A User's Guide to the evd Package (Version 1.1)

Alec Stephenson

Copyright ©2002

Department of Mathematics and Statistics,
Lancaster University,
Lancaster, LA1 4YF.

E-mail: a.stephenson@lancaster.ac.uk

10th May 2002

1 Introduction

1.1 What is the evd package?

The evd (extreme value distributions) package is an add-on package for the R (Ihaka and Gentleman, 1996) statistical computing system. The package extends simulation, distribution, quantile (inverse distribution) and density functions to univariate, bivariate and multivariate parametric extreme value distributions. It also provides fitting functions which calculate maximum likelihood estimates for univariate and bivariate models.

All comments, criticisms and queries on the package or associated documentation are gratefully received.

1.2 Obtaining the package/guide

The evd package can be downloaded from CRAN (The Comprehensive R Archive Network) at `http://cran.r-project.org/`. This guide (in pdf) will be in the directory `evd/doc/` underneath wherever the package is installed. It can also be downloaded directly from `http://www.maths.lancs.ac.uk/~stephena/` (in postscript or pdf).

1.3 Contents

This guide contains examples on the use of the evd package. The examples do not include any theoretical justification. See Coles (2001) for an introduction to the statistics of extreme values. See Kotz and Nadarajah (2000) for a theoretical treatment of univariate and multivariate extreme value distributions. Section 2 covers the standard functions for univariate extreme value distributions. Sections 3 and 4 do the same for bivariate and multivariate models. Maximum likelihood fitting of univariate and bivariate models is discussed in Sections 5 and 6 respectively. Two extended examples, one univariate and one bivariate, using the data sets `oxford` and `sealevel` (both included in the package) are given in Sections 7 and 8.

This guide should not be viewed as an alternative to the help files included within the package. These remain the definitive source of help. A reference manual containing all the help files can be downloaded from `http://www.maths.lancs.ac.uk/~stephena/` or from CRAN.

All of the examples presented in this guide are called with `options(digits = 4)`.

1.4 Citing the package/guide

To cite this guide or the evd package in publications please use the following bibliographic database entry.

```
@MANUAL{key,
TITLE = {A User's Guide to the evd Package (Version 1.1)},
AUTHOR = {Stephenson, A. G.},
YEAR = {2002},
NOTE = {Available from \verb+http://www.maths.lancs.ac.uk/~stephena/+}
}
```

1.5 Caveat

I have checked these functions as best I can but, as ever, they may contain bugs. If you find a bug or suspected bug in the code or the documentation please report it to me at `a.stephenson@lancaster.ac.uk`. If you do find a bug and are the first person to report it, I guarantee to buy you the drink of your choice. If you ever manage to find me.

1.6 Legalese

1.7 Acknowledgments

Thanks to Paulo Ribeiro Jr. for much needed advice.

2 Standard Univariate Functions

The Gumbel, Fréchet and (reversed) Weibull distribution functions are respectively given by

$$G(z) = \exp\left\{-\exp\left[-\left(\frac{z-a}{b}\right)\right]\right\}, \quad -\infty < z < \infty \tag{1}$$

$$G(z) = \begin{cases} 0, & z \le a, \\ \exp\left\{-\left(\frac{z-a}{b}\right)^{-\alpha}\right\}, & z > a, \end{cases} \tag{2}$$

$$G(z) = \begin{cases} \exp\left\{-\left[-\left(\frac{z-a}{b}\right)\right]^{\alpha}\right\}, & z < a, \\ 1, & z \ge a, \end{cases} \tag{3}$$

where a is a location parameter, $b > 0$ is a scale parameter and $\alpha > 0$ is a shape parameter. The distribution (3) is often referred to as the Weibull distribution. To avoid confusion I will call this the reversed Weibull, since it is related by a change of sign to the three parameter Weibull distribution used in survival analysis.

The GEV (Generalized Extreme Value) distribution function is given by

$$G(z) = \exp\left\{-\left[1 + \xi\left(z - \mu\right)/\sigma\right]_+^{-1/\xi}\right\}, \quad (4)$$

where (μ, σ, ξ) are the location, scale and shape parameters respectively, $\sigma > 0$ and $h_+ = \max(h, 0)$. The parametric form of the GEV encompasses that of the Gumbel, Frèchet and reversed Weibull distributions. The Gumbel distribution is obtained in the limit as $\xi \to 0$. The Fréchet and Weibull distributions are obtained when $\xi > 0$ and $\xi < 0$ respectively. To recover the parameterization of the Fréchet distribution (2) set $\xi = 1/\alpha > 0$, $\sigma = b/\alpha > 0$ and $\mu = a + b$. To recover the parameterization of the reversed Weibull distribution (3) set $\xi = -1/\alpha < 0$, $\sigma = b/\alpha > 0$ and $\mu = a - b$.

It is standard practice within R to concatenate the letters r, p, q and d with an abbreviated distribution name to yield the names of the corresponding simulation, distribution, quantile (inverse distribution) and density functions respectively. The evd package follows this convention. Each of the four distributions defined above has an associated set of functions, as shown here for the GEV.

```
rgev(n, loc = 0, scale = 1, shape = 0)
pgev(q, loc = 0, scale = 1, shape = 0, lower.tail = TRUE)
qgev(p, loc = 0, scale = 1, shape = 0, lower.tail = TRUE)
dgev(x, loc = 0, scale = 1, shape = 0, log = FALSE)
```

Parameters arguments can be vectors* (the standard recycling rules are applied). Some examples are given below. They should be familiar to those who have had previous experience with R.

```
> rgev(6, loc = c(2,1), scale = .5, shape = 1)
[1] 1.6355 2.6562 5.7309 0.6468 2.3299 3.0512

> qrweibull(seq(0.1, 0.4, 0.1), 2, 0.5, 1, lower.tail = FALSE)
> qrweibull(seq(0.9, 0.6, -0.1), loc = 2, scale = 0.5, shape = 1)
# Both give
[1] 1.947 1.888 1.822 1.745

> pfrechet(2:6, 2, 0.5, 1)
[1] 0.0000 0.6065 0.7788 0.8465 0.8825
> pfrechet(2:6, 2, 0.5, 1, low = FALSE)
[1] 1.0000 0.3935 0.2212 0.1535 0.1175

> drweibull(-1:3, 2, 0.5, log = TRUE)
[1] -5.307 -3.307 -1.307   -Inf   -Inf
> dgumbel(-1:3, 0, 1)
[1] 0.17937 0.36788 0.25465 0.11820 0.04737
```

*With one exception; the `shape` parameter of the GEV functions cannot be a vector.

Let F be an arbitrary distribution function, and let $X_1, \ldots, X_m$ be a random sample from F. Define $U_m = \max\{X_1, \ldots, X_m\}$ and $L_m = \min\{X_1, \ldots, X_m\}$. The distributions of U_m and L_m are given by

$$\Pr(U_m \leq x) = [F(x)]^m \tag{5}$$
$$\Pr(L_m \leq x) = 1 - [1 - F(x)]^m. \tag{6}$$

Simulation, distribution, quantile and density functions for the distributions of U_m and L_m, given an integer m and an arbitrary distribution function F, are provided by

```
rext(n, quantfun, ..., distn, mlen = 1, largest = TRUE)
pext(q, distnfun, ..., distn, mlen = 1, largest = TRUE, lower.tail = TRUE)
qext(p, quantfun, ..., distn, mlen = 1, largest = TRUE, lower.tail = TRUE)
dext(x, densfun, distnfun, ..., distn, mlen = 1, largest = TRUE, log = FALSE)
```

The integer m should be given to the argument `mlen`. The distribution F can be specified by passing the corresponding quantile, distribution and density functions to `quantfun`, `distnfun` and `densfun` respectively (both the distribution and density functions are required for `dext`). Alternatively, a string can be passed to `distn` such that the name of the quantile/distribution/density function is derived when the string is prefixed by the letter q/p/d. If the distribution of U_m is required use `largest = TRUE` (the default). If the distribution of L_m is required use `largest = FALSE`. Some examples should make this clear.

```
> rext(4, qexp, rate = 1, mlen = 5)
> rext(4, distn = "exp", rate = 1, mlen = 5)
> rext(4, distn = "exp", mlen = 5)
# All simulate from the same distribution
[1] 2.2001 0.8584 4.5595 3.9397

> rext(1, distn = "norm", sd = 2, mlen = 20, largest = FALSE)
[1] -4.403
# Simulates from the same distribution as
> min(rnorm(20, mean = 0, sd = 2))
[1] -2.612

> pext(c(.4, .5), distn = "norm", mean = 0.5, sd = c(1, 2), mlen = 4)
[1] 0.04484 0.06250

> dext(c(1, 4), distn = "gamma", shape = 1, scale = 0.3, mlen = 100)
[1] 0.3261328 0.0005398
```

Parameters can be given as vectors assuming that this is implemented in the functions passed as arguments (either directly, using `[quant/dist/dens]fun`, or indirectly, using `distn`). If any of the parameters of F are omitted the defaults defined in the corresponding distribution/density/quantile function are used. If default values do not exist an error occurs. Although the examples above use functions provided in R, user defined functions can be specified. Density functions must have `log` arguments.[†]

Let $X_{(1)} \geq X_{(2)} \geq \cdots \geq X_{(m)}$ be the order statistics of the random sample $X_1, \ldots, X_m$. The distribution of the jth largest order statistic, for $j = 1, \ldots, m$, is

$$\Pr(X_{(j)} \leq x) = \sum_{k=0}^{j-1} \binom{m}{k} [F(x)]^{m-k} [1 - F(x)]^k. \tag{7}$$

[†]A simple wrapper can always be constructed to achieve this.

The distribution of the jth smallest order statistic is obtained by setting $j = m+1-j$. Simulation, distribution and density functions for the distribution of $X_{(j)}$ for given integers m and $j \in \{1, \dots, m\}$, and for an arbitrary distribution function F, are provided by

```
rorder(n, quantfun, ..., distn, mlen = 1, j = 1, largest = TRUE)
porder(q, distn, ..., mlen = 1, j = 1, largest = TRUE, lower.tail = TRUE)
dorder(x, densfun, distnfun, ..., distn, mlen = 1, j = 1, largest = TRUE,
       log = FALSE)
```

The integer m should again be given to the argument `mlen`. If `largest = TRUE` (the default) the distribution of the `j`th largest order statistic $X_{(j)}$ is used. If `largest = FALSE` the distribution of the `j`th smallest order statistic $X_{(m+j-1)}$ is used.

For computational reasons it is better to specify `j` to be an integer in the interval $[1, \texttt{ceiling(mlen/2)}]$. This can always be achieved using the argument `largest`. Some examples are given below.

```
> rorder(1, distn = "norm", mlen = 20, j = 2)
> rorder(1, distn = "norm", mlen = 20, j = 19, largest = FALSE)
# Both simulate the second largest order statistic from 20 standard normals
# The first expression is preferred since j is in the interval [1,10]
[1] 2.284

> porder(c(1, 2), distn = "gamma", shape = c(.5, .7), mlen = 10, j = 2)
[1] 0.5177 0.8259
> dorder(c(1, 2), distn = "gamma", shape = c(.5, .7), mlen = 10, j = 2)
[1] 0.7473 0.3081
```

3 Standard Bivariate Functions

The evd package contains functions associated with eight (parametric) bivariate extreme value distributions. The univariate marginal distributions in each case are GEV, with marginal parameters (μ_1, σ_1, ξ_1) and (μ_2, σ_2, ξ_2).

