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_fd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`frechet_k1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_fdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

frechet_k1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
frechet_k1_ldda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

frechet_k1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
frechet_k1_lddda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

`frechet_k1_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
frechet_k1_logf(params, x, kloc)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`kloc`	the known location parameter

Value

Scalar value.

`frechet_k1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_logfdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

frechet_k1_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

frechet_k1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
frechet_k1_mu1fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

frechet_k1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
frechet_k1_mu2fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

frechet_k1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
frechet_k1_p1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

frechet_k1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
frechet_k1_p2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

frechet_k1_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

frechet_k1_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_k1_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

frechet_k1_waic *Waic*

Description

Waic

Usage

```
frechet_k1_waic(waicscores, x, v1hat, v2hat, kloc, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

`frechet_loglik` *log-likelihood function*

Description

log-likelihood function

Usage

```
frechet_loglik(vv, x, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
kloc	the known location parameter

Value

Scalar

`frechet_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
frechet_logscores(logscores, x, kloc)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
kloc	the known location parameter

Value

Two scalars

frechet_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
frechet_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2, kloc)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

frechet_p2k1_cp *Frechet Distribution with Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.

- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qfrechet_p2k1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  kloc = 0,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rfrechet_p2k1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  kloc = 0,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE
)

dfrechet_p2k1_cp(
  x,
  t,
```

```
  t0 = NA,
  n0 = NA,
  y = x,
  kloc = 0,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pfrechet_p2k1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  kloc = 0,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tfrechet_p2k1_cp(n, x, t, kloc = 0, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
kloc	the known location parameter
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated

centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.

- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Frechet distribution with predictor has distribution function

$$F(x; a, b, \lambda) = \exp\left(-\left(\frac{x-\mu}{\sigma(a,b)}\right)^{-\lambda}\right)$$

where $x > \mu$ is the random variable, $\sigma = e^{a+bt}$ is the scale parameter, modelled as a function of parameters a, b and predictor t, and $\lambda > 0$ is the shape parameter. We consider μ to be known (hence the k1 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),

- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d071frechet_p2k1_example_data_v1_x
tt=fitdistcp::d071frechet_p2k1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qfrechet_p2k1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qfrechet_p2k1_cp)",
 main="Frechet w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

frechet_p2k1_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
frechet_p2k1_f1fa(x, t0, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Vector

frechet_p2k1_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
frechet_p2k1_f1fw(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Vector

`frechet_p2k1_f2fa` *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
frechet_p2k1_f2fa(x, t0, v1, v2, v3, kloc)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_p2k1_f2fw` *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
frechet_p2k1_f2fw(x, t, v1, v2, v3, kloc)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_p2k1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_fd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Vector

`frechet_p2k1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

---

frechet_p2k1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_ldda(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
frechet_p2k1_lddda(x, t, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

3d array

frechet_p2k1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
frechet_p2k1_logf(params, x, t, kloc)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kloc	the known location parameter

Value

Scalar value.

frechet_p2k1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

frechet_p2k1_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

---

frechet_p2k1_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
frechet_p2k1_loglik(vv, x, t, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kloc	the known location parameter

Value

Scalar

frechet_p2k1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
frechet_p2k1_logscores(logscores, x, t, kloc)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
kloc	the known location parameter

Value

Two scalars

frechet_p2k1_means *frechet_k1 distribution: RHP mean*

Description

frechet_k1 distribution: RHP mean

Usage

```
frechet_p2k1_means(
  means,
  t0,
  ml_params,
  lddi,
  lddd,
  lambdad_rhp,
  nx,
  dim,
  kloc
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

---

frechet_p2k1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
frechet_p2k1_mu1fa(alpha, t0, v1, v2, v3, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Vector

frechet_p2k1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
frechet_p2k1_mu2fa(alpha, t0, v1, v2, v3, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Matrix

frechet_p2k1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
frechet_p2k1_p1fa(x, t0, v1, v2, v3, kloc)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kloc	the known location parameter

Value

Vector

`frechet_p2k1_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
frechet_p2k1_p2fa(x, t0, v1, v2, v3, kloc)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kloc`	the known location parameter

Value

Matrix

`frechet_p2k1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_pd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

---

frechet_p2k1_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
frechet_p2k1_pdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

frechet_p2k1_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
frechet_p2k1_predictordata(predictordata, x, t, t0, params, kloc)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kloc	the known location parameter

Value

Two vectors

frechet_p2k1_waic *Waic*

Description

Waic

Usage

```
frechet_p2k1_waic(
  waicscores,
  x,
  t,
  v1hat,
  v2hat,
  v3hat,
  kloc,
  lddi,
  lddd,
  lambdad
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

---

gamma_cp *Gamma Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgamma_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

rgamma_cp(
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
```

```
)

tgamma_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.

- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Gamma distribution has probability density function

$$f(x;\alpha,\sigma) = \frac{1}{\sigma^{\alpha}\Gamma(\alpha)} x^{\alpha-1} e^{-x/\sigma}$$

where $x \geq 0$ is the random variable and $\alpha > 0, \sigma > 0$ are the parameters.

The calibrating prior we use is

$$\pi(\alpha,\sigma) \propto \frac{1}{\alpha\sigma}$$

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d100gamma_example_data_v1
p=c(1:9)/10
q=qgamma_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgamma_cp)",
 main="Gamma: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

gamma_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gamma_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

gamma_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gamma_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gamma_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gamma_f2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

gamma_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gamma_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gamma_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gamma_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gamma_gg — *Second derivative matrix of the expected log-likelihood*

Description

Second derivative matrix of the expected log-likelihood

Usage

```
gamma_gg(v1, fd1, v2, fd2)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

gamma_gmn *One component of the second derivative of the expected log-likelihood*

Description

One component of the second derivative of the expected log-likelihood

Usage

```
gamma_gmn(alpha, v1, fd1, v2, fd2, mm, nn)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gamma_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gamma_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

gamma_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gamma_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gamma_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gamma_lddd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

gamma_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gamma_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

gamma_lmn — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gamma_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

gamma_lmnp — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gamma_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

gamma_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gamma_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

`gamma_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`gamma_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gamma_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

gamma_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gamma_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

gamma_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gamma_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

gamma_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
gamma_means(means, ml_params, lddi, lddd, lambdad_cp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_cp	derivative of the log prior
nx	length of training data
dim	number of parameters

Value

Two scalars

gamma_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gamma_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`gamma_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gamma_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`gamma_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
gamma_p1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

gamma_p2f *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
gamma_p2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

gamma_waic *Waic*

Description

Waic

Usage

```
gamma_waic(waicscores, x, v1hat, fd1, v2hat, fd2, lddi, lddd, lambdad, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gev_boot *Bootstrap*

Description

Bootstrap

Usage

```
gev_boot(x, n)
```

Arguments

x	a vector of training data values
n	number of random samples required

Value

A list containing a matrix of simulated parameter values

---

gev_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_checkmle(ml_params, minxi = -1, maxxi = 1)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

---

gev_cp *Generalized Extreme Value Distribution, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.

- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0),
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  customprior = 0,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  pwm = FALSE,
  debug = FALSE
)

rgev_cp(
  n,
  x,
  ics = c(0, 0, 0),
  minxi = -1,
  maxxi = 1,
  method = "rust",
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dgev_cp(
  x,
```

```
  y = x,
  ics = c(0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 1000,
  debug = FALSE
)

pgev_cp(
  x,
  y = x,
  ics = c(0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 1000,
  debug = FALSE
)

tgev_cp(method, n, x, ics = c(0, 0, 0), extramodels = FALSE, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)

customprior	a custom value for the slope of the log prior at the maxlik estimate
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
pwm	logical for whether to include PWM results (longer runtime)
debug	logical for turning on debug messages
n	the number of random samples required
method	character string that indicates whether to use rust method=rust or bootstrap method=boot
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions
boot	logical that indicates whether bootstrap-based posterior sampling calculations should be run or not (longer run time)
nboot	the number of posterior samples used in the bootstrap calculations

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution has distribution function

$$F(x;\mu,\sigma,\xi) = \exp\left(-t(x;\mu,\sigma,\xi)\right)$$

where

$$t(x;\mu,\sigma,\xi) = \begin{cases} \left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable and $\mu, \sigma > 0, \xi$ are the parameters.

The calibrating prior we use is given by

$$\pi(\mu,\sigma,\xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.

- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
shape=-0.4
x=fitdistcp::d110gev_example_data_v1
p=c(1:9)/10
q=qgev_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_cp)",
 main="GEVD: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_f1fa(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_f2fa(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k12_ppm_minusloglik *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
gev_k12_ppm_minusloglik(x)
```

Arguments

x	a vector of training data values

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

gev_k3_cp *Generalized Extreme Value Distribution with Known Shape, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fdalpha = 0.01,
  kshape = 0,
  means = FALSE,
  waicscores = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rgev_k3_cp(n, x, kshape = 0, rust = FALSE, mlcp = TRUE, debug = FALSE)
```

```
dgev_k3_cp(x, y = x, kshape = 0, rust = FALSE, nrust = 1000, debug = FALSE)

pgev_k3_cp(x, y = x, kshape = 0, rust = FALSE, nrust = 1000, debug = FALSE)

tgev_k3_cp(n, x, kshape = 0, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kshape	the known shape parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.

- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with known shape has distribution function

$$F(x;\mu,\sigma) = \exp\left(-t(x;\mu,\sigma)\right)$$

where

$$t(x;\mu,\sigma) = \begin{cases} \left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu, \sigma > 0$ are the parameters and ξ is known (hence the k3 in the name).

The calibrating prior we use is given by

$$\pi(\mu,\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),

- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
kshape=-0.4
x=fitdistcp::d053gev_k3_example_data_v1
p=c(1:9)/10
q=qgev_k3_cp(x,p,kshape=kshape,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_k3_cp)",
 main="GEV: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
muhat=q$ml_params[1]
sghat=q$ml_params[2]
xi=kshape
qmax=ifelse(xi<0,muhat-sghat/xi,Inf)
cat(" ml_params=",q$ml_params,",")
cat(" qmax=",qmax,"\n")
```

---

gev_k3_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gev_k3_f1fa(x, v1, v2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Vector

---

gev_k3_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gev_k3_f2fa(x, v1, v2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Matrix

gev_k3_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_k3_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_k3_ldda(x, v1, v2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Matrix

gev_k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_k3_lddda(x, v1, v2, kshape)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

3d array

gev_k3_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_k3_logf(params, x, kshape)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kshape	the known shape parameter

Value

Scalar value.

gev_k3_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_k3_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_k3_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gev_k3_loglik(vv, x, kshape)
```

Arguments

vv	parameters
x	a vector of training data values
kshape	the known shape parameter

Value

Scalar

gev_k3_means *MLE and RHP means*

Description

MLE and RHP means

Usage

```
gev_k3_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2, kshape)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kshape	the known shape parameter

Value

Two scalars

gev_k3_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_k3_mu1fa(alpha, v1, v2, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kshape	the known shape parameter

Value

Vector

`gev_k3_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_k3_mu2fa(alpha, v1, v2, kshape)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`v2`	second parameter
`kshape`	the known shape parameter

Value

Matrix

`gev_k3_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_pd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`gev_k3_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_k3_pdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gev_k3_waic` *Waic*

Description

Waic

Usage

```
gev_k3_waic(waicscores, x, v1hat, v2hat, kshape, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
kshape	the known shape parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

---

gev_ld12a *The combined derivative of the normalized log-likelihood*

Description

The combined derivative of the normalized log-likelihood

Usage

```
gev_ld12a(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_lda *The first derivative of the normalized log-likelihood*

Description

The first derivative of the normalized log-likelihood

Usage

```
gev_lda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_ldda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_lddda(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

gev_logfd — *First derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gev_loglik — *log-likelihood function*

Description

log-likelihood function

Usage

```
gev_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

gev_means *Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_means(
  means,
  ml_params,
  lddi,
  lddd,
  lambdad_rh_flat,
  lambdad_custom,
  nx,
  dim = 3
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_custom	custom value of the derivative of the log prior
nx	length of training data
dim	number of parameters

Value

Two scalars

gev_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_mu1fa(alpha, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gev_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_mu2fa(alpha, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gev_p1_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_p1_checkmle(ml_params, minxi = -1, maxxi = 1)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gev_p1_cp *Generalized Extreme Value Distribution with a Single Predictor on the Location, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0),
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rgev_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  ics = c(0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dgev_p1_cp(
  x,
```

```
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pgev_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p1_cp(n, x, t, ics = c(0, 0, 0, 0), extramodels = FALSE, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with a predictor has distribution function

$$F(x; a, b, \sigma, \xi) = \exp\left(-t(x; \mu(a, b), \sigma, \xi)\right)$$

where

$$t(x; \mu(a, b), \sigma, \xi) = \begin{cases} \left[1 + \xi\left(\frac{x-\mu(a,b)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(a,b)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, modelled as a function of parameters a, b and predictor t, and $\sigma > 0, \xi$ are the scale and shape parameters.

The calibrating prior we use is given by

$$\pi(a, b, \sigma, \xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will switch to maximum likelihood prediction if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. If this happens, it is reported in the revert2ml flag. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The `cp` outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x
tt=fitdistcp::d150gev_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qgev_p1_cp(x=x,t=tt,n0=n0,t0=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p1_cp)",
 main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_p1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

gev_p1_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gev_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

gev_p1_means *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p1_means(means, t0, ml_params, lddi, lddd, lambdad_rh_flat, nx, dim = 4)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
nx	length of training data
dim	number of parameters

Value

Two scalars

`gev_p1_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

`predictordata`	logical that indicates whether to calculate and return predictordata
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf

Value

Two vectors

`gev_p1_setics` *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_p1_setics(x, t, ics)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`ics`	initial conditions for the maximum likelihood search

Value

Vector

gev_p1_waic *Waic*

Description

Waic

Usage

```
gev_p1_waic(waicscores, x, t0, v1hat, v2hat, v3hat, v4hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
v4hat	fourth parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

gev_p12_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_p12_checkmle(ml_params, minxi = -1, maxxi = 1)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

---

gev_p12_cp *Generalized Extreme Value Distribution with Two Predictors, Predictions based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
```

```
  t02 = NA,
  n01 = NA,
  n02 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0, 0),
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rgev_p12_cp(
  n,
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  ics = c(0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE
)

dgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
```

```
  ics = c(0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 10,
  centering = TRUE,
  debug = FALSE
)

pgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p12_cp(
  n,
  x,
  t1,
  t2,
  ics = c(0, 0, 0, 0, 0),
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)

n02	an index for the predictor (specify either t02 or n02 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The GEV distribution with two predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, \xi) = \exp\left(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi)\right)$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi) = \begin{cases} \left[1 + \xi\left(\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1, $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2, and ξ is the shape parameter.

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, \xi) \propto 1$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

`q****` optionally returns the following, for EVT models only:

- `cp_pdf`: the density function at quantiles corresponding to input probabilities `p`. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The `cp` outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with linear predictor on the mean (`norm_p1`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),

- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
# example 1
x=fitdistcp::d151gev_p12_example_data_v1_x
tt=fitdistcp::d151gev_p12_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
p=c(1:9)/10
n01=10
n02=10
q=qgev_p12_cp(x=x,t1=t1,t2=t2,n01=n01,n02=n02,t01=NA,t02=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p12_cp)",
 main="GEVD w/ p12: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_p12_cp *Generalized Extreme Value Distribution with Two Predictors, Predictions based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y

- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0, 0),
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rgev_p12_cp(
  n,
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  ics = c(0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
```

```
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE
)

dgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 10,
  centering = TRUE,
  debug = FALSE
)

pgev_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p12_cp(
  n,
  x,
  t1,
```

```
  t2,
  ics = c(0, 0, 0, 0, 0),
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with two predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, \xi) = \exp\left(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi)\right)$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi) = \begin{cases} \left[1 + \xi\left(\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1, $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2, and ξ is the shape parameter.

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, \xi) \propto 1$$

as given in Jewson et al. (2025).

The code will switch to maximum likelihood prediction if the input data gives a maximum likelihood value for the shape parameter that lies outside the range `(minxi,maxxi)`, since outside this range there may be numerical problems. If this happens, it is reported in the `revert2ml` flag. Such values seldom occur in real observed data for maxima.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

`q****` optionally returns the following, for EVT models only:

- `cp_pdf`: the density function at quantiles corresponding to input probabilities `p`. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The `cp` outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),

- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
# example 1
x=fitdistcp::d151gev_p12_example_data_v1_x
tt=fitdistcp::d151gev_p12_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
p=c(1:9)/10
n01=10
n02=10
q=qgev_p12_cp(x=x,t1=t1,t2=t2,n01=n01,n02=n02,t01=NA,t02=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p12_cp)",
 main="GEVD w/ p12: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

gev_p12_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gev_p12_f1fa(x, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gev_p12_f1fw(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

---

gev_p12_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gev_p12_f2fa(x, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gev_p12_f2fw(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_fd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

---

gev_p12_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_fdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p12_ldda(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p12_lddda(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

---

gev_p12_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p12_logf(params, x, t1, t2)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

Value

Scalar value.

gev_p12_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_logfdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_logfddd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

---

gev_p12_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gev_p12_loglik(vv, x, t1, t2)
```

Arguments

vv	parameters
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

Value

Scalar

gev_p12_means — *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p12_means(means, t01, t02, ml_params, nx)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
ml_params	parameters
nx	length of training data

Value

Two scalars

gev_p12_mu1fa — *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p12_mu1fa(alpha, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

---

gev_p12_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p12_mu2fa(alpha, t01, t02, v1, v2, v3, v4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_pd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12_pdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

---

gev_p12_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p12_predictordata(predictordata, x, t1, t2, t01, t02, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
params	model parameters for calculating logf

Value

Two vectors

gev_p12_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_p12_setics(x, t1, t2, ics)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ics	initial conditions for the maximum likelihood search

Value

Vector

gev_p12_waic *Waic*

Description

Waic

Usage

```
gev_p12_waic(
  waicscores,
  x,
  t1,
  t2,
  v1hat,
  v2hat,
  v3hat,
  v4hat,
  v5hat,
  lddi,
  lddd,
  lambdad
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
v4hat	fourth parameter
v5hat	fifth parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

---

gev_p123_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_p123_checkmle(ml_params, minxi = -1, maxxi = 1, t1, t2, t3)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape

Value

No return value (just a message to the screen).

gev_p123_cp *Generalized Extreme Value Distribution with Three Predictors, Predictions based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0, 0, 0),
  fdalpha = 0.01,
  minxi = -1,
```

```
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
  debug = FALSE
)

rgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  ics = c(0, 0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE
)

dgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  minxi = -1,
```

```
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 10,
  centering = TRUE,
  debug = FALSE
)

pgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  ics = c(0, 0, 0, 0, 0, 0),
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t3	a vector of predictors for the shape, such that length(t3)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)

t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
n03	an index for the predictor (specify either t03 or n03 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.

- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with three predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, a_3, b_3) = \exp\left(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)\right)$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)) = \begin{cases} \left[1 + \xi(a_3, b_3)\left(\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right)\right]^{-1/\xi(a_3,b_3)} & \text{if } \xi(a_3, b_3) \neq 0 \\ \exp\left(-\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right) & \text{if } \xi(a_3, b_3) = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1, $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2, and $\xi = a_3 + b_3 t_3$ is the shape parameter, modelled as a function of parameters a_3, b_3 and predictor t_3.

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, a_3, b_3) \propto 1$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range `(minxi,maxxi)`, since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),

- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d152gev_p123_example_data_v1_x
tt=fitdistcp::d152gev_p123_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
t3=tt[,3]
p=c(1:9)/10
n01=10
n02=10
n03=10
q=qgev_p123_cp(x=x,t1=t1,t2=t2,t3=t3,n01=n01,n02=n02,n03=n03,t01=NA,t02=NA,t03=NA,
 p=p,rust=FALSE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p123_cp)",
 main="GEVD w/ p123: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_p123_cp *Generalized Extreme Value Distribution with Three Predictors, Predictions based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0, 0, 0),
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
```

```
  nrust = 1e+05,
  centering = TRUE,
  debug = FALSE
)

rgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  ics = c(0, 0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE
)

dgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 10,
  centering = TRUE,
  debug = FALSE
)
```

```
pgev_p123_cp(
  x,
  t1,
  t2,
  t3,
  t01 = NA,
  t02 = NA,
  t03 = NA,
  n01 = NA,
  n02 = NA,
  n03 = NA,
  y = x,
  ics = c(0, 0, 0, 0, 0, 0),
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p123_cp(
  n,
  x,
  t1,
  t2,
  t3,
  ics = c(0, 0, 0, 0, 0, 0),
  extramodels = FALSE,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t3	a vector of predictors for the shape, such that length(t3)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
n03	an index for the predictor (specify either t03 or n03 but not both)

p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.

- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with three predictors has distribution function

$$F(x; a_1, b_1, a_2, b_2, a_3, b_3) = \exp\left(-t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)\right)$$

where

$$t(x; \mu(a_1, b_1), \sigma(a_2, b_2), \xi(a_3, b_3)) = \begin{cases} \left[1 + \xi(a_3, b_3)\left(\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right)\right]^{-1/\xi(a_3,b_3)} & \text{if } \xi(a_3, b_3) \neq 0 \\ \exp\left(-\frac{x-\mu(a_1,b_1)}{\sigma(a_2,b_2)}\right) & \text{if } \xi(a_3, b_3) = 0 \end{cases}$$

where x is the random variable, $\mu = a_1 + b_1 t_1$ is the location parameter, modelled as a function of parameters a_1, b_1 and predictor t_1, $\sigma = e^{a_2 + b_2 t_2}$ is the scale parameter, modelled as a function of parameters a_2, b_2 and predictor t_2, and $\xi = a_3 + b_3 t_3$ is the shape parameter, modelled as a function of parameters a_3, b_3 and predictor t_3.

The calibrating prior we use is given by

$$\pi(a_1, b_1, a_2, b_2, a_3, b_3) \propto 1$$

as given in Jewson et al. (2025).

The code will switch to maximum likelihood prediction if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. If this happens, it is reported in the revert2ml flag. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

`q****` optionally returns the following, for EVT models only:

- `cp_pdf`: the density function at quantiles corresponding to input probabilities `p`. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The `cp` outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d152gev_p123_example_data_v1_x
tt=fitdistcp::d152gev_p123_example_data_v1_t
t1=tt[,1]
t2=tt[,2]
t3=tt[,3]
p=c(1:9)/10
n01=10
n02=10
n03=10
q=qgev_p123_cp(x=x,t1=t1,t2=t2,t3=t3,n01=n01,n02=n02,n03=n03,t01=NA,t02=NA,t03=NA,
 p=p,rust=FALSE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p123_cp)",
 main="GEVD w/ p123: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_p123_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gev_p123_f1fa(x, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

---

gev_p123_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gev_p123_f1fw(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gev_p123_f2fa(x, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gev_p123_f2fw(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

---

gev_p123_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_fd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p123_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_fdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p123_ldda(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

---

gev_p123_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p123_lddda(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

3d array

gev_p123_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p123_logf(params, x, t1, t2, t3)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape

Value

Scalar value.

gev_p123_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_logfdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter

v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

`gev_p123_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_logfddd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

3d array

gev_p123_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gev_p123_loglik(vv, x, t1, t2, t3)
```

Arguments

vv	parameters
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape

Value

Scalar

gev_p123_means *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p123_means(means, t01, t02, t03, ml_params, nx)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
ml_params	parameters
nx	length of training data

Value

Two scalars

`gev_p123_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p123_mu1fa(alpha, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`t03`	a single value of the predictor (specify either `t03` or `n03` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`v5`	fifth parameter
`v6`	sixth parameter

Value

Vector

gev_p123_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p123_mu2fa(alpha, t01, t02, t03, v1, v2, v3, v4, v5, v6)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_pd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

---

gev_p123_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p123_pdd(x, t1, t2, t3, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p123_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p123_predictordata(x, t1, t2, t3, t01, t02, t03, params)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
params	model parameters for calculating logf

Value

Two vectors

gev_p123_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_p123_setics(x, t1, t2, t3, ics)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ics	initial conditions for the maximum likelihood search

Value

Vector

`gev_p123_waic` *Waic*

Description

Waic

Usage

```
gev_p123_waic(
  waicscores,
  x,
  t1,
  t2,
  t3,
  v1h,
  v2h,
  v3h,
  v4h,
  v5h,
  v6h,
  lddi,
  lddd,
  lambdad
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t3`	a vector of predictors for the shape
`v1h`	first parameter
`v2h`	second parameter
`v3h`	third parameter
`v4h`	fourth parameter
`v5h`	fifth parameter
`v6h`	sixth parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior

Value

Two numeric values.

