User's Guide for `evmix` Package in `R`

Yang Hu

July 6, 2013

1 Introduction

1.1 What is the `evmix` package for?

`evmix` is an add-on package for the statistical software (R Development Core Team, 2006). The main purpose of this package is to implement extreme value mixture models and other threshold diagnostics to make these new tools accessible for researchers and practitioners in this field. This packages includes the density, distribution, quantile and random number generation function of most of the extreme value mixture distributions defined in the current literature, including extensions of many of them to provide more flexibility. This package provides likelihood inference for these extreme value mixture models, and will later be extended to provide Bayesian inference via MCMC for some of the models which have likelihoods which are challenging to numerically maximise. A reasonably comprehensive review of extreme value mixture models can be found in Scarrott and MacDonald (2012).

Extreme Value Mixture Distributions

The following is a list of all the mixture models which are currently implemented and a reference to where I believe they were first proposed or applied. Very few extreme value mixture models in the literature are constrained to be continuous at the threshold. So to provided greater flexibility we have provided both the original version of each mixture model and where appropriate, an extended version which is constrained to continuous at the threshold is also provided (with the function names post-fixed with `con`).

The implemented extreme value mixture models with a **parametric** model for the bulk component are:

- Normal GPD mixture distribution (Behrens et al., 2004)
- Normal GPD with single continuity constraint mixture distribution
- Gamma GPD mixture distribution (Behrens et al., 2004)
- Gamma GPD with single continuity constraint mixture distribution
- Weibull GPD mixture distribution (Behrens et al., 2004)
- Weibull GPD with single continuity constraint mixture distribution
- Lognormal GPD mixture distribution (Behrens et al., 2004)
- Lognormal GPD with single continuity constraint mixture distribution
- Beta GPD mixture distribution (Behrens et al., 2004)
- Beta GPD with single continuity constraint mixture distribution
- GPD-Normal-GPD(GNG) mixture distribution (Zhao et al., 2010)
- GPD-normal-GPD (GNG) with single continuity constraint mixture distribution

- Dynamically weighted mixture distribution (Frigessi et al., 2002)
- Hybrid Pareto distribution (Carreau and Bengio, 2008)
- Hybrid Pareto with single continuity constraint distribution

The implemented extreme value mixture models with a **semi-parametric** model for the bulk component are:

- Mixture of gamma GPD distribution (Nascimento et al., 2011)

The implemented extreme value mixture models with a **non-parametric** model for the bulk component are:

- Kernel GPD distribution (MacDonald et al., 2011)
- Kernel GPD with single continuity constraint distribution
- Two tailed kernel GPD distribution (MacDonald et al., 2011)
- Boundary correction kernel GPD distribution (MacDonald et al., 2013)

1.2 Obtaining the package/guide

The package can be downloaded from CRAN (The Comprehensive R Archive Network) at http://cran.r-project.org/. The PDF version of this guide will be in package installation directory and available from the help system, type `help(evmix)` and click `browse directory`.

1.3 Citing the package/guide

The package citation is available from `citation(evmix)`. The current citation is:

Hu, Y. and Scarrott, C.J. (2013). evmix: `R` Package for Extreme Value Mixture Modelling and Threshold Estimation. Available on CRAN. URL: http://www.math.canterbury.ac.nz/ c.scarrott/evmix

1.4 Caveat

These functions are provided on a best attempt basis, use at your own risk. They are well tried and tested and have many error checking facilities to ensure users can't do silly things and useful error messages are reported back where needed. However, there will almost certainly be bugs. If you do find a possible bug in the code or documentation then do report it to Carl Scarrott, `carl.scarrott@canterbury.ac.nz`. For bugs, please provide a reproducible example.

1.5 Legalese

2 Generalized Pareto distribution(GPD)

Coles (2001) states that if $X_1, \ldots, X_n$ be a sequence of independent random variables, for some large enough threshold u, the excess $X - u$ may be well approximated by a generalized Pareto distribution.

The distribution function of GPD is given by:

$$G(x|u, \sigma_u, \xi) = \begin{cases} 1 - \left[1 + \xi \left(\frac{x-u}{\sigma_u}\right)\right]_+^{-1/\xi} & \xi \neq 0, \\ 1 - \exp\left[-\left(\frac{x-u}{\sigma_u}\right)\right]_+ & \xi = 0. \end{cases}$$

The shape parameter ξ determines the tail behavior of the GPD:

- $\xi < 0$: short tail with finite upper end point of $u - \frac{\sigma_u}{\xi}$;
- $\xi = 0$: exponential tail; and
- $\xi > 0$: heavy tail.

It is common to use letters `d`, `p`, `q`, `r` with a distribution name to represent the corresponding density, distribution, quantile and random number generation functions respectively. This package follows this guideline. The parameter definitions and ordering are explicitly designed to be similar to those used in the `evd` package, such that most code would be interchangeable between them.

```
# User can specify whether density or log density should be evaluated by setting the log=FALSE or log=TRUE.
dgpd(0.1, u = 0, sigmau = 1, xi = c(0.05, 0.1, 0.15), log = TRUE)
[1] -0.1047384 -0.1094536 -0.1141460

# User can specify whether cumulative or exceedance probability by setting option lower.tail=FALSE or TRUE.
pgpd(0.1,u = 0, sigmau = c(0.99, 1, 1.01), xi = 0.1, lower.tail = FALSE)
[1] 0.9043821 0.9052870 0.9061749

rgpd(5,u = 0, sigmau = 1, xi = 0)
[1] 2.1070377 0.1620293 0.3380177 2.6219259 0.2536274
```

3 Parametric Mixture Models

This section will cover several existing parametric form models in the literature and some extensions.

3.1 Gamma/Normal/Weibull/Log-Normal/Beta GPD Model

Behrens et al. (2004) developed a simple extreme value mixture model by combining a parametric bulk model below the threshold with GPD above the threshold. This has been implemented using gamma, Weibull, normal distribution as the bulk model. There is no standardized framework for defining the renormalizing constant of combining the two distributions in the literature. This

package implements two approaches for specifying the tail fraction ϕ_u either defined using the parameters from the bulk component or as a separate parameter to be estimated, to be detailed below.

The distribution function of the parametric form of the of the bulk component of the extreme value mixture mixture model can be defined as:

Bulk Model Based Tail Fraction Approach

$$F(x|u,\theta,\sigma_u,\xi,\phi_u) = \begin{cases} H(x|\theta) & x \leq u, \\ H(u|\theta) + (1 - H(u|\theta)) \times G(x|u,\sigma_u,\xi) & x > u. \end{cases} \tag{1}$$

Parameterised Tail Fraction Approach

$$F(x|u,\theta,\sigma_u,\xi,\phi_u) = \begin{cases} (1-\phi_u) \times \frac{H(x|\theta)}{H(u|\theta)} & x \leq u, \\ (1-\phi_u) + \phi_u \times G(x|u,\sigma_u,\xi) & x > u, \end{cases} \tag{2}$$

where ϕ_u is the proportion of data above the threshold and $0 < \phi_u < 1$. $H(.|\theta)$ could be parametric distribution function such as gamma, Weibull or normal distribution function and θ is the parameter vector of the bulk distribution. $G(.|u,\sigma_u,\xi)$ represents the GPD distribution function where u is the threshold, ξ is the shape parameter and σ_u is the scale parameter.

The latter approach in Equation (2) explicitly makes it clear that when conditional modeling the upper tail using the GPD also requires the proportion of excesses to obtain the unconditional quantities of interest. The maximum likelihood estimator of this tail fraction parameter is of course the usual sample proportion. The former approach in Equation (1) is included in Equation (2) as a special case when ϕ_u is set to $1 - H(u|\theta)$.

Base Function

The density function of the mixture model:

- `dgammagpd(x, gshape = 1, gscale = 1, u = qgamma(0.9, gshape, gscale), sigmau = sqrt(gshape) * gscale, xi = 0, phiu = TRUE, log = FALSE)` ,

  where `gshape` and `grate` are represented the gamma shape and rate parameter.

- `dweibullgpd(x, wshape = 1, wscale = 1, u = qweibull(0.9, wshape, wscale), sigmau = sqrt(wscale^2 * gamma(1 + 2/wshape) - (wscale * gamma(1 + 1/wshape))^2), xi = 0, phiu = TRUE, log = FALSE)`,

  where `wshape` and `wscale` are represented the Weibull shape and scale parameter.

- `dnormgpd(x, nmean = 0, nsd = 1, u = qnorm(0.9, nmean, nsd), sigmau = nsd, xi = 0, phiu = TRUE, log = FALSE))`,

  where `nmean` and `nsd` are represented the normal mean and standard deviation parameter. The `xi` and `sigmau` are the usual GPD shape and scale parameter. The use of the name `sigmau` reminds us of the dependence between the threshold and GPD scale parameter. If `phiu = TRUE`, the bulk model based approach is adopted. If `phiu` is between 0 and 1 (exclusive), the parameterised approach is adopted.

