evmix: An R package for Extreme Value Mixture Modelling, Threshold Estimation and Boundary Corrected Kernel Density Estimation

Yang Hu
University of Canterbury, New Zealand

Carl Scarrott
University of Canterbury, New Zealand

Abstract

evmix is an R package (R Core Team 2013) with two interlinked toolsets: i) for extreme value modeling and ii) kernel density estimation. A key issue in univariate extreme value modelling is the choice of threshold beyond which the asymptotically motivated extreme value models provide a suitable tail approximation. The package provides diagnostics plots to aid threshold choice, along with functions for fitting extreme value mixture models which permit objective threshold estimation and uncertainty quantification. The implemented mixture models cover the spectrum of parametric, semiparametric and nonparametric estimators to describe the distribution below the threshold.

Kernel density estimators using a range of potential kernels are provided, including cross-validation maximum likelihood inference for the bandwidth. A key contribution over existing kernel smoothing packages in R is that a wide range of boundary corrected kernel density estimators are provided, which cope with populations with lower and/or upper bounded support.

The full complement of density, distribution, quantile and random number generation functions are provided along with parameter estimation by likelihood inference and standard model diagnostics, for both the mixture models and kernel density estimators. This paper describes the key extreme value mixture models and boundary corrected kernel density estimators and demonstrate how they are implemented in the package.

Keywords: extreme value mixture model, threshold estimation, boundary corrected kernel density estimation.

1. Introduction

Extreme value theory is used to derive asymptotically justified models for the tails of distributions, see Coles (2001) for an introduction. A classic asymptotically motivated model for the exceedances of a suitably high threshold u is the generalised Pareto distribution (GPD). Suppose the random variable variable for an exceedances X of a threshold u follows a GPD, parameterised by scale $\sigma_u > 0$ and shape ξ, with a cumulative distribution function given by:

$$G(x|u,\sigma_u,\xi) = \Pr(X \le x|X > u) = \begin{cases} 1 - \left[1 + \xi\left(\dfrac{x-u}{\sigma_u}\right)\right]_+^{-1/\xi}, & \xi \neq 0, \\ 1 - \exp\left[-\left(\dfrac{x-u}{\sigma_u}\right)\right]_+, & \xi = 0, \end{cases} \quad (1)$$

where $x_+ = \max(x, 0)$. When $\xi < 0$ there is an upper end point so that $u < x < u - \sigma_u/\xi$. In this formulation the threshold is sometimes described as the location parameter. The GPD is easily reformulated to describe excesses above the threshold $Y = X - u$ by replacing the numerator by $y = x - u$ and so $\Pr(Y < y|Y > 0) = G(y|0, \sigma_u, \xi)$. In this case the special case of $\xi = 0$, defined by continuity in the limit $\xi \to 0$, reduces to the usual exponential tail. In practice, the GPD is applied as an approximation to the upper tail of the population distribution above a sufficiently high threshold.

Implicitly underlying the GPD is a third parameter $\phi_u = \Pr(X > u)$, the threshold exceedance probability. We refer to this parameter as the "tail fraction" required in calculating quantities like the unconditional survival probability:

$$\Pr(X > x) = \phi_u \left[1 - \Pr(X \leq x|X > u)\right]. \tag{2}$$

This representation is often referred to as a Poisson-GPD, as it explicitly accounts for the constant Poisson rate of exceedance occurrences. Relatively mild conditions are required for the GPD to be a suitable limiting tail excess model, see Coles (2001) for details.

The first stage in GPD tail modelling is to choose the threshold u, which is essentially a bias against variance trade-off. A sufficiently high threshold is needed for the asymptotics underlying the GPD to provide a reliable model thus reducing bias, but also increasing the variance of parameters estimates due to the reduced sample size. By contrast, too low a threshold may mean the GPD is not a good tail approximation, leading to bias, but providing a larger sample size for inference thus reducing the variance.

Scarrott and MacDonald (2012) provides a reasonably comprehensive review of threshold estimation approaches, from which this package was conceived. Direct inference for the threshold is not straightforward due to the varying sample size. Traditionally, graphical diagnostics evaluating aspects of the model fit were used to choose a threshold, which was then fixed and assumed known and so the uncertainty associated with it were ignored. Some commonly used diagnostics for the so called "**fixed threshold approach**" are implemented in the package and described in Section 2 below. Over the last decade there has been increasing interest in extreme value mixture models which combine a tail model above the threshold with a suitable model below the threshold, for which standard inference approaches can be applied as the sample size does not vary. Most of these mixture models treat the threshold as a parameter to be estimated and thus the associated uncertainty on tail inferences can be directly accounted for. The majority of the mixture models in the literature have been implemented in the **evmix** package (Hu and Scarrott 2013), as detailed in Section 3.2.

A particularly flexible type of extreme value mixture models uses a nonparametric density estimator for the below the threshold, following Tancredi, Anderson, and O'Hagan (2006), MacDonald, Scarrott, Lee, Darlow, Reale, and Russell (2011) and MacDonald, Scarrott, and Lee (2013). Hence, a wide range of kernel density estimation approaches have also been implemented in the package. These can be used as standalone functionality or as part of an extreme value mixture model. Standard kernel density estimation with a constant bandwidth are described in Section 7, followed by a range of boundary corrected kernel density estimation approaches in Section 8. The extreme value mixture models combining the kernel density estimator with GPD tail model are described in Section 9.

2. Graphical diagnostics for threshold choice

The GPD exhibits some threshold stability properties. Suppose the excesses $X - u$ above u follow a GPD(σ_u, ξ) then the expected value of the excesses $X - v$ for all higher thresholds $v > u$ follow a GPD(σ_v, ξ). The shape parameter is invariant to the threshold, but the scale parameter is a linear function of the threshold $\sigma_v = \sigma_u + \xi(v - u)$. A common diagnostic is the threshold stability plot where estimated shape parameters (including pointwise uncertainty intervals) are plotted against a range of possible thresholds. The threshold is chosen as the lowest possible value such that the shape parameter is stable for all higher values, once sample uncertainty is taken in account. A similar plot of modified scale parameter estimates given by $\hat{\sigma}_v^* = \hat{\sigma}_v - \xi v$ is occasionally considered, although the correlation between the scale and shape parameter mean there is often little extra information.

The mean residual life (MRL) function is also insightful as a threshold choice diagnostic for the GPD. The mean residual life, or equivalently "mean excess", is then:

$$E(X - v|X > v) = \frac{\sigma_u + \xi(v - u)}{1 - \xi}, \text{ for } \xi < 1 \tag{3}$$

and is infinite for $\xi \geq 1$. The MRL is linear in the excess of the higher threshold $v - u$ above the suitable threshold u with gradient $\xi/(1 - \xi)$ and intercept $\sigma_u/(1 - \xi)$, once the sample uncertainty is accounted for. The sample mean excess is plotted against potential thresholds (including pointwise uncertainty intervals) to give the sample mean residual life plot.

Examples of these diagnostic plots are given in Figure 1 for the Fort Collins precipitation data which is available in the **extRemes** package (Gilleland and Katz 2011). The estimates and 95% confidence intervals are shown by solid and dashed lines respectively, with Wald approximation based sampling density estimates as the background greyscale image. Three potential thresholds are considered using the `try.thresh` argument. The threshold of $u = 0.395$ was recommended by Gilleland and Katz (2005), with both 0.85 and 1.2 as viable alternatives using the above guidelines as the GPD fits are increasingly consistent with the uncertainty estimates at higher thresholds.

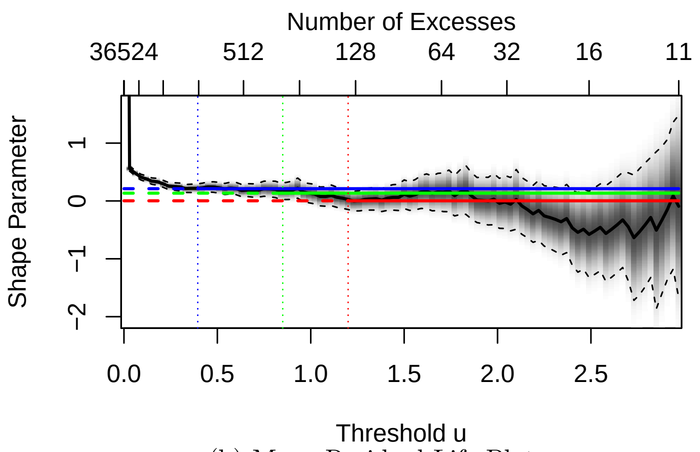
```
R> library("evmix")
R> data("Fort", package = "extRemes")
R> mrlplot(Fort[,6], try.thresh = c(0.395, 0.85, 1.2), legend = NULL)

R> tshapeplot(Fort[,6], try.thresh = c(0.395, 0.85, 1.2), legend = NULL)
```

In this case there is clearly large uncertainty in the threshold choice and the resultant quantile estimates as the implied shape parameters are very different, which is ignored in the "fixed threshold approach". Many authors (e.g., Coles (2001)) have commented on the subjectivity and substantial experience needed in interpreting these diagnostics, thus leading to the development of extreme value mixture models which potentially provide a more objective threshold estimate and can account for the resultant uncertainty.