`gev_p12k3_f1fa` *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gev_p12k3_f1fa(x, t01, t02, v1, v2, v3, v4, kshape)
```

Arguments

`x`	a vector of training data values
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`kshape`	the known shape parameter

Value

Vector

`gev_p12k3_f1fw` *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gev_p12k3_f1fw(x, t1, t2, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Vector

---

gev_p12k3_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gev_p12k3_f2fa(x, t01, t02, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

gev_p12k3_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gev_p12k3_f2fw(x, t1, t2, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

gev_p12k3_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_fd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

---

gev_p12k3_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_fdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p12k3_ldda(x, t1, t2, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

gev_p12k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p12k3_lddda(x, t1, t2, v1, v2, v3, v4, kshape)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

3d array

---

gev_p12k3_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_logfdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p12k3_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_logfddd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

gev_p12k3_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p12k3_mu1fa(alpha, t01, t02, v1, v2, v3, v4, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Vector

---

gev_p12k3_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p12k3_mu2fa(alpha, t01, t02, v1, v2, v3, v4, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
kshape	the known shape parameter

Value

Matrix

gev_p12k3_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_pd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p12k3_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p12k3_pdd(x, t1, t2, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

---

gev_p1a_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gev_p1a_f1fa(x, t0, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1a_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gev_p1a_f1fw(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1a_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gev_p1a_f2fa(x, t0, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

`gev_p1a_f2fw` *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gev_p1a_f2fw(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`gev_p1a_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1a_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

---

gev_p1a_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1a_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1a_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1a_ldda(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1a_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p1a_lddda(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

`gev_p1a_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1a_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`gev_p1a_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1a_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

`gev_p1a_mu1fa` *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p1a_mu1fa(alpha, t0, v1, v2, v3, v4)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

gev_p1a_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p1a_mu2fa(alpha, t0, v1, v2, v3, v4)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1a_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1a_pd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

`gev_p1a_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1a_pdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`gev_p1b_f1fa` *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gev_p1b_f1fa(x, t0a, t0b, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

---

gev_p1b_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gev_p1b_f1fw(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p1b_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gev_p1b_f2fa(x, t0a, t0b, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p1b_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gev_p1b_f2fw(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

---

gev_p1b_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1b_fd(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p1b_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1b_fdd(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p1b_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1b_ldda(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

---

gev_p1b_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p1b_lddda(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

gev_p1b_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1b_logfdd(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p1b_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1b_logfddd(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

3d array

---

gev_p1b_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p1b_mu1fa(alpha, t0a, t0b, v1, v2, v3, v4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

gev_p1b_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p1b_mu2fa(alpha, t0a, t0b, v1, v2, v3, v4, v5)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p1b_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1b_pd(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Vector

---

gev_p1b_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1b_pdd(x, ta, tb, v1, v2, v3, v4, v5)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter

Value

Matrix

gev_p1c_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gev_p1c_f1fa(x, t0a, t0b, t0c, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
t0c	a single value of the predictor, for the third column of the predictor (specify either t0c or n0c but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p1c_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gev_p1c_f1fw(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

---

gev_p1c_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gev_p1c_f2fa(x, t0a, t0b, t0c, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
t0c	a single value of the predictor, for the third column of the predictor (specify either t0c or n0c but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

`gev_p1c_f2fw` *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gev_p1c_f2fw(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

`x`	a vector of training data values
`ta`	a vector of predictors for the mean (first column)
`tb`	a vector of predictors for the mean (second column)
`tc`	a vector of predictors for the mean (third column)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`v5`	fifth parameter
`v6`	sixth parameter

Value

Matrix

gev_p1c_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1c_fd(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p1c_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1c_fdd(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

---

gev_p1c_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1c_ldda(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

gev_p1c_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p1c_lddda(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

3d array

gev_p1c_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1c_logfdd(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

---

gev_p1c_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1c_logfddd(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

3d array

gev_p1c_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p1c_mu1fa(alpha, t0a, t0b, t0c, v1, v2, v3, v4, v5, v6)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
t0c	a single value of the predictor, for the third column of the predictor (specify either t0c or n0c but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

gev_p1c_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p1c_mu2fa(alpha, t0a, t0b, t0c, v1, v2, v3, v4, v5, v6)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
t0c	a single value of the predictor, for the third column of the predictor (specify either t0c or n0c but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Matrix

---

gev_p1c_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1c_pd(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

x	a vector of training data values
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter
v5	fifth parameter
v6	sixth parameter

Value

Vector

`gev_p1c_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1c_pdd(x, ta, tb, tc, v1, v2, v3, v4, v5, v6)
```

Arguments

`x`	a vector of training data values
`ta`	a vector of predictors for the mean (first column)
`tb`	a vector of predictors for the mean (second column)
`tc`	a vector of predictors for the mean (third column)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter
`v5`	fifth parameter
`v6`	sixth parameter

Value

Matrix

gev_p1k3_cp *GEV Distribution with Known Shape with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  fdalpha = 0.01,
  kshape = 0,
  means = FALSE,
  waicscores = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
```

```
  centering = TRUE,
  debug = FALSE
)

rgev_p1k3_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  kshape = 0,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE
)

dgev_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  kshape = 0,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pgev_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  kshape = 0,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p1k3_cp(n, x, t, kshape = 0, debug = FALSE)
```

Arguments

x	a vector of training data values

t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kshape	the known shape parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with known shape with a predictor has distribution function

$$F(x; a, b, \sigma) = \exp\left(-t(x; \mu(a, b), \sigma)\right)$$

where

$$t(x; a, b, \sigma) = \begin{cases} \left[1 + \xi\left(\frac{x-\mu(a,b)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(a,b)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, $\sigma > 0$ is the shape parameter and ξ is known (hence the k3 in the name).

The calibrating prior we use is given by

$$\pi(\mu, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x #use data for 150
tt=fitdistcp::d150gev_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qgev_p1k3_cp(x=x,t=tt,n0=n0,t0=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p1k3_cp)",
 main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gev_p1k3_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gev_p1k3_f1fa(x, t0, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Vector

---

gev_p1k3_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gev_p1k3_f1fw(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Vector

---

gev_p1k3_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gev_p1k3_f2fa(x, t0, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

`gev_p1k3_f2fw` *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gev_p1k3_f2fw(x, t, v1, v2, v3, kshape)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`kshape`	the known shape parameter

Value

Matrix

`gev_p1k3_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

---

gev_p1k3_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_fdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

gev_p1k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gev_p1k3_ldda(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

gev_p1k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gev_p1k3_lddda(x, t, v1, v2, v3, kshape)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

3d array

`gev_p1k3_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p1k3_logf(params, x, t, kshape)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`kshape`	the known shape parameter

Value

Scalar value.

`gev_p1k3_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`gev_p1k3_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

3d array

`gev_p1k3_loglik` *GEV-with-known-shape-with-p1 observed log-likelihood function*

Description

GEV-with-known-shape-with-p1 observed log-likelihood function

Usage

```
gev_p1k3_loglik(vv, x, t, kshape)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kshape	the known shape parameter

Value

Scalar

---

gev_p1k3_means *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p1k3_means(means, t0, ml_params, kshape, nx)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
kshape	the known shape parameter
nx	length of training data

Value

Two scalars

gev_p1k3_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gev_p1k3_mu1fa(alpha, t0, v1, v2, v3, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Vector

gev_p1k3_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gev_p1k3_mu2fa(alpha, t0, v1, v2, v3, kshape)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kshape	the known shape parameter

Value

Matrix

`gev_p1k3_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_pd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Vector

`gev_p1k3_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_p1k3_pdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

---

gev_p1k3_predictordata

Predicted Parameter and Generalized Residuals

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p1k3_predictordata(predictordata, x, t, t0, params, kshape)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kshape	the known shape parameter

Value

Two vectors

gev_p1k3_waic *Waic*

Description

Waic

Usage

```
gev_p1k3_waic(
  waicscores,
  x,
  t,
  v1hat,
  v2hat,
  v3hat,
  kshape,
  lddi,
  lddd,
  lambdad
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
kshape	the known shape parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

gev_p1n_checkmle *Check MLE*

Description

Check MLE

Usage

```
gev_p1n_checkmle(ml_params, minxi = -1, maxxi = 1)
```

Arguments

ml_params	parameters
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gev_p1n_cp *Generalized Extreme Value Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1n_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rgev_p1n_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dgev_p1n_cp(
  x,
  t,
  t0 = NA,
```

```
  n0 = NA,
  y = x,
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pgev_p1n_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p1n_cp(n, x, t, extramodels = FALSE, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)

pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer run-time)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The GEV distribution with a 2d predictor has distribution function

$$F(x; a, b_1, b_2, \sigma, \xi) = \exp\left(-t(x; \mu(a, b_1, b_2), \sigma, \xi)\right)$$

where

$$t(x; \mu(a, b_1, b_2), \sigma, \xi) = \begin{cases} \left[1 + \xi\left(\frac{x-\mu(a,b_1,b_2)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(a,b_1,b_2)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = a + b_1 t_1 + b_2 t_2$ is the location parameter, modelled as a function of parameters a, b_1, b_2 and predictor t_1, t_2, and $\sigma > 0, \xi$ are the scale and shape parameters.

The calibrating prior we use is given by

$$\pi(a, b_1, b_2, \sigma, \xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2024) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x
t1=fitdistcp::d150gev_p1_example_data_v1_t
t2=sample(t1)
t=cbind(t1,t2)
p=c(1:9)/10
```

```
n0=c(10,10)
q=qgev_p1n_cp(x=x,t=t,n0=n0,t0=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p1n_cp)",
 main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

`gev_p1n_cp` *Generalized Extreme Value Distribution with Multiple Predictors on the Location, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgev_p1n_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
```

```
  p = seq(0.1, 0.9, 0.1),
  fdalpha = 0.01,
  minxi = -1,
  maxxi = 1,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rgev_p1n_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dgev_p1n_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pgev_p1n_cp(
  x,
  t,
```

```
  t0 = NA,
  n0 = NA,
  y = x,
  minxi = -1,
  maxxi = 1,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgev_p1n_cp(n, x, t, extramodels = FALSE, debug = FALSE)
```

Arguments

x	a vector of training data values
t	predictors, which can be a vector, or a matrix with 1, 2 or 3 columns
t0	a single value for each predictor, as 1, 2 or 3 scalars (specify t0 or n0 but not both)
n0	an index for the each predictor, as 1, 2 or 3 integers (specify t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages

n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

1, 2 or 3 predictors on the location parameter are supported. For instance, the GEV distribution with 2 predictors has distribution function

$$F(x;\alpha,\beta_1,\beta_2,\sigma,\xi) = \exp\left(-t(x;\mu(\alpha,\beta_1,\beta_2),\sigma,\xi)\right)$$

where

$$t(x;\mu(\alpha,\beta_1,\beta_2),\sigma,\xi) = \begin{cases} \left[1+\xi\left(\frac{x-\mu(\alpha,\beta_1,\beta_2)}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu(\alpha,\beta_1,\beta_2)}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable, $\mu = \alpha + \beta_1 t_1 + \beta_2 t_2$ is the location parameter, modelled as a function of parameters α, β_1, β_2 and predictor t_1, t_2, and $\sigma > 0, \xi$ are the scale and shape parameters.

The calibrating prior we use is given by

$$\pi(\alpha,\beta_1,\beta_2,\sigma,\xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will switch to maximum likelihood prediction if the input data gives a maximum likelihood value for the shape parameter that lies outside the range `(minxi,maxxi)`, since outside this range there may be numerical problems. If this happens, it is reported in the `revert2ml` flag. Such values seldom occur in real observed data for maxima.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
# example 1
x=fitdistcp::d150gev_p1_example_data_v1_x
t1=fitdistcp::d150gev_p1_example_data_v1_t
t2=sample(t1)
t=cbind(t1,t2)
p=c(1:9)/10
n0=c(10,10)
q=qgev_p1n_cp(x=x,t=t,n0=n0,t0=NA,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgev_p1n_cp)",
 main="GEVD w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

gev_p1n_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gev_p1n_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

gev_p1n_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gev_p1n_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

`gev_p1n_means` *Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gev_p1n_means(
  means,
  t0,
  ml_params,
  lddi,
  lddd,
  lambdad_rh_flat,
  nx,
  dim = (nt + 3)
)
```

Arguments

`means`	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ml_params`	parameters
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad_rh_flat`	derivative of the log CRHP-FLAT prior
`nx`	length of training data
`dim`	number of parameters

Value

Two scalars

gev_p1n_n1_exampledata *GEV_p1n n=1 example data*

Description

GEV_p1n n=1 example data

Usage

```
gev_p1n_n1_exampledata(iseed)
```

Arguments

iseed	The random seed

Value

A list containing data to run an example

gev_p1n_n2_exampledata *GEV_p1n n=2 example data*

Description

GEV_p1n n=2 example data

Usage

```
gev_p1n_n2_exampledata(iseed)
```

Arguments

iseed	The random seed

Value

A list containing data to run an example

gev_p1n_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
gev_p1n_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

gev_p1n_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_p1n_setics(x, t)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors

Value

Vector

gev_p1n_waic *Waic*

Description

Waic

Usage

```
gev_p1n_waic(waicscores, x, t0, vhat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
vhat	vector of all parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

gev_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gev_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gev_pdd(x, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gev_pwm_params` *PWM parameter estimation*

Description

PWM parameter estimation

Usage

```
gev_pwm_params(x)
```

Arguments

`x`	a vector of training data values

Value

Vector

gev_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
gev_setics(x, ics)
```

Arguments

x	a vector of training data values
ics	initial conditions for the maximum likelihood search

Value

Vector

gev_waic *Waic*

Description

Waic

Usage

```
gev_waic(waicscores, x, v1hat, v2hat, v3hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

gnorm_k3_cp *Generalized Normal Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgnorm_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kbeta = 4,
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)
```

```
rgnorm_k3_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dgnorm_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pgnorm_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kbeta = 4,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tgnorm_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kbeta = 4, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kbeta	the known beta parameter
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter

means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The generalized normal distribution has probability density function

$$f(x;\mu,\alpha) = \frac{\beta}{2\alpha\Gamma(1/\beta)} e^{-(|x-\mu|/\alpha)^\beta}$$

where x is the random variable, $\mu, \alpha > 0$ are the parameters and we consider β to be known (hence the k3 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha) \propto \frac{1}{\alpha}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d032gnorm_k3_example_data_v1
p=c(1:9)/10
q=qgnorm_k3_cp(x,p,kbeta=4,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgnorm_k3_cp)",
 main="gnorm: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`gnorm_k3_f1f` *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
gnorm_k3_f1f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kbeta`	the known beta parameter

Value

Matrix

gnorm_k3_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gnorm_k3_f1fa(x, v1, v2, kbeta)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kbeta	the known beta parameter

Value

Vector

gnorm_k3_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
gnorm_k3_f2f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

gnorm_k3_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gnorm_k3_f2fa(x, v1, v2, kbeta)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kbeta	the known beta parameter

Value

Matrix

Matrix

gnorm_k3_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gnorm_k3_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gnorm_k3_ldd — *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
gnorm_k3_ldd(x, v1, d1, v2, fd2, kbeta)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Square scalar matrix

`gnorm_k3_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gnorm_k3_ldda(x, v1, v2, kbeta)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kbeta`	the known beta parameter

Value

Matrix

`gnorm_k3_lddd` *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
gnorm_k3_lddd(x, v1, d1, v2, fd2, kbeta)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kbeta`	the known beta parameter

Value

Cubic scalar array

`gnorm_k3_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gnorm_k3_lddda(x, v1, v2, kbeta)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter
`kbeta`	the known beta parameter

Value

3d array

`gnorm_k3_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gnorm_k3_lmn(x, v1, d1, v2, fd2, kbeta, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

---

gnorm_k3_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gnorm_k3_logf(params, x, kbeta)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kbeta	the known beta parameter

Value

Scalar value.

gnorm_k3_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gnorm_k3_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gnorm_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gnorm_k3_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gnorm_k3_loglik(vv, x, kbeta)
```

Arguments

vv	parameters
x	a vector of training data values
kbeta	the known beta parameter

Value

Scalar

gnorm_k3_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gnorm_k3_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, kbeta, aderivs)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

gnorm_k3_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
gnorm_k3_mu1f(alpha, v1, d1, v2, fd2, kbeta)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

Matrix

gnorm_k3_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
gnorm_k3_mu2f(alpha, v1, d1, v2, fd2, kbeta)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

`gnorm_k3_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
gnorm_k3_p1f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kbeta`	the known beta parameter

Value

Matrix

`gnorm_k3_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
gnorm_k3_p2f(y, v1, d1, v2, fd2, kbeta)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter

Value

3d array

---

gnorm_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
gnorm_lmnp(x, v1, d1, v2, fd2, kbeta, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

gnorm_waic *Waic for RUST*

Description

Waic for RUST

Usage

```
gnorm_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  kbeta,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kbeta	the known beta parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

gpd_k1_checkmle *Check MLE*

Description

Check MLE

Usage

```
gpd_k1_checkmle(ml_params, kloc, minxi = -1, maxxi = 2)
```

Arguments

ml_params	parameters
kloc	the known location parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

No return value (just a message to the screen).

gpd_k1_cp *Generalized Pareto Distribution with Known Location Parameter, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgpd_k1_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kloc = 0,
  ics = c(0, 0),
  fdalpha = 0.01,
  customprior = 0,
  minxi = -1,
  maxxi = 2,
  means = FALSE,
  waicscores = FALSE,
  extramodels = FALSE,
  pdf = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rgpd_k1_cp(
  n,
  x,
  kloc = 0,
  ics = c(0, 0),
  minxi = -1,
  maxxi = 2,
  extramodels = FALSE,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dgpd_k1_cp(
  x,
  y = x,
  kloc = 0,
  ics = c(0, 0),
  customprior = 0,
  minxi = -1,
```

```
  maxxi = 2,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE
)

pgpd_k1_cp(
  x,
  y = x,
  kloc = 0,
  ics = c(0, 0),
  customprior = 0,
  minxi = -1,
  maxxi = 2,
  extramodels = FALSE,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE
)

tgpd_k1_cp(n, x, kloc = 0, ics = c(0, 0), extramodels = FALSE, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kloc	the known location parameter
ics	initial conditions for the maximum likelihood search
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
customprior	a custom value for the slope of the log prior at the maxlik estimate
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)

rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Details

The GP distribution has exceedcance distribution function

$$S(x;\mu,\sigma,\xi) = \begin{cases} \left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi} & \text{if } \xi \neq 0 \\ \exp\left(-\frac{x-\mu}{\sigma}\right) & \text{if } \xi = 0 \end{cases}$$

where x is the random variable and $\mu, \sigma > 0, \xi$ are the parameters.

The calibrating prior we use is given by

$$\pi(\mu,\sigma,\xi) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

The code will stop with an error if the input data gives a maximum likelihood value for the shape parameter that lies outside the range (minxi,maxxi), since outside this range there may be numerical problems. Such values seldom occur in real observed data for maxima.

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible

- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.

- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d120gpd_k1_example_data_v1
p=c(1:9)/10
q=qgpd_k1_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgpd_k1_cp)",
 main="GPD: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue",lwd=2)
cat(" ml_params=",q$ml_params,"\n")
```

---

gpd_k1_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gpd_k1_f1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

---

gpd_k1_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gpd_k1_f2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k1_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gpd_k1_ldda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gpd_k1_lddda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

gpd_k1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gpd_k1_logf(params, x, kloc)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kloc	the known location parameter

Value

Scalar value.

gpd_k1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k1_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gpd_k1_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gpd_k1_loglik(vv, x, kloc)
```

Arguments

vv	parameters
x	a vector of training data values
kloc	the known location parameter

Value

Scalar

gpd_k1_means *Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1*

Description

Analytical Expressions for Predictive Means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
gpd_k1_means(
  means,
  ml_params,
  lddi,
  lddd,
  lambdad_rh_flat,
  lambdad_jp,
  nx,
  dim = 2,
  kloc = 0
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
nx	length of training data
dim	number of parameters
kloc	the known location parameter

Value

Two scalars

gpd_k1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gpd_k1_mu1fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

gpd_k1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gpd_k1_mu2fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k1_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k1_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k1_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k1_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
gpd_k1_setics(x, ics)
```

Arguments

x	a vector of training data values
ics	initial conditions for the maximum likelihood search

Value

Vector

gpd_k1_waic *Waic*

Description

Waic

Usage

```
gpd_k1_waic(waicscores, x, v1hat, v2hat, kloc, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
kloc	the known location parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

gpd_k13_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gpd_k13_f1fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

gpd_k13_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gpd_k13_f2fa(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k13_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k13_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k13_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gpd_k13_ldda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k13_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gpd_k13_lddda(x, v1, v2, kloc)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

3d array

gpd_k13_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gpd_k13_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

gpd_k13_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gpd_k13_mu1fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Vector

gpd_k13_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gpd_k13_mu2fa(alpha, v1, v2, kloc)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter
kloc	the known location parameter

Value

Matrix

gpd_k13_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_pd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gpd_k13_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gpd_k13_pdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gumbel_cp *Gumbel Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgumbel_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rgumbel_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE)

dgumbel_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)
```

```
pgumbel_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

tgumbel_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Gumbel distribution has distribution function

$$F(x;\mu,\sigma) = \exp\left(-\exp\left(-\frac{x-\mu}{\sigma}\right)\right)$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d050gumbel_example_data_v1
p=c(1:9)/10
q=qgumbel_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgumbel_cp)",
 main="Gumbel: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

gumbel_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
gumbel_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

---

gumbel_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
gumbel_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`gumbel_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`gumbel_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

---

gumbel_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gumbel_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

---

gumbel_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gumbel_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

---

gumbel_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
gumbel_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

---

gumbel_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

---

gumbel_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

---

gumbel_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
gumbel_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

---

gumbel_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gumbel_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

---

gumbel_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
gumbel_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

gumbel_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gumbel_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

gumbel_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gumbel_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_p1_cp *Gumbel Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qgumbel_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rgumbel_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dgumbel_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)
```

```
pgumbel_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tgumbel_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Gumbel distribution with a predictor has distribution function

$$F(x; a, b, \sigma) = \exp\left(-\exp\left(-\frac{x - \mu(a, b)}{\sigma}\right)\right)$$

where x is the random variable, $\mu = a + bt$ is the shape parameter as a function of parameters a, b and predictor t, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),

- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d070gumbel_p1_example_data_v1_x
tt=fitdistcp::d070gumbel_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qgumbel_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qgumbel_p1_cp)",
 main="Gumbel w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

gumbel_p1_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
gumbel_p1_f1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

---

gumbel_p1_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
gumbel_p1_f1fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
gumbel_p1_f2fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gumbel_p1_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
gumbel_p1_f2fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gumbel_p1_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

gumbel_p1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
gumbel_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

gumbel_p1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
gumbel_p1_lddda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`gumbel_p1_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
gumbel_p1_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

`gumbel_p1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gumbel_p1_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`gumbel_p1_loglik` *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
gumbel_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

`gumbel_p1_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
gumbel_p1_logscores(logscores, x, t)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Two scalars

`gumbel_p1_means` *Gumbel distribution: RHP mean*

Description

Gumbel distribution: RHP mean

Usage

```
gumbel_p1_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

---

gumbel_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
gumbel_p1_mu1fa(alpha, t0, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
gumbel_p1_mu2fa(alpha, t0, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gumbel_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
gumbel_p1_p1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

gumbel_p1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
gumbel_p1_p2fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

gumbel_p1_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`gumbel_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_p1_pdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`gumbel_p1_predictordata`
Predicted Parameter and Generalized Residuals

Description

Predicted Parameter and Generalized Residuals

Usage

```
gumbel_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

---

gumbel_p1_waic *Waic*

Description

Waic

Usage

```
gumbel_p1_waic(waicscores, x, t, v1hat, v2hat, v3hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

gumbel_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
gumbel_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gumbel_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
gumbel_p2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

gumbel_pdd — *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
gumbel_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

gumbel_waic *Waic*

Description

Waic

Usage

```
gumbel_waic(waicscores, x, v1hat, v2hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

halfnorm_cp *Half-Normal Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y

- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qhalfnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rhalfnorm_cp(
  n,
  x,
  fd1 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dhalfnorm_cp(
  x,
  y = x,
  fd1 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

phalfnorm_cp(
  x,
```

```
  y = x,
  fd1 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

thalfnorm_cp(n, x, fd1 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.

- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The half-normal distribution has probability density function

$$f(x;\theta) = \frac{2\theta}{\pi} e^{-\theta^2 x^2/\pi}$$

where $x \geq 0$ is the random variable and $\theta > 0$ is the parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\theta) \propto \frac{1}{\theta}$$

as given in Jewson et al. (2025). Some other authors may parametrize the half-normal differently.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d020halfnorm_example_data_v1
p=c(1:9)/10
q=qhalfnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles)
xmax=max(q$ml_quantiles,q$cp_quantiles)
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qhalfnorm_cp)",
 main="Halfnorm: quantile estimates")
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

halfnorm_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
halfnorm_f1f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

halfnorm_f1fa *The first derivative of the density*

Description

The first derivative of the density

The first derivative of the density

Usage

```
halfnorm_f1fa(x, v1)
```


```
halfnorm_f1fa(x, v1)
```


Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

Vector

halfnorm_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
halfnorm_f2f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

halfnorm_f2fa *The second derivative of the density*

Description

The second derivative of the density

The second derivative of the density

Usage

```
halfnorm_f2fa(x, v1)

halfnorm_f2fa(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

halfnorm_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_fd(x, v1)

halfnorm_fd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Vector

Vector

`halfnorm_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_fdd(x, v1)

halfnorm_fdd(x, v1)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter

Value

Matrix

Matrix

`halfnorm_gg` *Expected information matrix*

Description

Expected information matrix

Usage

```
halfnorm_gg(v1, fd1)
```

Arguments

v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

halfnorm_gg11 *Second derivative of the expected log-likelihood*

Description

Second derivative of the expected log-likelihood

Usage

```
halfnorm_gg11(alpha, v1, fd1)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

---

halfnorm_l111 *Third derivative of the normalized log-likelihood*

Description

Third derivative of the normalized log-likelihood

Usage

```
halfnorm_l111(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Scalar value

---

halfnorm_ldd *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
halfnorm_ldd(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

---

halfnorm_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

The second derivative of the normalized log-likelihood

Usage

```
halfnorm_ldda(x, v1)

halfnorm_ldda(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

---

halfnorm_lddd *Third derivative tensor of the log-likelihood*

Description

Third derivative tensor of the log-likelihood

Usage

```
halfnorm_lddd(x, v1, fd1)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

---

halfnorm_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

The third derivative of the normalized log-likelihood

Usage

```
halfnorm_lddda(x, v1)
```


```
halfnorm_lddda(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

3d array

`halfnorm_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
halfnorm_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

`halfnorm_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_logfdd(x, v1)

halfnorm_logfdd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

Matrix

Matrix

`halfnorm_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
halfnorm_logfddd(x, v1)

halfnorm_logfddd(x, v1)
```

Arguments

x	a vector of training data values
v1	first parameter

Value

3d array

3d array

halfnorm_loglik *Log-likelihood function*

Description

Log-likelihood function

Usage

```
halfnorm_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

halfnorm_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
halfnorm_logscores(logscores, x, fd1 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

halfnorm_means *MLE and RHP predictive means RHP mean based on the expectation of DMGS equation 2.1*

Description

MLE and RHP predictive means RHP mean based on the expectation of DMGS equation 2.1

Usage

```
halfnorm_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 1)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

halfnorm_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
halfnorm_mu1f(alpha, v1, fd1)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`halfnorm_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
halfnorm_mu2f(alpha, v1, fd1)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`halfnorm_p1f` *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
halfnorm_p1f(y, v1, fd1)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

halfnorm_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
halfnorm_p2f(y, v1, fd1)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

halfnorm_waic *Waic*

Description

Waic

Usage

```
halfnorm_waic(waicscores, x, v1hat, fd1, lddi, lddd, lambdad, aderivs)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`ifvectorthenmatrix` *If a variable is a vector, convert it to a matrix*

Description

If a variable is a vector, convert it to a matrix

Usage

```
ifvectorthenmatrix(t)
```

Arguments

`t`	a vector or matrix of predictors

Value

Vector

`invgamma_cp` *Inverse Gamma Distribution, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qinvgamma_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

rinvgamma_cp(
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dinvgamma_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

pinvgamma_cp(
```

```
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tinvgamma_cp(n, x, fd1 = 0.01, fd2 = 0.01, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.

- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Inverse Gamma distribution has probability density function

$$f(x;\alpha,\sigma) = \frac{1}{x\Gamma(\alpha)}\left(\frac{\sigma}{x}\right)^{\alpha} e^{-\sigma/x}$$

where $x \geq 0$ is the random variable and $\alpha > 0, \sigma > 0$ are the parameters.

The calibrating prior we use is

$$\pi(\alpha,\sigma) \propto \frac{1}{\alpha\sigma}$$

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d101invgamma_example_data_v1
p=c(1:9)/10
q=qinvgamma_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qinvgamma_cp)",
 main="Invgamma: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

invgamma_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
invgamma_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgamma_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
invgamma_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgamma_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
invgamma_f2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgamma_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
invgamma_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgamma_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_ldd — *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
invgamma_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

invgamma_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
invgamma_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgamma_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
invgamma_lddd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

invgamma_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
invgamma_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

invgamma_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgamma_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

`invgamma_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgamma_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate
`rr`	an index for which derivative to calculate

Value

Scalar value

`invgamma_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
invgamma_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

`invgamma_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`invgamma_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgamma_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

---

invgamma_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
invgamma_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

---

invgamma_logscores *Log scores for MLE and cp predictions calculated using leave-one-out*

Description

Log scores for MLE and cp predictions calculated using leave-one-out

Usage

```
invgamma_logscores(logscores, x, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`invgamma_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
invgamma_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgamma_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
invgamma_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgamma_p1f *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
invgamma_p1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`invgamma_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
invgamma_p2f(y, v1, fd1, v2, fd2)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`invgamma_waic` *Waic*

Description

Waic

Usage

```
invgamma_waic(
  waicscores,
  x,
  v1hat,
  fd1,
  v2hat,
  fd2,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

invgauss_cp *Inverse Gauss Distribution, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qinvgauss_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  fd1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

rinvgauss_cp(
  n,
  x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  prior = "type 1",
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dinvgauss_cp(
  x,
  y = x,
```

```
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

pinvgauss_cp(
  x,
  y = x,
  fd1 = 0.01,
  fd2 = 0.01,
  rust = FALSE,
  nrust = 1000,
  prior = "type 1",
  debug = FALSE,
  aderivs = TRUE
)

tinvgauss_cp(n, x, fd1 = 0.01, fd2 = 0.01, prior = "type 1", debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
prior	logical indicating which prior to use
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.

n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Inverse Gaussian distribution has probability density function

$$f(x;\mu,\phi) = \left(\frac{1}{2\pi\phi x^3}\right)^{1/2} \exp\left(-\frac{(x-\mu)^2}{2\mu^2\phi x}\right)$$

where $x \geq 0$ is the random variable and $\mu > 0, \phi > 0$ are the parameters.