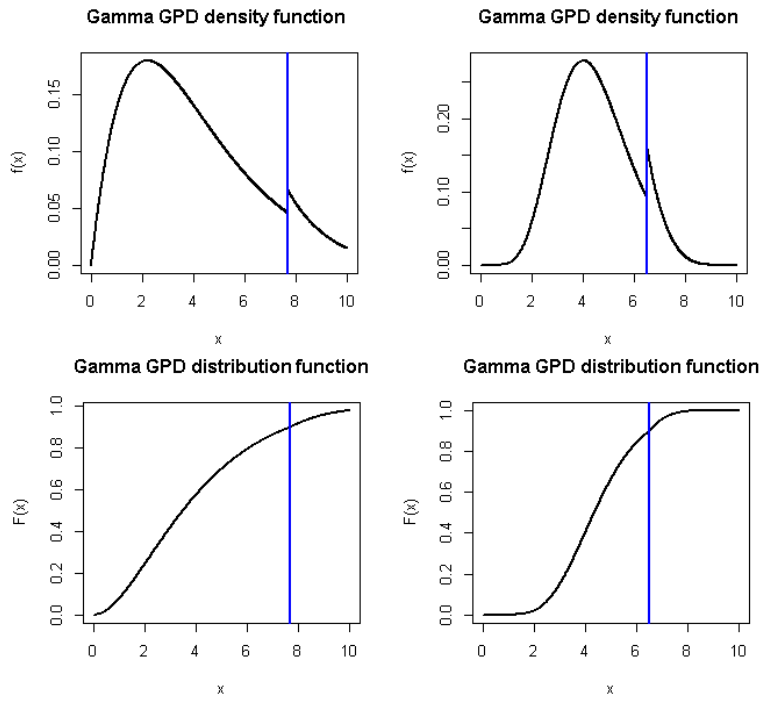

Figure 1: The density function of the parametric form GPD mixture model may have a discontinuity at the threshold. However, the corresponding distribution function (or CDF) is always continuous at the threshold. The vertical dashed line indicates the threshold.

```
dnormgpd(1.1, c(0.02, 0), 1, 1.28, 0.5173, -0.1489, phiu = TRUE)
[1] 0.2226535 0.2178522

pnormgpd(1.4, 0, 1,c(1.1,1.28), 0.5173, c(-0.1,-0.1489), phiu=0.1)
[1] 0.9449776 0.9210278

par(mfrow=c(2,2))
x = seq(0,10,0.01)
f = dgammagpd(x, 2.1, 2, 7.7, 1.52, -0.07, phiu = 0.1)
plot(x, f, type = 'l', lty = 1, lwd = 2, ylab = "f(x)", main = "Gamma GPD density function")
abline(v = 7.7, lty = 3, lwd = 2)

f = dgammagpd(x, 9, 0.5, 6.5, 0.62, -0.21, phiu = 0.1)
plot(x, f, type = 'l', lty = 1, lwd = 2, ylab = "f(x)", main = "Gamma GPD density function")
abline(v = 6.5, lty = 3, lwd = 2)

f = pgammagpd(x, 2.1, 2, 7.7, 1.52, -0.07, phiu = 0.1)
plot(x, f, type = 'l', lty = 1, lwd = 2, ylab = "F(x)", main = "Gamma GPD distribution function")
abline(v = 7.7, lty = 3, lwd = 2)

f = pgammagpd(x, 9, 0.5, 6.5, 0.62, -0.21, phiu = 0.1)
plot(x, f, type = 'l', lty = 1, lwd = 2, ylab = "F(x)", main = "Gamma GPD distribution function")
abline(v = 6.5, lty = 3, lwd = 2)
```

Likelihood Function

The negative log likelihood function is pre-fixed by `nl` with a distribution name (e.g. `nlweibullgpd` and and the corresponding log likelihood functions prefixed by `l` (e.g.`lweibullgpd`). It is worthwhile to notice that these likelihood functions are implemented allowing the two approaches for the parameter ϕ_u, where phiu=FALSE gives uses the sample proportion estimate

and phiu=TRUE gives the bulk model estimate.

An example of the syntax for the negative log likelihood function is explicitly designed for use in the optimisation, so all the parameters are input as a vector (except ϕ_u) and is just a wrapper for the likelihood function itself::

- `nlgammagpd(pvector, x, phiu = TRUE, finitelik = FALSE)`, `pvector` has 5 arguments with gamma shape, gamma scale, threshold, GPD scale and GPD shape respectively.

The log likelihood function is pre-fixed by `l` with a distribution name (e.g. `lweibullgpd` and `lnormgpd`). The log likelihood function syntax is similar to that of the usual distribution functions::

- `lgammagpd(x, gshape = 1, gscale = 1, u = qgamma(0.9, gshape, 1/gscale), sigmau = sqrt(gshape) * gscale, xi = 0, phiu = TRUE, log = TRUE)`

Now `x` is the data and `phiu` is the logical input indicating either parameterised approach or bulk model based approach. If let `phiu = TRUE`, the likelihood function will apply the bulk model based approach. If `phiu = FALSE`, the likelihood function will adopt the parameterised approach, and estimate `phiu` using the sample proportion above the threshold.

Fitting Function

The maximum likelihood based fitting function is pre-fixed by `f` with a distribution name (e.g. `fweibullgpd`). An example of the syntax for the MLE fitting function of the mixture model:

- `fgammagpd(x, phiu = TRUE, pvector = NULL, std.err = TRUE, method = "BFGS", control = list(maxit = 10000), finitelik = TRUE, ...)`,

The `pvector` is a vector of initial values of the mixture model and when `pvector = NULL` then the default initial values are used. `x` is a vector of sample data. The default value for `phiu = TRUE` so that the tail fraction is specified by gamma distribution $\phi_u = 1 - H(.|\theta)$, where $H(.|\theta)$ could be parametric distribution function. When `phiu = FALSE` then the tail fraction is treated as an extra parameter estimated using the MLE which is the sample proportion above the threshold. `std.err` is a logical variable, if `std.err = TRUE`, then the standard errors of the model parameters are calculated. `finitelik` is a logical variable, if `finitelik = TRUE`, then log-likelihood returns finite value for invalid parameters. The default optimisation algorithm is "BFGS", which requires a finite negative log-likelihood function evaluation `finitelik = TRUE`. For invalid parameters, a zero likelihood is replaced with `exp(-1e6)`. The "BFGS" optimisation algorithms require finite values for likelihood, so any user input for `finitelik` will be overridden and set to `finitelik = TRUE` if either of these optimisation methods is chosen. The `method`, `control` and `...` are optional inputs passed to `optim`.

```
x = rgamma(1000, shape = 2)
xx = seq(-1, 10, 0.01)
y = dgamma(xx, shape = 2)
par(xaxs="i", yaxs="i")

# Bulk model base tail fraction
fit = fgammagpd(x, phiu = TRUE, std.err = FALSE)
hist(x, breaks = 100, freq = FALSE, xlim = c(0, 10))
lines(xx, y)
lines(xx, dgammagpd(xx, gshape = fit$gshape, gscale = fit$gscale, u = fit$u,
   sigmau = fit$sigmau, xi = fit$xi, phiu = TRUE), col="red")
abline(v = fit$u)

# Parameterised tail fraction
fit2 = fgammagpd(x, phiu = FALSE, std.err = FALSE)
plot(xx, y, type = "l")
lines(xx, dgammagpd(xx, gshape = fit$gshape, gscale = fit$gscale, u = fit$u,
  sigmau = fit$sigmau, xi = fit$xi, phiu = TRUE), col="red")
lines(xx, dgammagpd(xx, gshape = fit2$gshape, gscale = fit2$gscale, u = fit2$u,
   sigmau = fit2$sigmau, xi = fit2$xi, phiu = fit2$phiu), col="blue")
abline(v = fit$u, col = "red")
abline(v = fit2$u, col = "blue")
legend("topright", c("True Density","Bulk T. F","Parameterised T. F"),
   col=c("black", "red", "blue"), lty = 1, cex= 0.8)
```

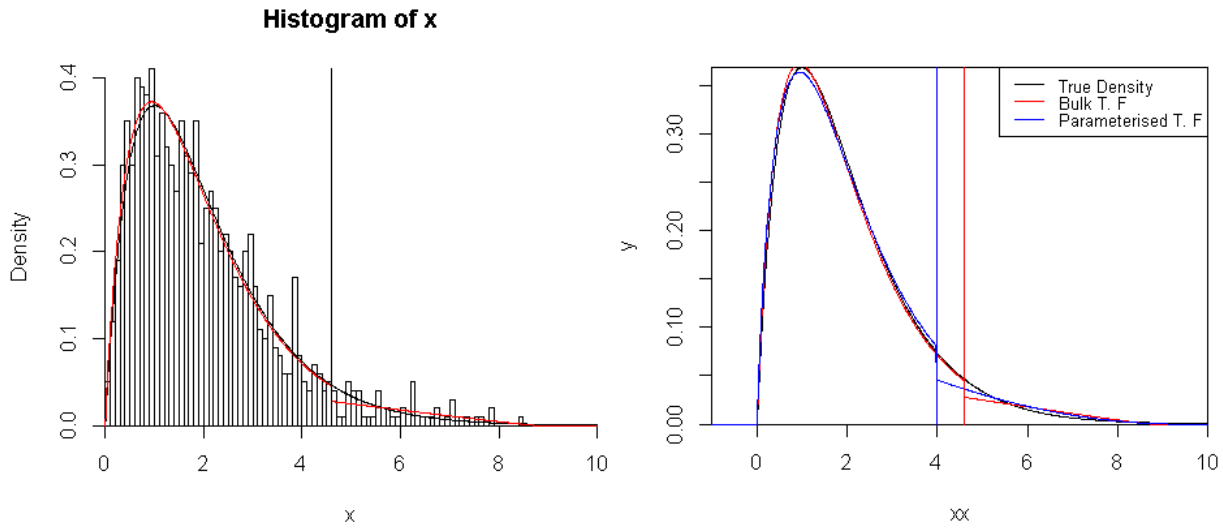

Figure 2: The plots from gamma MLE fitting examples.

3.2 Gamma/Normal/Weibull/Log-Normal/Beta GPD with Single Continuity Constraint Mixture Model

The continuity constraint is achieved by equating the bulk distribution and tail model at the threshold and replacing the GPD scale parameter in terms of other parameters. For the mixture extreme value models, the postfixes `con` means a continuity constraint at the threshold. For example, the Weibull distribution is defined as `weibull` in `R`, so the quantile function for Weibull GPD model with continuity constraint is therefore abbreviated as `qweibullgpdcon`.