3. Extreme value mixture models

A simple form of extreme value mixture model proposed by Behrens, Lopes, and Gamerman (2004) is depicted in Figure 2 which splices together a parametric model upto the threshold

(a) Shape Parameter Threshold Stability Plot



(b) Mean Residual Life Plot

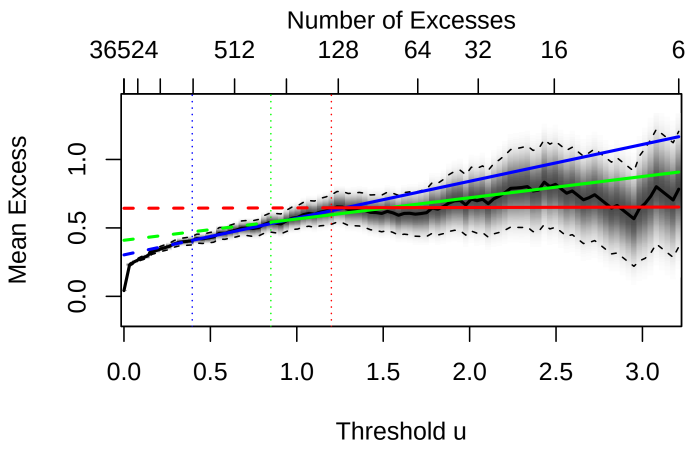


Figure 1: Shape parameter threshold stability plot and mean residual life plot for Fort Collins precipitation data. Estimate shape parameters and fitted MRL functions for three potential thresholds $u = 0.395, 0.85$ and 1.2.

with the usual GPD for the upper tail. We refer to the former as the "**bulk model**" and the latter as the "**tail model**". The cumulative distribution function (cdf) of the form:

$$F(x|\boldsymbol{\theta}, u, \sigma_u, \xi) = \begin{cases} H(x|\boldsymbol{\theta}) & \text{for } x \leq u, \\ H(u|\boldsymbol{\theta}) + [1 - H(u|\boldsymbol{\theta})]G(x|u, \sigma_u, \xi) & \text{for } x > u, \end{cases} \tag{4}$$

where $H(.)$ is the bulk model cdf with parameter vector $\boldsymbol{\theta}$.

In this definition, the tail fraction ϕ_u from the classical unconditional tail modelling approach in Equation 2 is replaced by the survival probability of the bulk model assuming it continues above threshold, i.e., $\phi_u = 1 - H(u|\theta)$. Hence, we will refer to this approach as the "**bulk model based tail fraction**" approach. Essentially, the tail fraction is borrowing information from the data in the bulk. However, it also exposes the tail estimation to model misspecification of the bulk and may not perform well if the bulk model's tail behaviour is very different to that of the population upper tail, which is to be discussed in Section 10.

MacDonald *et al.* (2011) consider a slightly more general form:

$$F(x|\theta, u, \sigma_u, \xi, \phi_u) = \begin{cases} \frac{(1-\phi_u)}{H(u|\boldsymbol{\theta})} H(x|\boldsymbol{\theta}) & \text{for } x \le u, \\ (1-\phi_u) + \phi_u G(x|u, \sigma_u, \xi) & \text{for } x > u, \end{cases} \tag{5}$$

which is closer to classical tail modelling approach in Equation 2, which has an explicit extra parameter for the tail fraction ϕ_u. Hu (2013) has shown this approach that the tail estimates are more robust to misspecification of the bulk model component. We refer to this as the "**parameterised tail fraction**" approach. It should be clear that either specification gives a proper density and that the parameterised tail fraction approach includes the bulk model based tail fraction as a special case, where $\phi_u = 1 - H(u|\boldsymbol{\theta})$.

```
R> x = seq(-5, 5, 0.001)
R> y = dnormgpd(x, u = 1.5, sigmau = 0.4)
R> plot(x, y, xlab = "x", ylab = "Density f(x)", type = "l")
R> abline(v = 1.5, lty = 2)
R> y = pnormgpd(x, u = 1.5, sigmau = 0.4)
R> plot(x, y, xlab = "x", ylab = "Distribution F(x)", type = "l")
R> abline(v = 1.5, lty = 2)
```

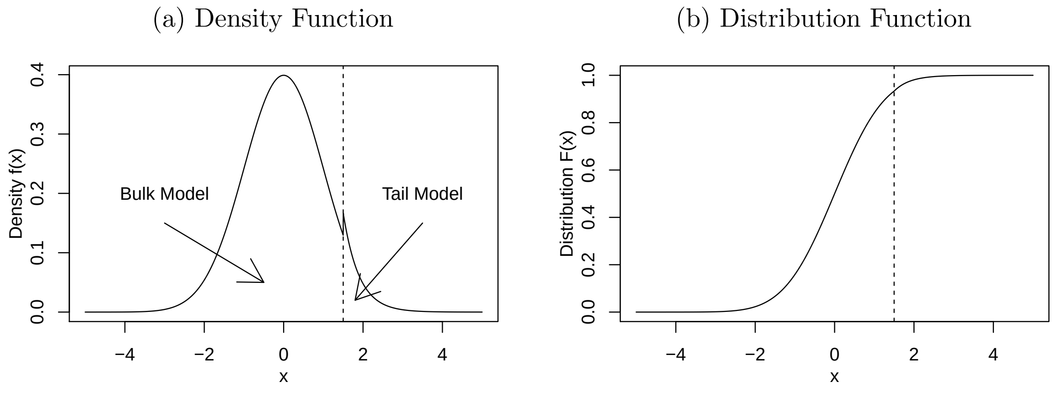


Figure 2: Density function and cumulative distribution functions of the parametric extreme value mixture model, with normal for bulk with $\boldsymbol{\theta} = (\mu = 0, \sigma = 1)$, threshold $u = 1.5$ and GPD for upper tail with $\sigma_u = 0.4$ and $\xi = 0$. The vertical dashed line indicates the threshold.

Figure 2 is an example density with a (truncated) normal for the bulk model, using the bulk model based tail fraction as in Equation 4. Notice that the density may be discontinuous at the threshold, but the cdf will be continuous provided the bulk model cdf is continuous.

3.1. Continuity constraint and other extensions

A straightforward adaptation can be made to include a constraint of continuity at the threshold. The density of the GPD at the threshold is simply $G(u|u, \sigma_u, \xi) = 1/\sigma_u$ so in the bulk model based tail fraction approach then continuity can be achieved by defining σ_u as:

$$\sigma_u = \frac{1 - H(u|\boldsymbol{\theta})}{h(u|\boldsymbol{\theta})} \tag{6}$$

and for the parameterised tail fraction approach:

$$\sigma_u = \frac{H(u|\boldsymbol{\theta})}{1-\phi_u} \frac{\phi_u}{h(u|\boldsymbol{\theta})}. \tag{7}$$

The right hand fraction in the latter formulation has the same structure as in Equation 6, with the bulk tail fraction $1 - H(u|\boldsymbol{\theta})$ replaced by the new parameter ϕ_u. Correspondingly, the left hand fraction is the reciprocal of the normalisation factor in Equation 5 which ensures the bulk component integrates to $1 - \phi_u$.

All of the extreme value mixture models of the general form presented in Equations 4 and 5 above have been implemented in the **evmix** package with two distinct versions: with and without a constraint of continuity at the threshold.