The calibrating prior we use by default is

$$\pi(\alpha,\sigma) \propto \frac{1}{\phi}$$

The prior

$$\pi(\alpha,\sigma) \propto \frac{1}{\mu\phi}$$

is also available as an option with `prior="type 2"`.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),
- Uniform (`unif`),
- Weibull (`weibull`),
- Weibull with linear predictor on the scale (`weibull_p2`),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine `reltest`.

Model selection among models can be demonstrated using the routines `ms_flat_1tail`, `ms_flat_2tail`, `ms_predictors_1tail`, and `ms_predictors_2tail`,

Examples

```
debug=FALSE
# example 1 can go wrong for small sample sizes, so I've increased to 50
#
# example 1
if(debug)cat("example 1\n")
x=fitdistcp::d102invgauss_example_data_v1
if(debug)cat("x=",x,"\n")
p=c(1:9)/10
q=qinvgauss_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qinvgauss_cp)",
 main="Invgauss: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

invgauss_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
invgauss_f1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

invgauss_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
invgauss_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgauss_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
invgauss_f2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

invgauss_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
invgauss_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

invgauss_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
invgauss_ldd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Square scalar matrix

invgauss_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
invgauss_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
invgauss_lddd(x, v1, fd1, v2, fd2)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Cubic scalar array

invgauss_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
invgauss_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

`invgauss_lmn` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgauss_lmn(x, v1, fd1, v2, fd2, mm, nn)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`mm`	an index for which derivative to calculate
`nn`	an index for which derivative to calculate

Value

Scalar value

`invgauss_lmnp` *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
invgauss_lmnp(x, v1, fd1, v2, fd2, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

invgauss_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
invgauss_logf(params, x, prior)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
prior	logical indicating which prior to use

Value

Scalar value.

invgauss_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

invgauss_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
invgauss_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

invgauss_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
invgauss_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

invgauss_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
invgauss_logscores(logscores, x, prior, fd1 = 0.01, fd2 = 0.01, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
prior	logical indicating which prior to use
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

invgauss_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
invgauss_means(means, ml_params, lddi, lddd, lambdad_cp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_cp	derivative of the log prior
nx	length of training data
dim	number of parameters

Value

Two scalars

invgauss_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
invgauss_mu1f(alpha, v1, fd1, v2, fd2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

`invgauss_mu2f` *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
invgauss_mu2f(alpha, v1, fd1, v2, fd2)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`v1`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`invgauss_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
invgauss_p1f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

Matrix

---

invgauss_p2f *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
invgauss_p2f(y, v1, fd1, v2, fd2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter

Value

3d array

`invgauss_waic` *Waic*

Description

Waic

Usage

```
invgauss_waic(
  waicscores,
  x,
  v1hat,
  fd1,
  v2hat,
  fd2,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`v1hat`	first parameter
`fd1`	the fractional delta used in the numerical derivatives with respect to the parameter
`v2hat`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

jpf2p *Jeffreys' Prior with two parameters*

Description

Jeffreys' Prior with two parameters

Usage

```
jpf2p(ggd, detg, ggi)
```

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 2 values

jpf3p *Jeffreys' Prior with three parameters*

Description

Jeffreys' Prior with three parameters

Usage

```
jpf3p(ggd, detg, ggi)
```

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 3 values

jpf4p *Jeffreys' Prior with four parameters*

Description

Jeffreys' Prior with four parameters

Usage

```
jpf4p(ggd, detg, ggi)
```

Arguments

ggd	gradient of the expected information matrix
detg	determinant of the expected information matrix
ggi	inverse of the expected information matrix

Value

Vector of 4 values

lnorm_cp *Log-normal Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rlnorm_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE)

dlnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

plnorm_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

tlnorm_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.

- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The log normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-(\log(x)-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d035lnorm_example_data_v1
p=c(1:9)/10
q=qlnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qlnorm_cp)",
 main="Log-normal: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`lnorm_dmgs_cp` *Log-normal Distribution Predictions Based on a Calibrating Prior, using DMGS (for testing only)*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_dmgs_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  debug = FALSE
)

rlnorm_dmgs_cp(n, x, mlcp = TRUE, debug = FALSE)

dlnorm_dmgs_cp(x, y = x, debug = FALSE)

plnorm_dmgs_cp(x, y, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.

- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The log normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma x} e^{-(\log(x)-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d035lnorm_example_data_v1
p=c(1:9)/10
q=qlnorm_dmgs_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qlnorm_dmgs_cp)",
 main="Log-normal_DMGS: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

---

lnorm_dmgs_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
lnorm_dmgs_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar value.

lnorm_dmgs_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_dmgs_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

lnorm_dmgs_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
lnorm_dmgs_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

lnorm_dmgs_waic *Waic*

Description

Waic

Usage

```
lnorm_dmgs_waic(waicscores, x, v1hat, v2hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

lnorm_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
lnorm_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

lnorm_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
lnorm_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

lnorm_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lnorm_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lnorm_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

lnorm_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
lnorm_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

lnorm_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

lnorm_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

lnorm_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
lnorm_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

lnorm_mu2fa — *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
lnorm_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_p1_cp — *Log-normal Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
  debug = FALSE
)

rlnorm_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dlnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

plnorm_p1_cp(
```

```
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tlnorm_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.

- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The log normal distribution with a predictor has probability density function

$$f(x; a, b, \sigma) = \frac{1}{\sqrt{2\pi}x\sigma} e^{-(\log(x)-\mu(a,b))^2/(2\sigma^2)}$$

where x is the random variable, $\mu = a + bt$ is the location parameter of the log of the random variable, modelled as a function of parameters a, b and predictor t, and $\sigma > 0$ is the scale parameter of the log of the random variable.

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b, \sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),

- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d061lnorm_p1_example_data_v1_x
tt=fitdistcp::d061lnorm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlnorm_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qlnorm_p1_cp)",
 main="Log-Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

lnorm_p1_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
lnorm_p1_f1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

lnorm_p1_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
lnorm_p1_f1fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

lnorm_p1_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
lnorm_p1_f2fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
lnorm_p1_f2fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`lnorm_p1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`lnorm_p1_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lnorm_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`lnorm_p1_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lnorm_p1_lddda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`lnorm_p1_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
lnorm_p1_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

`lnorm_p1_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`lnorm_p1_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

`lnorm_p1_loglik` *Log-normal-with-p1 observed log-likelihood function*

Description

Log-normal-with-p1 observed log-likelihood function

Usage

```
lnorm_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

`lnorm_p1_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lnorm_p1_logscores(logscores, x, t)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors

Value

Two scalars

lnorm_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
lnorm_p1_mu1fa(alpha, t0, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

lnorm_p1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
lnorm_p1_mu2fa(alpha, t0, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
lnorm_p1_p1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

lnorm_p1_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
lnorm_p1_p2fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lnorm_p1_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

lnorm_p1_pdd — *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_p1_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`lnorm_p1_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
lnorm_p1_predictordata(x, t, t0, params)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`params`	model parameters for calculating logf

Value

Two vectors

`lnorm_p1_waic` *Waic*

Description

Waic

Usage

```
lnorm_p1_waic(waicscores, x, t, v1hat, v2hat, v3hat)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1hat`	first parameter
`v2hat`	second parameter
`v3hat`	third parameter

Value

Two numeric values.

`lnorm_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
lnorm_p1fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

`lnorm_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
lnorm_p2fa(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

lnorm_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

lnorm_pdd — *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lnorm_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

lnorm_waic *Waic for RUST*

Description

Waic for RUST

Usage

```
lnorm_waic(waicscores, x, v1hat, v2hat)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter

Value

Two numeric values.

logis_cp *Logistic Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlogis_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rlogis_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE)

dlogis_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

plogis_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

tlogis_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required

mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The logistic distribution has distribution function

$$f(x;\mu,\sigma) = \frac{1}{1+e^{-(x-\mu)/\sigma}}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),

- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d040logis_example_data_v1
p=c(1:9)/10
q=qlogis_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qlogis_cp)",
 main="Logistic: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

logis_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
logis_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
logis_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
logis_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
logis_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

logis_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
logis_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

logis_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

logis_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

logis_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
logis_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

logis_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
logis_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

logis_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
logis_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

logis_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
logis_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

`logis_p1_cp` *Logistic Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlogis_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
```

```
)

rlogis_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dlogis_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

plogis_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tlogis_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The logistic distribution with a predictor has distribution function

$$f(x; a, b, \sigma) = \frac{1}{1 + e^{-(x-\mu(a,b))/\sigma}}$$

where x is the random variable, $\mu = a+bt$ is the location paramter, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)

- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d062logis_p1_example_data_v1_x
tt=fitdistcp::d062logis_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlogis_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qlogis_p1_cp)",
 main="Logistic w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

logis_p1_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
logis_p1_f1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
logis_p1_f1fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

logis_p1_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
logis_p1_f2fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

logis_p1_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
logis_p1_f2fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

logis_p1_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`logis_p1_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_fdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`logis_p1_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
logis_p1_ldda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`logis_p1_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
logis_p1_lddda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`logis_p1_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
logis_p1_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

logis_p1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

logis_p1_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

---

logis_p1_loglik *Logistic-with-p1 observed log-likelihood function*

Description

Logistic-with-p1 observed log-likelihood function

Usage

```
logis_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

---

logis_p1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
logis_p1_logscores(logscores, x, t)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors

Value

Two scalars

logis_p1_means *Logistic distribution: RHP mean*

Description

Logistic distribution: RHP mean

Usage

```
logis_p1_means(t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

logis_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
logis_p1_mu1fa(alpha, t0, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

---

logis_p1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
logis_p1_mu2fa(alpha, t0, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

logis_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
logis_p1_p1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`logis_p1_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
logis_p1_p2fa(x, t0, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`logis_p1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_pd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`logis_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_p1_pdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`logis_p1_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
logis_p1_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

---

logis_p1_waic *Waic*

Description

Waic

Usage

```
logis_p1_waic(waicscores, x, t, v1hat, v2hat, v3hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

logis_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
logis_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

logis_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
logis_p2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`logis_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_pd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

`logis_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
logis_pdd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Matrix

logis_waic *Waic*

Description

Waic

Usage

```
logis_waic(waicscores, x, v1hat, v2hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

lst_k3_cp *t Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y

- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlst_k3_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kdf = 5,
  d1 = 0.01,
  fd2 = 0.01,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE,
  aderivs = TRUE
)

rlst_k3_cp(
  n,
  x,
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlst_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  nrust = 1000,
```

```
  debug = FALSE,
  aderivs = TRUE
)

plst_k3_cp(
  x,
  y = x,
  d1 = 0.01,
  fd2 = 0.01,
  kdf = 5,
  rust = FALSE,
  nrust = 1000,
  debug = FALSE,
  aderivs = TRUE
)

tlst_k3_cp(n, x, d1 = 0.01, fd2 = 0.01, kdf = 5, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kdf	the known degrees of freedom parameter
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

`q****` returns a list containing at least the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_value:` the value of the log-likelihood at the maximum.
- `standard_errors:` estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles:` quantiles calculated using maximum likelihood.
- `cp_quantiles:` predictive quantiles calculated using a calibrating prior.
- `maic:` the AIC score for the maximum likelihood model, times -1/2.
- `cp_method:` a comment about the method used to generate the `cp` prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter:` the estimated value for parameter, as a function of the predictor.
- `adjustedx:` the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_deviates:` random deviates calculated using maximum likelihood.
- `cp_deviates:` predictive random deviates calculated using a calibrating prior.
- `cp_method:` a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_pdf:` density function from maximum likelihood.
- `cp_pdf:` predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method:` a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_cdf:` distribution function from maximum likelihood.
- `cp_cdf:` predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method:` a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples:` random samples from the parameter posterior.

Details of the Model

The t distribution (also known as the location-scale t distribution, hence the name `lst`), has probability density function

$$f(x;\mu,\sigma) = \frac{\Gamma((\nu+1)/2)}{\sqrt{\pi\nu}\sigma\Gamma(\nu/2)}\left(1+\frac{(x-\mu)^2}{\sigma^2\nu}\right)^{(\nu+1)/2}$$

where x is the random variable, $\mu, \sigma > 0$ are the parameters, and we consider the degrees of freedom ν to be known (hence the `k3` in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d041lst_k3_example_data_v1
p=c(1:9)/10
q=qlst_k3_cp(x,p,kdf=5,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qlst_k3_cp)",
 main="t: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

lst_k3_f1f *DMGS equation 3.3, f1 term*

Description

DMGS equation 3.3, f1 term

Usage

```
lst_k3_f1f(y, v1, d1, v2, fd2, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

---

lst_k3_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
lst_k3_f1fa(x, v1, v2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

Vector

lst_k3_f2f *DMGS equation 3.3, f2 term*

Description

DMGS equation 3.3, f2 term

Usage

```
lst_k3_f2f(y, v1, d1, v2, fd2, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_k3_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
lst_k3_f2fa(x, v1, v2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

Matrix

`lst_k3_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_fd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`lst_k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_fdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lst_k3_ldd *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
lst_k3_ldd(x, v1, d1, v2, fd2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Square scalar matrix

lst_k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lst_k3_ldda(x, v1, v2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_k3_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
lst_k3_lddd(x, v1, d1, v2, fd2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Cubic scalar array

lst_k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lst_k3_lddda(x, v1, v2, kdf)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_k3_lmn *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_k3_lmn(x, v1, d1, v2, fd2, kdf, mm, nn)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

lst_k3_lmnp *One component of the third derivative of the normalized log-likelihood*

Description

One component of the third derivative of the normalized log-likelihood

Usage

```
lst_k3_lmnp(x, v1, d1, v2, fd2, kdf, mm, nn, rr)
```

Arguments

x	a vector of training data values
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

---

lst_k3_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
lst_k3_logf(params, x, kdf)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kdf	the known degrees of freedom parameter

Value

Scalar value.

lst_k3_logfdd — *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_logfdd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

lst_k3_logfddd — *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_k3_logfddd(x, v1, v2, v3)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

lst_k3_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
lst_k3_loglik(vv, x, kdf)
```

Arguments

vv	parameters
x	a vector of training data values
kdf	the known degrees of freedom parameter

Value

Scalar

lst_k3_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lst_k3_logscores(logscores, x, d1 = 0.01, fd2 = 0.01, kdf, aderivs = TRUE)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
d1	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

lst_k3_mu1f *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
lst_k3_mu1f(alpha, v1, d1, v2, fd2, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_k3_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
lst_k3_mu2f(alpha, v1, d1, v2, fd2, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

`lst_k3_p1f` *DMGS equation 3.3, p1 term*

Description

DMGS equation 3.3, p1 term

Usage

```
lst_k3_p1f(y, v1, d1, v2, fd2, kdf)
```

Arguments

`y`	a vector of values at which to calculate the density and distribution functions
`v1`	first parameter
`d1`	the delta used in the numerical derivatives with respect to the parameter
`v2`	second parameter
`fd2`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter

Value

Matrix

`lst_k3_p2f` *DMGS equation 3.3, p2 term*

Description

DMGS equation 3.3, p2 term

Usage

```
lst_k3_p2f(y, v1, d1, v2, fd2, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

---

lst_k3_waic *Waic*

Description

Waic

Usage

```
lst_k3_waic(
  waicscores,
  x,
  v1hat,
  d1,
  v2hat,
  fd2,
  kdf,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter

fd2	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

`lst_p1k3_cp` *t Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qlst_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

rlst_p1k3_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

dlst_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
```

```
  fd3 = 0.01,
  kdf = 10,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

plst_p1k3_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE,
  aderivs = TRUE
)

tlst_p1k3_cp(
  n,
  x,
  t,
  d1 = 0.01,
  d2 = 0.01,
  fd3 = 0.01,
  kdf = 10,
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter

fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
kdf	the known degrees of freedom parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The t distribution with a predictor (also known as the location-scale t distribution with a predictor, hence the name lst), has probability density function

$$f(x; a, b, \sigma) = \frac{\Gamma((\nu+1)/2)}{\sqrt{\pi\nu}\sigma\Gamma(\nu/2)} \left(1 + \frac{(x - \mu(a,b))^2}{\sigma^2\nu}\right)^{(\nu+1)/2}$$

where x is the random variable, $\mu = a + bt$ is the location parameter, and $\sigma > 0$ is the scale parameter. We consider the degrees of freedom ν to be known (hence the k3 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d063lst_p1k3_example_data_v1_x
tt=fitdistcp::d063lst_p1k3_example_data_v1_t
p=c(1:9)/10
n0=10
q=qlst_p1k3_cp(x,tt,n0=n0,p=p,kdf=5,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qlst_p1k3_cp)",
 main="t w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

lst_p1k3_f1f *DMGS equation 2.1, f1 term*

Description

DMGS equation 2.1, f1 term

Usage

```
lst_p1k3_f1f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

`lst_p1k3_f1fa` *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
lst_p1k3_f1fa(x, t0, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Vector

lst_p1k3_f1fw *The first derivative of the densityfor WAIC*

Description

The first derivative of the densityfor WAIC

Usage

```
lst_p1k3_f1fw(x, t, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Vector

lst_p1k3_f2f *DMGS equation 2.1, f2 term*

Description

DMGS equation 2.1, f2 term

Usage

```
lst_p1k3_f2f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter

v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

`lst_p1k3_f2fa` *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
lst_p1k3_f2fa(x, t0, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_p1k3_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
lst_p1k3_f2fw(x, t, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_p1k3_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_fd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

`lst_p1k3_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_fdd(x, t, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`lst_p1k3_ldd` *Second derivative matrix of the normalized log-likelihood*

Description

Second derivative matrix of the normalized log-likelihood

Usage

```
lst_p1k3_ldd(x, t, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Square scalar matrix

---

lst_p1k3_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_ldda(x, t, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

Matrix

lst_p1k3_lddd *Third derivative tensor of the normalized log-likelihood*

Description

Third derivative tensor of the normalized log-likelihood

Usage

```
lst_p1k3_lddd(x, t, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Cubic scalar array

lst_p1k3_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lddda(x, t, v1, v2, v3, kdf)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
kdf	the known degrees of freedom parameter

Value

3d array

---

lst_p1k3_lmn — *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lmn(x, t, v1, d1, v2, d2, v3, fd3, kdf, mm, nn)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate

Value

Scalar value

lst_p1k3_lmnp *One component of the second derivative of the normalized log-likelihood*

Description

One component of the second derivative of the normalized log-likelihood

Usage

```
lst_p1k3_lmnp(x, t, v1, d1, v2, d2, v3, fd3, kdf, mm, nn, rr)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate

Value

Scalar value

lst_p1k3_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
lst_p1k3_logf(params, x, t, kdf)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors
kdf	the known degrees of freedom parameter

Value

Scalar value.

---

lst_p1k3_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_logfdd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

lst_p1k3_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
lst_p1k3_logfddd(x, t, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

lst_p1k3_loglik *LST-with-p1 observed log-likelihood function*

Description

LST-with-p1 observed log-likelihood function

Usage

```
lst_p1k3_loglik(vv, x, t, kdf)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors
kdf	the known degrees of freedom parameter

Value

Scalar

`lst_p1k3_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
lst_p1k3_logscores(logscores, x, t, d1, d2, fd3, kdf, aderivs)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`d1`	the delta used in the numerical derivatives with respect to the parameter
`d2`	the delta used in the numerical derivatives with respect to the parameter
`fd3`	the fractional delta used in the numerical derivatives with respect to the parameter
`kdf`	the known degrees of freedom parameter
`aderivs`	logical for whether to use analytic derivatives (instead of numerical)

Value

Two scalars

`lst_p1k3_mu1f` *DMGS equation 3.3, mu1 term*

Description

DMGS equation 3.3, mu1 term

Usage

```
lst_p1k3_mu1f(alpha, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

Matrix

---

lst_p1k3_mu2f *DMGS equation 3.3, mu2 term*

Description

DMGS equation 3.3, mu2 term

Usage

```
lst_p1k3_mu2f(alpha, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

lst_p1k3_p1f *DMGS equation 2.1, p1 term*

Description

DMGS equation 2.1, p1 term

Usage

```
lst_p1k3_p1f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

y	value of random variable
t0	value of predictor
v1	first parameter
d1	delta for numerical derivative
v2	second parameter
d2	delta for numerical derivative
v3	third parameter
fd3	fractional delta for numerical derivative
kdf	the known number of degrees of freedom

Value

Matrix

lst_p1k3_p2f *DMGS equation 2.1, p2 term*

Description

DMGS equation 2.1, p2 term

Usage

```
lst_p1k3_p2f(y, t0, v1, d1, v2, d2, v3, fd3, kdf)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter

Value

3d array

---

lst_p1k3_predictordata

Predicted Parameter and Generalized Residuals

Description

Predicted Parameter and Generalized Residuals

Usage

```
lst_p1k3_predictordata(predictordata, x, t, t0, params, kdf)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kdf	the known degrees of freedom parameter

Value

Two vectors

lst_p1k3_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
lst_p1k3_setics(x, t, ics)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
ics	initial conditions for the maximum likelihood search

Value

Vector

lst_p1k3_waic *Waic*

Description

Waic

Usage

```
lst_p1k3_waic(
  waicscores,
  x,
  t,
  v1hat,
  d1,
  v2hat,
  d2,
  v3hat,
  fd3,
  kdf,
  lddi,
  lddd,
  lambdad,
  aderivs
)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
v2hat	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
v3hat	third parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
kdf	the known degrees of freedom parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

Two numeric values.

---

make_cwaic *Make WAIC*

Description

Make WAIC

Usage

```
make_cwaic(x, fhatx, lddi, lddd, f1f, lambdad, f2f, dim)
```

Arguments

x	the training data
fhatx	density of x at the maximum likelihood parameters
lddi	inverse of the second derivative log-likelihood matrix
lddd	the third derivative log-likelihood tensor
f1f	the f1 term from DMGS equation 2.1
lambdad	the slope of the log prior
f2f	the f2 term from DMGS equation 2.1
dim	number of free parameters

Value

Two scalars

`make_maic` *Calculate MAIC*

Description

Calculate MAIC

Usage

```
make_maic(ml_value, nparams)
```

Arguments

`ml_value`	maximum of the likelihood
`nparams`	number of parameters

Value

Vector of 3 values Returns the two compoments of MAIC, and their sum

---

`make_se` *Make Standard Errors from lddi*

Description

Make Standard Errors from lddi

Usage

```
make_se(nx, lddi)
```

Arguments

`nx`	length of training data
`lddi`	the inverse log-likelihood matrix

Value

Vector

make_waic *Make WAIC*

Description

Make WAIC

Usage

```
make_waic(x, fhatx, lddi, lddd, f1f, lambdad, f2f, dim)
```

Arguments

x	the training data
fhatx	density of x at the maximum likelihood parameters
lddi	inverse of the second derivative log-likelihood matrix
lddd	the third derivative log-likelihood tensor
f1f	the f1 term from DMGS equation 2.1
lambdad	the slope of the log prior
f2f	the f2 term from DMGS equation 2.1
dim	number of free parameters

Value

Two scalars

makebetat0 *Calculate the location parameter when there are predictors (single time point)*

Description

Calculate the location parameter when there are predictors (single time point)

Usage

```
makebetat0(nt, params, t0)
```

Arguments

nt	the number of columns in t
params	model parameters for calculating logf
t0	a single value of the predictor (specify either t0 or n0 but not both)

Value

Vector

`makebetatm` *Calculate the location parameter when there are predictors (multiple time points)*

Description

Calculate the location parameter when there are predictors (multiple time points)

Usage

```
makebetatm(nt, params, t)
```

Arguments

`nt`	the number of columns in `t`
`params`	model parameters for calculating logf
`t`	a vector or matrix of predictors

Value

Vector

`makemuhat0` *Make muhat0*

Description

Make muhat0

Usage

```
makemuhat0(t0, n0, t, mle_params)
```

Arguments

`t0`	the value of the predictor vector at which to make the prediction (if n0 not specified)
`n0`	the position in the predictor vector at which to make the prediction (positive integer less than or equal to the length of x) (if t0 not specified)
`t`	predictor
`mle_params`	MLE params

Value

Scalar

makeq *Calculates quantiles from simulations by inverting the Hazen CDF*

Description

Calculates quantiles from simulations by inverting the Hazen CDF

Usage

```
makeq(yy, pp)
```

Arguments

yy	vector of samples
pp	vector of probabilities

Value

Vector

maket0 *Determine t0*

Description

Determine t0

Usage

```
maket0(t0, n0, t)
```

Arguments

t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
t	a vector or matrix of predictors

Value

Scalar

maketresid0 *Make ta0*

Description

Make ta0

Usage

```
maketresid0(t0, n0, t)
```

Arguments

t0	the value of the predictor vector at which to make the prediction (if n0 not specified)
n0	the position in the predictor vector at which to make the prediction (positive integer less than or equal to the length of x) (if t0 not specified)
t	predictor

Value

Scalar

man *A blank function I use for setting up the man page information*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
man(
  x,
  t,
  t1,
  t2,
  t3,
  t0,
  t01,
  t02,
  t03,
  t10,
  t20,
  n0,
  n01,
  n02,
  n03,
  n10,
  n20,
  p,
  n,
  y,
  ics,
  kloc,
  kscale,
  kshape,
  kdf,
  kbeta,
  d1,
  fd1,
  d2,
  fd2,
  d3,
  fd3,
  d4,
  fd4,
  d5,
  fd5,
  d6,
  fd6,
```

```
  fdalpha,
  minxi,
  maxxi,
  dlogpi,
  means,
  waicscores,
  logscores,
  extramodels,
  pdf,
  customprior,
  dmgs,
  mlcp,
  predictordata,
  centering,
  method,
  nonnegslopesonly,
  rnonnegslopesonly,
  prior,
  debug,
  rust,
  nrust,
  boot,
  nboot,
  pwm,
  unbiasedv,
  aderivs
)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t3	a vector of predictors for the shape, such that length(t3)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
t10	a single value of the predictor for the mean (specify either t10 or n10 but not both)
t20	a single value of the predictor for the sd (specify either t20 or n20 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)

n03	an index for the predictor (specify either t03 or n03 but not both)
n10	an index for the predictor for the mean (specify either t10 or n10 but not both)
n20	an index for the predictor for the sd (specify either t20 or n20 but not both)
p	a vector of probabilities at which to generate predictive quantiles
n	the number of random samples required
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
kloc	the known location parameter
kscale	the known scale parameter
kshape	the known shape parameter
kdf	the known degrees of freedom parameter
kbeta	the known beta parameter
d1	if aderivs=FALSE, the delta used for numerical derivatives with respect to the first parameter
fd1	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the first parameter
d2	if aderivs=FALSE, the delta used for numerical derivatives with respect to the second parameter
fd2	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the second parameter
d3	if aderivs=FALSE, the delta used for numerical derivatives with respect to the third parameter
fd3	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the third parameter
d4	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fourth parameter
fd4	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
d5	if aderivs=FALSE, the delta used for numerical derivatives with respect to the fifth parameter
fd5	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
d6	if aderivs=FALSE, the delta used for numerical derivatives with respect to the sixth parameter
fd6	if aderivs=FALSE, the fractional delta used for numerical derivatives with respect to the fourth parameter
fdalpha	if pdf=TRUE, the fractional delta used for numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
minxi	the minimum allowed value of the shape parameter (decrease with caution)
maxxi	the maximum allowed value of the shape parameter (increase with caution)

dlogpi	gradient of the log prior
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
pdf	logical that indicates whether to run additional calculations and return density functions evaluated at quantiles specified by the input probabilities (longer runtime)
customprior	a custom value for the slope of the log prior at the maxlik estimate
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
method	character string that indicates whether to use rust `method=rust` or bootstrap `method=boot`
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes
prior	logical indicating which prior to use
debug	logical for turning on debug messages
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
boot	logical that indicates whether bootstrap-based posterior sampling calculations should be run or not (longer run time)
nboot	the number of posterior samples used in the bootstrap calculations
pwm	logical for whether to include PWM results (longer runtime)
unbiasedv	logical for whether to include unbiased variance results in norm
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Optional Return Values (EVT models only)

q**** optionally returns the following, for EVT models only:

- cp_pdf: the density function at quantiles corresponding to input probabilities p. We provide this for EVD models, because direct estimation of the density function using the DMGS density equation is not possible.