In the Figure 3, the Weibull shape parameter is 2, the scale parameter is 0.5, the threshold is

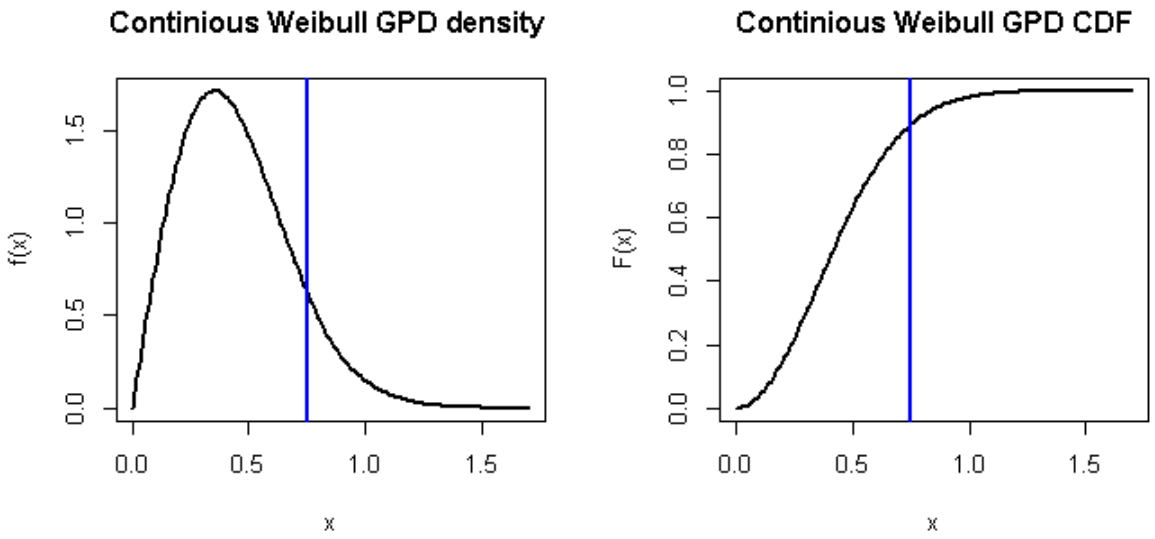

Figure 3: Example of density function and distribution function for continuous Weibull GPD distribution, where the blue line indicates the threshold.

fixed at 0.75 and GPD shape is -0.13. It can be seen that not only the density function for this mixture model is continuous at the threshold and distribution function will be continuous in first first derivative.

Base Function

The base functions of the parametric form continuous mixture model have very similar set up with the functions of parametric form mixture model, so will not be demonstrated again.

Examples of the syntax of the density function of parametric form continuous mixture model:

- `dgammagpdcon(x, gshape = 1, gscale = 1, u = qgamma(0.9, gshape, gscale), xi = 0, phiu = TRUE, log = FALSE)`, As usual, `phiu` can represent either bulk model based approach or parameterised approach.

Fitting and Likelihood Function

The fitting and likelihood functions of the continuity constraint mixture model has very similar set up with the fitting function of parametric form mixture model. Hence, the basic use of these functions will not be demonstrated again.

3.3 GPD-Normal-GPD (GNG) Model

Zhao et al. (2010) proposed a two tail mixture model, where the normal distribution describes the bulk and the GPD has been fitted to both tails separately. The parameter vector is $\Theta = (u_l, \sigma_{u_l}, \xi_l, \mu, \beta, u_r, \sigma_{u_r}, \xi_r)$.

Bulk Model Based Tail Fraction Approach

The distribution function is defined as:

$$F(x|\Theta) = \begin{cases} \phi_{u_l}\left\{1 - G(-x| - u_l, \sigma_{u_l}, \xi_l)\right\} & x < u_l, \\ H(x|\mu, \beta) & u_l \leq x \leq u_r, \\ (1 - \phi_{u_r}) + \phi_{u_r} G(x|u_r, \sigma_{u_r}, \xi_r) & x > u_r, \end{cases}$$

where $\phi_{u_l} = H(u_l|\mu, \beta)$ and $\phi_{u_r} = 1 - H(u_r|\mu, \beta)$ and $H(.|\mu, \beta)$ is the normal distribution function with mean μ and standard deviation β. $G(.| - u_l, \sigma_{u_l}, \xi_l)$ and $G(.|u_r, \sigma_{u_r}, \xi_r,)$ are GPD distribution functions for lower tail and upper tail.

Parameterised Tail Fraction Approach

It is possible to define $\phi_{u_l} = P(X < u_l)$ and $\phi_{u_r} = P(X > u_r)$. The corresponding distribution function is given by:

$$F(x|\Theta) = \begin{cases} \phi_{u_l}\left\{1 - G(-x| - u_l, \sigma_{u_l}, \xi_l)\right\} & x < u_l, \\ \phi_{u_l} + (1 - \phi_{u_l} - \phi_{u_r})\dfrac{H(x|\mu, \beta) - H(u_l|\mu, \beta)}{H(u_r|\mu, \beta) - H(u_l|\mu, \beta)} & u_l \leq x \leq u_r, \\ (1 - \phi_{u_r}) + \phi_{u_r} G(x|u_r, \sigma_{u_r}, \xi_r) & x > u_r, \end{cases}$$

Base function

The example of the density function of GNG model.

- `dgng(x, nmean = 0, nsd = 1,`
  `ul = qnorm(0.1, nmean, nsd), sigmaul = nsd, xil = 0, phiul = TRUE,`
  `ur = qnorm(0.9, nmean, nsd), sigmaur = nsd, xir = 0, phiur = TRUE, log = FALSE),`

The `xil` and `sigmaul` are GPD shape and scale parameter for the lower tail and the corresponding value `xir` and `sigmaur` for the upper tail.

```
f = dgng(x, 0, 1, -1.28, 0.42, 0.2, phiul = TRUE, 1.28, 0.42, 0.2, phiur = TRUE)
plot(x, f, ylim = c(0, 0.5), type = 'l', lty = 1, ylab = "Density", main = "GNG density function")
abline(v = -1.28, col = "blue")
abline(v =1.28, col = "blue")

f1 = dgng(x, 0, 1, -1.28, 0.6, -0.11, phiul = TRUE, 1.28, 0.6, -0.12, phiur = TRUE)
plot(x, f1, ylim = c(0, 0.5), type = 'l',lty = 1, ylab = "Density", main = "GNG density function")
abline(v = -1.28, col = "blue")
abline(v = 1.28, col = "blue")
```

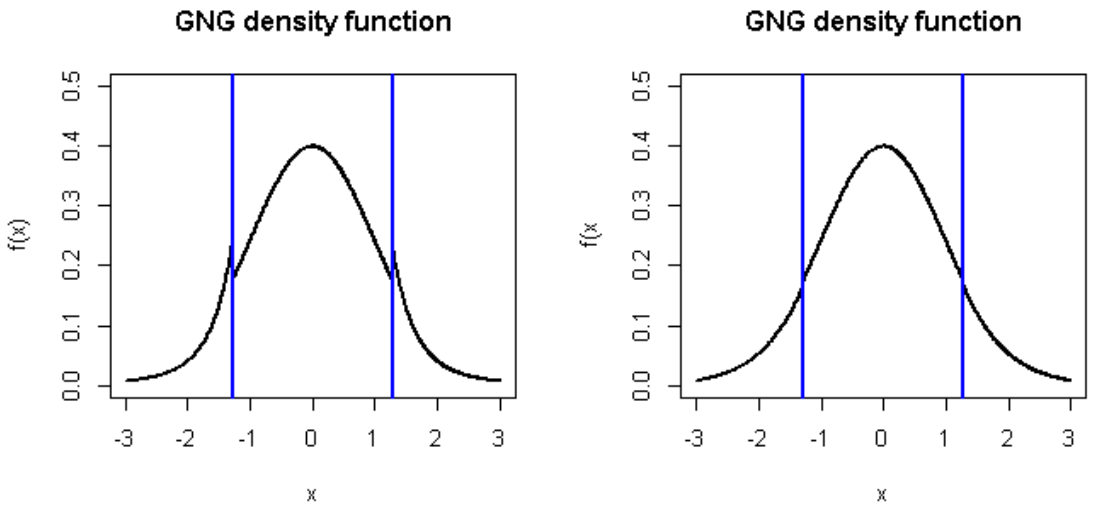

Figure 4: The density function of GNG model may have a discontinuity at either the lower threshold u_l or the upper threshold u_r.

Likelihood and Fitting Function

The fitting and likelihood functions are similar to those of the single tailed models, with the only difference that they now take parameters for both tails and the tail fraction can take either parameterisation for both tails.