Further extensions to this basic form of extreme value mixture model are possible. For example, constraints of continuity in first and second derivatives (Carreau and Bengio 2009), a point process representation of the tail excess model (MacDonald *et al.* 2011) or use of alternative tail models like a Pareto (Solari and Losada 2004) and extended GPD (Papastathopoulos and Tawn 2013). As none of these have yet been implemented in the package, they will not be discussed further here. In Section 6 we outline some related extreme value mixture models, which do not fit within the above the framework.

3.2. Implemented mixture models

The bulk model's considered in the literature cover the full spectrum of parametric, semi-parametric and non-parametric forms. All parametric bulk models we are aware of from the literature have been implemented, namely:

- normal with GPD for upper tail (Behrens *et al.* 2004), see `help(normgpd)`;
- gamma with GPD for upper tail (Behrens *et al.* 2004), see `help(gammagpd)`;
- Weibull with GPD for upper tail (Behrens *et al.* 2004), see `help(weibullgpd)`;
- log-normal with GPD for upper tail (Solari and Losada 2004), see `help(lognormgpd)` ;
- beta with GPD for upper tail (MacDonald 2012), see `help(betagpd)`;
- normal with GPD for both upper and lower tails (Zhao, Scarrott, Reale, and Oxley 2010) and (Mendes and Lopes 2004), see `help(gng)`.

The following semi-parametric bulk model has been implemented:

- mixture of gammas with GPD for upper tail (do Nascimento, Gamerman, and Lopes 2012), see `help(mgammagpd)`.

The semi-parametric bulk model using a mixture of exponentials of Lee, Li, and Wong (2012) is not implemented directly, as it is a special case of the mixture of gammas. Examples of the implementation of thee above bulk models are given the following sections.

The following non-parametric bulk models have been implemented:

- standard kernel density estimator using a constant bandwidth with GPD for upper tail (MacDonald *et al.* 2011) with a wide range of kernels, see `help(kdengpd)`.
- standard kernel density estimator using a constant bandwidth with GPD for both upper and lower tails (MacDonald *et al.* 2013), see `help(gkg)`.
- boundary corrected kernel density estimators with GPD for upper tail (MacDonald *et al.*, 2013 and MacDonald, 2012) with a range of boundary correction approaches and kernel functions, see `help(bckdengpd)`.

These will be discussed in Section 9 after description of the implemented kernel density estimation methods in Section 7.

Naming conventions

A common naming convention is provided for all the distributions functions for the extreme value mixture models, e.g., for the normal with GPD for upper tail:

- `dnormgpd` - density function;
- `pnormgpd` - cumulative distribution function;
- `qnormgpd` - quantile function; and
- `rnormgpd` - random number generation;

consistent with the prefixes of `d, p, q` and `r` and inputs in the **base** package (R Core Team 2013). These conventions are also extended to the likelihood inference related functions:

- `fnormgpd` - maximum likelihood estimation of parameters;
- `lnormgpd` - log-likelihood function;
- `nlnormgpd` - negative log-likelihood function; and
- `proflunormgpd`, `nlunormgpd` - profile and negative log-likelihood functions for given threshold u.

The bulk model parameters are specified first followed by the threshold and GPD parameters as in Equation 4 and the tail fraction specification. For example, `dnormgpd` has syntax:

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

Default values are provided for all parameters, where appropriate. The tail fraction input `phiu` can either be a specified as a valid probability to represent it's value under the parameterised tail fraction approach in Equation 5 or as the (default) logical value `TRUE` to indicate the bulk model based tail fraction as in Equation 4. In the maximum likelihood fitting function it can only be logical where `phiu=FALSE` indicates the tail fraction parameter should be estimated using the sample proportion of exceedances. In the likelihood functions it can either be logical (`TRUE` or `FALSE`) or a valid probability depending on the user's needs.

As noted above, all such mixture models are implemented in their simplest form with no continuity constraint and with continuity at the threshold (upper and lower thresholds for two tailed models). The convention for the versions with continuity constraint is to specify the GPD scale parameter `sigmau` using the above functions of the tail fraction and bulk model parameters, and to append "`con`" to the function name. For example:

- `dnormgpd` - density for normal bulk and GPD tail with no continuity constraint; and
- `dnormgpdcon` - with additional constraint of continuity at threshold.

4. Threshold inference

Optimisation of the likelihood function for these mixture models is challenging as there are frequently local modes, which is an inherent challenge as there are often many potential thresholds. Further, once a sufficiently high threshold is reached the GPD will be an appropriate fit for all higher ones (upto sample variation).

By default, the maximum likelihood estimate (MLE) is found using the `BFGS` optimiser for the entire parameter vector, including the threshold. A code snippet using the default settings of (truncated) normal bulk and GPD tail with bulk model based tail fraction approach is given by:

```
R> fit = fnormgpd(x)
```

which permits a parameterised tail fraction using:

```
R> fit = fnormgpd(x, phiu = FALSE)
```

Such black-box optimisation of the entire parameter vector is used in most of the examples below, but **users should be aware it can be rather sensitive to the initial parameter values** due to the local modes and so should consider some of the following alternatives.

4.1. Fixed threshold approach

The fixed threshold approach can be implemented using a pre-chosen threshold (e.g., from using the graphical diagnostics considered in Section 2) by setting the `useq` to the threshold and `fixedu=TRUE`. e.g., to fix the threshold to 1.5:

```
R> fit = fnormgpd(x, useq = 1.5, fixedu = TRUE)
```

4.2. Profile likelihood with or without fixed threshold

A grid search over a sequence of possible thresholds to find that which maximises the profile likelihood is carried out by specification of the `useq` input. The profile likelihood over u also exhibits local modes, hence why a grid search is used. The estimated threshold giving the maximum profile likelihood can either be treated as the initial value for the threshold in the default black-box optimiser, e.g., using sequences of thresholds from 0 to 2 with gaps of 0.1:

```
R> fit = fnormgpd(x, useq = seq(0, 2, 0.1))
```

or alternaively can be set in the fixed threshold approach:

```
R> fit = fnormgpd(x, useq = seq(0, 2, 0.1), fixedu = TRUE)
```

5. Parametric bulk model example

The following code provides an example of fitting a parametric bulk model. A sample of size 1000 from a standard normal is simulated and density histogram plotted:

```
R> set.seed(12345)
R> x = rnorm(1000)
R> xx = seq(-4, 4, 0.01)
R> y = dnorm(xx)
R> hist(x, breaks = 100, freq = FALSE, xlab = "x", ylab = "Density f(x)")
R> lines(xx, y)
```

The extreme value mixture model with a (truncated) normal distribution for the bulk and GPD upper tail, with bulk model based tail fraction is fit by default:

```
R> fit.bulk = fnormgpd(x)
R> with(fit.bulk, lines(xx, dnormgpd(xx, nmean, nsd, u, sigmau, xi), col = "red"))
R> abline(v = fit.bulk$u, col = "red", lty = 2)
```

and parameterised tail fraction requires the option `phiu=FALSE` to be set:

```
R> fit.par = fnormgpd(x, phiu = FALSE)
R> with(fit.par, lines(xx, dnormgpd(xx, nmean, nsd, u, sigmau, xi, phiu), col = "blue"))
R> abline(v = fit.par$u, col = "blue", lty = 2)
R> legend("topright", c("True Density", "Bulk Tail Fraction",
+    "Parameterised Tail Fraction"), col=c("black", "red", "blue"), lty = 1)
```

Notice in Figure 3 that in both cases the estimated thresholds are essentially the same, as are the GPD parameters. Though the fits on either side of the threshold are rather different.