Optional Return Values (some EVT models only)

q**** optionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_quantiles: predictive quantiles calculated from Bayesian integration with a flat prior.
- rh_ml_quantiles: predictive quantiles calculated from Bayesian integration with the calibrating prior, and the maximmum likelihood estimate for the shape parameter.
- jp_quantiles: predictive quantiles calculated from Bayesian integration with Jeffreys' prior.

r**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_deviates: predictive random deviates calculated using a Bayesian analysis with a flat prior.
- rh_ml_deviates: predictive random deviates calculated using a Bayesian analysis with the RHP-MLE prior.
- jp_deviates: predictive random deviates calculated using a Bayesian analysis with the JP.

d**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_pdf: predictive density function from a Bayesian analysis with the flat prior.
- rh_ml_pdf: predictive density function from a Bayesian analysis with the RHP-MLE prior.
- jp_pdf: predictive density function from a Bayesian analysis with the JP.

p**** additionally returns the following, for some EVT models only:

If extramodels=TRUE:

- flat_cdf: predictive distribution function from a Bayesian analysis with the flat prior.
- rh_ml_cdf: predictive distribution function from a Bayesian analysis with the RHP-MLE prior.
- jp_cdf: predictive distribution function from a Bayesian analysis with the JP.

These additional predictive distributions are included for comparison with the calibrating prior model. They generally give less good reliability than the calibrating prior.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (non-homogeneous models)

This model is not homogeneous, i.e. it does not have a transitive transformation group, and so there is no right Haar prior and no method for generating exactly reliable predictions. The cp outputs are generated using a prior that has been shown in tests to give reasonable reliability. See Jewson et al. (2025a) for discussion of the prior and test results. For non-homogeneous models, reliability is generally poor for small sample sizes (<20), but is still much better than maximum likelihood. For small sample sizes, it is advisable to check the level of reliability using the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

---

man1f *Return message for f1f, p1f, mu1f*

Description

Return message for f1f, p1f, mu1f

Usage

```
man1f()
```

Value

Matrix

man2f *Return message for f2f, p2f, mu2f*

Description

Return message for f2f, p2f, mu2f

Usage

```
man2f()
```

Value

3d array

manboot *Return message for boot*

Description

Return message for boot

Usage

```
manboot()
```

Value

A list containing a matrix of simulated parameter values

mancheckmle *Return message for checkmle*

Description

Return message for checkmle

Usage

```
mancheckmle()
```

Value

No return value (just a message to the screen).

mandsub *Return message for dsub*

Description

Return message for dsub

Usage

```
mandsub()
```

Value

A vector of parameter estimates, two pdf vectors, two cdf vectors

manf *Blank function I use for setting up the man page information for the functions*

Description

Blank function I use for setting up the man page information for the functions

Usage

```
manf(
  dim,
  vv,
  ml_params,
  nx,
  nxx,
  x,
  xx,
  t,
  nt,
  ta,
  tb,
  tc,
  t1,
  t2,
  t3,
  tt,
  tt1,
  tt2,
  tt3,
```

```
      tt2d,
      tt3d,
      t0,
      t0a,
      t0b,
      t0c,
      t01,
      t02,
      t03,
      t10,
      t20,
      t30,
      n0,
      n10,
      n20,
      p,
      n,
      y,
      ics,
      tresid,
      tresid0,
      muhat0,
      vhat,
      v1,
      v1hat,
      v1h,
      d1,
      fd1,
      v2,
      v2hat,
      v2h,
      d2,
      fd2,
      v3,
      v3hat,
      v3h,
      d3,
      fd3,
      v4,
      v4hat,
      v4h,
      d4,
      fd4,
      v5,
      v5hat,
      v5h,
      d5,
      v6,
```

```
v6hat,
v6h,
d6,
minxi,
maxxi,
ximin,
ximax,
fdalpha,
kscale,
kloc,
kshape,
kdf,
kbeta,
alpha,
ymn,
slope,
mu,
sigma,
sigma1,
sigma2,
scale,
shape,
xi,
xi1,
xi2,
lambda,
log,
mm,
nn,
rr,
lddi,
lddi_k2,
lddi_k3,
lddi_k4,
lddd,
lddd_k2,
lddd_k3,
lddd_k4,
lambdad,
lambdad_cp,
lambdad_rhp,
lambdad_flat,
lambdad_rh_mle,
lambdad_rh_flat,
lambdad_jp,
lambdad_custom,
means,
waicscores,
```

```
  logscores,
  extramodels,
  pdf,
  predictordata,
  nonnegslopesonly,
  rnonnegslopesonly,
  customprior,
  prior,
  params,
  yy,
  pp,
  dlogpi,
  debug,
  centering,
  aderivs
)
```

Arguments

dim	number of parameters
vv	parameters
ml_params	parameters
nx	length of training data
nxx	length of training data
x	a vector of training data values
xx	a vector of training data values
t	a vector or matrix of predictors
nt	the number of columns in t
ta	a vector of predictors for the mean (first column)
tb	a vector of predictors for the mean (second column)
tc	a vector of predictors for the mean (third column)
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
tt	a vector of predictors
tt1	a vector of predictors for the mean
tt2	a vector of predictors for the sd
tt3	a vector of predictors for the shape
tt2d	a matrix of predictors (nx by 2)
tt3d	a matrix of predictors (nx by 3)
t0	a single value of the predictor (specify either t0 or n0 but not both)

t0a	a single value of the predictor, for the first column of the predictor (specify either t0a or n0a but not both)
t0b	a single value of the predictor, for the second column of the predictor (specify either t0b or n0b but not both)
t0c	a single value of the predictor, for the third column of the predictor (specify either t0c or n0c but not both)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
t03	a single value of the predictor (specify either t03 or n03 but not both)
t10	a single value of the predictor for the mean (specify either t10 or n10 but not both)
t20	a single value of the predictor for the sd (specify either t20 or n20 but not both)
t30	a single value of the predictor for the shape (specify either t30 or n30 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
n10	an index for the predictor for the mean (specify either t10 or n10 but not both)
n20	an index for the predictor for the sd (specify either t10 or n10 but not both)
p	a vector of probabilities at which to generate predictive quantiles
n	number of random samples required
y	a vector of values at which to calculate the density and distribution functions
ics	initial conditions for the maximum likelihood search
tresid	predictor residuals
tresid0	predictor residual at the point being predicted
muhat0	muhat at the point being predicted
vhat	vector of all parameters
v1	first parameter
v1hat	first parameter
v1h	first parameter
d1	the delta used in the numerical derivatives with respect to the parameter
fd1	the fractional delta used in the numerical derivatives with respect to the parameter
v2	second parameter
v2hat	second parameter
v2h	second parameter
d2	the delta used in the numerical derivatives with respect to the parameter
fd2	the fractional delta used in the numerical derivatives with respect to the parameter
v3	third parameter
v3hat	third parameter

v3h	third parameter
d3	the delta used in the numerical derivatives with respect to the parameter
fd3	the fractional delta used in the numerical derivatives with respect to the parameter
v4	fourth parameter
v4hat	fourth parameter
v4h	fourth parameter
d4	the delta used in the numerical derivatives with respect to the parameter
fd4	the fractional delta used in the numerical derivatives with respect to the parameter
v5	fifth parameter
v5hat	fifth parameter
v5h	fifth parameter
d5	the delta used in the numerical derivatives with respect to the parameter
v6	sixth parameter
v6hat	sixth parameter
v6h	sixth parameter
d6	the delta used in the numerical derivatives with respect to the parameter
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
ximin	minimum value of shape parameter xi
ximax	maximum value of shape parameter xi
fdalpha	the fractional delta used in the numerical derivatives with respect to probability, for calculating the pdf as a function of quantiles
kscale	the known scale parameter
kloc	the known location parameter
kshape	the known shape parameter
kdf	the known degrees of freedom parameter
kbeta	the known beta parameter
alpha	a vector of values of alpha (one minus probability)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
scale	the scale parameter of the distribution
shape	the shape parameter of the distribution

xi	the shape parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution
lambda	the lambda parameter of the distribution
log	logical for the density evaluation
mm	an index for which derivative to calculate
nn	an index for which derivative to calculate
rr	an index for which derivative to calculate
lddi	inverse observed information matrix
lddi_k2	inverse observed information matrix, fixed shape parameter
lddi_k3	inverse observed information matrix, fixed shape parameter
lddi_k4	inverse observed information matrix, fixed shape parameter
lddd	third derivative of log-likelihood
lddd_k2	third derivative of log-likelihood, fixed shape parameter
lddd_k3	third derivative of log-likelihood, fixed shape parameter
lddd_k4	third derivative of log-likelihood, fixed shape parameter
lambdad	derivative of the log prior
lambdad_cp	derivative of the log prior
lambdad_rhp	derivative of the log RHP prior
lambdad_flat	derivative of the log flat prior
lambdad_rh_mle	derivative of the log CRHP-MLE prior
lambdad_rh_flat	derivative of the log CRHP-FLAT prior
lambdad_jp	derivative of the log JP prior
lambdad_custom	custom value of the derivative of the log prior
means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
extramodels	logical that indicates whether to add three additional prediction models
pdf	logical that indicates whether to return density functions evaluated at quantiles specified by input probabilities
predictordata	logical that indicates whether to calculate and return predictordata
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes

customprior	a custom value for the slope of the log prior at the maxlik estimate
prior	logical indicating which prior to use
params	model parameters for calculating logf
yy	vector of samples
pp	vector of probabilities
dlogpi	gradient of the log prior
debug	debug flag
centering	indicates whether the routine should center the data or not
aderivs	logical for whether to use analytic derivatives (instead of numerical)

Value

No return value

`manldd` *Return message for ldd*

Description

Return message for ldd

Usage

```
manldd()
```

Value

Square scalar matrix

`manlddd` *Return message for lddd*

Description

Return message for lddd

Usage

```
manlddd()
```

Value

Cubic scalar array

manlnn *Return message for lnn*

Description

Return message for lnn

Usage

```
manlnn()
```

Value

Scalar value

manlnnn *Return message for lnnn*

Description

Return message for lnnn

Usage

```
manlnnn()
```

Value

Scalar value

manlogf *Return message for Logf*

Description

Return message for Logf

Usage

```
manlogf()
```

Value

Scalar value.

manloglik *Return message for loglik*

Description

Return message for loglik

Usage

```
manloglik()
```

Value

Scalar

manlogscores *Return message for logscores*

Description

Return message for logscores

Usage

```
manlogscores()
```

Value

Two scalars

manmeans *Return message for means*

Description

Return message for means

Usage

```
manmeans()
```

Value

Two scalars

manpredictor *Return message for predictor.*

Description

Return message for predictor.

Usage

```
manpredictor()
```

Value

Two vectors

manvector *Return message for vector*

Description

Return message for vector

Usage

```
manvector()
```

Value

Vector

manwaic *Return message for WAIC*

Description

Return message for WAIC

Usage

```
manwaic()
```

Value

Two numeric values.

movexiawayfromzero *Move xi away from zero a bit*

Description

Move xi away from zero a bit

Usage

```
movexiawayfromzero(xi)
```

Arguments

xi	xi

Value

Scalar

ms_flat_1tail *Illustration of Model Selection Among 10 One Tail Distributions from the* fitdistcp *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, for 10 one tailed models in the fitdistcp package (although for the GPD, the logscore is NA for mathematical reasons).

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the fitdistcp routines.

The input data may be automatically shifted so that the minimum value is positive.

For the Pareto, the data may be further shifted so that the minimum value is slightly greater than 1.

Usage

```
ms_flat_1tail(
  x,
  index = 1,
  nyears = 10,
  plottype = "empirical",
  plottingposition = "Weibull",
  quiet = FALSE
)
```

Arguments

x	data vector
index	which data point to use for plotting positions
nyears	number of years for frequency calculations
plottype	What to plot? Possible values are 'both', 'empirical', 'cp'
plottingposition	Weibull or Hazen
quiet	logical for whether to print screen messages

Details

The 10 models are: exp, pareto_k2, halfnorm, lnorm, frechet_k1, weibull, gamma, invgamma, invgauss and gpd_k1.

Value

Plots QQ plots to the screen, for each of the models, and returns a data frame containing

- MLE parameter values
- AIC scores (times -0.5), AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores, logscore weights
- maximum likelihood and calibrating prior means
- maximum likelihood and calibrating prior standard deviations

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(1)
nx=50
x=rlnorm(nx)
print(ms_flat_1tail(x))
```

ms_flat_2tail *Illustration of Model Selection Among 18 Distributions from the* fitdistcp *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, for 7 two tailed models in the fitdistcp packages

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the fitdistcp routines.

Usage

```
ms_flat_2tail(x)
```

Arguments

x	data vector

Details

The 7 models are: norm, gnorm_k3, gumbel, logis, lst_k3, cauchy, gev

Value

Plots QQ plots to the screen, for each of the models, and returns a data frame containing

- AIC scores (times -0.5), AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores, logscore weights
- maximum likelihood and calibrating prior means
- maximum likelihood and calibrating prior standard deviations

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(1)
nx=50
x=rnorm(nx)
print(ms_flat_2tail(x))
```

`ms_predictors_1tail` *Model Selection Among 5 Distributions with predictors from the* `fitdistcp` *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, t, for 5 one tailed models with predictors in the `fitdistcp` package.

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the `fitdistcp` routines.

The input data may be automatically shifted so that the minimum value is positive.

For the Pareto, the data is so that the minimum value is slightly greater than 1.

Usage

```
ms_predictors_1tail(x, t)
```

Arguments

`x`	data vector
`t`	predictor vector

Details

The 5 models are: `exp_p1`, `pareto_p1k2`, `lnorm_p1`, `frechet_p2k1`, `weibull_p2`.

Value

Plots QQ plots to the screen, for each of the 5 models, and returns a data frame containing

- AIC scores, AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores and logscore weights

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(3)
nx=100
predictor=c(1:nx)/nx
x=rlnorm(nx,meanlog=predictor,sdlog=0.1)
print(ms_predictors_1tail(x,predictor))
```

---

ms_predictors_2tail *Model Selection Among 6 Distributions with predictors from the* fitdistcp *Package*

Description

Applies model selection using AIC, WAIC1, WAIC2 and leave-one-out logscore to the input data x, t, for 6 two tail models with predictors in the `fitdistcp` packages (although for the GEV, the logscore is NA for mathematical reasons).

The code is straightforward, and the point is to illustrate what is possible using the model selection outputs from the `fitdistcp` routines.

GEVD is temperamental in that it doesn't work if the shape parameter is extreme.

Usage

```
ms_predictors_2tail(x, t)
```

Arguments

`x`	data vector
`t`	predictor vector

Details

The 11 models are: `norm_p1`, `gumbel_p1`, `logis_p1`, `lst_k3_p1`, `cauchy_p1` and `gev_p1`.

Value

Plots QQ plots to the screen, for each of the 6 models, and returns a data frame containing

- AIC scores, AIC weights
- WAIC1 scores, WAIC1 weights
- WAIC2 scores, WAIC2 weights
- logscores and logscore weights

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

Examples

```
 # because it's too slow for CRAN
set.seed(2)
nx=100
predictor=c(1:nx)/nx
x=rnorm(nx,mean=predictor,sd=1)
print(ms_predictors_2tail(x,predictor))
```

---

nopdfcdfmsg *Message to explain why GEV and GPD* d*** *and* p*** *routines don't return DMGS pdfs and cdfs*

Description

Message to explain why GEV and GPD d*** and p*** routines don't return DMGS pdfs and cdfs

Usage

```
nopdfcdfmsg(yy, pp)
```

Arguments

yy	vector of samples
pp	vector of probabilities

Value

String

norm_boot *Bootstrap*

Description

Bootstrap

Usage

```
norm_boot(x, n)
```

Arguments

x	a vector of training data values
n	number of random samples required

Value

A list containing a matrix of simulated parameter values

norm_cp *Normal Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  unbiasedv = FALSE,
  debug = FALSE
)

rnorm_cp(n, x, method = "rust", rust = FALSE, mlcp = TRUE, debug = FALSE)

dnorm_cp(
  x,
  y = x,
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 1000,
  debug = FALSE
)

pnorm_cp(
  x,
  y = x,
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 1000,
  debug = FALSE
)

tnorm_cp(method, n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)

rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
unbiasedv	logical for whether to include unbiased variance results in norm
debug	logical for turning on debug messages
n	the number of random samples required
method	character string that indicates whether to use rust `method=rust` or bootstrap `method=boot`
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions
boot	logical that indicates whether bootstrap-based posterior sampling calculations should be run or not (longer run time)
nboot	the number of posterior samples used in the bootstrap calculations

Value

`q****` returns a list containing at least the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_value:` the value of the log-likelihood at the maximum.
- `standard_errors:` estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles:` quantiles calculated using maximum likelihood.
- `cp_quantiles:` predictive quantiles calculated using a calibrating prior.
- `maic:` the AIC score for the maximum likelihood model, times -1/2.
- `cp_method:` a comment about the method used to generate the `cp` prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter:` the estimated value for parameter, as a function of the predictor.
- `adjustedx:` the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_deviates:` random deviates calculated using maximum likelihood.
- `cp_deviates:` predictive random deviates calculated using a calibrating prior.
- `cp_method:` a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_pdf:` density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),

- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d030norm_example_data_v1
p=c(1:9)/10
q=qnorm_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qnorm_cp)",
 main="Normal: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

norm_dmgs_cp *Normal Distribution Predictions Based on a Calibrating Prior, using DMGS (for testing only)*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_dmgs_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  debug = FALSE
)

rnorm_dmgs_cp(n, x, mlcp = TRUE, debug = FALSE)

dnorm_dmgs_cp(x, y = x, debug = FALSE)

pnorm_dmgs_cp(x, y = x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The normal distribution has probability density function

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)}$$

where x is the random variable and $\mu, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)

- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with 1-3 linear predictors on the location (gev_p1n),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d030norm_example_data_v1
p=c(1:9)/10
q=qnorm_dmgs_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qnorm_dmgs_cp)",
 main="Normal_DMGS: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

---

norm_dmgs_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
norm_dmgs_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

---

norm_dmgs_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_dmgs_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

---

norm_dmgs_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
norm_dmgs_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

norm_dmgs_waic *Waic*

Description

Waic

Usage

```
norm_dmgs_waic(waicscores, x, v1hat, v2hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

norm_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
norm_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_f2fa — *The second derivative of the density*

Description

The second derivative of the density

Usage

```
norm_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_fd — *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_fdd — *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_ldda — *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
norm_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
norm_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

norm_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
norm_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

`norm_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`norm_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

norm_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

norm_ml_params *Maximum likelihood estimator*

Description

Maximum likelihood estimator

Usage

```
norm_ml_params(x)
```

Arguments

x	a vector of training data values

Value

Scalar

norm_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
norm_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

norm_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
norm_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

`norm_p1_cp` *Normal Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  centering = TRUE,
  debug = FALSE
)
```

```
rnorm_p1_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pnorm_p1_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tnorm_p1_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)

logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The normal distribution with a predictor has probability density function

$$f(x;\alpha,\beta,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu(\alpha,\beta))^2/(2\sigma^2)}$$

where x is the random variable, $\mu = \alpha + \beta t$ is the location parameter, modelled as a function of parameters α, β and predictor t, and $\sigma > 0$ is the scale parameter.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha,\beta,\sigma) \propto \frac{1}{\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible

- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),

- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d060norm_p1_example_data_v1_x
tt=fitdistcp::d060norm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qnorm_p1_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qnorm_p1_cp)",
 main="Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

norm_p1_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

The first derivative of the density

Usage

```
norm_p1_f1fa(x, t, v1, v2, v3)

norm_p1_f1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
norm_p1_f1fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`norm_p1_f2fa` *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

The second derivative of the density

Usage

```
norm_p1_f2fa(x, t, v1, v2, v3)
```

```
norm_p1_f2fa(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`norm_p1_f2fw` *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
norm_p1_f2fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`norm_p1_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_fd(x, t, v1, v2, v3)

norm_p1_fd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_fdd(x, t, v1, v2, v3)

norm_p1_fdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

The second derivative of the normalized log-likelihood

Usage

```
norm_p1_ldda(x, t, v1, v2, v3)

norm_p1_ldda(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

norm_p1_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

The third derivative of the normalized log-likelihood

Usage

```
norm_p1_lddda(x, t, v1, v2, v3)
```


```
norm_p1_lddda(x, t, v1, v2, v3)
```


Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

3d array

norm_p1_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
norm_p1_logf(params, x, t)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar value.

norm_p1_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_logfdd(x, t, v1, v2, v3)

norm_p1_logfdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`norm_p1_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_logfddd(x, t, v1, v2, v3)

norm_p1_logfddd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`norm_p1_loglik` *Normal-with-p1 observed log-likelihood function*

Description

Normal-with-p1 observed log-likelihood function

Usage

```
norm_p1_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

---

norm_p1_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
norm_p1_logscores(logscores, x, t)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors

Value

Two scalars

---

norm_p1_mlparams *Maximum likelihood estimator*

Description

Maximum likelihood estimator

Usage

```
norm_p1_mlparams(x, t)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors

Value

Vector

---

norm_p1_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Minus the first derivative of the cdf, at alpha

Usage

```
norm_p1_mu1fa(alpha, t, v1, v2, v3)
```

```
norm_p1_mu1fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

norm_p1_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Minus the second derivative of the cdf, at alpha

Usage

```
norm_p1_mu2fa(alpha, t, v1, v2, v3)
```

```
norm_p1_mu2fa(alpha, t, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

norm_p1_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

The first derivative of the cdf

Usage

```
norm_p1_p1fa(x, t, v1, v2, v3)
```

```
norm_p1_p1fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`norm_p1_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

The second derivative of the cdf

Usage

```
norm_p1_p2fa(x, t, v1, v2, v3)
```

```
norm_p1_p2fa(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`norm_p1_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_pd(x, t, v1, v2, v3)

norm_p1_pd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`norm_p1_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p1_pdd(x, t, v1, v2, v3)

norm_p1_pdd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

---

norm_p1_predictordata *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
norm_p1_predictordata(x, t, t0, params)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

norm_p1_waic *Waic*

Description

Waic

Usage

```
norm_p1_waic(waicscores, x, t, v1hat, v2hat, v3hat)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter

Value

Two numeric values.

norm_p12_boot *Bootstrap*

Description

Bootstrap

Usage

```
norm_p12_boot(x, t1, t2, n)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
n	number of random samples required

Value

A list containing a matrix of simulated parameter values

norm_p12_checkmle *Check MLE*

Description

Check MLE

Usage

```
norm_p12_checkmle(ml_params)
```

Arguments

ml_params	parameters

Value

No return value (just a message to the screen).

norm_p12_cp *Normal Distribution with Predictors on Mean and SD, with Parameter Uncertainty*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  extramodels = FALSE,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rnorm_p12_cp(
  n,
  x,
  t1,
  t2,
  n01 = NA,
  n02 = NA,
  t01 = NA,
  t02 = NA,
  ics = c(0, 0, 0, 0),
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dnorm_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
```

```
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 10,
  centering = TRUE,
  rnonnegslopesonly = FALSE,
  debug = FALSE
)

pnorm_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 10,
  centering = TRUE,
  rnonnegslopesonly = FALSE,
  debug = FALSE
)

tnorm_p12_cp(
  method,
  n,
  x,
  t1,
  t2,
  nonnegslopesonly = FALSE,
  ics = c(0, 0, 0, 0),
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)

n02	an index for the predictor (specify either t02 or n02 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions
boot	logical that indicates whether bootstrap-based posterior sampling calculations should be run or not (longer run time)
nboot	the number of posterior samples used in the bootstrap calculations
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes
method	character string that indicates whether to use rust method=rust or bootstrap method=boot
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.

- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),

- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d060norm_p1_example_data_v1_x
tt=fitdistcp::d060norm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qnorm_p12_cp(x,t1=tt,t2=tt,n01=n0,n02=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qnorm_p12_cp)",
 main="Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

norm_p12_cp *Normal Distribution with Predictors on both Mean and Standard Deviation, with Parameter Uncertainty*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qnorm_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  p = seq(0.1, 0.9, 0.1),
  ics = c(0, 0, 0, 0),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
```

```
  nrust = 1e+05,
  extramodels = FALSE,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rnorm_p12_cp(
  n,
  x,
  t1,
  t2,
  n01 = NA,
  n02 = NA,
  t01 = NA,
  t02 = NA,
  ics = c(0, 0, 0, 0),
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dnorm_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 10,
  centering = TRUE,
  rnonnegslopesonly = FALSE,
  debug = FALSE
)

pnorm_p12_cp(
  x,
  t1,
  t2,
  t01 = NA,
  t02 = NA,
  n01 = NA,
```

```
  n02 = NA,
  y = x,
  ics = c(0, 0, 0, 0),
  rust = FALSE,
  nrust = 1000,
  boot = FALSE,
  nboot = 10,
  centering = TRUE,
  rnonnegslopesonly = FALSE,
  debug = FALSE
)

tnorm_p12_cp(
  method,
  n,
  x,
  t1,
  t2,
  nonnegslopesonly = FALSE,
  ics = c(0, 0, 0, 0),
  debug = FALSE
)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean, such that length(t1)=length(x)
t2	a vector of predictors for the sd, such that length(t2)=length(x)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
n01	an index for the predictor (specify either t01 or n01 but not both)
n02	an index for the predictor (specify either t02 or n02 but not both)
p	a vector of probabilities at which to generate predictive quantiles
ics	initial conditions for the maximum likelihood search
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations

extramodels	logical that indicates whether to run additional calculations and add three additional prediction models (longer runtime)
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions
boot	logical that indicates whether bootstrap-based posterior sampling calculations should be run or not (longer run time)
nboot	the number of posterior samples used in the bootstrap calculations
rnonnegslopesonly	logical that indicates whether to disallow non-negative slopes
method	character string that indicates whether to use rust `method=rust` or bootstrap `method=boot`
nonnegslopesonly	logical that indicates whether to disallow non-negative slopes

Value

`q****` returns a list containing at least the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_value:` the value of the log-likelihood at the maximum.
- `standard_errors:` estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles:` quantiles calculated using maximum likelihood.
- `cp_quantiles:` predictive quantiles calculated using a calibrating prior.
- `maic:` the AIC score for the maximum likelihood model, times -1/2.
- `cp_method:` a comment about the method used to generate the cp prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter:` the estimated value for parameter, as a function of the predictor.
- `adjustedx:` the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_deviates:` random deviates calculated using maximum likelihood.
- `cp_deviates:` predictive random deviates calculated using a calibrating prior.
- `cp_method:` a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The normal distribution with predictors on both parameters has probability density function

$$f(x;\alpha,\beta,\gamma,\delta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu(\alpha,\beta))^2/(2\sigma(\gamma,\delta)^2)}$$

where x is the random variable, $\mu = \alpha + \beta t_1$ is the location parameter, modelled as a function of parameters α, β and predictor t_1, where t_1 is typically the ensemble mean, and $\sigma = \exp(\gamma + \delta \log(t_2))$ is the scale parameter, modelled as a function of parameters γ, δ and predictor t_2, where t_2 is typically the ensemble spread.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha,\beta,\gamma,\delta) \propto \frac{1}{\sigma}$$

as given in the Jewson et al. (2025) reference given below.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which introduces this model.