Examples

```
x = rnorm(1000)
xx = seq(-5, 5, 0.01)
y = dnorm(xx)

# Bulk model base tail fraction
fit = fgng(x, phiul = TRUE, phiur = TRUE, std.err = FALSE)
hist(x, breaks = 100, freq = FALSE, xlim = c(-5, 5), main = "N(0, 1)")
lines(xx, dgng(xx, nmean = fit$nmean, nsd = fit$nsd,
   ul = fit$ul, sigmaul = fit$sigmaul, xil = fit$xil, phiul = TRUE,
   ur = fit$ur, sigmaur = fit$sigmaur, xir = fit$xir, phiur = TRUE), col="blue", lwd = 2)
abline(v = c(fit$ul, fit$ur))

# Two tail model is safest if bulk has lower tail which is not normal tail
x = rt(3000, df = 3)
xx = seq(-10, 10, 0.01)
y = dt(xx, df = 3)
hist(x, breaks = 100, freq = FALSE, xlim = c(-10, 10), ylim= c(0, 0.5), main = "T (df=3)")

fit = fnormgpd(x, phiu = FALSE, std.err = FALSE)
fit2 = fgng(x, phiul = FALSE, phiur = FALSE, std.err = FALSE)
lines(xx, dnormgpd(xx, nmean = fit$nmean, nsd = fit$nsd,
   u = fit$u, sigmau = fit$sigmau, xi = fit$xi, phiu = fit$phiu), col="red", lwd = 2)
lines(xx, dgng(xx, nmean = fit2$nmean, nsd = fit2$nsd,
             ul = fit2$ul, sigmaul = fit2$sigmaul, xil = fit2$xil, phiul = fit2$phiul,
             ur = fit2$ur, sigmaur = fit2$sigmaur, xir = fit2$xir, phiur = fit2$phiur), col="blue", lwd = 2)
```

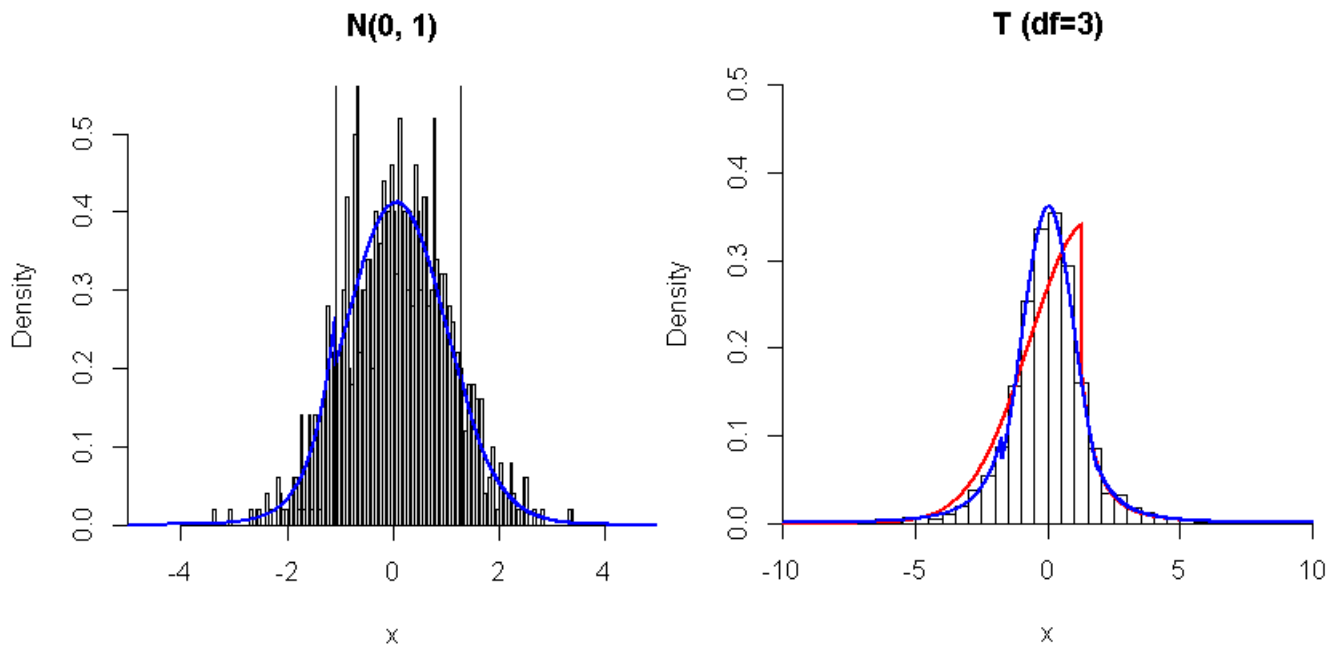

Figure 5: The plots from GNG fitting examples.

3.4 GNG with Single Continuity Constraint Model

The continuity constraints are achieved by equating the normal distribution with the GPD at the lower and upper threshold. The GPD scale parameters of the both tails are replaced by the other parameters. The function names are post-fixed by `dgngcon`, `pgngcon`, `qgngcon`, `rgngcon` to represent corresponding density, distribution, quantile and random number generation function respectively.

Base Function

These functions from the GNG continuous mixture model have very similar set up with the functions of GNG model. Hence, the details of these functions will not repeat again.

```
x = seq(-3, 3, 0.01)
f = dgngcon(x, 0, 1, -1.28, -0.23, phiul = TRUE, 1.28, 0.13, phiur = TRUE)
plot(x, f, type = 'l', lty = 1, ylim = c(0, 0.5), ylab = "f(x)", main = "Parameterised approach")
abline(v = -1.28, col = "darkgreen", lty = 3, lwd = 2)
abline(v = 1.28, col = "blue", lwd = 2)


f1 = dgngcon(x, 0, 1, -1.28, -0.23, phiul = 0.1, 1.28, 0.13, phiur = 0.1)
plot(x, f1, type = 'l', lty = 1, ylim = c(0, 0.5), ylab = "f(x)", main = "Bulk model approach")
abline(v = -1.28, col = "darkgreen", lty = 3, lwd = 2)
abline(v = 1.28, col = "blue", lwd = 2)
```

Figure 6 shows that the density function is not only continuous at the lower threshold but also at the upper threshold whatever the other parameters are.

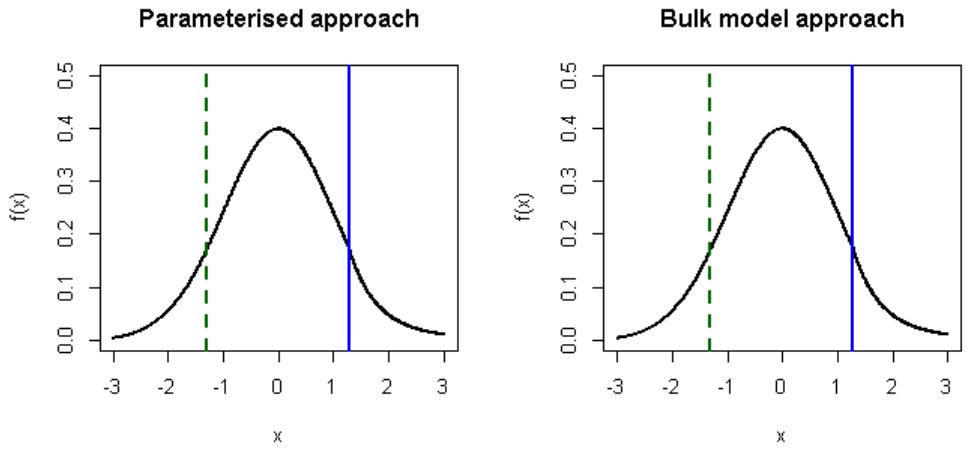

Figure 6: Examples of GNG continuous density function with asymmetric tail using two approaches

3.5 Dynamically Weighted Mixture Model

Frigessi et al. (2002) suggested a dynamically weighted mixture model by combining the Weibull for the bulk model with GPD for the tail model. Instead of explicitly defining the threshold, they use a Cauchy cumulative distribution function for the mixing function to make the transition between the bulk distribution and tail distribution.
The density function of the dynamic mixture model is given by:

$$l(x) = \frac{[1 - p(x|\theta)]\, f(x|\beta) + p(x|\theta) g(x|0, \sigma_u, \xi)}{Z(\theta, \beta, \sigma_u, \xi)},$$

where $f(x|\beta)$ denotes the Weibull density and $Z(\theta, \beta, \sigma_u, \xi)$ denotes the normalizing constant to make the density integrate to one. The mixing function $p(.|\theta)$ is a Cauchy distribution function with location parameter μ and scale parameter τ.

Base Function

This dynamic weighted mixture distribution is abbreviated as dwm and an example of the syntax is given by:

- `ddwm(x, wshape = 1, wscale = 1, cmu = 1, ctau = 1,`
  `sigmau = sqrt(wscale^2 * gamma(1 + 2/wshape) - (wscale * gamma(1 + 1/wshape))^2),`
  `xi = 0, log = FALSE),`

As before, the `wshape` and `wscale` are the Weibull shape and scale parameters. `cmu` and `ctau` are Cauchy distribution location and scale parameter. The quantile function `qdwm` requires the numerical integration of the distribution function `pdwm`, which very rarely fails due to integration difficulties. Further, the quantile function is not available in close form so require numerical inversion, which can also fail. The argument `qinit` is the initial quantile estimate, which can be used in case of failure. Hence, the parameter `qinit` is set up for user to choose some reasonable initial values for the quantile function.