The following code demonstrates the use of a grid search to find the threshold which maximises the profile likelihood, that is subsequently fixed in the inference:

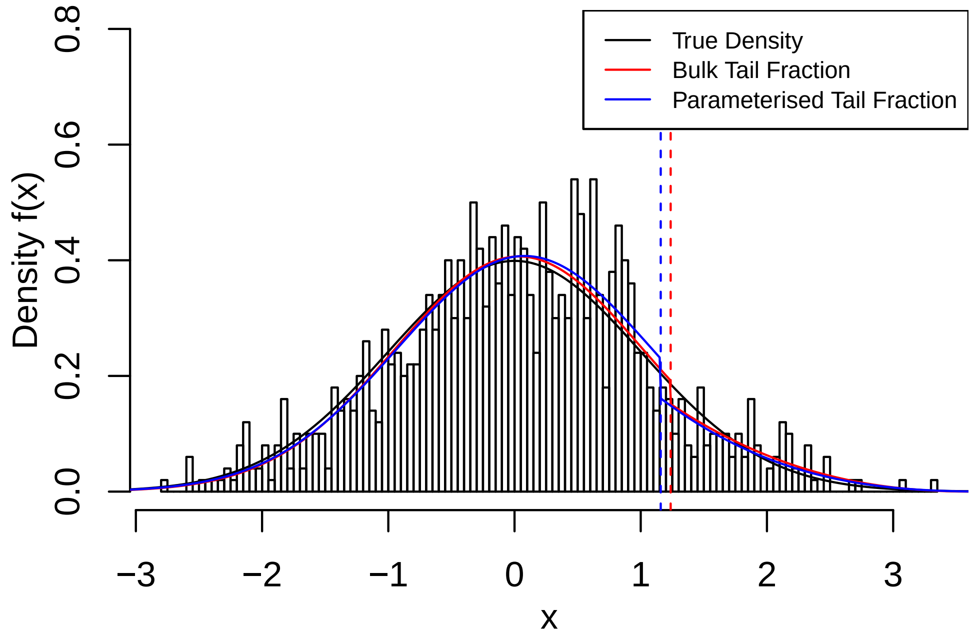


Figure 3: Sample density histogram of sample of 1000 standard normals, overlaid with fitted extreme value mixture model with (truncated) normal for bulk and GPD for upper tail. The bulk model based tail fraction in red and parameterised tail fraction in blue, with corresponding thresholds as vertical dashed lines.

```
R> hist(x, breaks = 100, freq = FALSE, xlab = "x", ylab = "Density f(x)")
R> lines(xx, y)
R> fit.fix = fnormgpd(x, useq = seq(0, 2, 0.1), fixedu = TRUE)
R> with(fit.fix, lines(xx, dnormgpd(xx, nmean, nsd, u, sigmau, xi), col = "blue"))
R> abline(v = fit.fix$u, col = "blue", lty = 2)
```

Notice that if we compare to the full likelihood approach:

```
R> with(fit.bulk, lines(xx, dnormgpd(xx, nmean, nsd, u, sigmau, xi), col = "red"))
R> abline(v = fit.bulk$u, col = "red", lty = 2)
R> legend("topright", c("True Density", "Black-box Optimisation",
+    "Fixed Threshold from Profile Lik"), col=c("black", "red", "blue"), lty = 1)
```

the threshold has changed to a much higher value, but the overall upper tail fit has not changed much. It is common for the threshold to move far out into the tail when the bulk model can approximate the tail well, e.g., when the bulk model is the same as the population distribution, as you would intuitively expect.

The standard diagnostic plots used in evaluating stationary univariate extreme value models are available using the `evmix.diag` function which is based on the `plot.uvevd` function of the **evd** library (Stephenson 2002). These diagnostic plot functions accept all the extreme value mixture models and kernel density estimation approaches outlined in the following section. In the case of the mixture models, the diagnostic plot functions default to focussing on the upper tail above the threshold (with at least the upper 10% shown). But the user can

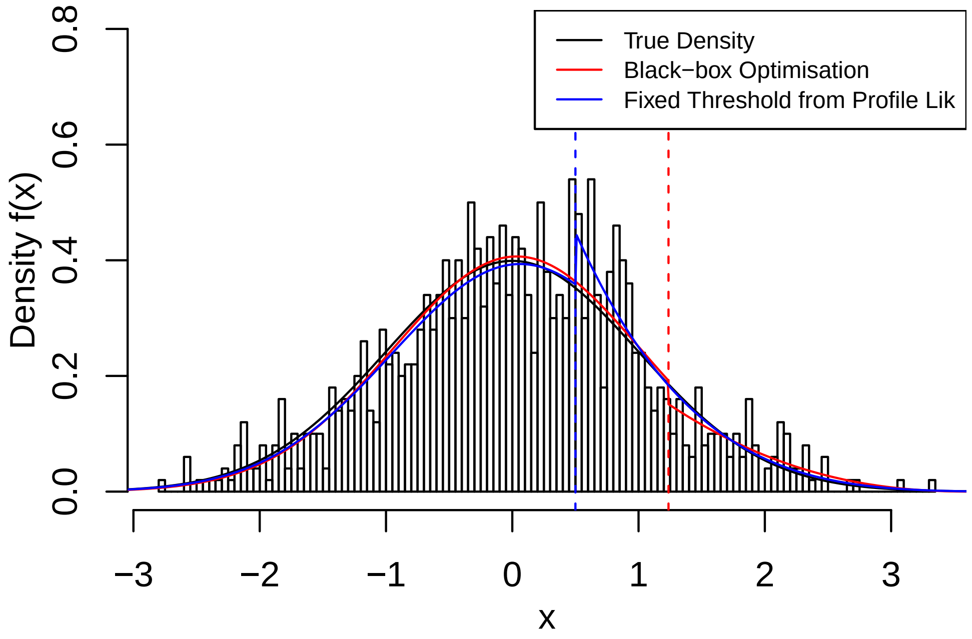


Figure 4: Same as Figure 4. The bulk model based tail fraction with black-box optimisation of all parameters in red and fixed threshold approach with threshold chosen by grid search to maximise profile likelihood over sequence of thresholds in blue.

set `upperfocus = FALSE` to evaluate the fit over the entire support. See the following code example and result in Figure 5. Each of these plots is available individually to allow flexibility for users, see `help(evmix.diag)` for details.

The only non-standard diagnostic plot outside of extreme value research community is the return level plot in the upper left. For independent and identically distributed observations the "return level" is defined as the quantile x_p which is exceeded with a survival probability of p, i.e., $P(X > x_p) = p$. In which case the waiting time between such events follows a geometric random variable and as such the expected waiting time is then defined $t = 1/p$ which is defined as the return period. The return levels described by the GPD for the upper tail from inverting Equation 1 are given by:

$$x_p = \begin{cases} u - \dfrac{\sigma_u}{\xi}\left[\left(\dfrac{p}{\phi_u}\right)^{-\xi} - 1\right] & \text{for } \xi \neq 0 \\ u - \sigma_u \log\left(\dfrac{p}{\phi_u}\right) & \text{for } \xi = 0. \end{cases} \tag{8}$$

The return level plot gives the return levels against the return period on a log scale, which gives a straight line in the case of an exponential upper tail ($\xi = 0$) thus aiding interpretation of the tail behaviour.

```
R> evmix.diag(fit.bulk)
```

In the return level plot, sample values are shown by the crosses and fit by the solid line. The upper right and lower left are the usual QQ and PP-plots, with line of equality. The pointwise

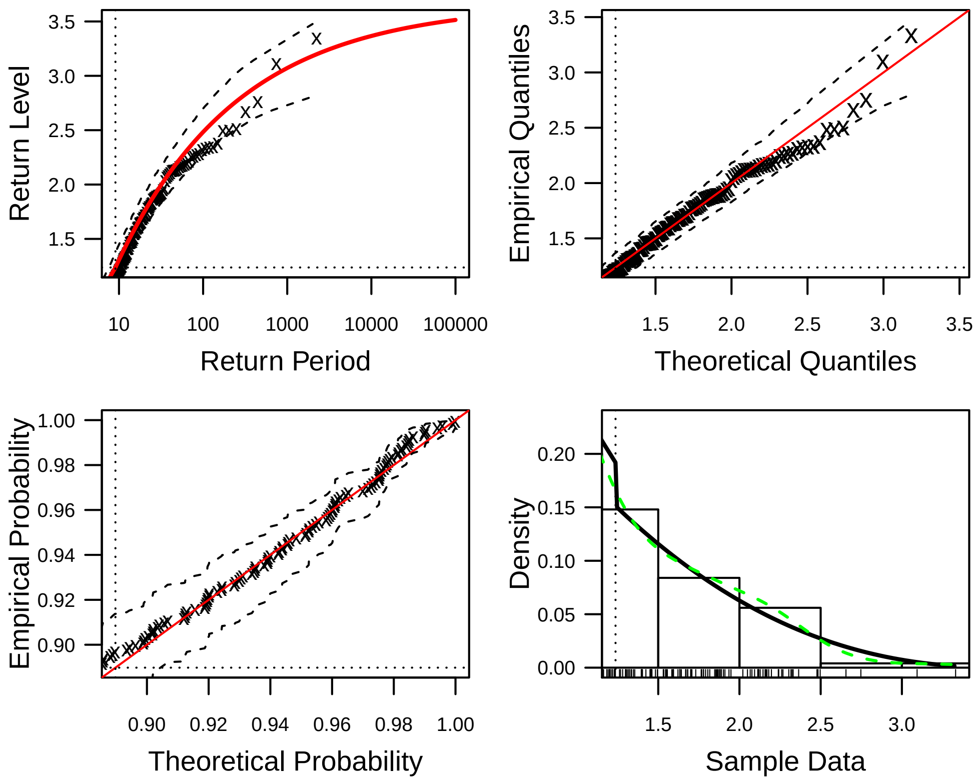


Figure 5: Example of diagnostic plots for assessing model fit.