- Jewson S., Olivetti L., Messori G., Northop P., Sweeting T. (2025): An Objective Bayesian Method for Including Parameter Uncertainty in Ensemble Model Output Statistics; QJRMS (Quarterly Journal of the Royal Meteorological Society).

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),

- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d060norm_p1_example_data_v1_x
tt=fitdistcp::d060norm_p1_example_data_v1_t
p=c(1:9)/10
n0=10
q=qnorm_p12_cp(x,t1=tt,t2=tt,n01=n0,n02=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qnorm_p12_cp)",
 main="Normal w/ p1: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

norm_p12_exampledata *Norm_p12 example data*

Description

Norm_p12 example data

Usage

```
norm_p12_exampledata(iseed)
```

Arguments

iseed	The random seed

Value

A list containing data to run an example

norm_p12_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
norm_p12_f1fa(x, t01, t02, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

norm_p12_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
norm_p12_f1fw(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

norm_p12_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
norm_p12_f2fa(x, t01, t02, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

`norm_p12_f2fw` *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
norm_p12_f2fw(x, t1, t2, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`norm_p12_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p12_fd(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

---

norm_p12_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p12_fdd(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

norm_p12_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
norm_p12_ldda(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

norm_p12_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
norm_p12_lddda(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

`norm_p12_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
norm_p12_logf(params, x, t1, t2, nonnegslopesonly = FALSE)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`nonnegslopesonly`	logical that indicates whether to disallow non-negative slopes

Value

Scalar value.

`norm_p12_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p12_logfdd(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

---

norm_p12_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p12_logfddd(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

3d array

norm_p12_loglik *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
norm_p12_loglik(vv, x, t1, t2)
```

Arguments

vv	parameters
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd

Value

Scalar

norm_p12_logscores *Log scores for 5 predictions calculated using leave-one-out*

Description

Log scores for 5 predictions calculated using leave-one-out

Usage

```
norm_p12_logscores(logscores, x, t1, t2, ics)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ics	initial conditions for the maximum likelihood search

Value

Two scalars

norm_p12_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
norm_p12_mu1fa(alpha, t01, t02, v1, v2, v3, v4)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

norm_p12_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
norm_p12_mu2fa(alpha, t01, t02, v1, v2, v3, v4)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

`norm_p12_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
norm_p12_p1fa(x, t01, t02, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t01`	a single value of the predictor (specify either `t01` or `n01` but not both)
`t02`	a single value of the predictor (specify either `t02` or `n02` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Vector

`norm_p12_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
norm_p12_p2fa(x, t01, t02, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Matrix

---

norm_p12_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p12_pd(x, t1, t2, v1, v2, v3, v4)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
v1	first parameter
v2	second parameter
v3	third parameter
v4	fourth parameter

Value

Vector

`norm_p12_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_p12_pdd(x, t1, t2, v1, v2, v3, v4)
```

Arguments

`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter
`v4`	fourth parameter

Value

Matrix

`norm_p12_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
norm_p12_predictordata(predictordata, x, t1, t2, t01, t02, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
params	model parameters for calculating logf

Value

Two vectors

---

norm_p12_setics *Set initial conditions*

Description

Set initial conditions

Usage

```
norm_p12_setics(x, t1, t2, ics)
```

Arguments

x	a vector of training data values
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ics	initial conditions for the maximum likelihood search

Value

Vector

`norm_p12_waic` *Waic*

Description

Waic

Usage

```
norm_p12_waic(
  waicscores,
  x,
  t1,
  t2,
  v1hat,
  v2hat,
  v3hat,
  v4hat,
  lddi,
  lddd,
  lambdad
)
```

Arguments

`waicscores`	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
`x`	a vector of training data values
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`v1hat`	first parameter
`v2hat`	second parameter
`v3hat`	third parameter
`v4hat`	fourth parameter
`lddi`	inverse observed information matrix
`lddd`	third derivative of log-likelihood
`lambdad`	derivative of the log prior

Value

Two numeric values.

norm_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
norm_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
norm_p2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_pd — *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

norm_pdd — *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
norm_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

norm_unbiasedv_params *Method of moments estimator*

Description

Method of moments estimator

Usage

```
norm_unbiasedv_params(x)
```

Arguments

x	a vector of training data values

Value

Vector

norm_waic *Waic*

Description

Waic

Usage

```
norm_waic(waicscores, x, v1hat, v2hat)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter

Value

Two numeric values.

pareto_k2_cp *Pareto Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qpareto_k2_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  kscale = 1,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rpareto_k2_cp(n, x, kscale = 1, rust = FALSE, mlcp = TRUE, debug = FALSE)

dpareto_k2_cp(x, y = x, kscale = 1, rust = FALSE, nrust = 1000, debug = FALSE)
```

```
ppareto_k2_cp(x, y = x, kscale = 1, rust = FALSE, nrust = 1000, debug = FALSE)

tpareto_k2_cp(n, x, kscale = 1, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
kscale	the known scale parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.

- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Pareto distribution has various forms. The form we are using has exceedance distribution function

$$S(x;\alpha) = \left(\frac{\sigma}{x}\right)^{\alpha}$$

where $x \geq \sigma$ is the random variable and $\alpha > 0, \sigma > 0$ are the shape and scale parameters. We consider the scale parameter σ to be known (hence the k2 in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(\alpha) \propto \frac{1}{\alpha}$$

as given in Jewson et al. (2025). Some others authors may refer to the shape and scale parameters as the scale and location parameters, respectively.

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),

- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d011pareto_k2_example_data_v1
p=c(1:9)/10
q=qpareto_k2_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles)
xmax=max(q$ml_quantiles,q$cp_quantiles)
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qpareto_k2_cp)",
 main="Pareto: quantile estimates")
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

pareto_k2_f1fa *The first derivative of the density*

Description

The first derivative of the density

The first derivative of the density

Usage

```
pareto_k2_f1fa(x, v1, kscale)

pareto_k2_f1fa(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Vector

Vector

pareto_k2_f2fa *The second derivative of the density*

Description

The second derivative of the density

The second derivative of the density

Usage

```
pareto_k2_f2fa(x, v1, kscale)

pareto_k2_f2fa(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Matrix

Matrix

`pareto_k2_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_fd(x, v1, v2)

pareto_k2_fd(x, v1, v2)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`v2`	second parameter

Value

Vector

Vector

`pareto_k2_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_fdd(x, v1, v2)

pareto_k2_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

Matrix

`pareto_k2_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

The second derivative of the normalized log-likelihood

Usage

```
pareto_k2_ldda(x, v1, kscale)

pareto_k2_ldda(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Matrix

Matrix

pareto_k2_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

The third derivative of the normalized log-likelihood

Usage

```
pareto_k2_lddda(x, v1, kscale)

pareto_k2_lddda(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

3d array

3d array

pareto_k2_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
pareto_k2_logf(params, x, kscale)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values
kscale	the known scale parameter

Value

Scalar value.

pareto_k2_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_logfdd(x, v1, v2)

pareto_k2_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

Matrix

pareto_k2_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_logfdddd(x, v1, v2)

pareto_k2_logfdddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

3d array

`pareto_k2_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
pareto_k2_logscores(logscores, x, kscale)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
kscale	the known scale parameter

Value

Two scalars

pareto_k2_ml_params *Maximum likelihood estimator*

Description

Maximum likelihood estimator

Usage

```
pareto_k2_ml_params(x, kscale)
```

Arguments

x	a vector of training data values
kscale	the known scale parameter

Value

Scalar

pareto_k2_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Minus the first derivative of the cdf, at alpha

Usage

```
pareto_k2_mu1fa(alpha, v1, kscale)

pareto_k2_mu1fa(alpha, v1, kscale)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
kscale	the known scale parameter

Value

Vector

Vector

pareto_k2_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Minus the second derivative of the cdf, at alpha

Usage

```
pareto_k2_mu2fa(alpha, v1, kscale)
```

```
pareto_k2_mu2fa(alpha, v1, kscale)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
kscale	the known scale parameter

Value

Matrix

Matrix

pareto_k2_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

The first derivative of the cdf

Usage

```
pareto_k2_p1fa(x, v1, kscale)
```

```
pareto_k2_p1fa(x, v1, kscale)
```

Arguments

x	a vector of training data values
v1	first parameter
kscale	the known scale parameter

Value

Vector

Vector

`pareto_k2_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

The second derivative of the cdf

Usage

```
pareto_k2_p2fa(x, v1, kscale)

pareto_k2_p2fa(x, v1, kscale)
```

Arguments

`x`	a vector of training data values
`v1`	first parameter
`kscale`	the known scale parameter

Value

Matrix

Matrix

`pareto_k2_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_pd(x, v1, v2)

pareto_k2_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

Vector

`pareto_k2_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_k2_pdd(x, v1, v2)

pareto_k2_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

Matrix

pareto_k2_waic *Waic*

Description

Waic

Usage

```
pareto_k2_waic(waicscores, x, v1hat, kscale)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
kscale	the known scale parameter

Value

Two numeric values.

pareto_p1k2_cp *Pareto Distribution with a Predictor, Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qpareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  kscale = 1,
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rpareto_p1k2_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  kscale = 1,
  rust = FALSE,
  mlcp = TRUE,
  centering = TRUE,
  debug = FALSE
)

dpareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  kscale = 1,
  rust = FALSE,
```

```
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

ppareto_p1k2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  kscale = 1,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tpareto_p1k2_cp(n, x, t, kscale = 1, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
kscale	the known scale parameter
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Pareto distribution with a predictor has various forms. The form we are using has exceedance distribution function

$$S(x; a, b) = \left(\frac{\sigma}{x}\right)^{\alpha(a,b)}$$

where $x \geq \sigma$ is the random variable, $\alpha = \exp(-a - bt)$ is the shape parameter, modelled as a function of parameters a, b, and σ is the scale parameter. We consider the scale parameter σ to be known (hence the `k2` in the name).

The calibrating prior is given by the right Haar prior, which is

$$\pi(a, b) \propto 1$$

as given in Jewson et al. (2025). Note that others authors have referred to the shape and scale parameters as the scale and location parameters, respectively.

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),

- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d056pareto_p1k2_example_data_v1_x
tt=fitdistcp::d056pareto_p1k2_example_data_v1_t
p=c(1:9)/10
n0=10
q=qpareto_p1k2_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qpareto_p1k2_cp)",
 main="Pareto w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

pareto_p1k2_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
pareto_p1k2_f1fa(x, t0, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Vector

pareto_p1k2_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
pareto_p1k2_f1fw(x, t, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Vector

pareto_p1k2_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
pareto_p1k2_f2fa(x, t0, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Matrix

pareto_p1k2_f2fw *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
pareto_p1k2_f2fw(x, t, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Matrix

`pareto_p1k2_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_fd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`pareto_p1k2_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_fdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`pareto_p1k2_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
pareto_p1k2_ldda(x, t, v1, v2, kscale)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`kscale`	the known scale parameter

Value

Matrix

`pareto_p1k2_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
pareto_p1k2_lddda(x, t, v1, v2, kscale)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`kscale`	the known scale parameter

Value

3d array

`pareto_p1k2_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
pareto_p1k2_logf(params, x, t, kscale)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`kscale`	the known scale parameter

Value

Scalar value.

`pareto_p1k2_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_logfdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`pareto_p1k2_logfdddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_logfdddd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`pareto_p1k2_loglik` *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
pareto_p1k2_loglik(vv, x, t, kscale)
```

Arguments

`vv`	parameters
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`kscale`	the known scale parameter

Value

Scalar

`pareto_p1k2_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
pareto_p1k2_logscores(logscores, x, t, kscale, debug)
```

Arguments

`logscores`	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
`x`	a vector of training data values
`t`	a vector or matrix of predictors
`kscale`	the known scale parameter
`debug`	debug flag

Value

Two scalars

`pareto_p1k2_means` *pareto_k1 distribution: RHP mean*

Description

pareto_k1 distribution: RHP mean

Usage

```
pareto_p1k2_means(
  means,
  t0,
  ml_params,
  lddi,
  lddd,
  lambdad_rhp,
  nx,
  dim = 2,
  kscale
)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters
kscale	the known scale parameter

Value

Two scalars

---

pareto_p1k2_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
pareto_p1k2_mu1fa(alpha, t0, v1, v2, kscale)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Vector

pareto_p1k2_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
pareto_p1k2_mu2fa(alpha, t0, v1, v2, kscale)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Matrix

pareto_p1k2_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
pareto_p1k2_p1fa(x, t0, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Vector

pareto_p1k2_p2fa *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
pareto_p1k2_p2fa(x, t0, v1, v2, kscale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
kscale	the known scale parameter

Value

Matrix

pareto_p1k2_pd *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_pd(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`pareto_p1k2_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
pareto_p1k2_pdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`pareto_p1k2_predictordata`
Predicted Parameter and Generalized Residuals

Description

Predicted Parameter and Generalized Residuals

Usage

```
pareto_p1k2_predictordata(predictordata, x, t, t0, params, kscale)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf
kscale	the known scale parameter

Value

Two vectors

---

pareto_p1k2_waic *Waic*

Description

Waic

Usage

```
pareto_p1k2_waic(waicscores, x, t, v1hat, v2hat, kscale, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
kscale	the known scale parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

pcauchy_p1 *Cauchy-with-p1 distribution function*

Description

Cauchy-with-p1 distribution function

Usage

```
pcauchy_p1(x, t0, ymn, slope, scale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

pexp_p1 *Exponential-with-p1 distribution function*

Description

Exponential-with-p1 distribution function

Usage

```
pexp_p1(x, t0, ymn, slope)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

pfrechet_p2k1 *Frechet_k1-with-p2 distribution function*

Description

Frechet_k1-with-p2 distribution function

Usage

```
pfrechet_p2k1(x, t0, ymn, slope, lambda, kloc)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
lambda	the lambda parameter of the distribution
kloc	the known location parameter

Value

Vector

pgev_p1 *GEVD-with-p1: Distribution function*

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p1(y, t0, ymn, slope, sigma, xi)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

---

pgev_p12 *GEVD-with-p1: Distribution function*

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p12(y, t1, t2, ymn, slope, sigma1, sigma2, xi)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

---

pgev_p123 *GEVD-with-p1: Distribution function*

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p123(y, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution

Value

Vector

---

pgev_p1k3 *GEV-with-known-shape-with-p1 distribution function*

Description

GEV-with-known-shape-with-p1 distribution function

Usage

```
pgev_p1k3(x, t0, ymn, slope, sigma, kshape)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
kshape	the known shape parameter

Value

Vector

pgev_p1n *GEVD-with-p1: Distribution function*

Description

GEVD-with-p1: Distribution function

Usage

```
pgev_p1n(y, t0, params)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Vector

pgumbel_p1 *Gumbel-with-p1 distribution function*

Description

Gumbel-with-p1 distribution function

Usage

```
pgumbel_p1(x, t0, ymn, slope, sigma)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

plnorm_p1 *Normal-with-p1 distribution function*

Description

Normal-with-p1 distribution function

Usage

```
plnorm_p1(x, t0, ymn, slope, sigma)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

plogis_p1 *Logistic-with-p1 distribution function*

Description

Logistic-with-p1 distribution function

Usage

```
plogis_p1(x, t0, ymn, slope, scale)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

plst_p1k3 *LST-with-p1 distribution function*

Description

LST-with-p1 distribution function

Usage

```
plst_p1k3(x, t0, ymn, slope, sigma, kdf)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
kdf	the known degrees of freedom parameter

Value

Vector

pnorm_p1 *Normal-with-p1 distribution function*

Description

Normal-with-p1 distribution function

Usage

```
pnorm_p1(x, t0, ymn, slope, sigma)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

pnorm_p1_formula *Linear regression formula, densities*

Description

Linear regression formula, densities

Usage

```
pnorm_p1_formula(y, tresid, tresid0, nx, muhat0, v3hat)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
tresid	predictor residuals
tresid0	predictor residual at the point being predicted
nx	length of training data
muhat0	muhat at the point being predicted
v3hat	third parameter

Value

Vector

pnorm_p12 *Normal-with-p12: Distribution function*

Description

Normal-with-p12: Distribution function

Usage

```
pnorm_p12(y, t01, t02, ymn, slope, sigma1, sigma2)
```

Arguments

y	a vector of values at which to calculate the density and distribution functions
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution

Value

Vector

`ppareto_p1k2` *pareto_k1-with-p2 distribution function*

Description

pareto_k1-with-p2 distribution function

Usage

```
ppareto_p1k2(x, t0, ymn, slope, kscale)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`kscale`	the known scale parameter

Value

Vector

`punif_formula` *Predictive CDFs*

Description

Predictive CDFs

Usage

```
punif_formula(x, y)
```

Arguments

`x`	a vector of training data values
`y`	a vector of values at which to calculate the density and distribution functions

Value

Two vectors

pweibull_p2 *Weibull-with-p1 distribution function*

Description

Weibull-with-p1 distribution function

Usage

```
pweibull_p2(x, t0, shape, ymn, slope)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
shape	the shape parameter of the distribution
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

qcauchy_p1 *Cauchy-with-p1 quantile function*

Description

Cauchy-with-p1 quantile function

Usage

```
qcauchy_p1(p, t0, ymn, slope, scale)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

qexp_p1 *-with-p1 quantile function*

Description

-with-p1 quantile function

Usage

```
qexp_p1(p, t0, ymn, slope)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

qfrechet_p2k1 *Frechet_k1-with-p2 quantile function*

Description

Frechet_k1-with-p2 quantile function

Usage

```
qfrechet_p2k1(p, t0, ymn, slope, lambda, kloc)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
lambda	the lambda parameter of the distribution
kloc	the known location parameter

Value

Vector

qgamma_k1_ppm *Temporary dummy for one of the cp models*

Description

Temporary dummy for one of the cp models

Usage

```
qgamma_k1_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgamma_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgamma_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgev_k12_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_k12_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

qgev_mpd_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_mpd_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgev_p1 *GEVD-with-p1: Quantile function*

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p1(p, t0, ymn, slope, sigma, xi)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution

Value

Vector

qgev_p1_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_p1_ppm(x, t, n0, p)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

`qgev_p12` *GEVD-with-p1: Quantile function*

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p12(p, t1, t2, ymn, slope, sigma1, sigma2, xi)
```

Arguments

`p`	a vector of probabilities at which to generate predictive quantiles
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`ymn`	the location parameter of the function of the predictor
`slope`	the slope of the function of the predictor
`sigma1`	first coefficient for the sigma parameter of the distribution
`sigma2`	second coefficient for the sigma parameter of the distribution
`xi`	the shape parameter of the distribution

Value

Vector

qgev_p123 *GEVD-with-p1: Quantile function*

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p123(p, t1, t2, t3, ymn, slope, sigma1, sigma2, xi1, xi2)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t1	a vector of predictors for the mean
t2	a vector of predictors for the sd
t3	a vector of predictors for the shape
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution
xi1	first coefficient for the shape parameter of the distribution
xi2	second coefficient for the shape parameter of the distribution

Value

Vector

qgev_p1k3 *GEV-with-known-shape-with-p1 quantile function*

Description

GEV-with-known-shape-with-p1 quantile function

Usage

```
qgev_p1k3(p, t0, ymn, slope, sigma, kshape)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
kshape	the known shape parameter

Value

Vector

---

qgev_p1n *GEVD-with-p1: Quantile function*

Description

GEVD-with-p1: Quantile function

Usage

```
qgev_p1n(p, t0, params)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Vector

qgev_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgev_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.

- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgpd_k1_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qgpd_k1_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

---

qgumbel_p1 *Gumbel-with-p1 quantile function*

Description

Gumbel-with-p1 quantile function

Usage

```
qgumbel_p1(p, t0, ymn, slope, sigma)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

---

qlnorm_p1 *Normal-with-p1 quantile function*

Description

Normal-with-p1 quantile function

Usage

```
qlnorm_p1(p, t0, ymn, slope, sigma)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

qlogis_p1 *Logistic-with-p1 quantile function*

Description

Logistic-with-p1 quantile function

Usage

```
qlogis_p1(p, t0, ymn, slope, scale)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
scale	the scale parameter of the distribution

Value

Vector

qlst_p1k3 *LST-with-p1 quantile function*

Description

LST-with-p1 quantile function

Usage

```
qlst_p1k3(p, t0, ymn, slope, sigma, kdf)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution
kdf	the known degrees of freedom parameter

Value

Vector

qnorm_p1 *Normal-with-p1 quantile function*

Description

Normal-with-p1 quantile function

Usage

```
qnorm_p1(p, t0, ymn, slope, sigma)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma	the sigma parameter of the distribution

Value

Vector

qnorm_p1_formula *Linear regression formula, quantiles*

Description

Linear regression formula, quantiles

Usage

```
qnorm_p1_formula(alpha, tresid, tresid0, nx, muhat0, v3hat)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
tresid	predictor residuals
tresid0	predictor residual at the point being predicted
nx	length of training data
muhat0	muhat at the point being predicted
v3hat	third parameter

Value

Vector

qnorm_p12 *Normal-with-p12: Quantile function*

Description

Normal-with-p12: Quantile function

Usage

```
qnorm_p12(p, t01, t02, ymn, slope, sigma1, sigma2)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t01	a single value of the predictor (specify either t01 or n01 but not both)
t02	a single value of the predictor (specify either t02 or n02 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
sigma1	first coefficient for the sigma parameter of the distribution
sigma2	second coefficient for the sigma parameter of the distribution

Value

Vector

qntt_ppm *Temporary dummy for one of the ppm models*

Description

Temporary dummy for one of the ppm models

Usage

```
qntt_ppm(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

`q****` returns a list containing at least the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_value`: the value of the log-likelihood at the maximum.
- `standard_errors`: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- `ml_quantiles`: quantiles calculated using maximum likelihood.
- `cp_quantiles`: predictive quantiles calculated using a calibrating prior.
- `maic`: the AIC score for the maximum likelihood model, times -1/2.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

For models with predictors, `q****` additionally returns:

- `predictedparameter`: the estimated value for parameter, as a function of the predictor.
- `adjustedx`: the detrended values of `x`

`r****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_deviates`: random deviates calculated using maximum likelihood.
- `cp_deviates`: predictive random deviates calculated using a calibrating prior.
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

qpareto_p1k2 *pareto_k1-with-p2 quantile function*

Description

pareto_k1-with-p2 quantile function

Usage

```
qpareto_p1k2(p, t0, ymn, slope, kscale)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor
kscale	the known scale parameter

Value

Vector

qunif_formula *Predictive Quantiles*

Description

Predictive Quantiles

Usage

```
qunif_formula(x, p)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles

Value

Two vectors

qweibull_p2 *Weibull-with-p1 quantile function*

Description

Weibull-with-p1 quantile function

Usage

```
qweibull_p2(p, t0, shape, ymn, slope)
```

Arguments

p	a vector of probabilities at which to generate predictive quantiles
t0	a single value of the predictor (specify either t0 or n0 but not both)
shape	the shape parameter of the distribution
ymn	the location parameter of the function of the predictor
slope	the slope of the function of the predictor

Value

Vector

reltest *Evaluation of Reliability for Models in the* fitdistcp *Package*

Description

Uses simulations to evaluate the reliability of the predictive quantiles produced by the q****_cp routines in the fitdistcp package.

Usage

```
reltest(
  model = "exp",
  ntrials = 1000,
  nrepeats = 3,
  nx = 20,
  params = NA,
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE,
  dmgs = TRUE,
  debug = FALSE,
```

```
  aderivs = TRUE,
  unbiasedv = FALSE,
  pwm = FALSE,
  minxi = -10,
  maxxi = 10
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k1", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k4", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data to use.
params	values for the parameters for the specified distribution
alpha	the exceedance probability values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off
dmgs	logical to turn DMGS calculations on and off (to optimize speed for maxlik only calculations)
debug	logical for turning debug messages on and off
aderivs	logical for whether to use analytic derivatives (instead of numerical)
unbiasedv	logical for whether to use the unbiased variance instead of maxlik (for the normal)
pwm	logical for whether to use PWM instead of maxlik (for the GEV)
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

For "exp", "pareto_k1", "unif", "norm", "lnorm", "norm_p1" and "lnorm_p1", the calibrating prior quantiles are calculated using the right Haar prior and an exact solution for the Bayesian prediction integral. They will converge towards exact reliability with a large enough number of trials, for any sample size.

For "halfnorm", "norm_dmgs", "lnorm_dmgs", "gnorm_k3", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "gumbel_p1", "logis_p1" and

"lst_p1k4" "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5" the calibrating prior quantiles are calculated using the right Haar prior, with the DMGS asymptotic solution for the Bayesian prediction integral. They will converge towards good reliability with a large enough number of trials, with the only deviation from exact reliability being due to the neglect of higher order terms in the asymptotic expansion. They will converge towards exact reliability with a large enough number of trials and a large enough sample size.

For "gamma", "invgamma", "invgauss", "gev", "gpd_k1" and "gev_p1", "gev_p12", "gev_p123", the calibrating prior quantiles are calculated using the "fitdistcp" recommended calibrating priors, with the DMGS asymptotic solution for the Bayesian prediction integral. The chosen priors give reasonably good reliability with a large enough number of trials, and for large sample sizes, but may give poor reliability for small sample sizes (e.g., n<20).

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),

- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the exponential distribution
reltest()
```

reltest *Evaluation of Reliability for Models in the* fitdistcp *Package*

Description

Uses simulations to evaluate the reliability of the predictive quantiles produced by the q****_cp routines in the fitdistcp package.