There are three symmetric models, given by

$$G(z_1, z_2) = \exp\left\{-(y_1^{1/\alpha} + y_2^{1/\alpha})^{\alpha}\right\}, \quad 0 < \alpha \leq 1, \tag{8}$$

$$G(z_1, z_2) = \exp\left\{-y_1 - y_2 + (y_1^{-r} + y_2^{-r})^{-1/r}\right\}, \quad r > 0, \tag{9}$$

$$G(z_1, z_2) = \exp\left(-y_1\Phi\{\lambda^{-1} + \tfrac{1}{2}\lambda[\log(y_1/y_2)]\} - y_2\Phi\{\lambda^{-1} + \tfrac{1}{2}\lambda[\log(y_2/y_1)]\}\right), \quad \lambda > 0,$$

known as the logistic (Gumbel, 1960), negative logistic (Galambos, 1975) and Hüsler-Reiss (Hüsler and Reiss, 1989) models respectively, where

$$y_j = y_j(z_j) = \{1 + \xi_j(z_j - \mu_j)/\sigma_j\}_+^{-1/\xi_j} \tag{10}$$

for $j = 1, 2$. Independence* is obtained when $\alpha = 1$, $r \downarrow 0$ or $\lambda \downarrow 0$. Complete dependence† is obtained when $\alpha \downarrow 0$, $r \to \infty$ or $\lambda \to \infty$.

*Independence occurs when $G(z_1, z_2) = \exp\{-(y_1 + y_2)\}$.

†Complete dependence occurs when $G(z_1, z_2) = \exp\{-\max(y_1, y_2)\}$.

The distributions (8) and (9) have asymmetric extensions, given by

$$G(z_1,z_2) = \exp\left\{-(1-\theta_1)y_1 - (1-\theta_2)y_2 - [(\theta_1 y_1)^{1/\alpha} + (\theta_2 y_2)^{1/\alpha}]^{\alpha}\right\}, \quad 0 < \alpha \le 1, \qquad (11)$$

$$G(z_1,z_2) = \exp\left\{-y_1 - y_2 + [(\theta_1 y_1)^{-r} + (\theta_2 y_2)^{-r}]^{-1/r}\right\}, \quad r > 0,$$

known as the asymmetric logistic (Tawn, 1988) and asymmetric negative logistic (Joe, 1990) models respectively, where the asymmetry parameters $0 \le \theta_1, \theta_2 \le 1$. For the asymmetric logistic model independence is obtained when either $\alpha = 1$, $\theta_1 = 0$ or $\theta_2 = 0$. Different limits occur when θ_1 and θ_2 are fixed and $\alpha \downarrow 0$. For the asymmetric negative logistic model independence is obtained when either $r \downarrow 0$, $\theta_1 \downarrow 0$ or $\theta_2 \downarrow 0$. Different limits occur when θ_1 and θ_2 are fixed and $r \to \infty$.

Any bivariate extreme value distribution function can be expressed as (de Haan, 1984)

$$G(z_1,z_2) = \exp\left\{-\int_0^1 \max\{y_1 f_1(x), y_2 f_2(x)\}\,\mathrm{d}x\right\}$$

where (y_1, y_2) are again defined by the transformations (10), and where f_1 and f_2 are density functions with support [0,1].

In particular, if we take the beta densities $f_1(x) = (1-\alpha)x^{-\alpha}$ and $f_2(x) = (1-\beta)(1-x)^{-\beta}$ we obtain

$$G(z_1,z_2) = \exp\left\{-\int_0^1 \max\{y_1(1-\alpha)x^{-\alpha}, y_2(1-\beta)(1-x)^{-\beta}\}\,\mathrm{d}x\right\}, \quad \alpha,\beta < 1.$$

If we further constrain the parameters to be non-negative we obtain the bivariate bilogistic model proposed by Smith (1990), which can also be expressed as

$$G(z_1,z_2) = \exp\left\{-y_1\gamma^{1-\alpha} - y_2(1-\gamma)^{1-\beta}\right\}, \quad 0 < \alpha,\beta < 1,$$

where $\gamma = \gamma(y_1, y_2; \alpha, \beta)$ solves $(1-\alpha)y_1(1-\gamma)^{\beta} = (1-\beta)y_2\gamma^{\alpha}$. The logistic model is obtained when $\alpha = \beta$. Independence is obtained as $\alpha = \beta \to 1$, and when one of α, β is fixed and the other approaches one. Different limits occur when one of α, β is fixed and the other approaches zero.

Alternatively, if we constrain both parameters to be non-positive and set $\alpha_0 = -\alpha > 0$ and $\beta_0 = -\beta > 0$ we obtain the negative bilogistic model (Coles and Tawn, 1994) which has the representation

$$G(z_1,z_2) = \exp\left\{-y_1 - y_2 + y_1\gamma^{1+\alpha_0} + y_2(1-\gamma)^{1+\beta_0}\right\}, \quad \alpha_0,\beta_0 > 0,$$

where $\gamma = \gamma(y_1, y_2; -\alpha_0, -\beta_0)$. The negative logistic model is obtained when $\alpha_0 = \beta_0$. Independence is obtained as $\alpha_0 = \beta_0 \to \infty$, and when one of α_0, β_0 is fixed and the other tends to ∞. Different limits occur when one of α_0, β_0 is fixed and the other approaches zero.

The Coles-Tawn model[‡] (Coles and Tawn, 1991) is the final model that is considered in the evd package. The distribution function is given by

$$G(z_1,z_2) = \exp\{-y_1[1 - \mathrm{Be}(u;\alpha+1,\beta)] - y_2\,\mathrm{Be}(u;\alpha,\beta+1)\}, \quad \alpha,\beta > 0,$$

where $u = \alpha_1 y_2/(\alpha_1 y_2 + \alpha_2 y_1)$ and Be is the incomplete beta function, given by

$$\mathrm{Be}(u;\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\int_0^u x^{\alpha-1}(1-x)^{\beta-1}\,\mathrm{d}x.$$

[‡]Coles and Tawn (1991) call this the Dirichelet model. I avoid this term because it could be confused with the Dirichelet distribution.

Complete dependence is obtained in the limit as $\alpha = \beta \to \infty$. Independence is obtained as $\alpha = \beta \to 0$ and when one of α, β is fixed and the other approaches zero. Different limits occur when one of α, β is fixed and the other tends to ∞.

Each of the eight models has a set of functions of the type given here for the asymmetric logistic.

```
rbvalog(n, dep, asy = c(1, 1), mar1 = c(0, 1, 0), mar2 = mar1)
pbvalog(q, dep, asy = c(1, 1), mar1 = c(0, 1, 0), mar2 = mar1)
dbvalog(x, dep, asy = c(1, 1), mar1 = c(0, 1, 0), mar2 = mar1, log = FALSE)
abvalog(x = 0.5, dep, asy = c(1, 1), plot = FALSE, border = TRUE, add = FALSE,
        lty = 1, blty = 3, xlim = c(0, 1), ylim = c(0.5, 1), xlab = "",
        ylab = "", ...)
```

The first three functions are for simulation and for the calculation of the distribution and density functions. The arguments `q` and `x` in `pbvalog` and `dbvalog` respectively should be vectors of length two or matrices with two columns, so that each row specifies a value for (z_1, z_2).

The argument `dep` is the dependence parameter. In this case `dep` represents the parameter α in distribution (11). The argument `asy` is a vector containing the asymmetry parameters (θ_1, θ_2). The marginal parameters (μ_1, σ_1, ξ_1) and (μ_2, σ_2, ξ_2) should be passed to `mar1` and `mar2` respectively. The arguments `mar1` and `mar2` can also be given as matrices with three columns, in which case each column represents a vector of values to be passed to the corresponding marginal parameter (the standard recycling rules are applied). Vector/matrix arguments for `dep` and `asy` are not implemented.

The `abvalog` function calculates (by default) or plots[§] the dependence function $A(\cdot)$, which is defined as follows. Any bivariate extreme value distribution function can be represented in the form

$$G(z_1, z_2) = \exp\left\{-(y_1 + y_2)A\left(\frac{y_1}{y_1 + y_2}\right)\right\},$$

so that $A(\omega) = -\log\{G(y_1^{-1}(\omega), y_2^{-1}(1-\omega))\}$, defined on $0 \le \omega \le 1$.[¶] It can be shown that $A(0) = A(1) = 1$, and that $A(\cdot)$ is a convex function with $\max(\omega, 1-\omega) \le A(\omega) \le 1$ for all $0 \le \omega \le 1$. $A(\cdot)$ is differentiable when the distribution is jointly continuous. The value $A(1/2) \in [0.5, 1]$ is returned by default. The lower and upper end points of $[0.5, 1]$ are obtained by $A(1/2)$ at complete dependence and at independence respectively. $A(\cdot)$ does not depend on the marginal parameters.

Non-parametric estimators of the dependence function can also be calculated or plotted, using the function `abvnonpar`. Suppose (z_{i1}, z_{i2}) for $i = 1, \ldots, n$ are n bivariate observations that are passed to `abvnonpar` using its `data` argument. The marginal parameters are estimated (under the assumption of independence) and the data is transformed using

$$y_{i1} = \{1 + \hat{\xi}_1(z_{i1} - \hat{\mu}_1)/\hat{\sigma}_1\}_+^{-1/\hat{\xi}_1}$$
$$y_{i2} = \{1 + \hat{\xi}_2(z_{i2} - \hat{\mu}_2)/\hat{\sigma}_2\}_+^{-1/\hat{\xi}_2}$$

for $i = 1, \ldots, n$, where $(\hat{\mu}_1, \hat{\sigma}_1, \hat{\xi}_1)$ and $(\hat{\mu}_2, \hat{\sigma}_2, \hat{\xi}_2)$ are the maximum likelihood estimates for the location, scale and shape parameters on the first and second margins. If non-stationary fitting is implemented using the `nsloc1` or `nsloc2` arguments (see Sections 5 and 6) the marginal location parameters may depend on i.

Three different estimators can be implemented. They are defined (on $0 \le \omega \le 1$) as follows.

[§] If plotted, the values used for the plot are returned invisibly.

[¶] Some authors (e.g. Pickands, 1981) use $A(\omega) = -\log\{G(y_1^{-1}(1-\omega), y_2^{-1}(\omega))\}$.

Pickands (1981)

$$A_p(\omega) = n\left\{\sum_{i=1}^{n} \min\left(\frac{y_{i1}}{\omega}, \frac{y_{i2}}{1-\omega}\right)\right\}^{-1}$$

Deheuvels (1991)

$$A_d(\omega) = n\left\{\sum_{i=1}^{n} \min\left(\frac{y_{i1}}{\omega}, \frac{y_{i2}}{1-\omega}\right) - \omega\sum_{i=1}^{n} y_{i1} - (1-\omega)\sum_{i=1}^{n} y_{i2} + n\right\}^{-1}$$

Capéraà *et al.* (1997); The Default Estimator

$$A_c(\omega) = \exp\left\{\{1-p(t)\}\int_0^t \frac{H_n(x)-x}{x(1-x)}\,\mathrm{d}x - p(t)\int_t^1 \frac{H_n(x)-x}{x(1-x)}\,\mathrm{d}x\right\}$$

In the estimator of Capéraà *et al.* (1997), $H_n(x)$ is the empirical distribution function of $x_1, \dots, x_n$, where $x_i = y_{i2}/(y_{i1} + y_{i2})$ for $i = 1, \dots, n$, and $p(t)$ is any bounded function on $[0, 1]$, which can be specified using the argument `wf`. By default $p(\cdot)$ is the identity function.