95% tolerance intervals shown by the dashed lines are obtained by Monte Carlo simulation from the fitted model. The vertical and horizontal lines show either the fitted threshold or tail fraction as appropriate. The lower right gives the sample density histogram, overlaid with the fitted density as a solid line and a kernel density estimate as a dashed green line.

6. Alternative parametric extreme value mixture models

Two extreme value mixture models do not fit into the general framework in Equation 5, namely the hybrid Pareto distribution proposed by Carreau and Bengio (2009) and the dynamically weighted mixture of Frigessi, Haug, and Rue (2003) both described below.

6.1. Hybrid Pareto

The hybrid Pareto is conceptually similar to that of Equation 5, with the bulk model as the (truncated) normal distribution. However, there are two key differences:

1. the scaling of the GPD by the tail fraction $\phi_u = P(X > u)$ is ignored, so the GPD is treated as an unconditional model for the upper tail, and
2. constraint of continuity and continuity in the first derivative are both included.

The latter does not affect performance much, though it can introduce more sensitivity of the tail fit to misspecification of the bulk model, to be discussed in Section 10. However, the dropping of the tail fraction scaling from Equation 5 has more substantive consequences. The following reformulation of the hybrid Pareto using the same notational framework as for the mixture models above will demonstrate why the GPD will dominate over the bulk model and hence why this model does not perform as expected.

The hybrid Pareto includes a normalisation constant γ to make it proper:

$$F(x|\mu,\sigma,\xi) = \begin{cases} \frac{1}{\gamma} H(x|\mu,\sigma) & \text{for } x \leq u, \\ \frac{1}{\gamma}[H(u|\mu,\sigma) + G(x|u,\sigma_u,\xi)] & \text{for } x > u. \end{cases} \tag{9}$$

where we use the notation $H(x|\mu,\sigma) = \Phi(x|\mu,\sigma)$ for the normal distribution cdf to clarify the relation with the usual mixture model in Equation 4. The normalisation constant:

$$\gamma = H(u|\mu,\sigma) + 1. \tag{10}$$

shows the reason why the GPD will dominate the fit after removal of the tail fraction scaling. As $H(u|\mu,\sigma) \leq 1$ the GPD will receive weight of at least a half. Notice that the cumulative distribution in Equation 9 is only dependent on three parameters (μ,σ,ξ), with the threshold u and GPD scale σ_u which appear on the right hand side prescribed to satisfy the constraints of continuity in zeroth and first derivative at the threshold, see Carreau and Bengio (2009) or type `help(hpd)` for full details. Due to these parameter constraints, there is insufficient flexibility in the GPD parameters to adapt to the lack of tail fraction scaling.

The hybrid Pareto is fit to 1000 simulated standard normal variates (using `fhpd` function) and overlaid on a density histogram using the following code:

```
R> set.seed(1)
R> x = rnorm(1000)
R> xx = seq(-4, 4, 0.01)
R> y = dnorm(xx)
R> hist(x, breaks = 100, freq = FALSE, xlab = "x", ylab = "Density f(x)")
R> lines(xx, y)

R> fit.hpd = fhpd(x)
R> with(fit.hpd, lines(xx, dhpd(xx, nmean, nsd, xi), col = "red"))
R> abline(v = fit.hpd$u, col = "red", lty = 2)
```

On first sight one may expect the hybrid Pareto would fit a normal population well, due to the normal being used for the bulk distribution. Unfortunately, this is not the case as can be seen in the example in Figure 6. For comparison, the usual extreme value mixture model in Equation 4 with a (truncated) normal distribution for the bulk and GPD for the upper tail is now fit using the `fnormgpdcon` function and overlaid on plot as blue line.

```
R> fit.evmix = fnormgpdcon(x, phiu = FALSE)
R> with(fit.evmix, lines(xx, dnormgpdcon(xx, nmean, nsd, u, xi, phiu), col = "blue"))
R> abline(v = fit.evmix$u, col = "blue", lty = 2)
R> legend("topright", c("True Density", "Hybrid Pareto",
+    "Normal + GPD with Cont. Density"), col=c("black", "red", "blue"), lty = 1)
```

In the latter, the bulk model based tail fraction approach is used, including a constraint of continuous density at the threshold to enable a more direct comparison. Notice the poor fit of the hybrid Pareto and the adequate fit of the more usual extreme value mixture model.

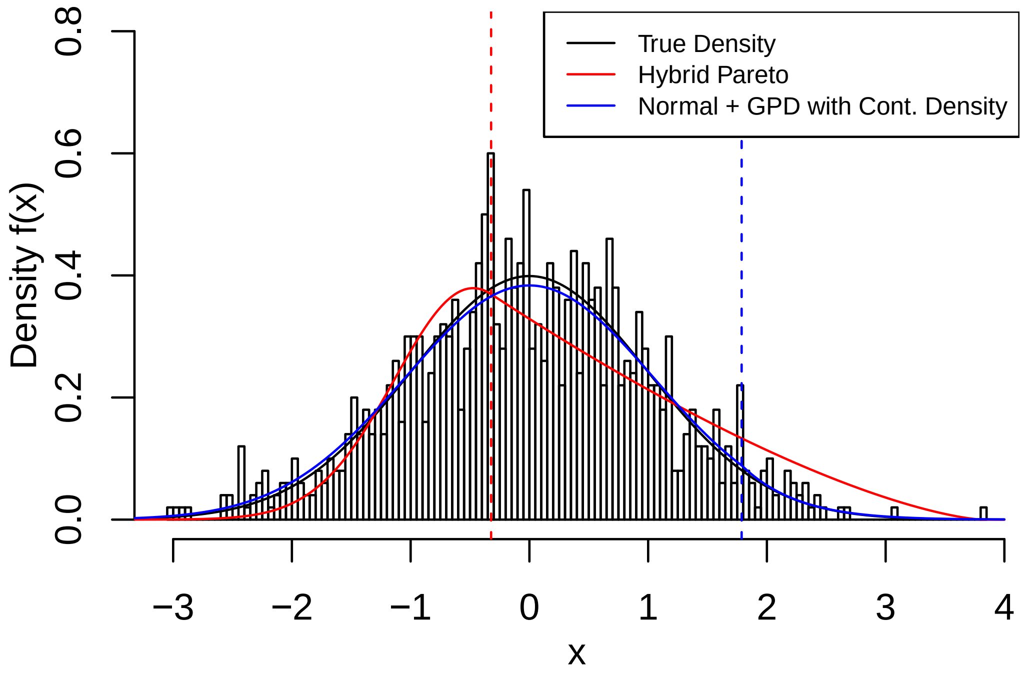


Figure 6: Sample density histogram of sample of 1000 standard normals, overlaid with fitted hybrid Pareto and extreme value mixture model with (truncated) normal for bulk and GPD for upper tail with continuous density at threshold. The hybrid Pareto in red and usual extreme value mixture model in blue, with corresponding thresholds as vertical dashed lines.

For more direct comparability a simplified hybrid Pareto with a single continuity constraint at the threshold is implemented in the package, see `help(hpdcon)` function. In this case, the GPD scale σ_u is defined using the other parameters as:

$$\sigma_u = \frac{1}{h(u|\mu,\sigma)} = \frac{1}{\phi(u|\mu,\sigma)}. \tag{11}$$

Notice that the only difference between this constraint compared to the usual extreme value mixture model in Equation 6 is that the numerator is now one due to the lack of tail fraction scaling of the GPD. Although not shown here for brevity, a comparison by replacing `fnormgpd` by `fnormgpdcon` in the above code snippet confirms that the poor performance is due to this lack of tail fraction scaling of the GPD and is easily resolved by it's inclusion.