Usage

```
reltest(
  model = "exp",
  ntrials = 1000,
  nrepeats = 3,
  nx = 20,
  params = NA,
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE,
  unbiasedv = FALSE,
  pwm = FALSE,
  minxi = -10,
  maxxi = 10
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k1", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k4", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data to use.
params	values for the parameters for the specified distribution
alpha	the exceedance probability values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off

dmgs	logical to turn DMGS calculations on and off (to optimize speed for maxlik only calculations)
debug	logical for turning debug messages on and off
aderivs	logical for whether to use analytic derivatives (instead of numerical)
unbiasedv	logical for whether to use the unbiased variance instead of maxlik (for the normal)
pwm	logical for whether to use PWM instead of maxlik (for the GEV)
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

For "exp", "pareto_k1", "unif", "norm", "lnorm", "norm_p1" and "lnorm_p1", the calibrating prior quantiles are calculated using the right Haar prior and an exact solution for the Bayesian prediction integral. They will converge towards exact reliability with a large enough number of trials, for any sample size.

For "halfnorm", "norm_dmgs", "lnorm_dmgs", "gnorm_k3", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k3", "gumbel_p1", "logis_p1" and "lst_p1k4" "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k4", "norm_p12", "lst_p12k5" the calibrating prior quantiles are calculated using the right Haar prior, with the DMGS asymptotic solution for the Bayesian prediction integral. They will converge towards good reliability with a large enough number of trials, with the only deviation from exact reliability being due to the neglect of higher order terms in the asymptotic expansion. They will converge towards exact reliability with a large enough number of trials and a large enough sample size.

For "gamma", "invgamma", "invgauss", "gev", "gpd_k1" and "gev_p1", "gev_p12", "gev_p123", the calibrating prior quantiles are calculated using the "fitdistcp" recommended calibrating priors, with the DMGS asymptotic solution for the Bayesian prediction integral. The chosen priors give reasonably good reliability with a large enough number of trials, and for large sample sizes, but may give poor reliability for small sample sizes (e.g., n<20).

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),

- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the exponential distribution
reltest()
```

---

reltest_makeep *Calculate EP from one model*

Description

Calculate EP from one model

Usage

```
reltest_makeep(model, pred1, tt0, tt10, tt20, tt30, params)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k3", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
pred1	quantile predictions
tt0	value of the predictor
tt10	value of predictor 1

tt20	value of predictor 2
tt30	value of predictor 3
params	the model parameters

Value

Vector

`reltest_makemaxep` *Calculate MaxEP from one model*

Description

Calculate MaxEP from one model

Usage

```
reltest_makemaxep(model, ml_max, tt0, tt10, tt20, tt30, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
ml_max	predicted max value
tt0	value of the predictor
tt10	value of predictor 1
tt20	value of predictor 2
tt30	value of predictor 3
params	the model parameters

Value

Vector

reltest_params *Set default params for the chosen model*

Description

Set default params for the chosen model

Usage

```
reltest_params(model = "exp", params)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k3", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
params	values for the parameters for the specified distribution

Value

Vector

reltest_predict *Make prediction from one model*

Description

Make prediction from one model

Usage

```
reltest_predict(
  model,
  xx,
  tt,
  tt1,
  tt2,
  tt3,
  n0,
  n10,
  n20,
```

```
  n30,
  pp,
  params,
  dmgs = TRUE,
  debug = FALSE,
  aderivs = TRUE,
  unbiasedv = FALSE,
  pwm = FALSE,
  minxi = -10,
  maxxi = 10
)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "exp_p1k4", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
xx	training data
tt	predictor vector
tt1	predictor vector 1
tt2	predictor vector 2
tt3	predictor vector 3
n0	index for predictor vector
n10	index for predictor vector 1
n20	index for predictor vector 2
n30	index for predictor vector 2
pp	probabilites at which to make quantile predictions
params	model parameters
dmgs	flag for whether to run dmgs calculations or not
debug	flag for turning debug messages on
aderivs	a logical for whether to use analytic derivatives (instead of numerical)
unbiasedv	a logical for whether to use the unbiased variance instead of maxlik (for the normal)
pwm	a logical for whether to use PWM instead of maxlik (for the GEV)
minxi	minimum value for EVT shape parameter
maxxi	maximum value for EVT shape parameter

Value

Two vectors

`reltest_simulate` *Random training data from one model*

Description

Random training data from one model

Usage

```
reltest_simulate(
  model = "exp",
  nx = 20,
  tt,
  tt1,
  tt2,
  tt3,
  params,
  minxi = -10,
  maxxi = -10
)
```

Arguments

`model`	which distribution to test. Possibles values are "exp", "pareto_k2", "halfnorm", "unif", "norm", "norm_dmgs", "gnorm_k3", "lnorm", "lnorm_dmgs", "logis", "lst_k3", "cauchy", "gumbel", "frechet_k1", "weibull", "gev_k3", "exp_p1", "pareto_p1k2", "norm_p1", "lnorm_p1", "logis_p1", "lst_p1k3", "cauchy_p1", "gumbel_p1", "frechet_p2k1", "weibull_p2", "gev_p1k3", "norm_p12", "lst_p12k3", "gamma", "invgamma", "invgauss", "gev", "gpd_k1", "gev_p1". "gev_p12". "gev_p123".
`nx`	the length of the training data to use.
`tt`	predictor vector
`tt1`	predictor vector 1
`tt2`	predictor vector 2
`tt3`	predictor vector 2
`params`	values for the parameters for the specified distribution
`minxi`	minimum value for EVT shape parameter
`maxxi`	maximum value for EVT shape parameter

Value

Vector

reltest2 *Evaluation of Reliability for Certain Additional Models in the* fitdistcp *Package*

Description

This routine is mainly for reproducing certain results in Jewson et al. (2025), and not of general interest.

It uses simulations to evaluate the reliability of the predictive quantiles produced by the qgev_cp, ggpd_cp and qgev_p1_cp routines in the fitdistcp package. For each model, results for 5 models are calculated. This is to illustrate that the calibrating prior predictions dominate the ml, flat, crhp_ml and jp predictions, in terms of reliability.

Usage

```
reltest2(
  model = "gev",
  ntrials = 100,
  nrepeats = 3,
  nx = 50,
  params = c(0, 1, 0),
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE
)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_p1".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data.
params	values for the parameters for the specified distribution
alpha	the alpha values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

The cp predictive quantiles generally give reasonably good reliability, especially for sample sizes of ~100. The other predictions generally give poor reliability.

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to fitdistcp, with more examples, is given on this webpage.

The fitdistcp package currently includes the following models (in alphabetical order):

- Cauchy (cauchy),
- Cauchy with linear predictor on the mean (cauchy_p1),
- Exponential (exp),
- Exponential with log-linear predictor on the scale (exp_p1),
- Frechet with known location parameter (frechet_k1),
- Frechet with log-linear predictor on the scale and known location parameter (frechet_p2k1),
- Gamma (gamma),
- Generalized normal (gnorm),
- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),

- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the GEV distribution
reltest2(nrepeats=1)
```

---

reltest2 *Evaluation of Reliability for Certain Additional Models in the* fitdistcp *Package*

Description

This routine is mainly for reproducing certain results in Jewson et al. (2025), and not of general interest.

It uses simulations to evaluate the reliability of the predictive quantiles produced by the qgev_cp, ggpd_cp and qgev_p1_cp routines in the fitdistcp package. For each model, results for 5 models are calculated. This is to illustrate that the calibrating prior predictions dominate the ml, flat, crhp_ml and jp predictions, in terms of reliability.

Usage

```
reltest2(
  model = "gev",
  ntrials = 100,
  nrepeats = 3,
  nx = 50,
  params = c(0, 1, 0),
  alpha = seq(0.005, 0.995, 0.005),
  plotflag = TRUE,
  verbose = TRUE
)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_p1".
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence. 3 is a good choice.
nx	the length of the training data.
params	values for the parameters for the specified distribution
alpha	the alpha values at which to test
plotflag	logical to turn the plotting on and off
verbose	logical to turn loop counting on and off

Details

The maximum likelihood quantiles (plotted in blue) do not give good reliability. They typically underestimate the tails (see panel (f)).

The cp predictive quantiles generally give reasonably good reliability, especially for sample sizes of ~100. The other predictions generally give poor reliability.

Value

A plot showing 9 different reliability checks, and a list containing various outputs, including the probabilities shown in the plot.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),

- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
set.seed(1)
# example 1
# -runs the default settings, which test reliability for the GEV distribution
reltest2(nrepeats=1)
```

---

reltest2_cases *Cases*

Description

Cases

Usage

```
reltest2_cases(model = "gev", nx = 50, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
nx	length of training data
params	model parameters

Value

Two integers

reltest2_makeep *Cases*

Description

Cases

Usage

```
reltest2_makeep(model, pred1, tt0, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
pred1	quantile predictions
tt0	value of predictor vector
params	model parameters

Value

Vector

reltest2_plot *Plotting routine for reltest2*

Description

Plots 9 diagnostics related to predictive probability matching.

Usage

```
reltest2_plot(
  model,
  ntrials,
  nrepeats,
  nx,
  params,
  nmethods,
  alpha,
  freqexceeded,
  case
)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd", "gev_p1".
ntrials	the number of trials o run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence
nx	the length of the training data.
params	values for the parameters for the specified distribution
nmethods	the number of methods being tested
alpha	the values of alpha being tested
freqexceeded	the exceedance counts
case	there are 3 cases (must be set to case=1 except for my testing)

Value

Plots the results of reliability testing

---

reltest2_predict *Make prediction from one model*

Description

Make prediction from one model

Usage

```
reltest2_predict(model = "gev", xx, tt, n0, pp, params, case, nmethods)
```

Arguments

model	which distribution to test. Possibles values are "exp", "pareto_k1", "halfnorm", "norm", "lnorm", "gumbel", "frechet_k1", "weibull", "gev_k3", "logis", "lst_k3", "cauchy", "norm_p1", "lnorm_p1", "logis_p1", "lst_k3p1", "gumbel_p1", "norm_p12", "gev", "gpd", "gev_p1".
xx	training data
tt	predictor vector
n0	index for predictor vector
pp	probabilities to predict
params	model parameters
case	the case number: different models have different lists of methods
nmethods	the number of methods: different models have different numbers of methods

Value

Vector

reltest2_simulate *Random training data from one model*

Description

Random training data from one model

Usage

```
reltest2_simulate(model = "gev", nx = 50, tt, params)
```

Arguments

model	which distribution to test. Possibles values are "gev", "gpd_k1", "gev_pred1".
nx	the length of the training data.
tt	the predictor
params	values for the parameters for the specified distribution

Value

Vector

rgev_minmax *rgev but with maxlik xi guaranteed within bounds*

Description

rgev but with maxlik xi guaranteed within bounds

Usage

```
rgev_minmax(nx, mu = 0, sigma = 1, xi = 0, minxi = -1, maxxi = 1)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

Vector

rgev_p1_minmax *rgev for gev_p1 but with maxlik xi within bounds*

Description

rgev for gev_p1 but with maxlik xi within bounds

Usage

```
rgev_p1_minmax(
  nx,
  mu = 0,
  sigma = 1,
  xi = 0,
  tt,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

nx	length of training data
mu	the location parameter of the distribution
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
tt	a vector of predictors
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi
centering	indicates whether the routine should center the data or not

Value

Vector

`rgev_p12_minmax` *rgev for gev_p12 but with maxlik xi within bounds*

Description

rgev for gev_p12 but with maxlik xi within bounds

Usage

```
rgev_p12_minmax(
  nx,
  mu = 0,
  sigma = 1,
  xi = 0,
  t1,
  t2,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

`nx`	length of training data
`mu`	the location parameter of the distribution
`sigma`	the sigma parameter of the distribution
`xi`	the shape parameter of the distribution
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi
`centering`	indicates whether the routine should center the data or not

Value

Vector

`rgev_p123_minmax` *rgev for gev_p123 but with maxlik xi within bounds*

Description

rgev for gev_p123 but with maxlik xi within bounds

Usage

```
rgev_p123_minmax(
  nx,
  mu = 0,
  sigma = 1,
  xi = 0,
  t1,
  t2,
  t3,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

`nx`	length of training data
`mu`	the location parameter of the distribution
`sigma`	the sigma parameter of the distribution
`xi`	the shape parameter of the distribution
`t1`	a vector of predictors for the mean
`t2`	a vector of predictors for the sd
`t3`	a vector of predictors for the shape
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi
`centering`	indicates whether the routine should center the data or not

Value

Vector

`rgev_p1n_minmax` *rgev for gev_p1n but with maxlik xi within bounds*

Description

rgev for gev_p1n but with maxlik xi within bounds

Usage

```
rgev_p1n_minmax(
  nx,
  mu = 0,
  sigma = 1,
  xi = 0,
  tt,
  minxi = -0.45,
  maxxi = 0.45,
  centering = TRUE
)
```

Arguments

`nx`	length of training data
`mu`	the location parameter of the distribution
`sigma`	the sigma parameter of the distribution
`xi`	the shape parameter of the distribution
`tt`	a vector of predictors
`minxi`	minimum value of shape parameter xi
`maxxi`	maximum value of shape parameter xi
`centering`	indicates whether the routine should center the data or not

Value

Vector

rgpd_k1_minmax *rgpd for gpd_k1 but with maxlik xi within bounds*

Description

rgpd for gpd_k1 but with maxlik xi within bounds

Usage

```
rgpd_k1_minmax(nx, kloc, sigma, xi, minxi = -0.45, maxxi = 0.45)
```

Arguments

nx	length of training data
kloc	the known location parameter
sigma	the sigma parameter of the distribution
xi	the shape parameter of the distribution
minxi	minimum value of shape parameter xi
maxxi	maximum value of shape parameter xi

Value

Vector

rhp_dmgs_cpmethod *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
rhp_dmgs_cpmethod()
```

Value

String

rust_pumethod *Generates a comment about the method*

Description

Generates a comment about the method

Usage

```
rust_pumethod()
```

Value

String

testppm_plot *Plotting routine for testppm*

Description

Plots 9 diagnostics related to predictive probability matching.

Usage

```
testppm_plot(
  model,
  ntrials,
  nrepeats,
  nx,
  params,
  nmethods,
  alpha,
  freqexceeded
)
```

Arguments

model	which distribution to test. Possibles values are
ntrials	the number of trials to run. 5000 typically gives good results.
nrepeats	the number of entire repeats of the test to run, to check for convergence
nx	the length of the training data.
params	values for the parameters for the specified distribution
nmethods	the number of methods being tested
alpha	the values of alpha being tested
freqexceeded	the exceedance counts

Value

Plots the results of reliability testing

`unif_cp` *Uniform Distribution Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qunif_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  debug = FALSE,
  aderivs = TRUE
)

runif_cp(n, x, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dunif_cp(x, y = x, debug = FALSE, aderivs = TRUE)

punif_cp(x, y = x, debug = FALSE, aderivs = TRUE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The uniform distribution has probability density function

$$f(x; min, max) = \frac{1}{max - min}$$

and zero otherwise, where $min \leq x \leq max$ is the random variable and min, max are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\lambda) \propto \frac{1}{max - min}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),

- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d025unif_example_data_v1
cat("length(x)=",length(x),"\n")
p=c(1:9)/10
q=qunif_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qunif_cp)",
 main="unif: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

unif_cp *Uniform Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qunif_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  debug = FALSE,
  aderivs = TRUE
)

runif_cp(n, x, mlcp = TRUE, debug = FALSE, aderivs = TRUE)

dunif_cp(x, y = x, debug = FALSE, aderivs = TRUE)

punif_cp(x, y = x, debug = FALSE, aderivs = TRUE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
debug	logical for turning on debug messages
aderivs	(for code testing only) logical for whether to use analytic derivatives (instead of numerical). By default almost all models now use analytical derivatives.
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).

- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The uniform distribution has probability density function

$$f(x; min, max) = \frac{1}{max - min}$$

and zero otherwise, where $min \leq x \leq max$ is the random variable and min, max are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(\lambda) \propto \frac{1}{max - min}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (analytic integration)

For this model, the Bayesian prediction equation is integrated analytically.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),

- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with predictor on the mean (norm_p1),
- Normal with predictors on the mean and sd (norm_p12),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d025unif_example_data_v1
cat("length(x)=",length(x),"\n")
p=c(1:9)/10
q=qunif_cp(x,p)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qunif_cp)",
 main="unif: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
```

weibull_cp *Weibull Distribution Predictions Based on a Calibrating Prior*

Description

The fitdistcp package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data x. For model **** the five functions are as follows:

- q****_cp returns predictive quantiles at the specified probabilities p, and various other diagnostics.
- r****_cp returns n random deviates from the predictive distribution.
- d****_cp returns the predictive density function at the specified values y
- p****_cp returns the predictive distribution function at the specified values y
- t****_cp returns n random deviates from the posterior distribution of the model parameters.

The q, r, d, p routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
)

rweibull_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE)

dweibull_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)
```

```
pweibull_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

tweibull_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Weibull distribution has exceedance distribution function

$$S(x; k, \sigma) = \exp\left(-\left(\frac{x}{\sigma}\right)^k\right)$$

where $x \geq 0$ is the random variable and $k > 0, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d052weibull_example_data_v1
p=c(1:9)/10
q=qweibull_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
```

```
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qweibull_cp)",
 main="Weibull: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`weibull_cp` *Weibull Distribution Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_cp(
  x,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  debug = FALSE
```

```
)

rweibull_cp(n, x, rust = FALSE, mlcp = TRUE, debug = FALSE)

dweibull_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

pweibull_cp(x, y = x, rust = FALSE, nrust = 1000, debug = FALSE)

tweibull_cp(n, x, debug = FALSE)
```

Arguments

x	a vector of training data values
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_pdf: density function from maximum likelihood.
- cp_pdf: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

p*** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_cdf: distribution function from maximum likelihood.
- cp_cdf: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- cp_method: a comment about the method used to generate the cp prediction.

t*** returns a list containing the following:

- theta_samples: random samples from the parameter posterior.

Details of the Model

The Weibull distribution has exceedance distribution function

$$S(x; k, \sigma) = \exp\left(-\left(\frac{x}{\sigma}\right)^k\right)$$

where $x \geq 0$ is the random variable and $k > 0, \sigma > 0$ are the parameters.

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k\sigma}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d052weibull_example_data_v1
p=c(1:9)/10
q=qweibull_cp(x,p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qweibull_cp)",
 main="Weibull: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

weibull_f1fa *The first derivative of the density*

Description

The first derivative of the density

Usage

```
weibull_f1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

weibull_f2fa *The second derivative of the density*

Description

The second derivative of the density

Usage

```
weibull_f2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_fd *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_fd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

weibull_fdd *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_fdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_ldda *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
weibull_ldda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_lddda *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
weibull_lddda(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

weibull_logf *Logf for RUST*

Description

Logf for RUST

Usage

```
weibull_logf(params, x)
```

Arguments

params	model parameters for calculating logf
x	a vector of training data values

Value

Scalar value.

weibull_logfdd *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_logfdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_logfddd *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_logfddd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

3d array

weibull_loglik *log-likelihood function*

Description

log-likelihood function

Usage

```
weibull_loglik(vv, x)
```

Arguments

vv	parameters
x	a vector of training data values

Value

Scalar

weibull_logscores *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
weibull_logscores(logscores, x)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values

Value

Two scalars

weibull_means *MLE and RHP predictive means*

Description

MLE and RHP predictive means

Usage

```
weibull_means(means, ml_params, lddi, lddd, lambdad_rhp, nx, dim = 2)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

weibull_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
weibull_mu1fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Vector

weibull_mu2fa *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
weibull_mu2fa(alpha, v1, v2)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
v1	first parameter
v2	second parameter

Value

Matrix

weibull_p1fa *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
weibull_p1fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

`weibull_p2_cp` *weibull Distribution with a Predictor on the Scale Parameter, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
```

```
)

rweibull_p2_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tweibull_p2_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)

waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

r**** returns a list containing the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_deviates: random deviates calculated using maximum likelihood.
- cp_deviates: predictive random deviates calculated using a calibrating prior.
- cp_method: a comment about the method used to generate the cp prediction.

d**** returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_pdf`: density function from maximum likelihood.
- `cp_pdf`: predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params`: maximum likelihood estimates for the parameters.
- `ml_cdf`: distribution function from maximum likelihood.
- `cp_cdf`: predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method`: a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples`: random samples from the parameter posterior.

Details of the Model

The Weibull distribution with predictor on the scale parameter has exceedance distribution function

$$S(x; k, a, b) = \exp\left(-\left(\frac{x}{\sigma(a,b)}\right)^k\right)$$

where $x \geq 0$ is the random variable, $k > 0$ is the shape parameter and $\sigma = e^{a+bt}$ is the scale parameter, modelled as a function of parameters a, b and predictor t.

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k}$$

as given in Jewson et al. (2025).

Optional Return Values

`q****` optionally returns the following:

If `rust=TRUE`:

- `ru_quantiles`: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on `nrust` samples.

If `waicscores=TRUE`:

- `waic1`: the WAIC1 score for the calibrating prior model.
- `waic2`: the WAIC2 score for the calibrating prior model.

If `logscores=TRUE`:

- `ml_oos_logscore`: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- `cp_oos_logscore`: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If `means=TRUE`:

- `ml_mean`: analytic estimate of the mean of the MLE predictive distribution, where possible
- `cp_mean`: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

`r****` optionally returns the following:

If `rust=TRUE`:

- `ru_deviates`: `nrust` predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

`d****` optionally returns the following:

If `rust=TRUE`:

- `ru_pdf`: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` density functions.

`p****` optionally returns the following:

If `rust=TRUE`:

- `ru_cdf`: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over `nrust` distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the `cp` results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the `q****_cp` routines in `fitdistcp` can be quantified using repeated simulations with the routine `reltest`.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting `rust=TRUE` and looking at the `ru` outputs. The performance for any sample size, in terms of reliability, can be tested using `reltest`.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option `rust=TRUE`. `fitdistcp` then calls Paul Northrop's `rust` package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2024): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),

- GEV (gev),
- GEV with linear predictor on the location (gev_p1),
- GEV with linear predictor on the location and log-linear prediction on the scale (gev_p12),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (gev_p123),
- GEV with linear predictor on the location and known shape (gev_p1k3),
- GEV with known shape (gev_k3),
- GPD with known location (gpd_k1),
- Gumbel (gumbel),
- Gumbel with linear predictor on the mean(gumbel_p1),
- Half-normal (halfnorm),
- Inverse gamma (invgamma),
- Inverse Gaussian (invgauss),
- t distribution with unknown location and scale and known DoF (lst_k3),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (lst_p1k3),
- Logistic (logis),
- Logistic with linear predictor on the location (logis_p1),
- Log-normal (lnorm),
- Log-normal with linear predictor on the location (lnorm_p1),
- Normal (norm),
- Normal with linear predictor on the mean (norm_p1),
- Pareto with known scale (pareto_k2),
- Pareto with log-linear predictor on the shape and known scale (pareto_p1k2),
- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d073weibull_p2_example_data_v1_x
tt=fitdistcp::d073weibull_p2_example_data_v1_t
p=c(1:9)/10
n0=10
```

```
q=qweibull_p2_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qweibull_p2_cp)",
 main="Weibull w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

`weibull_p2_cp` *weibull Distribution with a Predictor on the Scale Parameter, Predictions Based on a Calibrating Prior*

Description

The `fitdistcp` package contains functions that generate predictive distributions for various statistical models, with and without parameter uncertainty. Parameter uncertainty is included by using Bayesian prediction with a type of objective prior known as a calibrating prior. Calibrating priors are chosen to give predictions that give good reliability (i.e., are well calibrated), for any underlying true parameter values.

There are five functions for each model, each of which uses training data `x`. For model `****` the five functions are as follows:

- `q****_cp` returns predictive quantiles at the specified probabilities `p`, and various other diagnostics.
- `r****_cp` returns `n` random deviates from the predictive distribution.
- `d****_cp` returns the predictive density function at the specified values `y`
- `p****_cp` returns the predictive distribution function at the specified values `y`
- `t****_cp` returns `n` random deviates from the posterior distribution of the model parameters.

The `q, r, d, p` routines return two sets of results, one based on maximum likelihood, and the other based on a calibrating prior. The prior used depends on the model, and is given under Details below.

Where possible, the Bayesian prediction integral is solved analytically. Otherwise, DMGS asymptotic expansions are used. Optionally, a third set of results is returned that integrates the prediction integral by sampling the parameter posterior distribution using the RUST rejection sampling algorithm.

Usage

```
qweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  p = seq(0.1, 0.9, 0.1),
  means = FALSE,
```

```
  waicscores = FALSE,
  logscores = FALSE,
  dmgs = TRUE,
  rust = FALSE,
  nrust = 1e+05,
  predictordata = TRUE,
  centering = TRUE,
  debug = FALSE
)

rweibull_p2_cp(
  n,
  x,
  t,
  t0 = NA,
  n0 = NA,
  rust = FALSE,
  mlcp = TRUE,
  debug = FALSE
)

dweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

pweibull_p2_cp(
  x,
  t,
  t0 = NA,
  n0 = NA,
  y = x,
  rust = FALSE,
  nrust = 1000,
  centering = TRUE,
  debug = FALSE
)

tweibull_p2_cp(n, x, t, debug = FALSE)
```

Arguments

x	a vector of training data values
t	a vector of predictors, such that length(t)=length(x)
t0	a single value of the predictor (specify either t0 or n0 but not both)
n0	an index for the predictor (specify either t0 or n0 but not both)
p	a vector of probabilities at which to generate predictive quantiles
means	logical that indicates whether to run additional calculations and return analytical estimates for the distribution means (longer runtime)
waicscores	logical that indicates whether to run additional calculations and return estimates for the WAIC1 and WAIC2 scores (longer runtime)
logscores	logical that indicates whether to run additional calculations and return leave-one-out estimates of the log-score (much longer runtime, non-EVT models only)
dmgs	logical that indicates whether DMGS calculations should be run or not (longer run time)
rust	logical that indicates whether RUST-based posterior sampling calculations should be run or not (longer run time)
nrust	the number of posterior samples used in the RUST calculations
predictordata	logical that indicates whether predictordata should be calculated
centering	logical that indicates whether the predictor should be centered
debug	logical for turning on debug messages
n	the number of random samples required
mlcp	logical that indicates whether maxlik and parameter uncertainty calculations should be performed (turn off to speed up RUST)
y	a vector of values at which to calculate the density and distribution functions

Value

q**** returns a list containing at least the following:

- ml_params: maximum likelihood estimates for the parameters.
- ml_value: the value of the log-likelihood at the maximum.
- standard_errors: estimates of the standard errors on the parameters, from the inverse observed information matrix.
- ml_quantiles: quantiles calculated using maximum likelihood.
- cp_quantiles: predictive quantiles calculated using a calibrating prior.
- maic: the AIC score for the maximum likelihood model, times -1/2.
- cp_method: a comment about the method used to generate the cp prediction.

For models with predictors, q**** additionally returns:

- predictedparameter: the estimated value for parameter, as a function of the predictor.
- adjustedx: the detrended values of x

`r****` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_deviates:` random deviates calculated using maximum likelihood.
- `cp_deviates:` predictive random deviates calculated using a calibrating prior.
- `cp_method:` a comment about the method used to generate the `cp` prediction.

`d****` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_pdf:` density function from maximum likelihood.
- `cp_pdf:` predictive density function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method:` a comment about the method used to generate the `cp` prediction.

`p***` returns a list containing the following:

- `ml_params:` maximum likelihood estimates for the parameters.
- `ml_cdf:` distribution function from maximum likelihood.
- `cp_cdf:` predictive distribution function calculated using a calibrating prior (not available in EVT routines, for mathematical reasons, unless using RUST).
- `cp_method:` a comment about the method used to generate the `cp` prediction.

`t***` returns a list containing the following:

- `theta_samples:` random samples from the parameter posterior.

Details of the Model

The Weibull distribution with predictor on the scale parameter has exceedance distribution function

$$S(x; k, a, b) = \exp\left(-\left(\frac{x}{\sigma(a,b)}\right)^k\right)$$

where $x \geq 0$ is the random variable, $k > 0$ is the shape parameter and $\sigma = e^{a+bt}$ is the scale parameter, modelled as a function of parameters a, b and predictor t.

The calibrating prior is given by the right Haar prior, which is

$$\pi(k, \sigma) \propto \frac{1}{k}$$

as given in Jewson et al. (2025).

Optional Return Values

q**** optionally returns the following:

If rust=TRUE:

- ru_quantiles: predictive quantiles calculated using a calibrating prior, using posterior sampling with the RUST algorithm, based on inverting an empirical CDF based on nrust samples.

If waicscores=TRUE:

- waic1: the WAIC1 score for the calibrating prior model.
- waic2: the WAIC2 score for the calibrating prior model.

If logscores=TRUE:

- ml_oos_logscore: the leave-one-out logscore for the maximum likelihood prediction (not available in EVT routines, for mathematical reasons)
- cp_oos_logscore: the leave-one-out logscore for the parameter uncertainty model available in EVT routines, for mathematical reasons)

If means=TRUE:

- ml_mean: analytic estimate of the mean of the MLE predictive distribution, where possible
- cp_mean: analytic estimate of the mean of the calibrating prior predictive distribution, where mathematically possible. Can be compared with the mean estimated from random deviates.

r**** optionally returns the following:

If rust=TRUE:

- ru_deviates: nrust predictive random deviatives calculated using a calibrating prior, using posterior sampling with RUST.

d**** optionally returns the following:

If rust=TRUE:

- ru_pdf: predictive density calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust density functions.

p**** optionally returns the following:

If rust=TRUE:

- ru_cdf: predictive probability calculated using a calibrating prior, using posterior sampling with RUST, averaging over nrust distribution functions.

Selecting these additional outputs increases runtime. They are optional so that runtime for the basic outputs is minimised. This facilitates repeated experiments that evaluate reliability over many thousands of repeats.

Details (homogeneous models)

This model is a homogeneous model, and the cp results are based on the right Haar prior. For homogeneous models (models with sharply transitive transformation groups), a Bayesian prediction based on the right Haar prior gives exact reliability, as shown by Severini et al. (2002), even when the true parameters are unknown. This means that probabilities in the prediction will correspond to frequencies of future outcomes in repeated trials (if the model is correct).

Maximum likelihood prediction does not give reliable predictions, even when the model is correct, because it does not account for parameter uncertainty. In particular, maximum likelihood predictions typically underestimate the tail in repeated trials.

The reliability of the maximum likelihood and the calibrating prior predictive quantiles produced by the q****_cp routines in fitdistcp can be quantified using repeated simulations with the routine reltest.

Details (DMGS integration)

For this model, the Bayesian prediction equation cannot be solved analytically, and is approximated using the DMGS asymptotic expansions given by Datta et al. (2000). This approximation seems to work well for medium and large sample sizes, but may not work well for small sample sizes (e.g., <20 data points). For small sample sizes, it may be preferable to use posterior sampling by setting rust=TRUE and looking at the ru outputs. The performance for any sample size, in terms of reliability, can be tested using reltest.

Details (RUST)

The Bayesian prediction equation can also be integrated using ratio-of-uniforms-sampling-with-transformation (RUST), using the option rust=TRUE. fitdistcp then calls Paul Northrop's rust package (Northrop, 2023). The RUST calculations are slower than the DMGS calculations.

For small sample sizes (e.g., n<20), and the very extreme tail, the DMGS approximation is somewhat poor (although always better than maximum likelihood) and it may be better to use RUST. For medium sample sizes (30+), DMGS is reasonably accurate, except for the very far tail.

It is advisable to check the RUST results for convergence versus the number of RUST samples.

It may be interesting to compare the DMGS and RUST results.

Author(s)

Stephen Jewson <stephen.jewson@gmail.com>

References

If you use this package, we would be grateful if you would cite the following reference, which gives the various calibrating priors, and tests them for reliability:

- Jewson S., Sweeting T. and Jewson L. (2025): Reducing Reliability Bias in Assessments of Extreme Weather Risk using Calibrating Priors; ASCMO Advances in Statistical Climatology, Meteorology and Oceanography), https://ascmo.copernicus.org/articles/11/1/2025/.

See Also

An introduction to `fitdistcp`, with more examples, is given on this webpage.

The `fitdistcp` package currently includes the following models (in alphabetical order):

- Cauchy (`cauchy`),
- Cauchy with linear predictor on the mean (`cauchy_p1`),
- Exponential (`exp`),
- Exponential with log-linear predictor on the scale (`exp_p1`),
- Frechet with known location parameter (`frechet_k1`),
- Frechet with log-linear predictor on the scale and known location parameter (`frechet_p2k1`),
- Gamma (`gamma`),
- Generalized normal (`gnorm`),
- GEV (`gev`),
- GEV with linear predictor on the location (`gev_p1`),
- GEV with 1-3 linear predictors on the location (`gev_p1n`),
- GEV with linear predictor on the location and log-linear prediction on the scale (`gev_p12`),
- GEV with linear predictor on the location, log-linear prediction on the scale, and linear predictor on the shape (`gev_p123`),
- GEV with linear predictor on the location and known shape (`gev_p1k3`),
- GEV with known shape (`gev_k3`),
- GPD with known location (`gpd_k1`),
- Gumbel (`gumbel`),
- Gumbel with linear predictor on the mean(`gumbel_p1`),
- Half-normal (`halfnorm`),
- Inverse gamma (`invgamma`),
- Inverse Gaussian (`invgauss`),
- t distribution with unknown location and scale and known DoF (`lst_k3`),
- t distribution with unknown location and scale, linear predictor on the location, and known DoF (`lst_p1k3`),
- Logistic (`logis`),
- Logistic with linear predictor on the location (`logis_p1`),
- Log-normal (`lnorm`),
- Log-normal with linear predictor on the location (`lnorm_p1`),
- Normal (`norm`),
- Normal with predictor on the mean (`norm_p1`),
- Normal with predictors on the mean and sd (`norm_p12`),
- Pareto with known scale (`pareto_k2`),
- Pareto with log-linear predictor on the shape and known scale (`pareto_p1k2`),

- Uniform (unif),
- Weibull (weibull),
- Weibull with linear predictor on the scale (weibull_p2),

The level of predictive probability matching achieved by the maximum likelihood and calibrating prior quantiles, for any model, sample size and true parameter values, can be demonstrated using the routine reltest.

Model selection among models can be demonstrated using the routines ms_flat_1tail, ms_flat_2tail, ms_predictors_1tail, and ms_predictors_2tail,

Examples