Let $A_n(\cdot)$ be any estimator of $A(\cdot)$. The estimator proposed by Deheuvels (1991) satisfies $A_n(0) = A_n(1) = 1$. $A_c(\cdot)$ satisfies this constraint when $p(0) = 0$ and $p(1) = 1$. None of the estimators satisfy $\max(\omega, 1-\omega) \le A_n(\omega) \le 1$ for all $0 \le \omega \le 1$. An obvious modification is

$$A_n^{'}(\omega) = \min(1, \max\{A_n(\omega), \omega, 1-\omega\}).$$

Another estimator $A_n^{''}(\omega)$ can be derived by taking the convex hull of $A_n^{'}(\omega)$. These modifications can be implemented using the `modify` argument. Set `modify` $= 1$ to plot or calculate $A_n^{'}(\omega)$. Set `modify` $= 2$ to plot or calculate $A_n^{''}(\omega)$. Please refer to Section 8 for examples.

Some of the functions outlined in this section are illustrated below.

```
> rbvalog(3, dep = .8, asy = c(.4, 1))
         [,1]    [,2]
[1,]  0.07876 -0.7971
[2,]  0.01091 -0.8113
[3,] -0.10491 -0.8831

> rbvnegbilog(3, alpha = .5, beta = 1.2, mar1 = c(1, 1, 1))
       [,1]  [,2]
[1,] 0.7417 1.085
[2,] 0.8391 1.825
[3,] 2.0142 2.280

> pbvaneglog(c(1, 1.2), dep = .4, asy = c(.4, .6), mar1 = c(1, 1, 1))
[1] 0.173
> tmp.quant <- matrix(c(1,1.2,1,2),ncol = 2, byrow = TRUE)
> tmp.mar <- matrix(c(1,1,1,1.2,1.2,1.2), ncol = 3, byrow = TRUE)
> pbvaneglog(tmp.quant, dep = .4, asy = c(.4, .6), mar1 = tmp.mar)
[1] 0.173 0.175

> dbvct(c(1, 1.2), alpha = .2, beta = .6, mar1 = c(1, 1, 1))
[1] 0.1213
> dbvct(tmp.quant, alpha = 0.2, beta = 0.6, mar1 = tmp.mar)
```

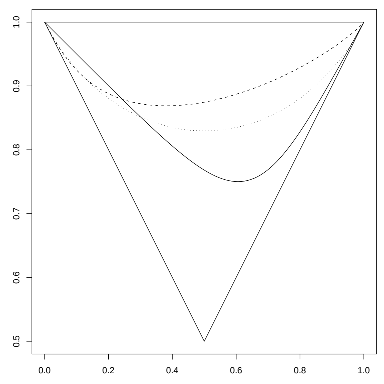

Figure 1: Dependence functions for various bivariate extreme value distributions. All dependence functions must be convex and must lie within the triangular region.

```
[1] 0.1213 0.0586

> abvlog(dep = .3)
[1] 0.6156
> abvlog(seq(0, 1, 0.25), dep = .3)
[1] 1.0000 0.7557 0.6156 0.7557 1.0000

> abvalog(dep = .3, asy = c(.5, .9), plot = TRUE, blty = 1)
> abvbilog(alpha = .5, beta = .9, add = TRUE, lty = 2)
> abvhr(dep = 1.05, add = TRUE, lty = 3)
```

The last three lines of code produce Figure 1.

The simulation functions `rbvlog` and `rbvalog` use bivariate versions of Algorithms 1.1 and 1.2 in Stephenson (2002). All other simulation functions use a root finding algorithm to generate random vectors from the conditional distribution function. The simulation functions `rbvbilog` and `rbvnegbilog` for the bilogistic and negative bilogistic models are relatively slow (about 2.8 seconds per 1000 random vectors on a 450MHz PIII, 512Mb RAM) because each evaluation of either distribution function requires a root finding algorithm to evaluate γ.

4 Standard Multivariate Functions

Let $z = (z_1, \ldots, z_d)$. The d-dimensional logistic model (Gumbel, 1960) is given by

$$G(z) = \exp\left\{-\left(\sum\nolimits_{j=1}^{d} y_j^{-1/\alpha}\right)^{\alpha}\right\} \tag{12}$$

where $\alpha \in (0, 1]$ and $(y_1, \ldots, y_d)$ is defined by the transformations (10).

This distribution can be extended to an asymmetric form. Let B be the set of all non-empty subsets of $\{1, \ldots, d\}$, let $B_1 = \{b \in B : |b| = 1\}$ and let $B_{(i)} = \{b \in B : i \in b\}$. The multivariate asymmetric logistic model (Tawn, 1990) is given by

$$G(z) = \exp\left\{-\sum\nolimits_{b \in B}\left[\sum\nolimits_{i \in b}(\theta_{i,b} y_i)^{1/\alpha_b}\right]^{\alpha_b}\right\}$$

where the dependence parameters $\alpha_b \in (0, 1]$ for all $b \in B \setminus B_1$, and the asymmetry parameters $\theta_{i,b} \in [0, 1]$ for all $b \in B$ and $i \in b$. The constraints $\sum_{b \in B_{(i)}} \theta_{i,b} = 1$ for $i = 1, \ldots, d$ ensure that the marginal distributions are GEV. There exists further constraints which arise from the possible redundancy of asymmetry parameters in the expansion of the distributional form. Specifically, if $\alpha_b = 1$ for some $b \in B \setminus B_1$ then $\theta_{i,b} = 0$ for all $i \in b$. Let $b_{-i_0} = \{i \in b : i \neq i_0\}$. If, for some $b \in B \setminus B_1$, $\theta_{i,b} = 0$ for all $i \in b_{-i_0}$ then $\theta_{i_0,b} = 0$. The model contains $2^d - d - 1$ dependence parameters and $d2^{d-1}$ asymmetry parameters (excluding the constraints). The logistic model (12) can be obtained by setting $\theta_{i,12\ldots d} = 1$ for all $i = 1, \ldots, d$ (which implies, using the sum constraints, that $\theta_{i,b} = 0$ whenever $|b| < d$).

The evd package provides the following functions for simulating from and calculating the distribution function of these models.

```
rmvlog(n, dep, d = 2, mar = c(0, 1, 0))
pmvlog(q, dep, d = 2, mar = c(0, 1, 0))
rmvalog(n, dep, asy, d = 2, mar = c(0, 1, 0))
pmvalog(q, dep, asy, d = 2, mar = c(0, 1, 0))
```

The argument `mar` represents the GEV marginal parameters for every univariate margin, and may again be a matrix. The simulation functions `rmvlog` and `rmvalog` use Algorithms 2.1 and 2.2 in Stephenson (2002).

For the symmetric model `dep` $= \alpha$. Some examples are given below.

```
> rmvlog(3, dep = .6, d = 5)
        [,1]    [,2]    [,3]     [,4]  [,5]
[1,]  0.1335  0.2878 1.07886  1.55515 1.310
[2,]  1.7100  0.9453 1.02070 -0.02553 1.527
[3,] -0.3376 -0.5814 0.07426  0.10906 2.827

> tmp.mar <- matrix(c(1,1,1,1,1,1.5,1,1,2), ncol = 3, byrow = TRUE)
> rmvlog(3, dep = .6, d = 5, mar = tmp.mar)
      [,1]   [,2]   [,3]   [,4]  [,5]
[1,] 2.803 4.6415 1.8531 3.5569 8.854
[2,] 0.751 0.9704 2.3328 2.6537 1.233
[3,] 4.641 1.4321 0.5825 0.6041 2.021

> tmp.quant <- matrix(rep(c(1,1.5,2), 5), ncol = 5)
> pmvlog(tmp.quant, dep = .6, d = 5, mar = tmp.mar)
[1] 0.07233 0.16387 0.21949
```

For the asymmetric model `dep` should be a vector of length $2^{\mathtt{d}} - \mathtt{d} - 1$ containing the dependence parameters. Specifically, when $\mathtt{d} = 4$

$$\mathtt{dep} = (\alpha_{12}, \alpha_{13}, \alpha_{14}, \alpha_{23}, \alpha_{24}, \alpha_{34}, \alpha_{123}, \alpha_{124}, \alpha_{134}, \alpha_{234}, \alpha_{1234}).$$

The asymmetry parameters should be passed to `asy` in a list with $2^{\mathtt{d}} - 1$ elements, where each element is a vector* corresponding to a set $b \in B$, containing $\{\theta_{i,b} : i \in b\}$. Specifically, when $\mathtt{d} = 4$

$$\begin{aligned}\mathtt{asy} = \mathtt{list}(&\theta_{1,1}, \theta_{2,2}, \theta_{3,3}, \theta_{4,4}, \mathtt{c}(\theta_{1,12}, \theta_{2,12}), \mathtt{c}(\theta_{1,13}, \theta_{3,13}), \mathtt{c}(\theta_{1,14}, \theta_{4,14}), \mathtt{c}(\theta_{2,23}, \theta_{3,23}),\\ &\mathtt{c}(\theta_{2,24}, \theta_{4,24}), \mathtt{c}(\theta_{3,34}, \theta_{4,34}), \mathtt{c}(\theta_{1,123}, \theta_{2,123}, \theta_{3,123}), \mathtt{c}(\theta_{1,124}, \theta_{2,124}, \theta_{4,124}),\\ &\mathtt{c}(\theta_{1,134}, \theta_{3,134}, \theta_{4,134}), \mathtt{c}(\theta_{2,234}, \theta_{3,234}, \theta_{4,234}), \mathtt{c}(\theta_{1,1234}, \theta_{2,1234}, \theta_{3,1234}, \theta_{4,1234})).\end{aligned}$$

*Including vectors of length one.

All the constraints, including $\sum_{b \in B_{(i)}} \theta_{i,b} = 1$ for $i = 1, \ldots, d$, must be satisfied or an error will occur.