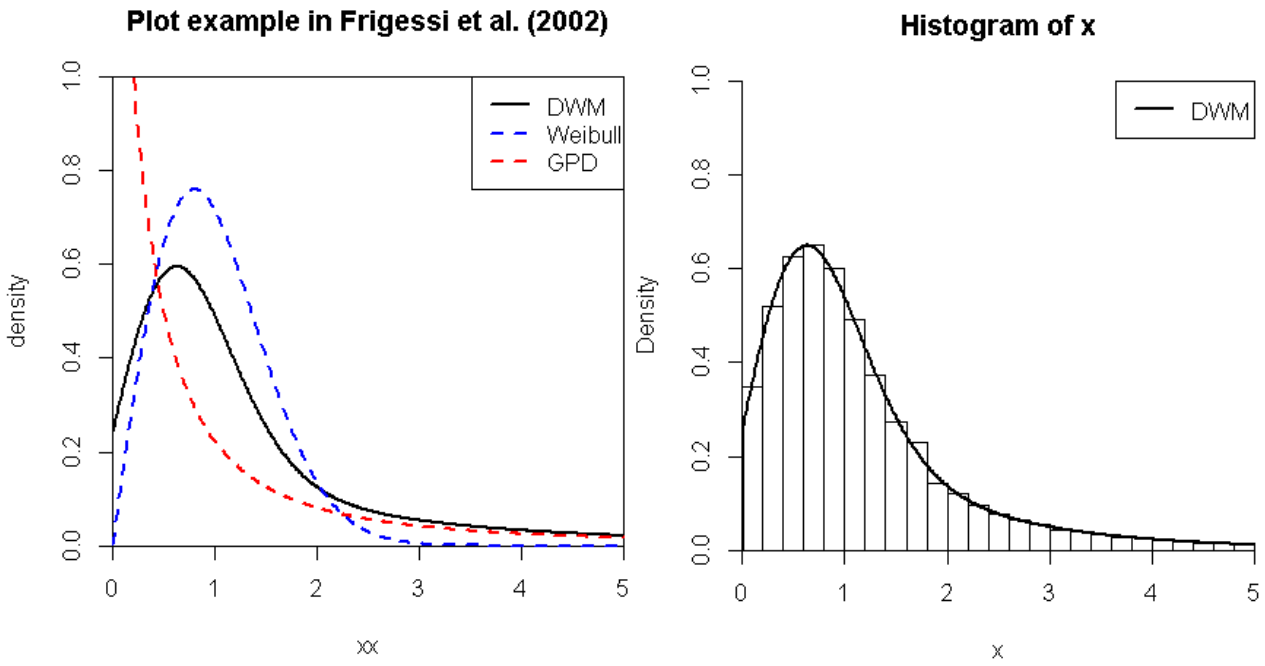

Figure 7: The plots from the dynamically weighted mixture model examples

```
xx = seq(0.001, 5, 0.01)
f = ddwm(xx, wshape = 2, wscale = 1/gamma(1.5), cmu = 1, ctau = 1, sigmau = 1, xi = 0.5)
plot(xx, f, ylim = c(0, 1), xlim = c(0, 5), type = 'l', lwd = 2,
    ylab = "density", main = "Plot example in Frigessi et al. (2002)")
lines(xx, dgpd(xx, xi = 1, sigmau = 0.5), col = "red", lty = 2, lwd = 2)
lines(xx, dweibull(xx, shape = 2, scale = 1/gamma(1.5)), col = "blue", lty = 2, lwd = 2)
legend('topright', c('DWM', 'Weibull', 'GPD'),
       col = c("black", "blue", "red"), lty = c(1, 2, 2), lwd = 2)

x = rdwm(10000, wshape = 2, wscale = 1/gamma(1.5), cmu = 1, ctau = 1, sigmau = 1, xi = 0.1)
xx = seq(0, 6, 0.01)
hist(x, freq = FALSE, breaks = 100, ylim = c(0, 1), xlim = c(0, 5))
lines(xx,ddwm(xx, wshape = 2, wscale = 1/gamma(1.5), cmu = 1, ctau = 1, sigmau = 1, xi = 0.1),
   lwd = 2, col = 'black')
legend('topright', 'DWM',
       col = "black", lty =1, lwd = 2)
```

Fitting and Likelihood Function

The fitting and likelihood functions are similar with other models so is not discussed here.

Examples

```
x = rweibull(1000, shape = 2, scale = 1.5)
xx = seq(0.1, 5, 0.01)
fit = fdwm(x, std.err = FALSE)
hist(x, 100, freq = FALSE, ylim = c(0, 1.3), main =" dwm example")

lines(xx, ddwm(xx, wshape = fit$wshape, wscale = fit$wscale, cmu = fit$cmu, ctau = fit$ctau,
              sigmau = fit$sigmau, xi = fit$xi),lwd=2)
lines(xx, dgpd(xx, sigmau = fit$sigmau, xi = fit$xi), col = "red", type = 'l', lwd = 2)
lines(xx, dweibull(xx, shape = fit$wshape, scale = fit$wscale), col = "blue", lwd = 2)
lines(xx, pcauchy(xx, location = fit$cmu, scale = fit$ctau), col = "green", lty = 2, lwd = 2)
legend("topleft", c("dwm", "GPD", "Weibull", "Cauchy"), col = c("black", "red", "blue", "green")
       ,lty = c(1, 1, 1, 2), lwd = c(2, 2, 2, 2))
```

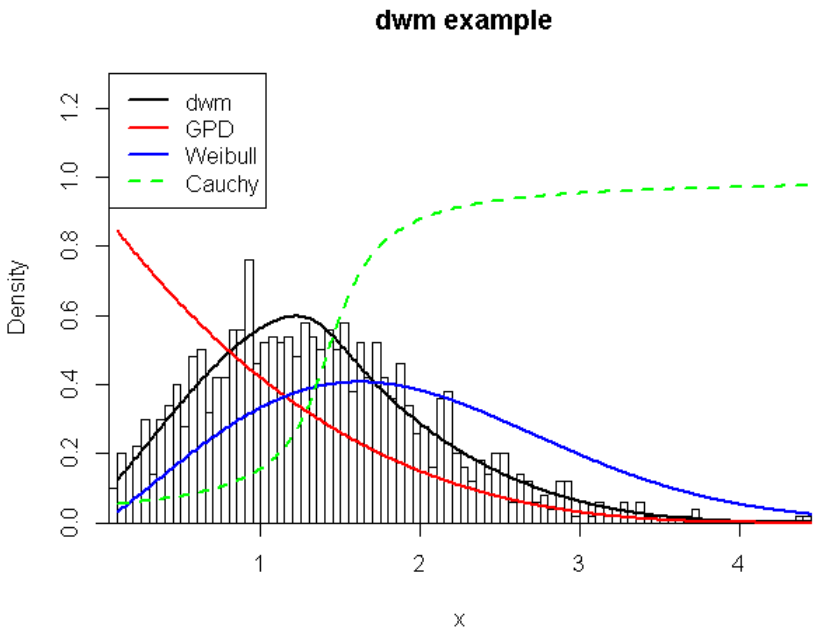

Figure 8: Example of simulated Weibull data with overlaid fitted dynamic weighted mixture model.

3.6 Hybrid Pareto Model

Carreau and Bengio (2008) proposed a hybrid Pareto model by splicing a normal distribution with GPD and set continuity constraint on the density and on its first derivative at the threshold. However, with a key difference to the usual mixture from Section 3.1 to Section 3.5 that it does not treat the GPD a conditional model, so the tail fraction scaling of the GPD $\phi_u = P(x > u)$ is not included. The unscaled GPD is simply spliced with the normal truncated at the threshold, so the density needs to be renormalized as it gets the contribution from the truncated normal upto the threshold and the entire unit integration of the GPD above the threshold.. The parameters have to adjust for the lack of tail fraction scaling. The GPD scale σ_u and threshold u are parameterised the other parameters to achieve two continuous constraints. Hence, they have reduced the initial 5 parameters to the normal mean μ and standard deviation β with the GPD shape ξ. The parameter vector is $\theta = (\mu, \beta, \xi)$. The hybrid Pareto mixture density is defined as:

$$f(x|\theta) = \begin{cases} \frac{1}{\tau} h(x|\mu, \beta) & x \leq u, \\ \\ \frac{1}{\tau} g(x|u, \sigma_u, \xi) & x > u. \end{cases}$$

μ is the normal mean and β is the standard deviation parameter. $h(.|\mu, \beta)$ is the normal density function and $g(.|u, \sigma_u, \xi)$ is the GPD density function. The τ is the usual normalising constant and $\tau = 1 + H(u|\mu, \beta)$, where the 1 comes from the integration of the unscaled GPD and second term is from the usual normal component.

The `condmixt` package written by one of the original authors of the hybrid Pareto model (Carreau and Bengio (2008)) also has similar functions for the hybrid Pareto `hpareto` and mixture of

hybrid Paretos `hparetomixt`, which are more flexible as they also permit the model to be truncated at zero.

Base Function

The hybrid Pareto distribution is abbreviated as hpd. Hence, the corresponding functions are defined as:

- `dhpd(x, nmean = 0, nsd = 1, xi = 0, log = FALSE)`,

Likelihood and Fitting Function

The negative log likelihood function of hpd is named as `nlhpd` and log likelihood function is named as `lhpd`. Similarly, the fitting function of hpd is named as `fhpd`. These functions have similar set up with other mixture models, so they are not detailed again.

Examples

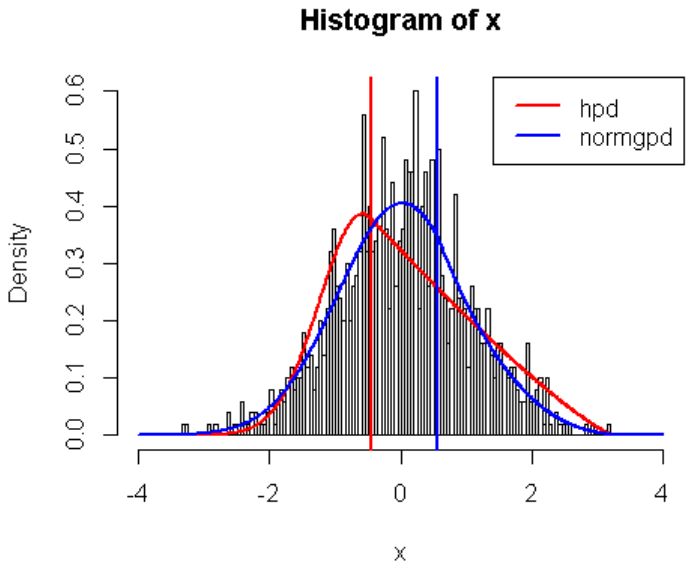

Figure 9: The plots from the hybrid Pareto model examples.

```
x = rnorm(1000)
xx = seq(-4, 4, 0.01)
y = dnorm(xx)

# Hybrid Pareto provides reasonable fit for asymmetric heavy tailed distribution
# but not for cases such as the normal distribution
fit = fhpd(x, std.err = FALSE)
hist(x, breaks = 100, freq = FALSE, xlim = c(-4, 4))
lines(xx, dhpd(xx, nmean = fit$nmean, nsd = fit$nsd,
xi = fit$xi), col="red", lwd = 2)
abline(v = fit$u, col="red", lwd = 2)

# Notice that if tail fraction is included a better fit is obtained
fit2 = fnormgpdcon(x, std.err = FALSE)
lines(xx, dnormgpdcon(xx, nmean = fit2$nmean, nsd = fit2$nsd, u = fit2$u,
xi = fit2$xi), col="blue", lwd = 2)
abline(v = fit2$u, col="blue", lwd = 2)
legend("topright", c("hpd", "normgpd" ), col = c("red", "blue")
       ,lty = c(1, 1), lwd = c(2, 2))
```

3.7 Hybrid Pareto with Single Continuous Constraint Model

A similar model can be suggested by splicing a normal distribution with GPD and set a continuity constraint on the density only not constraining it to also be continuous in first derivative at the threshold. The initial 5 parameters haven been reduced to threshold u, normal mean μ, standard deviation β and GPD shape ξ, which is consistent with previous continuity model. Again, note that the a key difference that ϕ_u is not included in this model, so there is no scaling of upper tail and bulk model. The parameter vector is $\theta = (u, \mu, \beta, \xi)$. The single continuous constraint hybrid Pareto mixture density is defined as:

$$f(x|\theta) = \begin{cases} \frac{1}{\tau} h(x|\mu, \beta) & x \leq u, \\ \\ \frac{1}{\tau} g(x|u, \sigma_u, \xi) & x > u, \end{cases}$$

where $h(.|\mu, \beta)$ is the normal density function and GPD density function is defined as $g(.|u, \sigma_u, \xi)$. The τ is the usual normalising constant and $\tau = 1 + H(u|\mu, \beta)$, where the 1 comes from the integration of the unscaled GPD and second term is from the usual normal component.