```
#
# example 1
x=fitdistcp::d073weibull_p2_example_data_v1_x
tt=fitdistcp::d073weibull_p2_example_data_v1_t
p=c(1:9)/10
n0=10
q=qweibull_p2_cp(x,tt,n0=n0,p=p,rust=TRUE,nrust=1000)
xmin=min(q$ml_quantiles,q$cp_quantiles);
xmax=max(q$ml_quantiles,q$cp_quantiles);
plot(q$ml_quantiles,p,xlab="quantile estimates",xlim=c(xmin,xmax),
 sub="(from qweibull_p2_cp)",
 main="Weibull w/ p2: quantile estimates");
points(q$cp_quantiles,p,col="red",lwd=2)
points(q$ru_quantiles,p,col="blue")
```

---

weibull_p2_f1fa *The first derivative of the density for DMGS*

Description

The first derivative of the density for DMGS

Usage

```
weibull_p2_f1fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

weibull_p2_f1fw *The first derivative of the density for WAIC*

Description

The first derivative of the density for WAIC

Usage

```
weibull_p2_f1fw(x, t, v1, v2, v3)
```

Arguments

x	a vector of training data values
t	a vector or matrix of predictors
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

weibull_p2_f2fa *The second derivative of the density for DMGS*

Description

The second derivative of the density for DMGS

Usage

```
weibull_p2_f2fa(x, t0, v1, v2, v3)
```

Arguments

x	a vector of training data values
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Matrix

`weibull_p2_f2fw` *The second derivative of the density for WAIC*

Description

The second derivative of the density for WAIC

Usage

```
weibull_p2_f2fw(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_fd` *First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_fd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`weibull_p2_fdd` *Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_fdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_ldda` *The second derivative of the normalized log-likelihood*

Description

The second derivative of the normalized log-likelihood

Usage

```
weibull_p2_ldda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_lddda` *The third derivative of the normalized log-likelihood*

Description

The third derivative of the normalized log-likelihood

Usage

```
weibull_p2_lddda(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`weibull_p2_logf` *Logf for RUST*

Description

Logf for RUST

Usage

```
weibull_p2_logf(params, x, t)
```

Arguments

`params`	model parameters for calculating logf
`x`	a vector of training data values
`t`	a vector or matrix of predictors

Value

Scalar value.

`weibull_p2_logfdd` *Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_logfdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_logfddd` *Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Third derivative of the log density Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_logfddd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

3d array

`weibull_p2_loglik` *observed log-likelihood function*

Description

observed log-likelihood function

Usage

```
weibull_p2_loglik(vv, x, t)
```

Arguments

vv	parameters
x	a vector of training data values
t	a vector or matrix of predictors

Value

Scalar

`weibull_p2_logscores` *Log scores for MLE and RHP predictions calculated using leave-one-out*

Description

Log scores for MLE and RHP predictions calculated using leave-one-out

Usage

```
weibull_p2_logscores(logscores, x, t)
```

Arguments

logscores	logical that indicates whether to return leave-one-out estimates estimates of the log-score (much longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors

Value

Two scalars

weibull_p2_means *weibull distribution: RHP mean*

Description

weibull distribution: RHP mean

Usage

```
weibull_p2_means(means, t0, ml_params, lddi, lddd, lambdad_rhp, nx, dim)
```

Arguments

means	logical that indicates whether to return analytical estimates for the distribution means (longer runtime)
t0	a single value of the predictor (specify either t0 or n0 but not both)
ml_params	parameters
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad_rhp	derivative of the log RHP prior
nx	length of training data
dim	number of parameters

Value

Two scalars

weibull_p2_mu1fa *Minus the first derivative of the cdf, at alpha*

Description

Minus the first derivative of the cdf, at alpha

Usage

```
weibull_p2_mu1fa(alpha, t0, v1, v2, v3)
```

Arguments

alpha	a vector of values of alpha (one minus probability)
t0	a single value of the predictor (specify either t0 or n0 but not both)
v1	first parameter
v2	second parameter
v3	third parameter

Value

Vector

`weibull_p2_mu2fa` *Minus the second derivative of the cdf, at alpha*

Description

Minus the second derivative of the cdf, at alpha

Usage

```
weibull_p2_mu2fa(alpha, t0, v1, v2, v3)
```

Arguments

`alpha`	a vector of values of alpha (one minus probability)
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_p1fa` *The first derivative of the cdf*

Description

The first derivative of the cdf

Usage

```
weibull_p2_p1fa(x, t0, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`weibull_p2_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
weibull_p2_p2fa(x, t0, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t0`	a single value of the predictor (specify either `t0` or `n0` but not both)
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_pd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Vector

`weibull_p2_pdd` *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_p2_pdd(x, t, v1, v2, v3)
```

Arguments

`x`	a vector of training data values
`t`	a vector or matrix of predictors
`v1`	first parameter
`v2`	second parameter
`v3`	third parameter

Value

Matrix

`weibull_p2_predictordata` *Predicted Parameter and Generalized Residuals*

Description

Predicted Parameter and Generalized Residuals

Usage

```
weibull_p2_predictordata(predictordata, x, t, t0, params)
```

Arguments

predictordata	logical that indicates whether to calculate and return predictordata
x	a vector of training data values
t	a vector or matrix of predictors
t0	a single value of the predictor (specify either t0 or n0 but not both)
params	model parameters for calculating logf

Value

Two vectors

---

weibull_p2_waic *Waic*

Description

Waic

Usage

```
weibull_p2_waic(waicscores, x, t, v1hat, v2hat, v3hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
t	a vector or matrix of predictors
v1hat	first parameter
v2hat	second parameter
v3hat	third parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

`weibull_p2fa` *The second derivative of the cdf*

Description

The second derivative of the cdf

Usage

```
weibull_p2fa(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

`weibull_pd` *First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

First derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_pd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Vector

weibull_pdd *Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol*

Description

Second derivative of the cdf Created by Stephen Jewson using Deriv() by Andrew Clausen and Serguei Sokol

Usage

```
weibull_pdd(x, v1, v2)
```

Arguments

x	a vector of training data values
v1	first parameter
v2	second parameter

Value

Matrix

weibull_waic *Waic for RUST*

Description

Waic for RUST

Usage