The dependence parameters used in the following trivariate asymmetric logistic model are $(\alpha_{12}, \alpha_{13}, \alpha_{23}, \alpha_{123}) = (.6, .5, .8, .3)$. The asymmetry parameters are $\theta_{1,1} = .4$, $\theta_{2,2} = 0$, $\theta_{3,3} = .6$, $(\theta_{1,12}, \theta_{2,12}) = (.3, .2)$, $(\theta_{1,13}, \theta_{3,13}) = (.1, .1)$, $(\theta_{2,23}, \theta_{3,23}) = (.4, .1)$ and finally $(\theta_{1,123}, \theta_{2,123}, \theta_{3,123}) = (.2, .4, .2)$. Notice that the constraints are satisfied.

```
> rmvalog(3, dep = c(.6,.5,.8,.3), asy =
  list(.4,0,.6,c(.3,.2),c(.1,.1),c(.4,.1),c(.2,.4,.2)), d = 3)
         [,1]    [,2]    [,3]
[1,]  0.52375 -0.8844  1.4898
[2,]  1.16174 -0.4368 -0.7404
[3,] -0.03737  1.5139 -0.5996

> pmvalog(c(2, 2, 2), dep = c(.6,.5,.8,.3), asy =
  list(.4,.0,.6,c(.3,.2),c(.1,.1),c(.4,.1),c(.2,.4,.2)), d = 3)
[1] 0.7223

> tmp.quant <- matrix(rep(c(1,1.5,2), 3), ncol = 3)
> pmvalog(tmp.quant, dep = c(.6,.5,.8,.3), asy =
  list(.4,.0,.6,c(.3,.2),c(.1,.1),c(.4,.1),c(.2,.4,.2)), d = 3)
[1] 0.4131 0.5849 0.7223
```

The dependence parameters used in the following four dimensional asymmetric logistic model are $\alpha_b = 1$ for $|b| = 2^\dagger$ and $(\alpha_{123}, \alpha_{124}, \alpha_{134}, \alpha_{234}, \alpha_{1234}) = (.7, .3, .8, .7, .5)$. The asymmetry parameters are $\theta_{i,b} = 0$ for all $i \in b$ such that $|b| \leq 2$, $(\theta_{1,123}, \theta_{2,123}, \theta_{3,123}) = (.2, .1, .2)$, $(\theta_{1,124}, \theta_{2,124}, \theta_{4,124}) = (.1, .1, .2)$, $(\theta_{1,134}, \theta_{3,134}, \theta_{4,134}) = (.3, .4, .1)$, $(\theta_{2,234}, \theta_{3,234}, \theta_{4,234}) = (.2, .2, .2)$ and finally $(\theta_{1,1234}, \theta_{2,1234}, \theta_{3,1234}, \theta_{4,1234}) = (.4, .6, .2, .5)$.

```
> rmvalog(3, dep = c(rep(1,6),.7,.3,.8,.7,.5), asy =
  list(0, 0, 0, 0, c(0,0), c(0,0), c(0,0), c(0,0), c(0,0), c(0,0),
  c(.2,.1,.2), c(.1,.1,.2), c(.3,.4,.1), c(.2,.2,.2), c(.4,.6,.2,.5)), d = 4)
        [,1]    [,2]    [,3]    [,4]
[1,] -0.5930 -0.1916  1.0211  0.6113
[2,]  4.3522 -1.0050  2.3618 -0.1875
[3,]  0.5805  0.4443 -0.5958  0.9717
```

I will end this section with some examples that may be helpful in deciphering errors.

```
> rmvalog(3, dep = c(.6,.5,.8,.3), asy =
  list(.4,0,.5,c(.3,.2),c(.1,.15),c(.4,.075),c(.2,.4,.25)), d = 3)
Error in rmvalog(3, dep = c(0.6, 0.5, 0.8, 0.3), asy = list(0.4, 0, 0.5,  :
        'asy' does not satisfy the appropriate constraints

# 0.5 + 0.15 + 0.075 + 0.25 does not equal one; the sum constraint on the third
margin is not satisfied.

> rmvalog(3, dep = c(.6,1,.8,.3), asy =
```

†The values taken by α_b when $|b| = 2$ are irrelevant here because $\theta_{i,b} = 0$ for all $i \in b$ such that $|b| = 2$.

```
  list(.4,0,.6,c(.3,.2),c(.1,.1),c(.4,.1),c(.2,.4,.2)), d = 3)
Error in rmvalog(3, dep = c(0.6, 1, 0.8, 0.3), asy = list(0.4, 0, 0.6,  :
        ‘asy’ does not satisfy the appropriate constraints

# A dependence parameter is equal to one but the corresponding asymmetry
parameters are not zero (the first ‘further constraint’).
# One possible alternative which preserves dep (and still satisfies the sum
constraints) is

> rmvalog(3, dep = c(.6,1,.8,.3), asy =
  list(.4,0,.6,c(.3,.2),c(0,0),c(.4,.1),c(.3,.4,.3)), d = 3)
       [,1]    [,2]    [,3]
[1,]  4.627  2.9125  0.9414
[2,]  1.200  0.1556  0.2048
[3,] -1.159 -0.8749 -1.0340

> rmvalog(3, dep = c(.6,.5,.8,.3), asy =
  list(.5,0,.6,c(.3,.2),c(0,.1),c(.4,.1),c(.2,.4,.2)), d = 3)
Error in rmvalog(3, dep = c(0.6, 0.5, 0.8, 0.3), asy = list(0.5, 0, 0.6,  :
        ‘asy’ does not satisfy the appropriate constraints

# The fifth element in asy contains exactly one non-zero asymmetry parameter
(the second ‘further constraint’).

> rmvalog(3, dep = c(.6,.5,.8,.3), asy =
  list(.4,0,.6,c(.3,.2),c(.1,.1),c(.4,.1),c(.2,.4,.2)))
Error in rmvalog(3, dep = c(0.6, 0.5, 0.8, 0.3), asy = list(0.4, 0, 0.6,  :
        ‘asy’ is not of the correct form

# asy is not of the correct form only because the dimension has been
mis-specified (the default dimension is 2).
```

5 Fitting Univariate Distributions by Maximum Likelihood

This section presents the functions which produce maximum likelihood estimates for the univariate distributions introduced in Section 2. Maximum likelihood estimates for bivariate distributions are discussed in Section 6. For illustrative purposes Sections 5 and 6 use only simulated data. Two extended examples (one univariate and one bivariate) using the data sets `oxford` and `sealevel` (both included in the evd package) are given in Sections 7 and 8.

The functions that produce maximum likelihood estimates for the distributions (1)–(4) are

```
ffrechet(x, start, ..., nsloc = NULL, std.err = TRUE, method = "BFGS")
frweibull(x, start, ..., nsloc = NULL, std.err = TRUE, method = "BFGS")
fgumbel(x, start, ..., nsloc = NULL, std.err = TRUE, method = "BFGS")
fgev(x, start, ..., nsloc = NULL, std.err = TRUE, method = "BFGS")
```

The argument `x` should be given a numeric vector containing data to be fitted. Missing values are allowed. If `start` is given it should be a named list containing starting values, the names of which should be the parameters over which the likelihood is to be maximized. If `start` is omitted the routine attempts to find good starting values for the optimization using moment estimators.

If some of the parameters are to be set to fixed values, they can be given as separate arguments. (The `fgumbel` function calls `fgev` with the shape parameter fixed at zero.) Any arguments that can be passed to the optimization function `optim` can also be specified. This includes the optimization method, which is explicitly passed using the argument `method`. If `std.err` = `TRUE` (the default), the asymptotic standard errors[*] of the maximum likelihood estimates are returned. The `nsloc` argument is explained subsequently.

Two examples of the `fgev` function are given below.

```
> data <- rgev(1000, loc = 0.13, scale = 1.1, shape = 0.2)

> fgev(data)
$estimate
   loc  scale  shape
0.1650 1.1368 0.2035

$std.err
    loc   scale   shape
0.04053 0.03215 0.02486

$deviance
[1] 3645

$counts
function gradient
      56       12

> fgev(data, loc = 0, scale = 1)
$estimate
 shape
0.2282

$std.err
  shape
0.02116

$deviance
[1] 3670

$counts
function gradient
      23        7
```

In the first example the likelihood is maximized over (`loc`, `scale shape`). In the second example the likelihood is maximized over `shape`, with the location and scale parameters fixed at zero and one respectively.

The value returned by the fitting function is a list of four elements. The `estimate` component gives the maximum likelihood estimates.[†]

[*]This is not strictly true. Read on for the details.
[†]Assuming a global maximum has been found.

Maximum likelihood estimators of GEV parameters do not necessarily have the usual asymptotic properties, since the (lower or upper) end point of the GEV distribution (given by $\mu - \sigma/\xi$) depends on the parameters. Smith (1985) shows that the usual asymptotic properties hold when $\xi > -0.5$. When $-1 < \xi \leq -0.5$ the maximum likelihood estimators do not have the standard asymptotic properties, but generally exist. When $\xi \leq -1$ maximum likelihood estimators do not often exist. This occurs because of the large mass near the upper end point. The likelihood increases without bound as the upper end point is estimated to be closer and closer to the largest observed value. In terms of the Weibull shape parameter α, the usual asymptotic properties hold when $\alpha > 2$, the asymptotic properties are not standard for $1 < \alpha \leq 2$, and maximum likelihood estimators do not often exist for $\alpha < 1$.

When the usual asymptotic properties hold the `std.err` component returns the asymptotic standard errors of the maximum likelihood estimates (approximated using the inverse of the observed information matrix). When the usual asymptotic properties do not hold the `std.err` component must be *interpreted with caution* (Smith, 1985). Confidence intervals based on profile likelihoods are discussed in Section 7.

The `deviance` component is minus twice the log-likelihood function, evaluated at `estimate`. Likelihood ratio tests can be performed by comparing with chi-squared distributions the difference between the deviances of nested models.[‡] We can compare the two examples given above to test the null hypothesis that the location parameter is zero and the scale parameter is one. The deviance difference is about $3670 - 3645 = 25$, which yields a p-value of 3.7×10^-6 when compared with a chi-squared distribution on two degrees of freedom.

The `counts` component returns the number of function and gradient evaluations (finite-difference approximations) used by `optim`.

By default the maximum likelihood estimates are calculated under the assumption that the data to be fitted are the observed values of independent random variables $Z_1, \ldots, Z_n$, where $Z_i \sim \text{GEV}(\mu, \sigma, \xi)$ for each $i = 1, \ldots, n$. The `nsloc` argument allows non-stationary models of the form $Z_i \sim \text{GEV}(\mu_i, \sigma, \xi)$, where

$$\mu_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}. \tag{13}$$

The parameters $(\beta_0, \ldots, \beta_k)$ are to be estimated. In matrix notation $\boldsymbol{\mu} = \boldsymbol{\beta_0} + X\boldsymbol{\beta}$, where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^T$, $\boldsymbol{\beta_0} = (\beta_0, \ldots, \beta_0)^T$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)^T$ and X is the $n \times k$ covariate matrix (excluding the intercept) with ijth element x_{ij}.

The `nsloc` argument must be a data frame containing the matrix X, or a numeric vector which is converted into a single column data frame with column name "trend". The column names of the data frame are used to derive names for the estimated parameters. This allows any of the $k + 3$ parameters $(\beta_0, \ldots, \beta_k, \sigma, \xi)$ to be set to fixed values within the optimization. The covariates must be (at least approximately) *centered and scaled*, not only for numerical reasons, but also because the starting value (if `start` is not given) for each corresponding coefficient is taken to be zero.

When a linear trend is present in the data, the location parameter is often modelled as

$$\mu_i = \beta_0 + \beta_1 t_i$$

where t_i is some centered and scaled version of the time of the ith observation. Non-stationary models are rarely fitted, but this is probably the most commonly used form of non-stationarity. More complex changes in μ may also be appropriate. For example, a change-point model

$$\mu_i = \beta_0 + \beta_1 x_i \qquad \text{where} \qquad x_i = \begin{cases} 0 & i \leq i_0 \\ 1 & i > i_0 \end{cases},$$

[‡]There are non-regular cases for which the asymptotic distribution of the test statistic is not chi-squared (see Section 8).

or a quadratic trend

$$\mu_i = \beta_0 + \beta_1 t_i + \beta_2 t_i^2.$$

See Sections 7 and 8 for examples of non-stationary modelling.