Some of the authors of the hybrid Pareto have also created the **condmixt** package (Carreau 2012) which implements the hybrid Pareto. It has some extensions not considered in the **evmix** package as yet. Firstly, their functions allow for the normal distribution for the bulk to be truncated at zero and appropriately renormalised. They also provide a mixture of hybrid Paretos, which also resolve the poor performance of the hybrid Pareto.

6.2. Dynamically weighted mixture model

Frigessi *et al.* (2003) avoid the need to determine a threshold altogether. They suggest using a dynamically weighted mixture model by combining the two distributions defined over the entire range of support:

- general distribution with light upper tail which well describes the bulk shape; and
- suitable tail model (e.g., GPD) with a threshold at the lower bound of the support.

The GPD tail model is defined over the whole support, so they focus application of their model for population distributions which are bounded below. Similarly, the general distribution for the bulk will have the same lower bound. Typically, the lower bound is zero.

A smooth transition function between these two components is also defined over the entire range of support. It is specified to give highest weight on the general distribution at low values in the range of support and give highest weight to the tail model in the upper range of the support, a smooth change in switching the weights provides the transition function. The authors suggest that conceptually the transition function could be a Heaviside side function, thus the density switches from the general bulk to GPD at a prescribed changepoint, thus defining the equivalent of the threshold as in the usual mixture models above. However, if using the Heaviside function alone there will be no scaling of the GPD by the tail fraction, so would suffer from the same performance issues as the hybrid Pareto above.

The density function of the dynamic mixture model is given by:

$$f(x) = \frac{[1 - p(x|\boldsymbol{\theta}_1)]\, h(x|\boldsymbol{\theta}_2) + p(x|\boldsymbol{\theta}_1)\ g(x|0, \sigma_u, \xi)}{Z(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \sigma_u, \xi)},$$

where $h(.)$ denotes the general bulk density, $g(.)$ is the usual GPD and $p(.)$ is the transition function. The denominator $Z(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \sigma_u, \xi)$ is the normalizing constant to ensure a proper density. In the application by Frigessi *et al.* (2003), they considered a Weibull for the general model, GPD for the tail model and shift-scale Cauchy distribution function as the transition function which can be fit using the `fdwm` function:

```
R> x = seq(0.01, 5, 0.01)
R> f = ddwm(x, wshape = 2, wscale = 1/gamma(1.5), cmu = 1, ctau = 1,
+    sigmau = 1, xi = 0.5)

R> plot(x, f, ylim = c(0, 1), type = 'l', ylab = "Density f(x)")
R> lines(x, dweibull(x, shape = 2, scale = 1/gamma(1.5)), col = "blue", lty = 2)
R> lines(x, dgpd(x, xi = 1, sigmau = 0.5), col = "red", lty = 2)
R> lines(x, pcauchy(x, loc = 1, sc = 1), col = "green")
R> legend('right', c('DWM', 'Weibull', 'GPD', "Cauchy CDF"), bg = "white",
+        col = c("black", "blue", "red", "green"), lty = c(1, 2, 2, 1))
```

The following code reproduces an example of the three components and the resultant density estimate are given in Figure 7 from Frigessi *et al.* (2003). A key drawback with this formulation is that the GPD contributes to the lower tail as it has a pole at the lower bound, shown by the mixture shown by the black line being much higher than the Weibull density in blue for

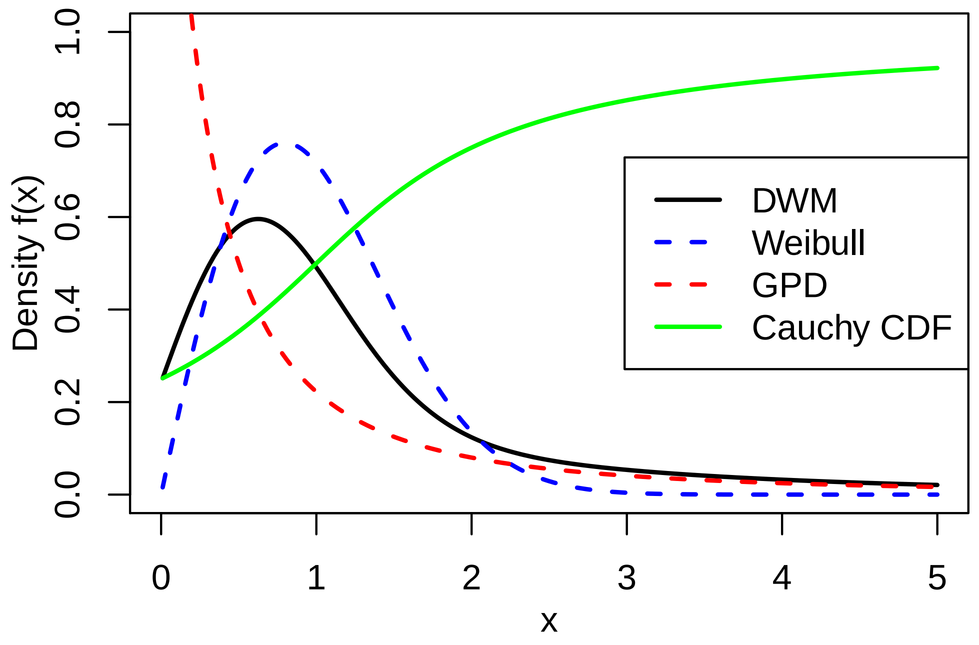


Figure 7: Example of dynamically weighted mixture model as in Frigessi *et al.* (2003) shown by solid black line. General (Weibull) model component shown in blue, GPD shown in red and standard Cauchy cdf transition function in green.

low values in the range of support. Thus there is a sensitivity of the upper tail fit to that of the lower tail, which is not ideal for extreme value models.

7. Kernel density estimation

Let $x_1, x_2, \ldots, x_n$ be a sample of a random variable X from a population density $f(x)$. The density of X can then be estimated by a kernel density estimator (KDE) as:

$$\hat{f}(x|\lambda) = \frac{1}{n}\sum_{i=1}^{n} K_\lambda(x - x_i) = \frac{1}{n\lambda}\sum_{i=1}^{n} K\left(\frac{x - x_i}{\lambda}\right) \tag{12}$$

where $K_\lambda(z) = K(z/\lambda)/\lambda$ is a kernel function with the following properties:

$$\text{(a)} \int K(z)\, dz = 1, \quad \text{(b)} \int z\, K(z)\, dz = 0 \text{ and (c)} \int z^2\, K(z)\, dz < \infty.$$

These imply the kernel function integrates to one, symmetric (mean zero) and has finite variance. Although not required, the kernel function is typically non-negative $K(z) \geq 0$ so could be a proper density function. The bandwidth $\lambda > 0$ is key in controlling the level of smoothing, which is a tradeoff between the bias and variance. Too small a bandwidth leads to undersmoothing and an increase in the variance, whereas too large a bandwidth will reduce the variance but increase the bias. Most kernel functions are defined on $|z| \leq 1$, making the bandwidth λ the half-width of the kernel function.

All the commonly used kernel functions are implemented, see `help(kernels)`:

- uniform (rectangular) function $K(z) = \frac{1}{2}\, I_{|z|\leq 1}(z)$, see `kduniform`;
- triangular function $K(z) = (1 - |z|)\, I_{|z|\leq 1}(z)$, see `kdtriangular`;
- Epanechnikov function $K(z) = \frac{3}{4}(1 - z^2)\, I_{|z|\leq 1}(z)$, see `kdepanechnikov`;
- biweight function $K(z) = \frac{15}{16}(1 - z^2)^2\, I_{|z|\leq 1}(z)$, see `kdbiweight`;
- triweight function $K(z) = \frac{35}{32}(1 - z^2)^3\, I_{|z|\leq 1}(z)$, see `kdtriweight`;
- tricube function $K(z) = \frac{70}{81}(1 - |z|^3)^3\, I_{|z|\leq 1}(z)$, see `kdtricube`;
- Parzen function $K(z) = 8(1-|z|)^3\, I_{0.5<|z|\leq 1}(z) + \frac{4}{3} - 8z^2 + 8|z|^3\, I_{|z|\leq 0.5}(z)$, see `kdparzen`;
- cosine function $K(z) = \frac{1}{2}(1 + \cos(\pi z))\, I_{|z|\leq 1}(z)$, see `kdcosine`; and
- optcosine function $K(z) = \frac{\pi}{4}\left(1 + \cos\left(\frac{\pi z}{2}\right)\right)\, I_{|z|\leq 1}(z)$, see `kdoptcosine`.