```
weibull_waic(waicscores, x, v1hat, v2hat, lddi, lddd, lambdad)
```

Arguments

waicscores	logical that indicates whether to return estimates for the waic1 and waic2 scores (longer runtime)
x	a vector of training data values
v1hat	first parameter
v2hat	second parameter
lddi	inverse observed information matrix
lddd	third derivative of log-likelihood
lambdad	derivative of the log prior

Value

Two numeric values.

Index

adhoc_dmgs_cpmethod, 24
analytic_cpmethod, 24

bayesian_dq_4terms_v1, 25

calc_revert2ml, 25
cauchy_cp, 26
cauchy_f1f, 33
cauchy_f1fa, 33
cauchy_f2f, 34
cauchy_f2fa, 34
cauchy_fd, 35
cauchy_fdd, 35
cauchy_ldd, 36
cauchy_ldda, 36
cauchy_lddd, 37
cauchy_lddda, 37
cauchy_lmn, 38
cauchy_lmnp, 38
cauchy_logf, 39
cauchy_logfdd, 40
cauchy_logfddd, 40
cauchy_loglik, 41
cauchy_logscores, 41
cauchy_mu1f, 42
cauchy_mu2f, 42
cauchy_p1_cp, 43
cauchy_p1_f1f, 51
cauchy_p1_f1fa, 51
cauchy_p1_f1fw, 52
cauchy_p1_f2f, 52
cauchy_p1_f2fa, 53
cauchy_p1_f2fw, 54
cauchy_p1_fd, 54
cauchy_p1_fdd, 55
cauchy_p1_ldd, 55
cauchy_p1_ldda, 56
cauchy_p1_lddd, 57
cauchy_p1_lddda, 57
cauchy_p1_lmn, 58
cauchy_p1_lmnp, 59
cauchy_p1_logf, 59
cauchy_p1_logfdd, 60
cauchy_p1_logfddd, 61
cauchy_p1_loglik, 61
cauchy_p1_logscores, 62
cauchy_p1_means, 62
cauchy_p1_mu1f, 63
cauchy_p1_mu2f, 64
cauchy_p1_p1f, 64
cauchy_p1_p2f, 65
cauchy_p1_predictordata, 66
cauchy_p1_waic, 66
cauchy_p1f, 67
cauchy_p2f, 68
cauchy_waic, 68
crhpflat_dmgs_cpmethod, 69

d010exp_example_data_v1, 69
d011pareto_k2_example_data_v1, 70
d020halfnorm_example_data_v1, 70
d025unif_example_data_v1, 70
d030norm_example_data_v1, 70
d031norm_dmgs_example_data_v1, 70
d032gnorm_k3_example_data_v1, 71
d035lnorm_example_data_v1, 71
d036lnorm_dmgs_example_data_v1, 71
d040logis_example_data_v1, 71
d041lst_k3_example_data_v1, 71
d042cauchy_example_data_v1, 72
d050gumbel_example_data_v1, 72
d051frechet_k1_example_data_v1, 72
d052weibull_example_data_v1, 72
d053gev_k3_example_data_v1, 72
d055exp_p1_example_data_v1_t, 73
d055exp_p1_example_data_v1_x, 73
d056pareto_p1k2_example_data_v1_t, 73
d056pareto_p1k2_example_data_v1_x, 73
d060norm_p1_example_data_v1_t, 73
d060norm_p1_example_data_v1_x, 74

d061lnorm_p1_example_data_v1_t, 74
d061lnorm_p1_example_data_v1_x, 74
d062logis_p1_example_data_v1_t, 74
d062logis_p1_example_data_v1_x, 74
d063lst_p1k3_example_data_v1_t, 75
d063lst_p1k3_example_data_v1_x, 75
d064cauchy_p1_example_data_v1_t, 75
d064cauchy_p1_example_data_v1_x, 75
d070gumbel_p1_example_data_v1_t, 75
d070gumbel_p1_example_data_v1_x, 76
d071frechet_p2k1_example_data_v1_t, 76
d071frechet_p2k1_example_data_v1_x, 76
d072weibull_p1_example_data_v1_t, 76
d072weibull_p1_example_data_v1_x, 76
d073weibull_p2_example_data_v1_t, 77
d073weibull_p2_example_data_v1_x, 77
d074gev_p1k3_example_data_v1_t, 77
d074gev_p1k3_example_data_v1_x, 77
d080norm_p12_example_data_v1_t1, 77
d080norm_p12_example_data_v1_t2, 78
d080norm_p12_example_data_v1_x, 78
d081lst_p12k3_example_data_v1_t1, 78
d081lst_p12k3_example_data_v1_t2, 78
d081lst_p12k3_example_data_v1_x, 78
d082weibull_p12_example_data_v1_t1, 79
d082weibull_p12_example_data_v1_t2, 79
d082weibull_p12_example_data_v1_x, 79
d100gamma_example_data_v1, 79
d101invgamma_example_data_v1, 79
d102invgauss_example_data_v1, 80
d105burr_example_data_v1, 80
d110gev_example_data_v1, 80
d120gpd_k1_example_data_v1, 80
d150gev_p1_example_data_v1_t, 80
d150gev_p1_example_data_v1_x, 81
d151gev_p12_example_data_v1_t, 81
d151gev_p12_example_data_v1_x, 81
d152gev_p123_example_data_v1_t, 81
d152gev_p123_example_data_v1_x, 81
dcauchy_cp *(*cauchy_cp*)*, 26
dcauchy_p1, 82
dcauchy_p1_cp *(*cauchy_p1_cp*)*, 43
dcauchy_p1sub, 82
dcauchysub, 83
deriv_copyfdd, 83
deriv_copyld2, 84
deriv_copyldd, 84
deriv_copylddd, 85
dexp_cp *(*exp_cp*)*, 114
dexp_p1, 85
dexp_p1_cp *(*exp_p1_cp*)*, 125
dexp_p1sub, 86
dexpsub, 86
dfrechet_k1_cp *(*frechet_k1_cp*)*, 148
dfrechet_p2k1, 87
dfrechet_p2k1_cp *(*frechet_p2k1_cp*)*, 164
dfrechet_p2k1sub, 87
dfrechetsub, 88
dgamma_cp *(*gamma_cp*)*, 185
dgammasub, 88
dgev_cp *(*gev_cp*)*, 207
dgev_k3_cp *(*gev_k3_cp*)*, 220
dgev_k3sub, 89
dgev_p1, 89
dgev_p12, 90
dgev_p123, 90
dgev_p123_cp *(*gev_p123_cp*)*, 286, 294
dgev_p123sub, 91
dgev_p12_cp *(*gev_p12_cp*)*, 255, 263
dgev_p12sub, 92
dgev_p1_cp *(*gev_p1_cp*)*, 242
dgev_p1k3, 93
dgev_p1k3_cp *(*gev_p1k3_cp*)*, 354
dgev_p1k3sub, 93
dgev_p1n, 94
dgev_p1n_cp *(*gev_p1n_cp*)*, 373, 382
dgev_p1nsub, 94
dgev_p1sub, 95
dgevsub, 95
dgnorm_k3_cp *(*gnorm_k3_cp*)*, 398
dgnorm_k3sub, 96
dgpd_k1_cp *(*gpd_k1_cp*)*, 418
dgpdsub, 97
dgumbel_cp *(*gumbel_cp*)*, 442
dgumbel_p1, 97
dgumbel_p1_cp *(*gumbel_p1_cp*)*, 455
dgumbel_p1sub, 98
dgumbelsub, 98
dhalfnorm_cp *(*halfnorm_cp*)*, 477
dhalfnormsub, 99
dinvgamma_cp *(*invgamma_cp*)*, 497
dinvgammasub, 99
dinvgauss_cp *(*invgauss_cp*)*, 516
dinvgausssub, 100
dlnorm_cp *(*lnorm_cp*)*, 538
dlnorm_dmgs_cp *(*lnorm_dmgs_cp*)*, 544

dlnorm_dmgssub, 100
dlnorm_p1, 101
dlnorm_p1_cp *(*lnorm_p1_cp*)*, 558
dlnorm_p1sub, 101
dlnormsub, 102
dlogis2sub, 103
dlogis_cp *(*logis_cp*)*, 578
dlogis_p1, 102
dlogis_p1_cp *(*logis_p1_cp*)*, 591
dlogis_p1sub, 103
dlst_k3_cp *(*lst_k3_cp*)*, 612
dlst_k3sub, 104
dlst_p1k3, 104
dlst_p1k3_cp *(*lst_p1k3_cp*)*, 632
dlst_p1k3sub, 105
dmgs, 105
dnorm_cp *(*norm_cp*)*, 691
dnorm_dmgs_cp *(*norm_dmgs_cp*)*, 697
dnorm_dmgssub, 106
dnorm_p1, 107
dnorm_p12, 108
dnorm_p12_cp *(*norm_p12_cp*)*, 733, 741
dnorm_p12dmgs, 108
dnorm_p1_cp *(*norm_p1_cp*)*, 712
dnorm_p1_formula, 107
dnorm_p1sub, 109
dnormsub, 110
dpareto_k2_cp *(*pareto_k2_cp*)*, 766
dpareto_k2_sub, 110
dpareto_p1k2, 111
dpareto_p1k2_cp *(*pareto_p1k2_cp*)*, 782
dpareto_p1k2sub, 111
dunif_cp *(*unif_cp*)*, 859, 865
dunif_formula, 112
dweibull_cp *(*weibull_cp*)*, 871, 877
dweibull_p2, 112
dweibull_p2_cp *(*weibull_p2_cp*)*, 891, 898
dweibull_p2sub, 113
dweibullsub, 113

exp_cp, 114
exp_f1fa, 120
exp_f2fa, 120
exp_fd, 121
exp_fdd, 121
exp_ldda, 122
exp_lddda, 123
exp_logf, 123
exp_logfdd, 124
exp_logfddd, 124
exp_logscores, 125
exp_p1_cp, 125
exp_p1_f1fa, 132
exp_p1_f1fw, 133
exp_p1_f2fa, 134
exp_p1_f2fw, 134
exp_p1_fd, 135
exp_p1_fdd, 135
exp_p1_ldda, 136
exp_p1_lddda, 136
exp_p1_logf, 137
exp_p1_logfdd, 137
exp_p1_logfddd, 138
exp_p1_loglik, 138
exp_p1_logscores, 139
exp_p1_means, 139
exp_p1_mu1fa, 140
exp_p1_mu2fa, 140
exp_p1_p1fa, 141
exp_p1_p2fa, 141
exp_p1_pd, 142
exp_p1_pdd, 142
exp_p1_predictordata, 143
exp_p1_waic, 143
exp_p1fa, 144
exp_p2fa, 144
exp_pd, 145
exp_pdd, 145
exp_waic, 146

findnt, 147
fixgevrange, 147
fixgpdrange, 148
frechet_k1_cp, 148
frechet_k1_f1fa, 154
frechet_k1_f2fa, 155
frechet_k1_fd, 156
frechet_k1_fdd, 156
frechet_k1_ldda, 157
frechet_k1_lddda, 157
frechet_k1_logf, 158
frechet_k1_logfdd, 158
frechet_k1_logfddd, 159
frechet_k1_mu1fa, 159
frechet_k1_mu2fa, 160
frechet_k1_p1fa, 160
frechet_k1_p2fa, 161
frechet_k1_pd, 161

frechet_k1_pdd, 162
frechet_k1_waic, 162
frechet_loglik, 163
frechet_logscores, 163
frechet_means, 164
frechet_p2k1_cp, 164
frechet_p2k1_f1fa, 172
frechet_p2k1_f1fw, 172
frechet_p2k1_f2fa, 173
frechet_p2k1_f2fw, 173
frechet_p2k1_fd, 174
frechet_p2k1_fdd, 174
frechet_p2k1_ldda, 175
frechet_p2k1_lddda, 176
frechet_p2k1_logf, 176
frechet_p2k1_logfdd, 177
frechet_p2k1_logfddd, 177
frechet_p2k1_loglik, 178
frechet_p2k1_logscores, 179
frechet_p2k1_means, 179
frechet_p2k1_mu1fa, 180
frechet_p2k1_mu2fa, 181
frechet_p2k1_p1fa, 181
frechet_p2k1_p2fa, 182
frechet_p2k1_pd, 182
frechet_p2k1_pdd, 183
frechet_p2k1_predictordata, 184
frechet_p2k1_waic, 184

gamma_cp, 185
gamma_f1f, 192
gamma_f1fa, 193
gamma_f2f, 193
gamma_f2fa, 194
gamma_fd, 194
gamma_fdd, 195
gamma_gg, 195
gamma_gmn, 196
gamma_ldd, 196
gamma_ldda, 197
gamma_lddd, 198
gamma_lddda, 198
gamma_lmn, 199
gamma_lmnp, 199
gamma_logf, 200
gamma_logfdd, 201
gamma_logfddd, 201
gamma_loglik, 202
gamma_logscores, 202
gamma_means, 203
gamma_mu1f, 203
gamma_mu2f, 204
gamma_p1f, 204
gamma_p2f, 205
gamma_waic, 206
gev_boot, 206
gev_checkmle, 207
gev_cp, 207
gev_f1fa, 216
gev_f2fa, 217
gev_fd, 217
gev_fdd, 218
gev_k12_ppm_minusloglik, 218
gev_k3_cp, 220
gev_k3_f1fa, 226
gev_k3_f2fa, 227
gev_k3_fd, 228
gev_k3_fdd, 228
gev_k3_ldda, 229
gev_k3_lddda, 229
gev_k3_logf, 230
gev_k3_logfdd, 230
gev_k3_logfddd, 231
gev_k3_loglik, 231
gev_k3_means, 232
gev_k3_mu1fa, 232
gev_k3_mu2fa, 233
gev_k3_pd, 233
gev_k3_pdd, 234
gev_k3_waic, 234
gev_ld12a, 235
gev_lda, 236
gev_ldda, 236
gev_lddda, 237
gev_logf, 237
gev_logfd, 238
gev_logfdd, 238
gev_logfddd, 239
gev_loglik, 239
gev_means, 240
gev_mu1fa, 241
gev_mu2fa, 241
gev_p123_checkmle, 285
gev_p123_cp, 286, 294
gev_p123_f1fa, 303
gev_p123_f1fw, 304
gev_p123_f2fa, 305

gev_p123_f2fw, 305
gev_p123_fd, 306
gev_p123_fdd, 307
gev_p123_ldda, 307
gev_p123_lddda, 308
gev_p123_logf, 309
gev_p123_logfdd, 309
gev_p123_logfddd, 310
gev_p123_loglik, 311
gev_p123_means, 311
gev_p123_mu1fa, 312
gev_p123_mu2fa, 313
gev_p123_pd, 313
gev_p123_pdd, 314
gev_p123_predictordata, 315
gev_p123_setics, 315
gev_p123_waic, 316
gev_p12_checkmle, 254
gev_p12_cp, 255, 263
gev_p12_f1fa, 272
gev_p12_f1fw, 272
gev_p12_f2fa, 273
gev_p12_f2fw, 274
gev_p12_fd, 274
gev_p12_fdd, 275
gev_p12_ldda, 276
gev_p12_lddda, 276
gev_p12_logf, 277
gev_p12_logfdd, 278
gev_p12_logfddd, 278
gev_p12_loglik, 279
gev_p12_means, 280
gev_p12_mu1fa, 280
gev_p12_mu2fa, 281
gev_p12_pd, 282
gev_p12_pdd, 282
gev_p12_predictordata, 283
gev_p12_setics, 284
gev_p12_waic, 284
gev_p12k3_f1fa, 317
gev_p12k3_f1fw, 317
gev_p12k3_f2fa, 318
gev_p12k3_f2fw, 319
gev_p12k3_fd, 319
gev_p12k3_fdd, 320
gev_p12k3_ldda, 321
gev_p12k3_lddda, 321
gev_p12k3_logfdd, 322
gev_p12k3_logfddd, 323
gev_p12k3_mu1fa, 323
gev_p12k3_mu2fa, 324
gev_p12k3_pd, 325
gev_p12k3_pdd, 325
gev_p1_checkmle, 242
gev_p1_cp, 242
gev_p1_logf, 251
gev_p1_loglik, 252
gev_p1_means, 252
gev_p1_predictordata, 253
gev_p1_setics, 253
gev_p1_waic, 254
gev_p1a_f1fa, 326
gev_p1a_f1fw, 327
gev_p1a_f2fa, 327
gev_p1a_f2fw, 328
gev_p1a_fd, 328
gev_p1a_fdd, 329
gev_p1a_ldda, 330
gev_p1a_lddda, 330
gev_p1a_logfdd, 331
gev_p1a_logfddd, 331
gev_p1a_mu1fa, 332
gev_p1a_mu2fa, 333
gev_p1a_pd, 333
gev_p1a_pdd, 334
gev_p1b_f1fa, 334
gev_p1b_f1fw, 335
gev_p1b_f2fa, 336
gev_p1b_f2fw, 336
gev_p1b_fd, 337
gev_p1b_fdd, 338
gev_p1b_ldda, 338
gev_p1b_lddda, 339
gev_p1b_logfdd, 340
gev_p1b_logfddd, 340
gev_p1b_mu1fa, 341
gev_p1b_mu2fa, 342
gev_p1b_pd, 342
gev_p1b_pdd, 343
gev_p1c_f1fa, 344
gev_p1c_f1fw, 344
gev_p1c_f2fa, 345
gev_p1c_f2fw, 346
gev_p1c_fd, 347
gev_p1c_fdd, 347
gev_p1c_ldda, 348

gev_p1c_lddda, 349
gev_p1c_logfdd, 349
gev_p1c_logfddd, 350
gev_p1c_mu1fa, 351
gev_p1c_mu2fa, 351
gev_p1c_pd, 352
gev_p1c_pdd, 353
gev_p1k3_cp, 354
gev_p1k3_f1fa, 361
gev_p1k3_f1fw, 362
gev_p1k3_f2fa, 362
gev_p1k3_f2fw, 363
gev_p1k3_fd, 363
gev_p1k3_fdd, 364
gev_p1k3_ldda, 365
gev_p1k3_lddda, 365
gev_p1k3_logf, 366
gev_p1k3_logfdd, 366
gev_p1k3_logfddd, 367
gev_p1k3_loglik, 367
gev_p1k3_means, 368
gev_p1k3_mu1fa, 369
gev_p1k3_mu2fa, 369
gev_p1k3_pd, 370
gev_p1k3_pdd, 370
gev_p1k3_predictordata, 371
gev_p1k3_waic, 372
gev_p1n_checkmle, 373
gev_p1n_cp, 373, 382
gev_p1n_logf, 391
gev_p1n_loglik, 391
gev_p1n_means, 392
gev_p1n_n1_exampledata, 393
gev_p1n_n2_exampledata, 393
gev_p1n_predictordata, 394
gev_p1n_setics, 394
gev_p1n_waic, 395
gev_pd, 395
gev_pdd, 396
gev_pwm_params, 396
gev_setics, 397
gev_waic, 397
gnorm_k3_cp, 398
gnorm_k3_f1f, 405
gnorm_k3_f1fa, 406
gnorm_k3_f2f, 406
gnorm_k3_f2fa, 407
gnorm_k3_fd, 407
gnorm_k3_fdd, 408
gnorm_k3_ldd, 408
gnorm_k3_ldda, 409
gnorm_k3_lddd, 409
gnorm_k3_lddda, 410
gnorm_k3_lmn, 410
gnorm_k3_logf, 411
gnorm_k3_logfdd, 412
gnorm_k3_logfddd, 412
gnorm_k3_loglik, 413
gnorm_k3_logscores, 413
gnorm_k3_mu1f, 414
gnorm_k3_mu2f, 414
gnorm_k3_p1f, 415
gnorm_k3_p2f, 415
gnorm_lmnp, 416
gnorm_waic, 417
gpd_k13_f1fa, 436
gpd_k13_f2fa, 436
gpd_k13_fd, 437
gpd_k13_fdd, 437
gpd_k13_ldda, 438
gpd_k13_lddda, 438
gpd_k13_logfdd, 439
gpd_k13_logfddd, 439
gpd_k13_mu1fa, 440
gpd_k13_mu2fa, 440
gpd_k13_pd, 441
gpd_k13_pdd, 441
gpd_k1_checkmle, 418
gpd_k1_cp, 418
gpd_k1_f1fa, 426
gpd_k1_f2fa, 427
gpd_k1_fd, 428
gpd_k1_fdd, 428
gpd_k1_ldda, 429
gpd_k1_lddda, 429
gpd_k1_logf, 430
gpd_k1_logfdd, 430
gpd_k1_logfddd, 431
gpd_k1_loglik, 431
gpd_k1_means, 432
gpd_k1_mu1fa, 433
gpd_k1_mu2fa, 433
gpd_k1_pd, 434
gpd_k1_pdd, 434
gpd_k1_setics, 435
gpd_k1_waic, 435

gumbel_cp, 442
gumbel_f1fa, 448
gumbel_f2fa, 448
gumbel_fd, 449
gumbel_fdd, 449
gumbel_ldda, 450
gumbel_lddda, 450
gumbel_logf, 451
gumbel_logfdd, 451
gumbel_logfddd, 452
gumbel_loglik, 452
gumbel_logscores, 453
gumbel_means, 453
gumbel_mu1fa, 454
gumbel_mu2fa, 455
gumbel_p1_cp, 455
gumbel_p1_f1fa, 462
gumbel_p1_f1fw, 463
gumbel_p1_f2fa, 464
gumbel_p1_f2fw, 464
gumbel_p1_fd, 465
gumbel_p1_fdd, 465
gumbel_p1_ldda, 466
gumbel_p1_lddda, 466
gumbel_p1_logf, 467
gumbel_p1_logfdd, 467
gumbel_p1_logfddd, 468
gumbel_p1_loglik, 468
gumbel_p1_logscores, 469
gumbel_p1_means, 469
gumbel_p1_mu1fa, 470
gumbel_p1_mu2fa, 471
gumbel_p1_p1fa, 471
gumbel_p1_p2fa, 472
gumbel_p1_pd, 472
gumbel_p1_pdd, 473
gumbel_p1_predictordata, 473
gumbel_p1_waic, 474
gumbel_p1fa, 475
gumbel_p2fa, 475
gumbel_pd, 476
gumbel_pdd, 476
gumbel_waic, 477

halfnorm_cp, 477
halfnorm_f1f, 484
halfnorm_f1fa, 485
halfnorm_f2f, 485
halfnorm_f2fa, 486
halfnorm_fd, 486
halfnorm_fdd, 487
halfnorm_gg, 487
halfnorm_gg11, 488
halfnorm_l111, 488
halfnorm_ldd, 489
halfnorm_ldda, 489
halfnorm_lddd, 490
halfnorm_lddda, 490
halfnorm_logf, 491
halfnorm_logfdd, 491
halfnorm_logfddd, 492
halfnorm_loglik, 493
halfnorm_logscores, 493
halfnorm_means, 494
halfnorm_mu1f, 494
halfnorm_mu2f, 495
halfnorm_p1f, 495
halfnorm_p2f, 496
halfnorm_waic, 496

ifvectorthenmatrix, 497
invgamma_cp, 497
invgamma_f1f, 504
invgamma_f1fa, 505
invgamma_f2f, 505
invgamma_f2fa, 506
invgamma_fd, 506
invgamma_fdd, 507
invgamma_ldd, 507
invgamma_ldda, 508
invgamma_lddd, 508
invgamma_lddda, 509
invgamma_lmn, 509
invgamma_lmnp, 510
invgamma_logf, 510
invgamma_logfdd, 511
invgamma_logfddd, 511
invgamma_loglik, 512
invgamma_logscores, 512
invgamma_mu1f, 513
invgamma_mu2f, 514
invgamma_p1f, 514
invgamma_p2f, 515
invgamma_waic, 515
invgauss_cp, 516
invgauss_f1f, 524
invgauss_f1fa, 524
invgauss_f2f, 525

invgauss_f2fa, 525
invgauss_fd, 526
invgauss_fdd, 526
invgauss_ldd, 527
invgauss_ldda, 527
invgauss_lddd, 528
invgauss_lddda, 528
invgauss_lmn, 529
invgauss_lmnp, 529
invgauss_logf, 530
invgauss_logfdd, 531
invgauss_logfddd, 531
invgauss_loglik, 532
invgauss_logscores, 532
invgauss_means, 533
invgauss_mu1f, 533
invgauss_mu2f, 534
invgauss_p1f, 534
invgauss_p2f, 535
invgauss_waic, 536

jpf2p, 537
jpf3p, 537
jpf4p, 538

lnorm_cp, 538
lnorm_dmgs_cp, 544
lnorm_dmgs_loglik, 550
lnorm_dmgs_logscores, 551
lnorm_dmgs_means, 551
lnorm_dmgs_waic, 552
lnorm_f1fa, 552
lnorm_f2fa, 553
lnorm_fd, 553
lnorm_fdd, 554
lnorm_ldda, 554
lnorm_lddda, 555
lnorm_logf, 555
lnorm_logfdd, 556
lnorm_logfddd, 556
lnorm_logscores, 557
lnorm_mu1fa, 557
lnorm_mu2fa, 558
lnorm_p1_cp, 558
lnorm_p1_f1fa, 565
lnorm_p1_f1fw, 566
lnorm_p1_f2fa, 566
lnorm_p1_f2fw, 567
lnorm_p1_fd, 567
lnorm_p1_fdd, 568
lnorm_p1_ldda, 568
lnorm_p1_lddda, 569
lnorm_p1_logf, 569
lnorm_p1_logfdd, 570
lnorm_p1_logfddd, 570
lnorm_p1_loglik, 571
lnorm_p1_logscores, 571
lnorm_p1_mu1fa, 572
lnorm_p1_mu2fa, 572
lnorm_p1_p1fa, 573
lnorm_p1_p2fa, 573
lnorm_p1_pd, 574
lnorm_p1_pdd, 574
lnorm_p1_predictordata, 575
lnorm_p1_waic, 575
lnorm_p1fa, 576
lnorm_p2fa, 576
lnorm_pd, 577
lnorm_pdd, 577
lnorm_waic, 578
logis_cp, 578
logis_f1fa, 584
logis_f2fa, 585
logis_fd, 585
logis_fdd, 586
logis_ldda, 586
logis_lddda, 587
logis_logf, 587
logis_logfdd, 588
logis_logfddd, 588
logis_loglik, 589
logis_logscores, 589
logis_mu1fa, 590
logis_mu2fa, 590
logis_p1_cp, 591
logis_p1_f1fa, 598
logis_p1_f1fw, 599
logis_p1_f2fa, 599
logis_p1_f2fw, 600
logis_p1_fd, 600
logis_p1_fdd, 601
logis_p1_ldda, 601
logis_p1_lddda, 602
logis_p1_logf, 602
logis_p1_logfdd, 603
logis_p1_logfddd, 603
logis_p1_loglik, 604

logis_p1_logscores, 604
logis_p1_means, 605
logis_p1_mu1fa, 605
logis_p1_mu2fa, 606
logis_p1_p1fa, 606
logis_p1_p2fa, 607
logis_p1_pd, 607
logis_p1_pdd, 608
logis_p1_predictordata, 608
logis_p1_waic, 609
logis_p1fa, 610
logis_p2fa, 610
logis_pd, 611
logis_pdd, 611
logis_waic, 612
lst_k3_cp, 612
lst_k3_f1f, 619
lst_k3_f1fa, 620
lst_k3_f2f, 621
lst_k3_f2fa, 621
lst_k3_fd, 622
lst_k3_fdd, 622
lst_k3_ldd, 623
lst_k3_ldda, 623
lst_k3_lddd, 624
lst_k3_lddda, 624
lst_k3_lmn, 625
lst_k3_lmnp, 625
lst_k3_logf, 626
lst_k3_logfdd, 627
lst_k3_logfddd, 627
lst_k3_loglik, 628
lst_k3_logscores, 628
lst_k3_mu1f, 629
lst_k3_mu2f, 629
lst_k3_p1f, 630
lst_k3_p2f, 630
lst_k3_waic, 631
lst_p1k3_cp, 632
lst_p1k3_f1f, 640
lst_p1k3_f1fa, 641
lst_p1k3_f1fw, 642
lst_p1k3_f2f, 642
lst_p1k3_f2fa, 643
lst_p1k3_f2fw, 644
lst_p1k3_fd, 644
lst_p1k3_fdd, 645
lst_p1k3_ldd, 645
lst_p1k3_ldda, 646
lst_p1k3_lddd, 647
lst_p1k3_lddda, 647
lst_p1k3_lmn, 648
lst_p1k3_lmnp, 649
lst_p1k3_logf, 649
lst_p1k3_logfdd, 650
lst_p1k3_logfddd, 651
lst_p1k3_loglik, 651
lst_p1k3_logscores, 652
lst_p1k3_mu1f, 652
lst_p1k3_mu2f, 653
lst_p1k3_p1f, 654
lst_p1k3_p2f, 654
lst_p1k3_predictordata, 655
lst_p1k3_setics, 656
lst_p1k3_waic, 656

make_cwaic, 657
make_maic, 658
make_se, 658
make_waic, 659
makebetat0, 659
makebetatm, 660
makemuhat0, 660
makeq, 661
maket0, 661
maketresid0, 662
man, 662
man1f, 672
man2f, 673
manboot, 673
mancheckmle, 673
mandsub, 674
manf, 674
manldd, 681
manlddd, 681
manlnn, 682
manlnnn, 682
manlogf, 682
manloglik, 683
manlogscores, 683
manmeans, 683
manpredictor, 684
manvector, 684
manwaic, 684
movexiawayfromzero, 685
ms_flat_1tail, 685
ms_flat_2tail, 687

ms_predictors_1tail, 688
ms_predictors_2tail, 689

nopdfcdfmsg, 690
norm_boot, 691
norm_cp, 691
norm_dmgs_cp, 697
norm_dmgs_loglik, 703
norm_dmgs_logscores, 703
norm_dmgs_means, 704
norm_dmgs_waic, 705
norm_f1fa, 705
norm_f2fa, 706
norm_fd, 706
norm_fdd, 707
norm_ldda, 707
norm_lddda, 708
norm_logf, 708
norm_logfdd, 709
norm_logfddd, 709
norm_logscores, 710
norm_ml_params, 710
norm_mu1fa, 711
norm_mu2fa, 711
norm_p12_boot, 732
norm_p12_checkmle, 733
norm_p12_cp, 733, 741
norm_p12_exampledata, 749
norm_p12_f1fa, 749
norm_p12_f1fw, 750
norm_p12_f2fa, 750
norm_p12_f2fw, 751
norm_p12_fd, 751
norm_p12_fdd, 752
norm_p12_ldda, 753
norm_p12_lddda, 753
norm_p12_logf, 754
norm_p12_logfdd, 754
norm_p12_logfddd, 755
norm_p12_loglik, 756
norm_p12_logscores, 756
norm_p12_mu1fa, 757
norm_p12_mu2fa, 757
norm_p12_p1fa, 758
norm_p12_p2fa, 758
norm_p12_pd, 759
norm_p12_pdd, 760
norm_p12_predictordata, 760
norm_p12_setics, 761
norm_p12_waic, 762
norm_p1_cp, 712
norm_p1_f1fa, 719
norm_p1_f1fw, 719
norm_p1_f2fa, 720
norm_p1_f2fw, 720
norm_p1_fd, 721
norm_p1_fdd, 722
norm_p1_ldda, 722
norm_p1_lddda, 723
norm_p1_logf, 724
norm_p1_logfdd, 724
norm_p1_logfddd, 725
norm_p1_loglik, 725
norm_p1_logscores, 726
norm_p1_mlparams, 726
norm_p1_mu1fa, 727
norm_p1_mu2fa, 728
norm_p1_p1fa, 728
norm_p1_p2fa, 729
norm_p1_pd, 730
norm_p1_pdd, 730
norm_p1_predictordata, 731
norm_p1_waic, 732
norm_p1fa, 763
norm_p2fa, 763
norm_pd, 764
norm_pdd, 764
norm_unbiasedv_params, 765
norm_waic, 765

pareto_k2_cp, 766
pareto_k2_f1fa, 772
pareto_k2_f2fa, 772
pareto_k2_fd, 773
pareto_k2_fdd, 773
pareto_k2_ldda, 774
pareto_k2_lddda, 775
pareto_k2_logf, 775
pareto_k2_logfdd, 776
pareto_k2_logfddd, 776
pareto_k2_logscores, 777
pareto_k2_ml_params, 778
pareto_k2_mu1fa, 778
pareto_k2_mu2fa, 779
pareto_k2_p1fa, 779
pareto_k2_p2fa, 780
pareto_k2_pd, 780
pareto_k2_pdd, 781

pareto_k2_waic, 782
pareto_p1k2_cp, 782
pareto_p1k2_f1fa, 790
pareto_p1k2_f1fw, 790
pareto_p1k2_f2fa, 791
pareto_p1k2_f2fw, 791
pareto_p1k2_fd, 792
pareto_p1k2_fdd, 792
pareto_p1k2_ldda, 793
pareto_p1k2_lddda, 793
pareto_p1k2_logf, 794
pareto_p1k2_logfdd, 794
pareto_p1k2_logfddd, 795
pareto_p1k2_loglik, 795
pareto_p1k2_logscores, 796
pareto_p1k2_means, 796
pareto_p1k2_mu1fa, 797
pareto_p1k2_mu2fa, 798
pareto_p1k2_p1fa, 798
pareto_p1k2_p2fa, 799
pareto_p1k2_pd, 799
pareto_p1k2_pdd, 800
pareto_p1k2_predictordata, 800
pareto_p1k2_waic, 801
pcauchy_cp (cauchy_cp), 26
pcauchy_p1, 802
pcauchy_p1_cp (cauchy_p1_cp), 43
pexp_cp (exp_cp), 114
pexp_p1, 802
pexp_p1_cp (exp_p1_cp), 125
pfrechet_k1_cp (frechet_k1_cp), 148
pfrechet_p2k1, 803
pfrechet_p2k1_cp (frechet_p2k1_cp), 164
pgamma_cp (gamma_cp), 185
pgev_cp (gev_cp), 207
pgev_k3_cp (gev_k3_cp), 220
pgev_p1, 803
pgev_p12, 804
pgev_p123, 804
pgev_p123_cp (gev_p123_cp), 286, 294
pgev_p12_cp (gev_p12_cp), 255, 263
pgev_p1_cp (gev_p1_cp), 242
pgev_p1k3, 805
pgev_p1k3_cp (gev_p1k3_cp), 354
pgev_p1n, 806
pgev_p1n_cp (gev_p1n_cp), 373, 382
pgnorm_k3_cp (gnorm_k3_cp), 398
pgpd_k1_cp (gpd_k1_cp), 418
pgumbel_cp (gumbel_cp), 442
pgumbel_p1, 806
pgumbel_p1_cp (gumbel_p1_cp), 455
phalfnorm_cp (halfnorm_cp), 477
pinvgamma_cp (invgamma_cp), 497
pinvgauss_cp (invgauss_cp), 516
plnorm_cp (lnorm_cp), 538
plnorm_dmgs_cp (lnorm_dmgs_cp), 544
plnorm_p1, 807
plnorm_p1_cp (lnorm_p1_cp), 558
plogis_cp (logis_cp), 578
plogis_p1, 807
plogis_p1_cp (logis_p1_cp), 591
plst_k3_cp (lst_k3_cp), 612
plst_p1k3, 808
plst_p1k3_cp (lst_p1k3_cp), 632
pnorm_cp (norm_cp), 691
pnorm_dmgs_cp (norm_dmgs_cp), 697
pnorm_p1, 808
pnorm_p12, 809
pnorm_p12_cp (norm_p12_cp), 733, 741
pnorm_p1_cp (norm_p1_cp), 712
pnorm_p1_formula, 809
ppareto_k2_cp (pareto_k2_cp), 766
ppareto_p1k2, 810
ppareto_p1k2_cp (pareto_p1k2_cp), 782
punif_cp (unif_cp), 859, 865
punif_formula, 810
pweibull_cp (weibull_cp), 871, 877
pweibull_p2, 811
pweibull_p2_cp (weibull_p2_cp), 891, 898

qcauchy_cp (cauchy_cp), 26
qcauchy_p1, 811
qcauchy_p1_cp (cauchy_p1_cp), 43
qexp_cp (exp_cp), 114
qexp_p1, 812
qexp_p1_cp (exp_p1_cp), 125
qfrechet_k1_cp (frechet_k1_cp), 148
qfrechet_p2k1, 812
qfrechet_p2k1_cp (frechet_p2k1_cp), 164
qgamma_cp (gamma_cp), 185
qgamma_k1_ppm, 813
qgamma_ppm, 814
qgev_cp (gev_cp), 207
qgev_k12_ppm, 815
qgev_k3_cp (gev_k3_cp), 220
qgev_mpd_ppm, 817
qgev_p1, 818

qgev_p12, 820
qgev_p123, 821
qgev_p123_cp *(gev_p123_cp)*, 286, 294
qgev_p12_cp *(gev_p12_cp)*, 255, 263
qgev_p1_cp *(gev_p1_cp)*, 242
qgev_p1_ppm, 819
qgev_p1k3, 821
qgev_p1k3_cp *(gev_p1k3_cp)*, 354
qgev_p1n, 822
qgev_p1n_cp *(gev_p1n_cp)*, 373, 382
qgev_ppm, 823
qgnorm_k3_cp *(gnorm_k3_cp)*, 398
qgpd_k1_cp *(gpd_k1_cp)*, 418
qgpd_k1_ppm, 824
qgumbel_cp *(gumbel_cp)*, 442
qgumbel_p1, 825
qgumbel_p1_cp *(gumbel_p1_cp)*, 455
qhalfnorm_cp *(halfnorm_cp)*, 477
qinvgamma_cp *(invgamma_cp)*, 497
qinvgauss_cp *(invgauss_cp)*, 516
qlnorm_cp *(lnorm_cp)*, 538
qlnorm_dmgs_cp *(lnorm_dmgs_cp)*, 544
qlnorm_p1, 826
qlnorm_p1_cp *(lnorm_p1_cp)*, 558
qlogis_cp *(logis_cp)*, 578
qlogis_p1, 827
qlogis_p1_cp *(logis_p1_cp)*, 591
qlst_k3_cp *(lst_k3_cp)*, 612
qlst_p1k3, 827
qlst_p1k3_cp *(lst_p1k3_cp)*, 632
qnorm_cp *(norm_cp)*, 691
qnorm_dmgs_cp *(norm_dmgs_cp)*, 697
qnorm_p1, 828
qnorm_p12, 829
qnorm_p12_cp *(norm_p12_cp)*, 733, 741
qnorm_p1_cp *(norm_p1_cp)*, 712
qnorm_p1_formula, 828
qntt_ppm, 829
qpareto_k2_cp *(pareto_k2_cp)*, 766
qpareto_p1k2, 831
qpareto_p1k2_cp *(pareto_p1k2_cp)*, 782
qunif_cp *(unif_cp)*, 859, 865
qunif_formula, 831
qweibull_cp *(weibull_cp)*, 871, 877
qweibull_p2, 832
qweibull_p2_cp *(weibull_p2_cp)*, 891, 898

rcauchy_cp *(cauchy_cp)*, 26
rcauchy_p1_cp *(cauchy_p1_cp)*, 43
reltest, 832, 836
reltest2, 844, 846
reltest2_cases, 849
reltest2_makeep, 850
reltest2_plot, 850
reltest2_predict, 851
reltest2_simulate, 852
reltest_makeep, 839
reltest_makemaxep, 840
reltest_params, 841
reltest_predict, 841
reltest_simulate, 843
rexp_cp *(exp_cp)*, 114
rexp_p1_cp *(exp_p1_cp)*, 125
rfrechet_k1_cp *(frechet_k1_cp)*, 148
rfrechet_p2k1_cp *(frechet_p2k1_cp)*, 164
rgamma_cp *(gamma_cp)*, 185
rgev_cp *(gev_cp)*, 207
rgev_k3_cp *(gev_k3_cp)*, 220
rgev_minmax, 852
rgev_p123_cp *(gev_p123_cp)*, 286, 294
rgev_p123_minmax, 855
rgev_p12_cp *(gev_p12_cp)*, 255, 263
rgev_p12_minmax, 854
rgev_p1_cp *(gev_p1_cp)*, 242
rgev_p1_minmax, 853
rgev_p1k3_cp *(gev_p1k3_cp)*, 354
rgev_p1n_cp *(gev_p1n_cp)*, 373, 382
rgev_p1n_minmax, 856
rgnorm_k3_cp *(gnorm_k3_cp)*, 398
rgpd_k1_cp *(gpd_k1_cp)*, 418
rgpd_k1_minmax, 857
rgumbel_cp *(gumbel_cp)*, 442
rgumbel_p1_cp *(gumbel_p1_cp)*, 455
rhalfnorm_cp *(halfnorm_cp)*, 477
rhp_dmgs_cpmethod, 857
rinvgamma_cp *(invgamma_cp)*, 497
rinvgauss_cp *(invgauss_cp)*, 516
rlnorm_cp *(lnorm_cp)*, 538
rlnorm_dmgs_cp *(lnorm_dmgs_cp)*, 544
rlnorm_p1_cp *(lnorm_p1_cp)*, 558
rlogis_cp *(logis_cp)*, 578
rlogis_p1_cp *(logis_p1_cp)*, 591
rlst_k3_cp *(lst_k3_cp)*, 612
rlst_p1k3_cp *(lst_p1k3_cp)*, 632
rnorm_cp *(norm_cp)*, 691
rnorm_dmgs_cp *(norm_dmgs_cp)*, 697
rnorm_p12_cp *(norm_p12_cp)*, 733, 741

rnorm_p1_cp (norm_p1_cp), 712
rpareto_k2_cp (pareto_k2_cp), 766
rpareto_p1k2_cp (pareto_p1k2_cp), 782
runif_cp (unif_cp), 859, 865
rust_pumethod, 858
rweibull_cp (weibull_cp), 871, 877
rweibull_p2_cp (weibull_p2_cp), 891, 898

tcauchy_cp (cauchy_cp), 26
tcauchy_p1_cp (cauchy_p1_cp), 43
testppm_plot, 858
texp_cp (exp_cp), 114
texp_p1_cp (exp_p1_cp), 125
tfrechet_k1_cp (frechet_k1_cp), 148
tfrechet_p2k1_cp (frechet_p2k1_cp), 164
tgamma_cp (gamma_cp), 185
tgev_cp (gev_cp), 207
tgev_k3_cp (gev_k3_cp), 220
tgev_p123_cp (gev_p123_cp), 286, 294
tgev_p12_cp (gev_p12_cp), 255, 263
tgev_p1_cp (gev_p1_cp), 242
tgev_p1k3_cp (gev_p1k3_cp), 354
tgev_p1n_cp (gev_p1n_cp), 373, 382
tgnorm_k3_cp (gnorm_k3_cp), 398
tgpd_k1_cp (gpd_k1_cp), 418
tgumbel_cp (gumbel_cp), 442
tgumbel_p1_cp (gumbel_p1_cp), 455
thalfnorm_cp (halfnorm_cp), 477
tinvgamma_cp (invgamma_cp), 497
tinvgauss_cp (invgauss_cp), 516
tlnorm_cp (lnorm_cp), 538
tlnorm_p1_cp (lnorm_p1_cp), 558
tlogis_cp (logis_cp), 578
tlogis_p1_cp (logis_p1_cp), 591
tlst_k3_cp (lst_k3_cp), 612
tlst_p1k3_cp (lst_p1k3_cp), 632
tnorm_cp (norm_cp), 691
tnorm_p12_cp (norm_p12_cp), 733, 741
tnorm_p1_cp (norm_p1_cp), 712
tpareto_k2_cp (pareto_k2_cp), 766
tpareto_p1k2_cp (pareto_p1k2_cp), 782
tweibull_cp (weibull_cp), 871, 877
tweibull_p2_cp (weibull_p2_cp), 891, 898

unif_cp, 859, 865

weibull_cp, 871, 877
weibull_f1fa, 883
weibull_f2fa, 884
weibull_fd, 884
weibull_fdd, 885
weibull_ldda, 885
weibull_lddda, 886
weibull_logf, 886
weibull_logfdd, 887
weibull_logfddd, 887
weibull_loglik, 888
weibull_logscores, 888
weibull_means, 889
weibull_mu1fa, 889
weibull_mu2fa, 890
weibull_p1fa, 890
weibull_p2_cp, 891, 898
weibull_p2_f1fa, 905
weibull_p2_f1fw, 906
weibull_p2_f2fa, 906
weibull_p2_f2fw, 907
weibull_p2_fd, 907
weibull_p2_fdd, 908
weibull_p2_ldda, 908
weibull_p2_lddda, 909
weibull_p2_logf, 909
weibull_p2_logfdd, 910
weibull_p2_logfddd, 910
weibull_p2_loglik, 911
weibull_p2_logscores, 911
weibull_p2_means, 912
weibull_p2_mu1fa, 912
weibull_p2_mu2fa, 913
weibull_p2_p1fa, 913
weibull_p2_p2fa, 914
weibull_p2_pd, 914
weibull_p2_pdd, 915
weibull_p2_predictordata, 915
weibull_p2_waic, 916
weibull_p2fa, 917
weibull_pd, 917
weibull_pdd, 918
weibull_waic, 918