The following functions produce maximum likelihood estimates for the distributions (5), (6) and (7).

```
fext(x, start, densfun, distnfun, ..., distn, mlen = 1, largest = TRUE,
     std.err = TRUE, method = "Nelder-Mead")
forder(x, start, densfun, distnfun, ..., distn, mlen = 1, j = 1, largest = TRUE,
     std.err = TRUE, method = "Nelder-Mead")
```

The `fext` function yields maximum likelihood estimates for the distributions (5) and (6) given an integer m and an arbitrary distribution function F.

The arguments `densfun`, `distnfun`, `distn`, `mlen` and `largest` are the same as those used in the density function `dext` (see Section 2). The argument `x` should be a numeric vector containing the data to be fitted, which should be assumed to represent maxima (if `largest` is `TRUE`, the default) or minima (if `largest` is `FALSE`). `start` (which cannot be missing) should be a named list containing starting values, the names of which should be the parameters over which the likelihood is to be maximized. If some of the parameters are to be set to fixed values, they can be given as separate arguments. If parameters are missing, they are fixed at their default values specified in the density/distribution function. Any of the arguments that can be given to `optim` can also be specified.

The optimizer will be affected by the way in which the density and distribution functions passed to `fext` behave when given values outside of the valid parameter space. Functions in R base generally produce `NaN` values which may result in warnings being printed.[§] These warnings can usually be ignored.

If the density and distribution functions are user defined, the order of the arguments must mimic those in R base (i.e. data first, parameters second). The density function must have a `log` argument.[¶]

Two examples are given below. The second example (incorrectly) takes F to be a gamma distribution with the shape parameter fixed at 0.5.

```
> data2 <- rext(100, qnorm, mean = 0.56, mlen = 365)
# Simulate yearly maxima using normal distribution

> fext(data2, list(mean = 0, sd = 1), distn = "norm", mlen = 365)
$estimate
  mean     sd
0.3749 1.0667

$std.err
   mean      sd
0.22111 0.08057

$deviance
[1] 96.98
```

[§]Arguments can be passed to `optim` to avoid this. See `optim`'s help page for details.

[¶]A simple wrapper can always be constructed to achieve this.

```
$counts
function gradient
      53       NA

> fext(data2, list(scale = 1), shape = 0.5, distn = "gamma", mlen = 365)
$estimate
scale
0.737

$std.err
  scale
0.01570

$deviance
[1] 155.8

$counts
function gradient
      28       NA
```

The `forder` function yields maximum likelihood estimates for the distribution (7) given integers m and $j \in \{1, \ldots, m\}$, and an arbitrary distribution function F. The arguments **`densfun`**, **`distnfun`**, **`distn`**, **`mlen`**, **`j`** and **`largest`** are the same as those used in the density function **`dorder`** (see Section 2). The argument **`x`** should be a numeric vector containing the data to be fitted, and **`start`** (which cannot be missing) should again be a named list containing starting values.

6 Fitting Bivariate Distributions by Maximum Likelihood

For each of the eight bivariate models introduced in Section 3 there is a function that produces maximum likelihood estimates. Each function has the same formal arguments. The function corresponding to the logistic model is

```
fbvlog(x, start, ..., nsloc1 = NULL, nsloc2 = NULL, std.err = TRUE,
    method = "BFGS")
```

The argument **`x`** should be given a numeric matrix (or a data frame) containing two columns of data to be fitted. Missing values are allowed on either or both margins/columns within any observation/row. If **`start`** is given it should be a named list containing starting values, the names of which should be the parameters over which the likelihood is to be maximized. See the help file for details. If **`start`** is omitted the routine attempts to find good starting values for the optimization using maximum likelihood estimators under the assumption of independence.

A separate name is associated with each individual parameter so that any parameter subset can be fixed at specified values. The parameters on the first margin can be fixed using the arguments **`loc1`**, **`scale1`** and **`shape1`**. The parameters on the second margin can be fixed using **`loc2`**, **`scale2`** and **`shape2`**. The asymmetry parameters in the asymmetric logistic and asymmetric negative logistic models can be fixed using **`asy1`** and **`asy2`**. As usual, any arguments that can passed to **`optim`** can be specified.

Bivariate Model	Constraints
Logistic	$0.05 \leq \alpha \leq 1$
Asymmetric Logistic	$0.05 \leq \alpha \leq 1$, $0 \leq \theta_1, \theta_2 \leq 1$
Hüsler-Reiss	$0.2 \leq \lambda \leq 10$
Negative Logistic	$0.05 \leq r \leq 8$
Asymmetric Negative Logistic	$0.05 \leq r \leq 8$, $0.001 \leq \theta_1, \theta_2 \leq 1$
Bilogistic	$0.1 \leq \alpha, \beta \leq 0.999$
Negative Bilogistic	$0.1 \leq \alpha, \beta \leq 20$
Coles-Tawn	$0.001 \leq \alpha, \beta \leq 30$

Table 1: For numerical reasons the parameters of each model are subject to the artificial constraints depicted here.

The `nsloc1` and `nsloc2` arguments allow non-stationary modelling of the location parameter on the first and second margin respectively. They should be used in the same manner as the `nsloc` argument in univariate fitting functions. Examples of bivariate models with non-stationary margins are given in Section 8. The `std.err` and `method` arguments are equivalent to those used in univariate fitting functions.

For numerical reasons the parameters of each model are subject to the artificial constraints depicted in Table 1. The scale parameters on each GEV margin are artificially constrained to be greater than or equal to 0.01. These constraints only apply to the functions discussed in this section.

The first example given below uses the `fbvlog` function to obtain maximum likelihood estimates for the (symmetric) logistic model. The second example constrains the model at independence (where `dep` = 1). The estimates produced in the second example are the same as those that would be produced if `fgev` was separately applied to each margin.

```
> bvdata <- rbvlog(100, dep = 0.6, mar1 = c(1.2,1.4,0), mar2 = c(1,1.6,0.1))

> fbvlog(bvdata)
$estimate
   loc1  scale1  shape1    loc2  scale2  shape2     dep
1.10657 1.41054 0.07183 0.80215 1.42536 0.19119 0.49549

$std.err
   loc1  scale1  shape1    loc2  scale2  shape2     dep
0.15358 0.12129 0.05261 0.15550 0.12519 0.05802 0.05018

$deviance
[1] 728

$counts
function gradient
      54       14

> fbvlog(bvdata, dep = 1)
$estimate
  loc1 scale1 shape1   loc2 scale2 shape2
1.0596 1.3683 0.1477 0.7683 1.4236 0.2466
```

```
$std.err
   loc1  scale1  shape1    loc2  scale2  shape2
0.15336 0.11757 0.07342 0.15688 0.12686 0.06897

$deviance
[1] 810

$counts
function gradient
       5        1
```

The discussion in Section 5 regarding the properties of maximum likelihood estimators for the GEV distribution applies to all bivariate models. The usual asymptotic properties hold only when the shape parameters on both margins are greater than -0.5. When the usual asymptotic properties do not hold the `std.err` component must be *interpreted with caution* (Smith, 1985).

The following snippet performs a likelihood ratio test.[*] The null hypothesis specifies that the margins are Gumbel distributions (`shape1` = `shape2` = 0). The deviance of the constrained model is compared with the deviance of the unconstrained model. The p-value is calculated to be 3.7×10^{-5}.

```
> fbvlog(bvdata, shape1 = 0, shape2 = 0)
$estimate
  loc1 scale1   loc2 scale2    dep
1.1731 1.3492 0.9800 1.6917 0.5436

$std.err
   loc1  scale1    loc2  scale2     dep
0.13994 0.10213 0.17543 0.13756 0.05264

$deviance
[1] 748.4

$counts
function gradient
      40       11

> pchisq(748.4 - 728, df = 2, lower.tail = FALSE)
[1] 3.717e-05
```

Boundary Problems

In the following example I attempt to fit the asymmetric logistic model to the simulated data set used above. The data set is distributed as symmetric logistic; both asymmetry parameters equal one. This illustrates the difficulties that arise when parameter estimates are on the edge of the parameter space.

```
> fbvalog(bvdata)
$estimate
   loc1  scale1  shape1    loc2  scale2  shape2    asy1    asy2     dep
```

[*]For illustrative purposes only. I know what the process generating the data is!

```
1.26980 1.57883 0.01057 1.16791 1.77395 0.15257 0.83117 0.99902 0.49552

$std.err
     loc1    scale1    shape1      loc2    scale2    shape2      asy1      asy2
1.750e-01 1.274e-01 8.314e-02 2.021e-01 1.495e-01 7.272e-02 1.421e-01 2.000e-06
      dep
6.759e-02

$deviance
[1] 784.2

$counts
function gradient
     113       11
```

A boundary of the parameter space has been reached; the maximum likelihood estimate for the second asymmetry parameter is (effectively) one. This will cause difficulties for the optimizer. There are two approaches to this problem. The parameter can be fixed at one, or the `L-BFGS-B` method can be used. The `L-BFGS-B` method allows box-constraints using the arguments `lower` and `upper`. The following snippet illustrates these approaches.

```
> fbvalog(bvdata, asy2 = 1)$estimate
   loc1   scale1  shape1    loc2  scale2  shape2    asy1     dep
 1.2739   1.5742 -0.0681  1.1412  1.7910  0.0906  0.8428  0.4947
> fbvalog(bvdata, method = "L-BFGS-B", lower = c(rep(-Inf, 6), 0, 0, -Inf),
  upper = c(rep(Inf, 6), 1, 1, Inf))$estimate
    loc1    scale1    shape1     loc2    scale2    shape2     asy1     asy2      dep
 1.27381   1.57413 -0.06811  1.14123  1.79098  0.09059  0.84280  1.00000  0.49467
```

Fitting Every Model

The `fbvall` function produces maximum likelihood estimates for all eight models.

```
fbvall(x, nsloc1 = NULL, nsloc2 = NULL, omit = NULL, boxcon = TRUE,
  std.err = TRUE, orderby = c("AIC", "BIC", "SC"), control = list(maxit = 250))
```

The argument `x` should again be given a numeric matrix (or a data frame) containing two columns of data to be fitted. The `nsloc1` and `nsloc2` arguments allow non-stationary modelling of the location parameters, for every fitted model. Models can be excluded from the fitting process using `omit`. The (column) order of the models in the components of the returned list is controlled by `orderby`. The `control` argument is passed to the optimization function `optim`. The `boxcon` argument defines the method used for the optimizations. If `boxcon` is `TRUE` (the default), the `L-BFGS-B` optimization method is used, which incorporates box-constraints. If `boxcon` is `FALSE` the `BFGS` method is used. The `BFGS` method is much faster, and should be preferred if the estimates are known to lie in the interior of the parameter space. The help page provides further details. An example is provided in Section 8.

7 Extended Example: Oxford Data

The numeric vector `oxford` contains annual maximum temperatures (in degrees Fahrenheit) at Oxford, England, from 1901 to 1980. It is included in the evd package, and can be made available using `data(oxford)`. The data has previously been analysed by Tabony (1983).