The commonly used Gaussian kernel has unbounded support with λ the standard deviation:

- Gaussian function $K(z) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right)$, see `kdgaussian`.

The kernel functions are shown in Figure 8.

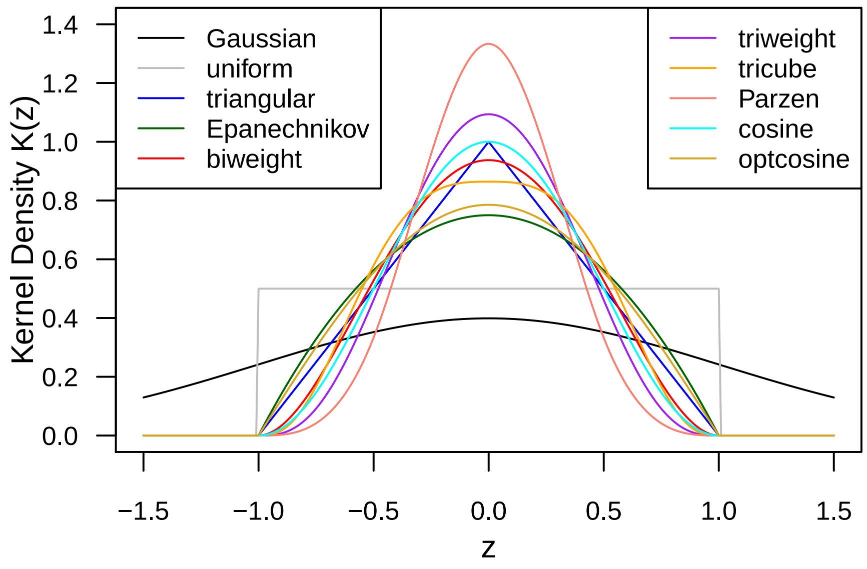


Figure 8: Standardised kernel functions $K(z)$ listed on page 17.

7.1. Bandwidth estimation

There are two commonly used definitions of the bandwidth as either the:

- kernel half-width λ, denoted as `lambda` and the default in the **evmix** package; or
- standard deviation of the kernel which is denoted `bw`.

The latter is used in the `density` function of the **stats** package (R Core Team 2013), hence we have used the same naming convention of`bw`. The `kbw` and `klambda` functions convert from one bandwidth definition to the other, based on the standard deviation of the underlying kernel. These functions allow the user to use the inbuilt bandwidth estimation functions, see `help(bw.nrd0)` for details, and convert them to the half-width equivalent.

To complement the inbuilt bandwidth estimation procedures we have implemented cross-validation likelihood inference, treating it as a parameter in a mixture model which was first proposed by Habbema, Hermans, and van den Broek (1974) and Duin (1976). To avoid the degeneracy in direct maximum likelihood estimation they suggest using leave one out cross-validation likelihood:

$$L(\lambda|\boldsymbol{X}) = \prod_{i=1}^{n} \frac{1}{(n-1)} \sum_{\substack{j=1 \\ j\neq i}}^{n} K_\lambda(x_i - x_j).$$

Functionality and conventions

The usual naming conventions have been adhered to `dkden`, `pkden`, `qkden`, `rkden`, `lkden`, `nlkden` and `fkden`. These functions differ from those of the mixture models above in that they take either definition of the bandwidth parameter (`lambda` or `bw`), defaulting to the former if both are provided, and the name of the kernel function (e.g., `kernel="gaussian"`. The maximum likelihood fitting function also outputs both bandwidth definitions for convenience. Unlike the mixture model functions the bandwidth must be input as a scalar.

These functions are implemented using the KDE definition in Equation 12 so are user readable and give direct evaluation at an arbitrary set of locations. However, this makes them inefficient when the evaluation points are a regular sequence (or approximated by such a sequence) in which case the Fourier transform makes the kernel convolution more computationally efficient. The following example demonstrates use of the `density` function to take advantage of this efficiency for a sequence of evaluation points, but using our maximum cross-validation likelihood bandwidth estimate.

Each implemented kernel has functions using the following the naming conventions:

- density for arbitrary bandwidth - with kernel name prefixed with `kd`, e.g., `kdgaussian` for Gaussian kernel density function; and
- distribution for arbitrary bandwidth - with kernel name prefixed with `kp`, e.g., `kpgaussian` for Gaussian kernel cumulative distribution function.

Extra functionality for an arbitrary kernel is provided by the following functions, which take the standardised values $z = \dfrac{x - x_i}{\lambda}$ and the kernel name (e.g., `kernel="gaussian"`) as inputs:

- `kdu` - standardised kernel density function;
- `kpu` - standardised kernel cumulative distribution function;
- `ka0` - $a_0(x) = \int_{-\lambda}^{x} K(z)\, dz$, i.e., kernel cdf at x;
- `ka1` - $a_1(x) = \int_{-\lambda}^{x} z\, K(z)\, dz$, i.e., partial contribution to first moment upto x; and
- `ka2` - $a_2(x) = \int_{-\lambda}^{x} z^2\, K(z)\, dz$, i.e., partial contribution to second moment upto x;

The latter three functions are used in the boundary correction techniques detailed in Section 8. Further, for the Gaussian kernel the lower limit of the integrals is $-\infty$, but is implemented as -5λ in the **evmix** package.

7.2. Example of kernel density estimation

The following code and result in Figure 9 demonstrate how to use maximum cross-validation likelihood estimation for the bandwidth using the `fkden` function, assuming the default Gaussian kernel:

```
R> set.seed(0)
R> x = rnorm(50)
R> fit = fkden(x)
```

The sample density histogram of the kernel centres is overlaid with the fit and true density:

```
R> xx = seq(-4, 4, 0.01)
R> hist(x, 10, freq = FALSE, xlim = c(-4, 4), ylim = c(0, 0.7))
R> rug(x)
R> lines(xx, dnorm(xx), col = "black")
R> with(fit, lines(xx, dkden(xx, x, lambda), col = "red"))
```

which can be compared to the KDE using the usual `density` function:

```
R> lines(density(x), lty = 2, col = "green")
R> with(fit, lines(density(x, bw = bw), lty = 2, col = "blue"))
R> legend("topright", c("True Density", "KDE fitted by MLE in evmix",
+    "KDE using density, default bw", "KDE using density, MLE bw"),
+    lty = c(1, 1, 2, 2), col = c("black", "red", "green", "blue"))
```

Notice that the estimated bandwidth is rather similar to that Silverman's rule of thumb used by default in the `density` function. Further, notice that if the MLE of the bandwidth is specified in the `density` function, the resultant density function (blue dashed line) is the same as that obtained from using the `dkden` (red solid line).

The principle drawback of maximum cross-validation likelihood for the bandwidth is that it is biased high (leading to oversmoothing) for heavy tailed distributions, as the distance between the upper and lower order statistics does not decay to zero as $n \to \infty$, see MacDonald *et al.* (2011) for details. An example of oversmoothing is given in the following code and Figure 10, where the blue line is clearly drastically oversmoothed. A robust estimator of bandwidth can

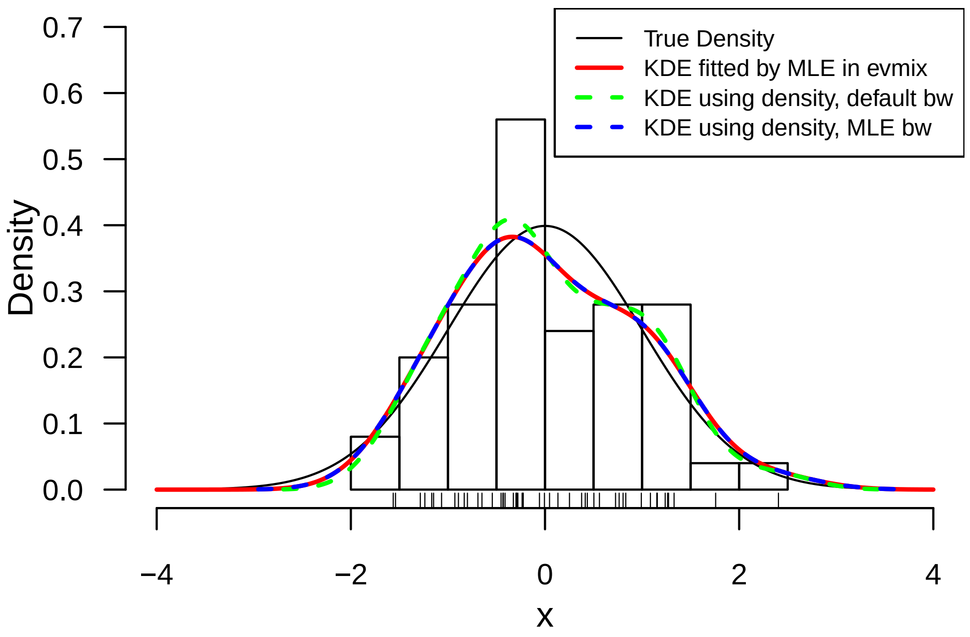


Figure 9: Example of fitted KDE for 50 simulated data from standard normal using **evmix** package in red. Comparison to `density` function using default bandwidth estimator in green and using maximum cross-validation likelihood estimate of bandwidth in blue to show efficiency of Fourier transform in `density` function.