Base Function

The standard density, distribution , quantile, random number generation functions of the single continuous constraint hybrid Pareto model are similar with the original hybrid Pareto. The example of density function of single continuous constraint hybrid Pareto model:

- `dhpdcon(x, nmean = 0, nsd = 1, u = qnorm(0.9, nmean, nsd), xi = 0, log = FALSE)`

Likelihood and Fitting Function

An example of fitting the hybrid Pareto (with and without the constraint of continuous first derivative) and the normal with GPD tail with single continuity constraint is given below.

Examples

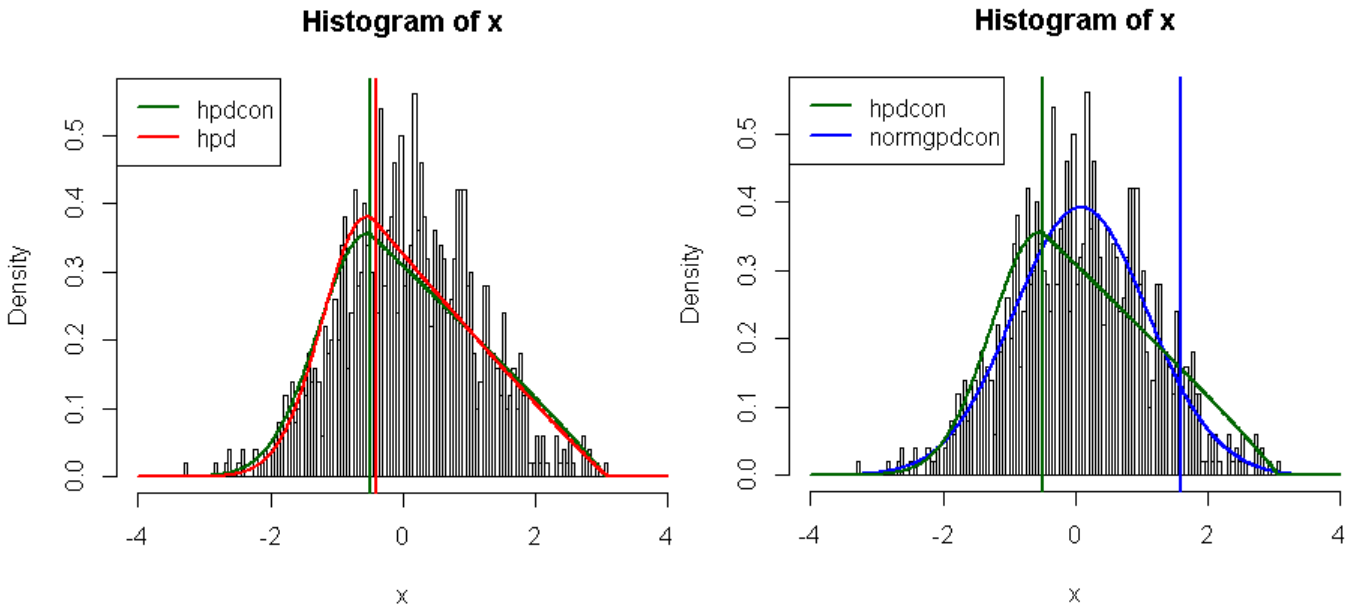

Figure 10: The plots from the examples.

```
x = rnorm(1000)
xx = seq(-4, 4, 0.01)
y = dnorm(xx)

# Two Hybrid Pareto models provide reasonable fit for asymmetric heavy tailed distribution
# but not for cases such as the normal distribution
fitcon = fhpdcon(x, std.err = FALSE)
fit = fhpd(x, std.err = FALSE)
hist(x, breaks = 100, freq = FALSE, xlim = c(-4, 4))
lines(xx, dhpdcon(xx, nmean = fitcon$nmean, nsd = fitcon$nsd, u = fitcon$u,
    xi = fitcon$xi), col="darkgreen", lwd = 2)
abline(v = fitcon$u, col="darkgreen", lwd = 2)
lines(xx, dhpd(xx, nmean = fit$nmean, nsd = fit$nsd, xi = fit$xi), col="red", lwd = 2)
abline(v = fit$u, col="red", lwd = 2)
legend("topleft", c("hpdcon", "hpd" ), col = c("darkgreen", "red")
       ,lty = c(1, 1), lwd = c(2, 2))

# Notice that if tail fraction is included a better fit is obtained
fit2 = fnormgpdcon(x, std.err = FALSE)
hist(x, breaks = 100, freq = FALSE, xlim = c(-4, 4))
lines(xx, dnormgpdcon(xx, nmean = fit2$nmean, nsd = fit2$nsd, u = fit2$u,
   xi = fit2$xi), col="blue", lwd = 2)
lines(xx, dhpdcon(xx, nmean = fitcon$nmean, nsd = fitcon$nsd, u = fitcon$u,
   xi = fitcon$xi), col="darkgreen", lwd = 2)
abline(v = fit2$u, col="blue", lwd = 2)
abline(v = fitcon$u, col="darkgreen", lwd = 2)
legend("topleft", c("hpdcon", "normgpdcon" ), col = c("darkgreen", "blue")
       ,lty = c(1, 1), lwd = c(2, 2))
```

4 Non-Parametric Mixture Models

MacDonald et al. (2011) and MacDonald et al. (2013) developed a non-parametric kernel density estimator based extreme value mixture models extending on that developed by Tancredi et al. (2006). The former has been implemented in the first version of this package.

4.1 Kernel GPD Model

MacDonald et al. (2011) constructed a standard kernel density estimator as the bulk model below the threshold with GPD for the tail model. The distribution function of standard kernel GPD mixture model is given by:

Bulk Model Based Tail Fraction Approach

$$F(x|X,\lambda,u,\sigma_u,\xi,\phi_u) = \begin{cases} H(x|X,\lambda) & x \leq u, \\ (1-\phi_u) + \phi_u \times G(x|u,\sigma_u,\xi) & x > u, \end{cases}$$

where $\phi_u = 1 - H(u|X,\lambda)$.

Parameterised Tail Fraction Approach

$$F(x|X,\lambda,u,\sigma_u,\xi,\phi_u) = \begin{cases} (1-\phi_u)\frac{H(x|X,\lambda)}{H(u|X,\lambda)} & x \leq u, \\ (1-\phi_u) + \phi_u \times G(x|u,\sigma_u,\xi) & x > u, \end{cases}$$

where $H(.|X,\lambda)$ is the distribution function of the kernel density estimator and $G(.|u,\sigma_u,\xi)$ is the distribution function of GPD. The traditional kernel density estimator is defined as:

$$h(x;\mathbf{X},\lambda) = \frac{1}{n\lambda}\sum_{i=1}^{n} K\left(\frac{x-x_i}{\lambda}\right),$$

where $K(.)$ is the kernel function and usually to be a symmetric and unimodal probability density function and lambda is the bandwidth parameter. The kernel function usually meets the following conditions: $K(x) \geq 0$ and $\int K(x)\,\mathrm{d}x = 1$.

Base Function

The density function of kernel GPD model is given by:

- `dkdengpd(x, kerncentres, lambda = NULL, u = as.vector(quantile(kerncentres, 0.9)), sigmau = sqrt(6*var(kerncentres))/pi, xi = 0, phiu = TRUE, log = FALSE)`

where `lambda` is the bandwidth parameter of the kernel density estimator and `kerncentres` is the sample of data. The `phiu` can be used for determining either bulk model based or parameterised approach.

Likelihood Function

The negative log likelihood and log likelihood functions are pre-fixed as usual:

- `nlkdengpd(pvector, x, phiu = TRUE, finitelik = FALSE)`

- `lkdengpd(x, lambda = NULL, u = 0, sigmau = 1, xi = 0, phiu = TRUE, log = TRUE)`

Fitting Function

Notice that the kernel centres are not needed in the likelihood functions as these are the data.

Two important practical issues arise with MLE for the kernel bandwidth:

- Cross-validation likelihood is needed for the kernel density estimator bandwidth parameter as the usual likelihood degenerates by Habbema et al. (1974) and Duin (1976). If the data has been heavily rounded, the bandwidth can be zero even if the using cross-validation likelihood. To overcome this issue an option to add a small jitter to the data has been included in the fitting inputs, using the `jitter` function, to remove the ties.
- For heavy tailed populations, the bandwidth is positively biased, giving a larger bandwidth. One solution to this problem is to splice the GPD to both the upper and lower tails (using the `gkg` function discussed below), as the bias comes from the lack of separation of the upper and lower order statistics in the two tails

The fitting function is given by:

- `fkdengpd(x, phiu = TRUE, pvector = NULL, add.jitter = FALSE, factor = 0.1, amount = NULL, std.err = TRUE, method = "BFGS", control = list(maxit = 10000), finitelik = TRUE, ...)`

`x` is a vector of sample data. `add.jitter` is a logical variable, if `add.jitter = TRUE`, then the jitter is needed for rounded data. `factor` and `amount` are the arguments passed to the `add.jitter`.