Always begin by plotting the data. The assumptions of stationarity and independence seem sensible, given the plot generated below. The plots generated during this example are presented (with better labelling) in Figure 2 at the end of this section.

```
> data(oxford)
# Load the data
> plot(1901:1980, oxford)
```

The following snippet fits two models, based on the GEV distribution. The first model allows for a trend term in the location parameter (even though the plot appears to show that this is unnecessary). The `nsloc` argument is centered and scaled so that the intercept `loc` represents the location parameter in 1950 and the trend `loctrend` represents the increase in the location parameter (or decrease, if negative) over a period of 100 years. The second model assumes stationarity.

```
> oxford.fit.trend <- fgev(oxford, nsloc = (1901:1980 - 1950)/100)

> oxford.fit.trend
$estimate
     loc loctrend    scale    shape
 83.6617  -1.8812   4.2233  -0.2841

$std.err
     loc loctrend    scale    shape
 0.55566  1.96754  0.36504  0.07068

$deviance
[1] 456.9

$counts
function gradient
      33       14

# the following expressions both produce the same object
> oxford.fit <- fgev(oxford)
> oxford.fit <- fgev(oxford, nsloc = (1901:1980 - 1950)/100, loctrend = 0)

> oxford.fit
$estimate
    loc   scale   shape
83.8392  4.2599 -0.2873

$std.err
    loc   scale   shape
0.52311 0.36579 0.06833

$deviance
[1] 457.8

$counts
function gradient
      29       11
```

The trend term not statistically significant (at any reasonable level). The stationary model `oxford.fit` is retained for further analysis. The fitted shape is negative, so the fitted distribution is Weibull. It is often of interest to test the hypothesis that the shape is zero (the Gumbel distribution). Using asymptotic normality, a 95% confidence interval for the shape parameter is approximately $(-0.42, -0.15)$. The corresponding Wald test can be performed by dividing the maximum likelihood estimate by its standard error. A likelihood ratio test is performed in the following snippet. The hypothesis is rejected at any significance level above about 0.0005.

```
> fgev(oxford, shape = 0)$deviance - oxford.fit$deviance
[1] 12
> pchisq(12, 1, low = FALSE)
[1] 0.000532
```

The following code calculates the (upper) end point* of the fitted distribution to be 98.67. The maximum recorded temperature was 95 degrees Fahrenheit.

```
> mle <- oxford.fit$estimate
> as.vector(mle[1] - mle[2]/mle[3])
[1] 98.67
> range(oxford)
[1] 75 95
```

Basic diagnostic plots can be produced using

```
> pvec <- seq(70, 100, len=200)
> plot(pvec, dgev(pvec, mle[1], mle[2], mle[3]), type = "l")
> rug(jitter(oxford))

> plot(c(70, sort(oxford), 100), c(seq(0, 1, len=81), 1),
  xlim = c(70, 100), type = "s")
> lines(pvec, pgev(pvec, mle[1], mle[2], mle[3]), lty = 2)

> plot(qgev(ppoints(oxford), mle[1], mle[2], mle[3]), sort(oxford))
> abline(0,1)
```

The plots compare parametric distributions, densities and quantiles to their empirical counterparts. The profile deviance (minus twice the profile likelihood) of the shape parameter can be plotted using

```
> shape.profile <- numeric(20)
> shapes <- seq(-0.5, 0.1, len = 20)
> for(i in 1:20)
  shape.profile[i] <- fgev(oxford, start = list(loc = 83.8, scale = 4.25),
  shape = shapes[i], method = "Nelder-Mead")$deviance
Warning messages:
[...]

> plot(sp <- spline(shapes, shape.profile), type = "l")
```

*The upper limit of the distribution function F is defined as $\sup\{x : F(x) < 1\}$.

```
> climit <- oxford.fit$deviance + qchisq(0.95, df = 1)
> climit
[1] 461.6
> abline(h = climit, lty = 2)
> approx(sp$y[sp$x < -0.29], sp$x[sp$x < -0.29], climit)$y
[1] -0.4143
> approx(sp$y[sp$x > -0.29], sp$x[sp$x > -0.29], climit)$y
[1] -0.1388
```

The `Nelder-Mead` optimization method is more robust. It is needed here because the starting values that have been passed to the optimizer occasionally lead to distributional end points that are inconsistant with the observed data[†] (this is specified in the warning messages that have been omitted from the output).

A horizontal line is drawn on the plot so that the two intersections of the line with the profile deviance yield a profile confidence interval, with confidence coefficient 0.95. The end points of the interval are calculated explicitly using `approx`, which performs linear interpolation. The profile confidence interval $(-0.41, -0.14)$ is approximately the same as the confidence interval based on asymptotic normality $(-0.42, -0.15)$, since the profile likelihood/deviance is approximately symmetric.

Profile deviances for the location and scale parameters can be calculated in the same manner.

```
> scale.profile <- numeric(20)
> scales <- seq(3, 6, len = 20)
> for(i in 1:20)
  scale.profile[i] <- fgev(oxford, start = list(loc = 83.8, shape = -0.28),
  scale = scales[i], method = "Nelder-Mead")$deviance
Warning message:
[...]
> plot(spline(scales, scale.profile), type = "l")
> abline(h = climit, lty = 2)

> loc.profile <- numeric(20)
> locs <- seq(82, 86, len = 20)
> for(i in 1:20)
  loc.profile[i] <- fgev(oxford, start = list(scale=4.25, shape=-0.28),
  loc = locs[i], method = "Nelder-Mead")$deviance
> plot(spline(locs, loc.profile), type = "l")
> abline(h = climit, lty = 2)
```

The joint profile deviance of the scale and shape parameters (which are often highly correlated) can be plotted using

```
> scaleshape.profile <- numeric(20^2)
> scaleshapes <- expand.grid(scales, shapes)
> for(i in 1:(20^2))
  scaleshape.profile[i] <- fgev(oxford, start = list(loc=83.8), scale =
  scaleshapes[i,1], shape = scaleshapes[i,2], method = "Nelder-Mead")$deviance
```

[†]Infinite values of the negative log-likelihood are converted to `1e6` when passed to the optimizer. If the optimizer cannot find its way to a ‘finite’ value the deviance component will be `2e6`.

```
[...]

> br.pts <- oxford.fit$deviance +
  qchisq(c(0,0.5,0.8,0.9,0.95,0.99,0.999,0.9999), df = 2)
> image(scales, shapes, matrix(scaleshape.profile, nrow = 20),
  col = heat.colors(8), breaks = c(br.pts,max(scaleshape.profile)))
```

The colours of the image plot represent confidence sets with different confidence coefficients. The lightest colour (ignoring the background colour) represents a confidence ellipse with coefficient 0.9999; the darkest colour represents a confidence ellipse with coefficient 0.5. It looks a bit 'blocky' though, because points were only evaluated on a 20 by 20 grid and no interpolation has been performed (the `spline` function cannot perform bivariate interpolation). The number of points evaluated could be increased, but interpolation is the better option. The following snippet requires the `akima` package. The `interp` function performs bivariate interpolation. The resolution can be increased further by increasing the `length` of the vector passed to `xo` and `yo`.

```
> library(akima)
> profile.interp <- interp(scaleshapes[,1], scaleshapes[,2], scaleshape.profile,
  xo = seq(3, 6, length = 75), yo = seq(-0.5, 0.1, length = 75))
> image(profile.interp, col = heat.colors(8), breaks =
  c(br.pts, max(scaleshape.profile)))
```

A return period plot can be produced using

```
> ret.period <- 10^(seq(0, 3, len = 100))[-1]
> plot(ret.period, qgev(1/ret.period, mle[1], mle[2], mle[3], low = FALSE),
  log = "x", type = "l")
> points(rev(1/ppoints(oxford)), sort(oxford))
```

Return values are simply quantiles in the upper tail. Return values can be generated using `qgev(probs, lower.tail = FALSE)`, where `probs` is a vector of probabilities. The return periods associated with these return values are `1/prob`. The return value associated with a return period of 100 years (or months, or whatever) is therefore defined as the upper quantile evaluated at the probability 0.01. The return period plot compares empirical return values with the fitted model. Confidence intervals for the return value at any specific return period can be calculated using e.g. the delta method, and profile deviances can be calculated by re-parameterizing the GEV likelihood.[‡]

Imagine that Oxford is on another planet (it's easy if you try), and that maximum *daily* temperatures are both stationary and independent. The snippet below fits normal and gamma distributions to the (unrecorded) daily observations.

```
> fext(oxford, start = list(mean = 40, sd = 1), distn = "norm", mlen = 365)
$estimate
 mean    sd
48.85 12.43

$std.err
```

[‡]Which can only be done by hacking the code or using another routine (or waiting until I decide to implement explicit diagnostic functions).

```
  mean     sd
2.7204 0.9928

$deviance
[1] 464.2

$counts
function gradient
      51       NA

> fext(oxford, start = list(scale = 1, shape = 1), distn = "gamma", mlen = 365)
$estimate
scale shape
 1.63 32.94

$std.err
 scale  shape
0.2407 6.0378

$deviance
[1] 465.9

$counts
function gradient
     353       NA

> plot(pvec, dgev(pvec, mle[1], mle[2], mle[3]), type = "l")
> lines(pvec, dext(pvec, mean = 48.85, sd = 12.43, distn = "norm", mlen = 365),
  lty = 2)
> lines(pvec, dext(pvec, scale = 1.63, shape = 32.94, distn = "gamma",
  mlen = 365), lty = 3)
> rug(jitter(oxford))
```

The fitted densities for the *annual* maxima, derived by passing normal and gamma parameter estimates to `dext`, are compared to the GEV model. The normal and gamma models yield very similar distributions, and both are marginally more right skewed than the GEV fit.[§] I'll admit that this example isn't particularly relevant for the `oxford` data, but it may be relevant for other data sets.

[§]Which will not be surprising to those who know about domains of attraction. The limiting distribution (as `mlen` tends to ∞) for both models is Gumbel.