Examples

```
x = rnorm(1000, 0, 1)
fit = fkdengpd(x, phiu = FALSE, std.err = FALSE)
hist(x, 100, freq = FALSE, xlim = c(-4, 4))
xx = seq(-4, 4, 0.01)
lines(xx, dkdengpd(xx, x, fit$lambda, fit$u, fit$sigmau, fit$xi, fit$phiu), col="blue", lwd = 2)
abline(v = fit$u, col="blue", lwd = 2)
legend("topright", "kdengpd", col = "blue",
       ,lty = 1, lwd = 2)

# Try a bimodal data
x <- c(rnorm(1000,-1,0.7),rnorm(1000,3,1))

fit = fkdengpd(x, phiu = FALSE, std.err = FALSE)
hist(x, 100, freq = FALSE, xlim = c(-4, 6.5), main="Bimodal data example")
xx = seq(-5, 6, 0.001)
lines(xx, dkdengpd(xx, x, fit$lambda, fit$u, fit$sigmau, fit$xi, fit$phiu), col="blue", lwd = 2)
abline(v = fit$u, col="blue", lwd = 2)
legend("topright", "kdengpd", col = "blue",
       ,lty = 1, lwd = 2)
```

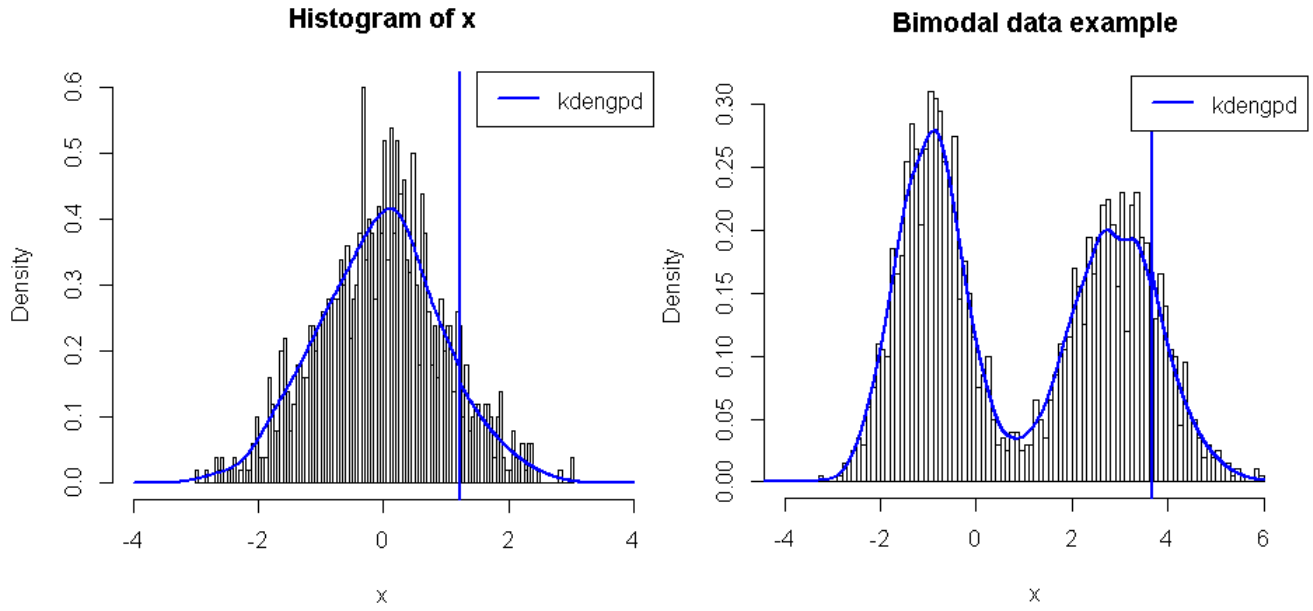

Figure 11: The plots from the examples.

4.2 Two Tailed Kernel GPD Model

MacDonald et al. (2011) introduced a two tailed mixture model by splicing the standard kernel density estimator with two extreme value tail models. This two-tailed extreme value mixture model also overcomes the inconsistency in the cross-validation likelihood estimation of the bandwidth for heavy tailed distributions, see MacDonald et al. (2011) for details.

The parameter vector is $\Theta = (X, \lambda, u_1, \sigma_{u_l}, \xi_l, \phi_{u_l}, u_2, \sigma_{u_r}, \xi_r, \phi_{u_r})$. The distribution function of two tailed mixture model is given by:

Parameterised Tail Fraction Approach

$$F(x|\Theta) = \begin{cases} \phi_{u_l}\left\{1 - G(-x| - u_l, \sigma_{u_l}, \xi_l)\right\} & x < u_l, \\ \phi_{u_l} + (1 - \phi_{u_l} - \phi_{u_r})\dfrac{H(x|X,\lambda) - H(u_l|X,\lambda)}{H(u_r|X,\lambda) - H(u_l|X,\lambda)} & u_l \le x \le u_r, \\ (1 - \phi_{u_r}) + \phi_{u_r} G(x|u_r, \sigma_{u_r}, \xi_r) & x > u_r, \end{cases}$$

where ϕ_{u_l} and ϕ_{u_r} are estimated as the sample proportions less than threshold u_l and above the threshold u_r respectively. $G(-x| - u_l, \sigma_{u_l}, \xi_l)$ is the unconditional GPD function for $x < u_l$ and $G(x|u_r, \sigma_{u_r}, \xi_r)$ is the unconditional GPD function for $x > u_r$.

Bulk Model Based Tail Fraction Approach

$$F(x|\Theta) = \begin{cases} \phi_{u_l}\left\{1 - G(-x| - u_l, \sigma_{u_l}, \xi_l)\right\} & x < u_l, \\ H(x|X,\lambda) & u_l \le x \le u_r, \\ (1 - \phi_{u_r}) + \phi_{u_r} G(x|u_r, \sigma_{u_r}, \xi_r) & x > u_r, \end{cases}$$

where ϕ_{u_l}=$H(u_l, X, \lambda)$ and $\phi_{u_r} = 1 - H(u_r, X, \lambda)$.

Base Function

The density function of two tailed kernel GPD model is defined as:

- `dgkg(x, kerncentres, lambda = NULL,`
  `ul = as.vector(quantile(kerncentres, 0.1)), sigmaul = sqrt(6*var(kerncentres))/pi,`
  `xil = 0, phiul = TRUE,`
  `ur = as.vector(quantile(kerncentres, 0.9)), sigmaur = sqrt(6*var(kerncentres))/pi,`
  `xir = 0, phiur = TRUE, log = FALSE),`

The basic use of the standard function will not repeated again.

Likelihood and Fitting Function

The likelihood and fitting functions are defined as usual:

- `nlgkg(pvector, x, phiul = TRUE, phiur = TRUE, finitelik = FALSE)`
- `lgkg(x, lambda = NULL,`
  `ul = as.vector(quantile(x, 0.1)), sigmaul = 1, xil = 0, phiul = TRUE,`
  `ur = as.vector(quantile(x, 0.9)), sigmaur = 1, xir = 0, phiur = TRUE, log = TRUE)`
- `fgkg(x, phiul = TRUE, phiur = TRUE, pvector = NULL, add.jitter = FALSE,`
  `factor = 0.1, amount = NULL, std.err = TRUE, method = "BFGS",`
  `control = list(maxit = 10000), finitelik = TRUE, ...)`