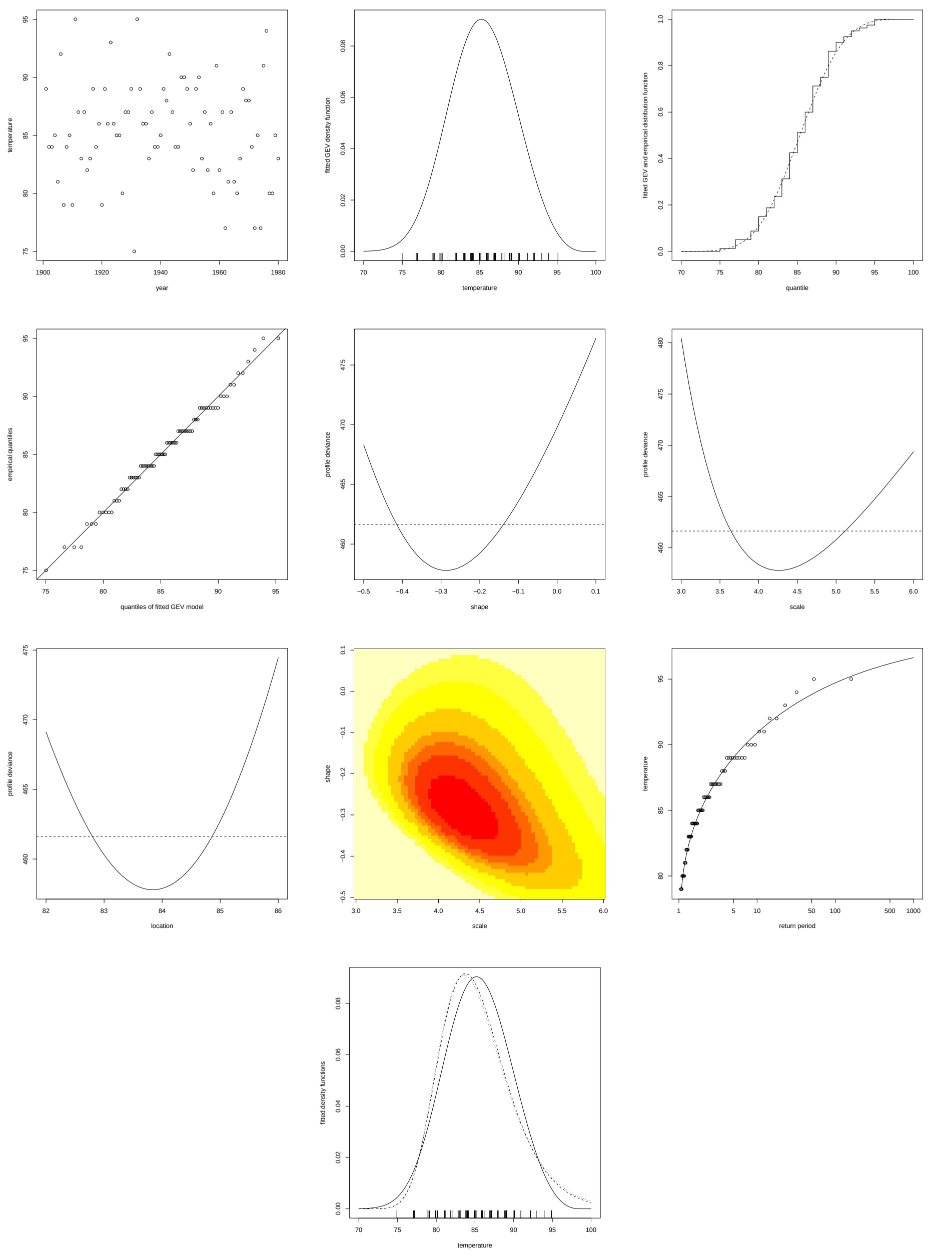

Figure 2: Lots of diagnostic plots. Reading from left to right: the data itself; the fitted GEV density including a rug plot of the (jittered) data; the fitted GEV and empirical distribution functions; a quantile-quantile plot; profile deviances for the shape, scale and location parameters respectively, including horizontal lines representing profile 95% confidence intervals; the joint profile deviance of the scale and shape parameters; a return level plot; and finally, the first plot is repeated along with fitted distributions for models assuming normal and gamma distributions for (unrecorded) daily maxima (valid under extremely tenuous assumptions).

8 Extended Example: Sea Level Data

The `sealevel` data frame (Coles and Tawn, 1990) has two columns containing annual sea level maxima from 1912 to 1992 at Dover and Harwich, two sites on the coast of Britain. It contains 39 missing maxima in total; nine at Dover and thirty at Harwich. There are three years for which the annual maximum is not available at either site.

As usual, I begin by plotting the data. The plot of the Harwich maxima against the Dover maxima shows a reasonable degree of dependence. The outlier corresponds to the 1953 flood resulting from a storm passing over the South-East coast of Britain on 1st February. An increasing trend is clearly depicted in the plot of the Dover maxima. An increasing trend is also arguably present in the plot of the Harwich maxima. The plots generated during this example are presented (with better labelling) in Figure 3 at the end of this section.

```
> data(sealevel)
# Load the data
> plot(sealevel)
> plot(1912:1992, sealevel[,1])
> plot(1912:1992, sealevel[,2])
> sl <- sealevel
# For lazy typists!
```

The following three expressions fit (symmetric) logistic models. The first model incorporates linear trend terms on both marginal location parameters. The second model incorporates a linear trend on the Dover margin only. The third model assumes stationarity. The `nsloc1` and `nsloc2` arguments are centered and scaled so that the intercepts `loc1` and `loc2` represent the marginal location parameters in 1950 and the linear trend parameters `loc1trend` and `loc2trend` represent the increase in the marginal location parameters (or decrease, if negative) over a period of 100 years.

```
> tvec <- (1912:1992 - 1950)/100
> m1 <- fbvlog(sl, nsloc1 = tvec, nsloc2 = tvec)
> m2 <- fbvlog(sl, nsloc1 = tvec)
> m3 <- fbvlog(sl)
```

I'll leave you to analyse the output. In particular, notice how the trend terms affect the parameter estimates. Marginal Weibull distributions (negative shapes) are estimated when the trends are not included, but marginal Fréchet distributions (positive shapes) are estimated upon their inclusion.

The maximum likelihood estimates of the parameters can be compared with their standard errors to perform Wald tests or construct confidence intervals based on asymptotic normality. Likelihood ratio tests can be performed using the following snippet. The p-values confirm the statistical significance of the linear trend terms.

```
> pchisq(m3$deviance - m2$deviance, df = 1, lower.tail = FALSE)
[1] 9.743e-06
> pchisq(m2$deviance - m1$deviance, df = 1, lower.tail = FALSE)
[1] 0.007048
> pchisq(m3$deviance - m1$deviance, df = 2, lower.tail = FALSE)
[1] 1.499e-06
```

The fitted lower limits[*] of the marginal distributions within the model `m1` can be calculated using

```
> mle <- m1$estimate
> as.vector(mle[1] + tvec * mle[2] - mle[3]/mle[4])
> as.vector(mle[5] + tvec * mle[6] - mle[7]/mle[8])
```

A quadratic trend for the location parameter on the Dover margin can be incorporated using the following code. The parameter corresponding to the quadratic term is not found to be statistically significant.

```
> tdframe <- data.frame(linear = tvec, quad = tvec^2)
> m4 <- fbvlog(sl, nsloc1 = tdframe, nsloc2 = tvec)
> m4$estimate[1:3]
      loc1 loc1linear   loc1quad
    3.5845     0.4570    -0.1858
> m4$std.err[1:3]
      loc1 loc1linear   loc1quad
   0.02742    0.08409    0.35635
> pchisq(m1$deviance - m4$deviance, df = 1, lower.tail = FALSE)
[1] 0.6047
```

Note that the following two expressions are equivalent (both produce the model `m1`). This demonstrates that any of the parameters, including those defined by `nsloc1` and `nsloc2`, can be set to fixed values.

```
> fbvlog(sl, nsloc1 = tvec, nsloc2 = tvec)
> fbvlog(sl, nsloc1 = tdframe, nsloc2 = tvec, loc1quad = 0)
```

The code given below compares two logistic models that are nested within `m1`. Model `m5` assumes that both marginal shape parameters are zero (or equivalently, that both marginal distributions are Gumbel). A likelihood ratio test of this hypothesis provides a p-value of 0.72. The hypothesis would not be rejected at any reasonable significance level.

Model `m6` assumes independence. The maximum likelihood estimates are the same as those that would be produced if `fgev` was separately applied to each margin. The deviance increase with respect to model `m1` is calculated to be 13.59. *The asymptotic distribution of the test statistic is not chi-squared.* The distribution is non-regular because the dependence parameter in the restricted (independence) model is fixed at the edge of the parameter space. Testing for the (symmetric) logistic model within the asymmetric logistic model also leads to non-regular behaviour. Tawn (1988) discusses non-regular testing procedures within bivariate extreme value models. In this case the increase in deviance is clearly too large to consider independence.

```
> m5 <- fbvlog(sl, nsloc1 = tvec, nsloc2 = tvec, shape1 = 0, shape2 = 0)
> pchisq(m5$deviance - m1$deviance, df = 2, lower.tail = FALSE)
[1] 0.7169

> m6 <- fbvlog(sl, nsloc1 = tvec, nsloc2 = tvec, dep = 1)
> m6$deviance - m1$deviance
# WARNING: The asymptotic distribution of this statistic is non-regular
[1] 13.59
```

[*]The lower limit of the distribution function F is defined as $\inf\{x : F(x) > 0\}$.

Models other than the logistic can be fitted in a similar fashion, using `fbvhr`, `fbvbilog`, `fbvct` and others. The function `fbvall` attempts to fit all bivariate models simultaneously (see Section 6). The code below provides an example.

```
> fbvall(sl, nsloc1 = tvec, nsloc2 = tvec)
[...]
$estimate
               log     alog  neglog  aneglog      hr   bilog      ct negbilog
loc1       3.57565  3.58502 3.57510  3.58541 3.57528 3.57548 3.57544  3.57516
loc1trend  0.43946  0.47028 0.44014  0.45709 0.44032 0.43871 0.43903  0.44104
scale1     0.15881  0.16978 0.15843  0.16964 0.15809 0.15857 0.15833  0.15863
shape1     0.06638 -0.02468 0.07259 -0.02685 0.07325 0.06623 0.06841  0.07315
loc2       2.54897  2.54968 2.55061  2.55069 2.55167 2.54913 2.54967  2.55054
loc2trend  0.53288  0.49724 0.53568  0.50586 0.53605 0.53504 0.53811  0.53379
scale2     0.20701  0.19881 0.20792  0.19997 0.20852 0.20718 0.20683  0.20784
shape2     0.04781  0.07397 0.03951  0.06370 0.03038 0.04898 0.04233  0.03889
asy1/alpha      NA  0.35214      NA  0.38798      NA 0.69068 0.49647  1.37009
asy2/beta       NA  0.22944      NA  0.26335      NA 0.71339 0.48443  1.54953
dep        0.70282  0.07742 0.68503  8.00000 1.05590      NA      NA       NA

$std.err
               log    alog  neglog  aneglog      hr   bilog      ct negbilog
loc1       0.02173 0.02249 0.02175 0.022029 0.02172 0.02185 0.02174  0.02174
loc1trend  0.07540 0.08247 0.07493 0.082913 0.07467 0.07508 0.07555  0.07489
scale1     0.01699 0.01776 0.01694 0.016893 0.01686 0.01716 0.01701  0.01702
shape1     0.09768 0.07413 0.09800 0.073448 0.09827 0.09840 0.09788  0.09738
loc2       0.03102 0.03078 0.03114 0.030690 0.03131 0.03102 0.03109  0.03113
loc2trend  0.17208 0.12836 0.17368 0.148925 0.17593 0.16241 0.17413  0.17488
scale2     0.02346 0.01820 0.02361 0.019038 0.02376 0.02353 0.02348  0.02350
shape2     0.07607 0.09012 0.07696 0.079492 0.07675 0.07728 0.07606  0.07705
asy1/alpha      NA 0.15832      NA 0.146384      NA 0.15609 0.45240  0.70585
asy2/beta       NA 0.14266      NA 0.111059      NA 0.13944 0.42810  0.72148
dep        0.09551 0.07708 0.19931 0.000002 0.22346      NA      NA       NA

$deviance
     log     alog   neglog  aneglog       hr    bilog       ct negbilog
  -36.50   -39.95   -35.91   -39.74   -35.38   -36.51   -36.22   -35.91

$criteria
        log    alog  neglog aneglog      hr   bilog      ct negbilog
AIC -18.498 -17.954 -17.905 -17.735 -17.376 -16.506 -16.218  -15.913
SC   -2.238   1.919  -1.645   2.138  -1.116   1.560   1.849    2.153
BIC   6.762  12.919   7.355  13.138   7.884  11.560  11.849   12.153

$dep.summary
          log    alog neglog aneglog     hr    bilog       ct negbilog
dep    0.3723  0.2293 0.3635  0.2619 0.3436  0.37284 0.359591  0.36404
intdep 0.5072  0.2765 0.5039  0.3102 0.4837  0.50794 0.495147  0.50458
intasy 0.0000 -0.2741 0.0000 -0.2742 0.0000 -0.02149 0.003764  0.01997

Warning message:
optimization may not have succeeded in: fbvaneglog
```

The `estimate` component contains the maximum likelihood estimates, including the marginal trend parameters. A warning is given that the optimization may not have succeeded for the asymmetric negative logistic model, since the dependence parameter estimate is at its artificial upper bound. The dependence parameter estimate for the asymmetric logistic model is very low; the asymmetric logistic and asymmetric negative logistic models are fitting very close to a (singular) distribution obtained in the limit.

Models that are not nested can be compared by adding penalty terms to the deviances. The penalty terms take into account the number of parameters fitted (if both models have the same number of parameters the deviances can be compared directly). Three commonly used penalty terms are $2p$ (Akaike's information criterion, or AIC), $p\log(n)$ (Schwarz's criterion, or SC) and $p\{1+\log(n)\}$ (Bayesian information criterion, or BIC), where p is the number of parameters estimated and n is the number of observations.[†] The AIC, SC and BIC values for each model are contained in the `criteria` component. By default, the (column) order of the models in the components of the returned list is based on AIC (lowest/best first, highest/worst last). The default behaviour can be changed using the argument `orderby`. The comparison of the models often depends largely on the criterion used. For the sea level data, the asymmetric logistic model has the second best AIC, but the worst BIC (excluding the suspect asymmetric negative logistic fit). The (symmetric) logistic model is seen to give the best fit under all three criteria.

The `dependence` component contains simple summary statistics of the dependence structure of each model, based on the dependence function $A(\cdot)$ defined in Section 3. These summaries are given by

$$\begin{aligned}
\texttt{dep} &= \chi = 2\{1 - A(1/2)\} \\
\texttt{intdep} &= \psi_d = 4\int_0^1 1 - A(x)\,\mathrm{d}x \\
\texttt{intasy} &= \psi_a = \frac{4}{3-2\sqrt{2}}\int_0^{1/2} A(x) - A(1-x)\,\mathrm{d}x
\end{aligned}$$

The two measures of dependence, χ (Coles *et al.*, 1999) and ψ_d, are contained in the closed interval [0,1]. At independence $\chi = \psi_d = 0$, and at complete dependence $\chi = \psi_d = 1$. The summary statistic ψ_a is a measure of asymmetry, and is contained in the closed interval [-1,1].[‡] If $A(\cdot)$ is symmetric $\psi_a = 0$. As a rough guide, any value $\psi_a \in (-0.25, 0.25)$ corresponds to a dependence structure that is close to symmetric.[§]

The dependence functions for the fitted models can be plotted using

```
> abvlog(dep = 0.70282, plot = TRUE)
> abvhr(dep = 1.05590, add = TRUE, lty = 2)
> abvneglog(dep = 0.68503 , add = TRUE, lty = 3)
> legend(0.05, 0.6, c("Logistic", "Husler-Reiss", "Neg Logistic"), lty = 1:3)

> abvbilog(alpha = 0.69068, beta = 0.71339, plot = TRUE)
> abvct(alpha = 0.49647, beta = 0.48443, add = TRUE, lty = 2)
> abvnegbilog(alpha = 1.37009, beta = 1.54953, add = TRUE, lty = 3)
> legend(0.05, 0.6, c("Bilogistic", "Coles-Tawn", "Neg Bilogistic"), lty = 1:3)
```

[†] Since fbvall compares models for the dependence structure, n is taken as the number of observations which are complete (i.e. not missing on either margin).

[‡] Conjecture.

[§] The integral measures ψ_d and ψ_a are not standard theory; I have created them for the purpose of this package. They have no underlying interpretation that I am aware of. I am open to suggestions of other simple summaries of the dependence structure.

```
> abvalog(dep = 0.07742, asy = c(0.35214, 0.22944), plot = TRUE)
> abvaneglog(dep = 8, asy = c(0.38798, 0.26335), add = TRUE, lty = 2)
> legend(0.05, 0.6, c("Asy Logistic", "Asy Neg Logistic"), lty = 1:2)
```

Non-parameteric estimators of the dependence function can be plotted using

```
> abvnonpar(data = sl, nsloc1 = tvec, nsloc2 = tvec, plot = TRUE, modify = 1,
  method = "p")
> abvnonpar(data = sl, nsloc1 = tvec, nsloc2 = tvec, add = TRUE, modify = 2,
  lty = 2, method = "p")

> abvnonpar(data = sl, nsloc1 = tvec, nsloc2 = tvec, plot = TRUE, modify = 1,
  method = "d")
> abvnonpar(data = sl, nsloc1 = tvec, nsloc2 = tvec, add = TRUE, modify = 2,
  lty = 2, method = "d")

> abvnonpar(data = sl, nsloc1 = tvec, nsloc2 = tvec, plot = TRUE, modify = 1)
> abvnonpar(data = sl, nsloc1 = tvec, nsloc2 = tvec, add = TRUE, modify = 2,
  lty = 2)
```

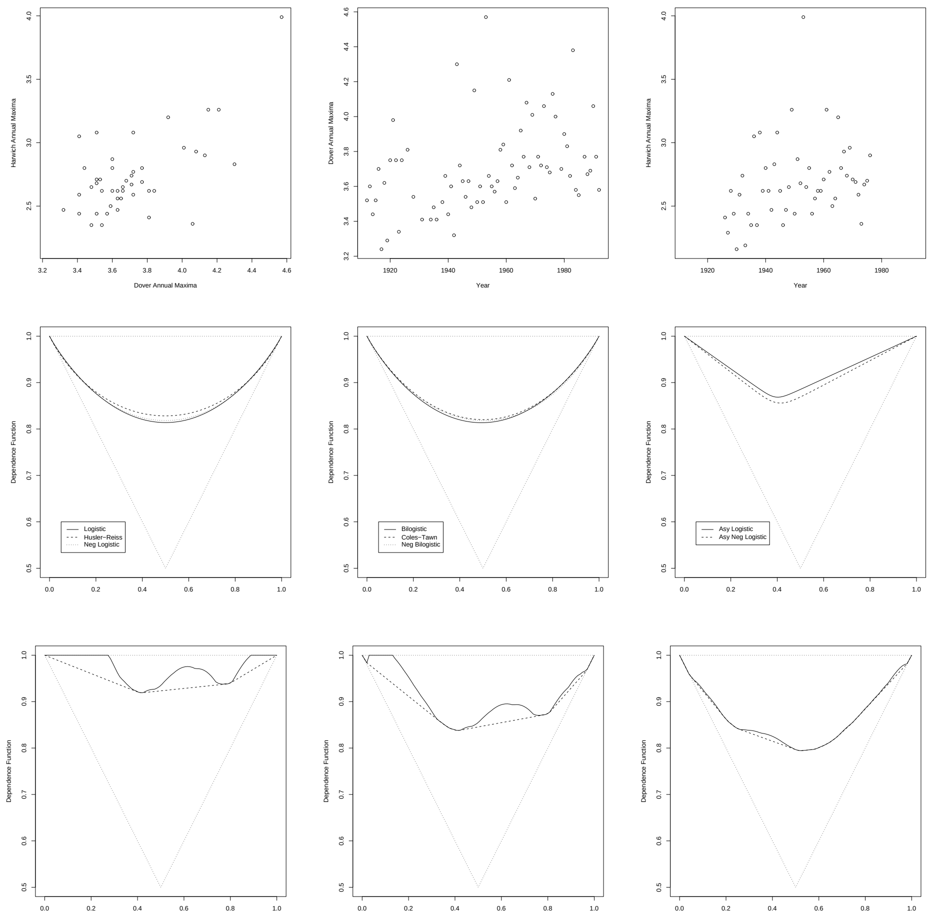

Figure 3: Top row: Harwich Maxima vs Dover Maxima, Dover Maxima vs Year and Harwich Maxima vs Year. Middle row: the dependence functions of fitted bivariate models (the likelihood for the asymmetric negative logistic model has not been properly maximized). Bottom row: non-parametric estimates of the dependence function; reading left to right, modifications of the estimators from Pickands (1981), Deheuvels (1991) and Capéraà *et al.* (1997) are shown.

References

Capéraà, P., Fougères, A.-L. and Genest, C. (1997) A non-parametric estimation procedure for bivariate extreme value copulas. *Biometrika*, **84**, 567–577.

Coles, S. G. (2001) *An Introduction to Statistical Modeling of Extreme Values*. London: Springer–Verlag.

Coles, S. G., Heffernan, J. and Tawn, J. A. (1999) Dependence measures for extreme value analyses. *Extremes*, **2**, 339–365.

Coles, S. G. and Tawn, J. A. (1990) Statistics of coastal flood prevention. *Phil. Trans. R. Soc. Lond., A*, **332**, 457–476.

Coles, S. G. and Tawn, J. A. (1991) Modelling extreme multivariate events. *J. Roy. Statist. Soc., B*, **53**, 377–392.

Coles, S. G. and Tawn, J. A. (1994) Statistical methods for multivariate extremes: an application to structural design (with discussion). *Appl. Statist.*, **43**, 1–48.

de Haan, L. (1984) A spectral representation for max-stable processes. *Ann. Probab.*, **12**, 1194–1204.

Deheuvels, P. (1991) On the limiting behaviour of the pickands estimator for bivariate extreme-value distributions. *Statist. Probab. Letters*, **12**, 429–439.

Galambos, J. (1975) Order statistics of samples from multivariate distributions. *J. Amer. Statist. Assoc.*, **70**, 674–680.

Gumbel, E. J. (1960) Distributions des valeurs extrêmes en plusieurs dimensions. *Publ. Inst. Statist. Univ. Paris*, **9**, 171–173.

Hüsler, J. and Reiss, R.-D. (1989) Maxima of normal random vectors: between independence and complete dependence. *Statist. Probab. Letters*, **7**, 283–286.

Ihaka, R. and Gentleman, R. (1996) R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, **5**, 299–314.

Joe, H. (1990) Families of min-stable multivariate exponential and multivariate extreme value distributions. *Statist. Probab. Letters*, **9**, 75–81.

Kotz, S. and Nadarajah, S. (2000) *Extreme Value Distributions*. London: Imperial College Press.

Pickands, J. (1981) Multivariate extreme value distributions. *Proc. 43rd Sess. Int. Statist. Inst.*, **49**, 859–878.

Smith, R. L. (1985) Maximum likelihood estimation in a class of non-regular cases. *Biometrika*, **72**, 67–90.

Smith, R. L. (1990) Extreme value theory. In *Handbook of Applicable Mathematics* (ed. W. Ledermann), vol. 7. Chichester: John Wiley, pp. 437–471.

Stephenson, A. G. (2002) Simulating multivariate extreme value distributions of logistic type. To be published - available on request.

Tabony, R. C. (1983) Extreme value analysis in meteorology. *The Meteorological Magazine*, **112**, 77–98.

Tawn, J. A. (1988) Bivariate extreme value theory: models and estimation. *Biometrika*, **75**, 397–415.

Tawn, J. A. (1990) Modelling multivariate extreme value distributions. *Biometrika*, **77**, 245–253.