Examples

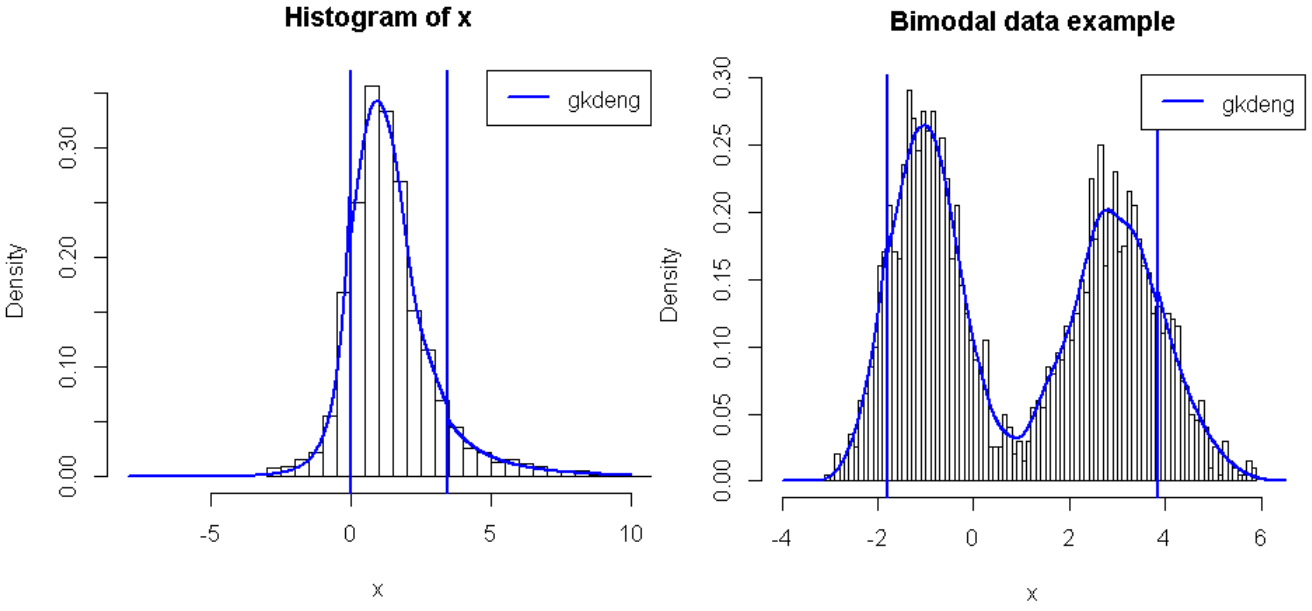

Figure 12: Distribution function plot using two approaches.

```
x = rt(1000, 3, 1)
fit = fgkg(x, phiul = FALSE, phiur = FALSE, std.err = FALSE)
hist(x, 100, freq = FALSE, xlim = c(-8, 10))
xx = seq(-8, 10, 0.01)
lines(xx, dgkg(xx, x, fit$lambda, fit$ul, fit$sigmaul, fit$xil, fit$phiul, fit$ur, fit$sigmaur, fit$xir, fit$phiur)
     , col="blue", lwd = 2)
abline(v = fit$ul, col="blue", lwd = 2)
abline(v = fit$ur, col="blue", lwd = 2)
legend("topright", "gkdeng", col = "blue",
       ,lty = 1, lwd = 2)

# Try a bimodal data
x <- c(rnorm(1000,-1,0.7),rnorm(1000,3,1))

fit = fgkg(x, phiul = FALSE, phiur = FALSE, std.err = FALSE)
hist(x, 100, freq = FALSE, xlim = c(-4, 6.5), main="Bimodal data example")
xx = seq(-4, 6.5, 0.01)
lines(xx, dgkg(xx, x, fit$lambda, fit$ul, fit$sigmaul, fit$xil, fit$phiul, fit$ur, fit$sigmaur, fit$xir, fit$phiur)
     , col="blue", lwd = 2)
abline(v = fit$ul, col="blue", lwd = 2)
abline(v = fit$ur, col="blue", lwd = 2)
legend("topright", "gkdeng", col = "blue",
       ,lty = 1, lwd = 2)
```

4.3 Boundary Corrected Kernel GPD Model

MacDonald et al. (2013) combine a boundary corrected kernel density estimator for bulk model sliced at the threshold with a point process representation of GPD tail model. The boundary correction kernel density estimator tries to reduce the inherent bias of the kernel density estimator. The distribution function of boundary corrected mixture model is given by:

Bulk Model Based Tail Fraction Approach

$$F(x|X, h_{BC}, u, \sigma_u, \xi, \phi_u) = \begin{cases} H_{BC}(x|X, \lambda_{BC}) & x \leq u, \\ (1 - \phi_u) + \phi_u \times G(x|u, \sigma_u, \xi) & x > u, \end{cases}$$

where $\phi_u = 1 - H_{BC}(u|X, h)$.

Parameterised Tail Fraction Approach

$$F(x|X, h_{BC}, u, \sigma_u, \xi, \phi_u) = \begin{cases} (1 - \phi_u)\frac{H_{BC}(x|X,\lambda_{BC})}{H_{BC}(u|X,\lambda_{BC})} & x \leq u, \\ (1 - \phi_u) + \phi_u \times G(x|u, \sigma_u, \xi) & x > u, \end{cases}$$

where $H_{BC}(x|X, \lambda_{BC})$ is the distribution function of the boundary correction kernel density estimator.

Base Function

The density functionis given by:

- `dbckdengpd(x, kerncentres, lambda = NULL, u = as.vector(quantile(kerncentres, 0.9)), sigmau = sqrt(6*var(kerncentres))/pi, xi = 0, phiu = TRUE, bcmethod = "simple", proper = TRUE, nn = "jf96", offset = 0, xmax = Inf, log = FALSE)`

The density, distribution, quantile function of boundary corrected kernel GPD model have very similar set up with the standard kernel GPD model. Hence, the basic use of these functions will not repeat here. See help(dbckdengpd) for details of boundary correction methods that have been implemented.

Likelihood and Fitting Function

The likelihood functions and fitting function are very similar with other mixture models.

- `nlbckdengpd(pvector, x, phiu = TRUE, finitelik = FALSE)`
- `lbckdengpd(x, lambda = NULL, u = 0, sigmau = 1, xi = 0, phiu = TRUE, log = TRUE)`
- `fbckdengpd(x, phiu = TRUE, pvector = NULL, add.jitter = FALSE, factor = 0.1, amount = NULL, bcmethod = "simple", proper = TRUE, nn = "jf96", offset = 0, xmax = Inf, std.err = TRUE, method = "BFGS", control = list(maxit = 10000), finitelik = TRUE, ...)`

Examples

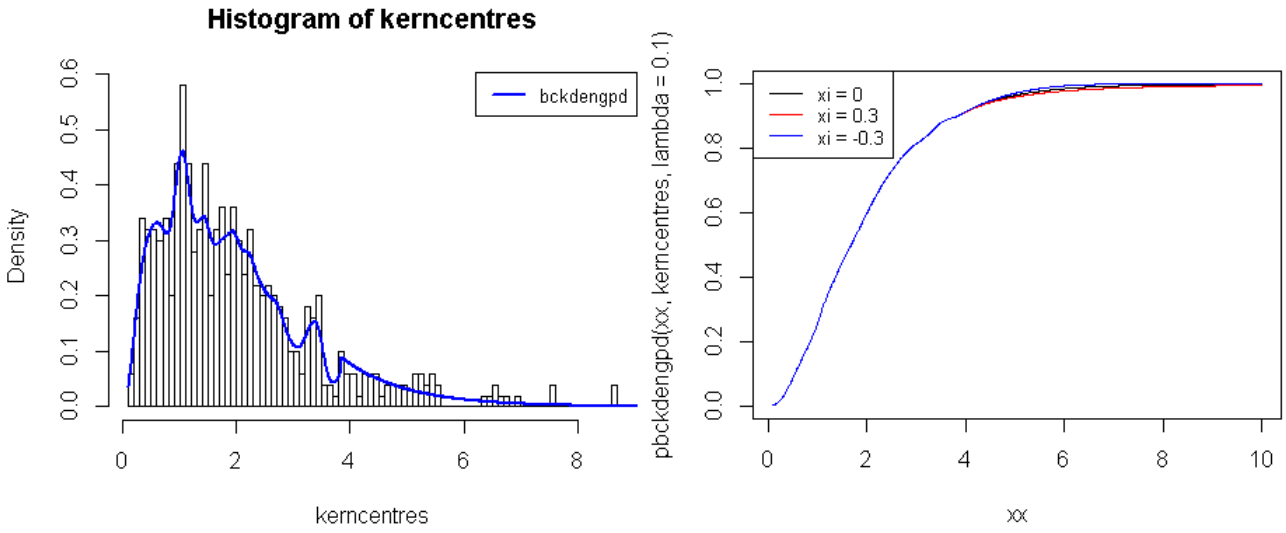

Figure 13: The plot from the examples.

```
kerncentres=rgamma(500, 2, 1)
xx = seq(0.1, 10, 0.01)
hist(kerncentres, breaks = 100, freq = FALSE)
lines(xx, dbckdengpd(xx, kerncentres, lambda = 0.1), col="blue", lwd = 2)
legend("topright", "bckdengpd",
       col="blue", lty = 1, cex = 0.8, lwd = 2)

plot(xx, pbckdengpd(xx, kerncentres, lambda = 0.1), type = "l")
lines(xx, pbckdengpd(xx, kerncentres, lambda = 0.1, xi = 0.3), col = "red")
lines(xx, pbckdengpd(xx, kerncentres, lambda = 0.1, xi = -0.3), col = "blue")
legend("topleft", paste("xi =",c(0, 0.3, -0.3)),
       col=c("black", "red", "blue"), lty = 1, cex = 0.8)
```

Bibliography

Behrens, C. N., H. F. Lopes, and D. Gamerman (2004). Bayesian analysis of extreme events with threshold estimation. *Statistical Modelling 4*(3), 227–244.

Carreau, J. and Y. Bengio (2008). A hybrid Pareto model for asymmetric fat-tailed data: the univariate case. *Extremes 12*(1), 53–76.

Coles, S. (2001). *An Introduction to Statistical Modeling of Extreme Values*. Springer Series in Statistics. Springer-Verlag:London.

Duin, R. (1976). On the choice of smoothing parameters for Parzen estimators of probability density functions. *IEEE Transactions on Computers 25*(11), 1175–1179.

Frigessi, A., O. Haug, and H. Rue (2002). A dynamic mixture model for unsupervised tail estimation without threshold selection. *Extremes 5*(3), 219–235.

Habbema, J., H. J, and v. d. B. K (1974). *A stepwise discriminant analysis program using density estimation*. Number 1. Physica-Verlag: Vienna.

MacDonald, A. E., C. J. Scarrott, and D. S. Lee (2013). Boundary correction, consistency and robustness of kernel densities using extreme value theory. *Submitted*.

MacDonald, A. E., C. J. Scarrott, D. S. Lee, B. Darlow, M. Reale, and G. Russell (2011). A flexible extreme value mixture model. *Computational Statistics and Data Analysis 55*(6), 2137–2157.

Nascimento, F. F., D. Gamerman, and H. F. Lopes (2011). A semiparametric Bayesian approach to extreme value estimation. *Statistics and Computing 22*(2), 661–675.

Scarrott, C. J. and A. E. MacDonald (2012). A review of extreme value threshold estimation and uncertainty quantification. *REVSTAT Statistical Journal 10*(1), 33–60.

Tancredi, A., C. Anderson, and A. O'Hagan (2006). Accounting for threshold uncertainty in extreme value estimation. *Extremes 9*, 87–106.

Zhao, X., C. J. Scarrott, L. Oxley, and M. Reale (2010). Extreme value modelling for forecasting market crisis impacts. *Applied Financial Economics 20*(1-2), 63–72.