be achieved by ignoring the heavy tails in the cross-validation likelihood, but note that the observations in the tails are still required in the KDE otherwise a boundary bias would occur. Denote the observations not in the tails in this example are given by $A = \{i : -3 \leq x_i \leq 3\}$ then the robust cross-validation likelihood is then:

$$L_A(\lambda|\boldsymbol{X}) = \prod_{A} \frac{1}{(n-1)} \sum_{\substack{j=1 \\ j \neq i}}^{n} K_\lambda(x_i - x_j),$$

where the product includes only observations in A, but the KDE sum includes the extra tail observations in $A^c = \{i : x_i < -3 \ \cap \ x_i > 3\}$. The reduced set of observations A which contributes to the cross-validation likelihood are specified by the usual `kerncentres` input and the extra tail observations which contribute to the KDE sum are given by the `extracentres` input. Notice that the bandwidth estimate obtained by ignoring the tails in the MLE is robust to the heavy tails (red line) in Figure 10. An alternative approach to overcoming the bandwidth bias for heavy tails, is to use an extreme value mixture model with the KDE for the bulk distribution and GPD for the tails discussed in Section 9.

The following code simulates 100 variates from a heavy tailed Student-$t(\nu = 2)$ and uses the entire sample in the cross-validation likelihood estimation:

```
R> set.seed(0)
R> x = rt(100, df = 2)
R> fit = fkden(x)
```

and re-fits using a trimmed estimate where data larger in magnitude than 3 are ignored:

```
R> intails = which((x <= -3) | (x >= 3))
R> fit.notails = fkden(x[-intails], extracentres = x[intails])
```

The trimmed bandwidth estimate is then applied across the entire sample to get the KDE:

```
R> xx=seq(-10, 10, 0.01)
R> hist(x, seq(floor(min(x)), ceiling(max(x)), 0.5), freq = FALSE, xlim = c(-10, 10))
R> rug(x)
R> lines(xx, dt(xx , df = 2), col = "black")
R> with(fit, lines(xx, dkden(xx, x, lambda), col = "blue"))
R> with(fit.notails, lines(xx, dkden(xx, x, lambda), lwd = 2, col = "red"))
R> legend("topright", c("True Density", "KDE fitted by MLE in evmix",
+    "KDE if tails ignored in MLE"), col = c("black", "blue", "red"))
```

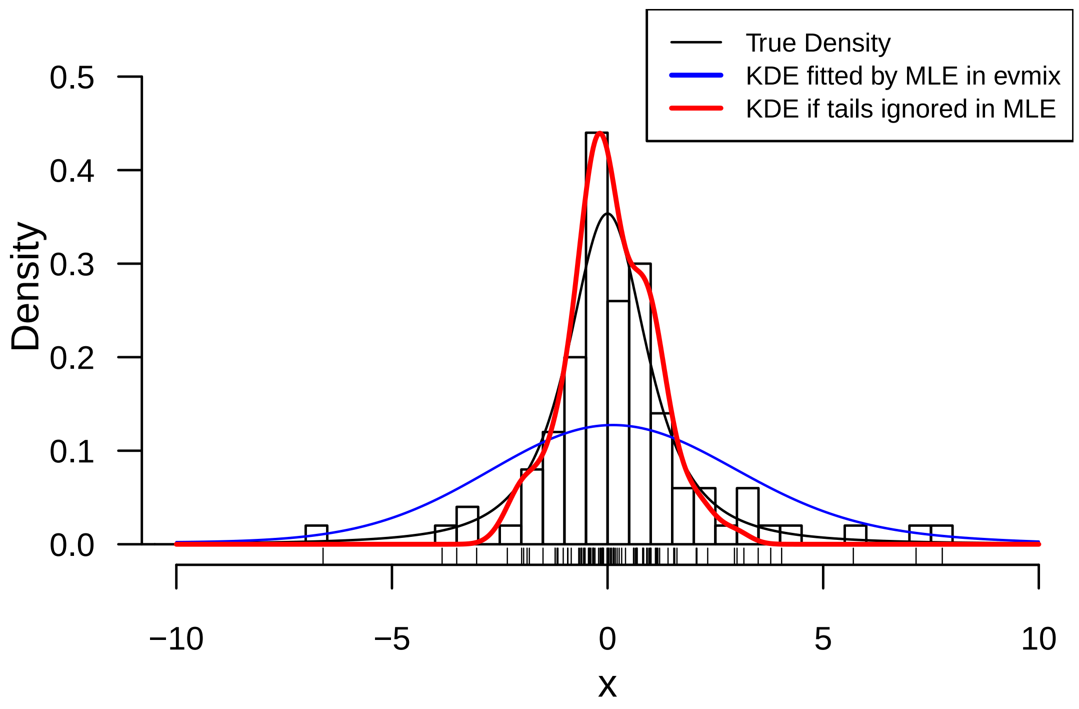


Figure 10: Example of fitted KDE for simulation from Student-t($\nu = 2$) using **evmix** in blue. Robust bandwidth estimator with heavy tails ignored in the cross-validation likelihood in red.

8. Boundary corrected kernel density estimators

Many population distributions have bounded support, for example the Weibull distribution is defined on $[0; \infty)$. In such situations it is clearly desirable that the KDE $\hat{f}(x)$ has the same support. Direct application of the standard KDE as in Equation 12 will not necessarily satisfy this constraint. A wide range of boundary corrected kernel density estimators (BCKDE) have been proposed in the literature. The approaches implemented in the **evmix** package are listed below. The only BCKDE we have been able to find in the packages on `CRAN` is the log transformation method available in the `kde` function in the **ks** package (Duong 2013). So the **evmix** package is providing much new functionality.

For brevity, the details of the implemented BCKDE are not given here for brevity, but the key references are given for the interested reader. Each approach is listed by their names in `bcmethod` input, see `help(bckden)` for further details. The following methods assume the support is bounded below at zero $x \in [0, \infty)$:

- `bcmethod="simple"` (default) applies the simple boundary correction method in Equation 3.4 of Jones (1993) and is equivalent to kernel weighted local linear fitting near the boundary. The kernel function $K(z)$ in Equation 12 is replaced by:

$$K_p(z) = \frac{a_2(p) - a_1(p)z}{a_2(p)a_0(p) - a_1^2(p)} K(z) \tag{13}$$

  where $p = x/\lambda$ is the standardies distance of the evaluation point x to the boundary. The coefficients $a_0(.), a_1(.)$ and $a_2(.)$ are detailed in the list of kernel functions on page 18. The boundary correction only applies for $x \in [0, \lambda)$ as beyond this range they reduce to the usual kernel function $K(z)$ as $a_1(p) = 0$ and $a_0(p) = 0$ for $p > 1$. The boundary corrected kernels in Equation 13 can be negative, so a negative density estimate can result. The non-negative correction method of Jones and Foster (1996) has been implemented and is applied by default and chosen by the input `nn="jf96"`. Further, the density estimate may not be proper and so renormalisation to unity is applied by default by the input `proper=TRUE`.

- `bcmethod="renorm"` applies the first order bias correction method of Diggle (1985), where the kernel density estimate itself is renormalised near boundary:

$$\tilde{f}(x) = \frac{1}{a_0(p)} \hat{f}(x). \tag{14}$$

  As with the `"simple"` method the resultant density may need renormalisation to unity.

- `bcmethod="cutnorm"` applies the cut and normalisation method of Gasser and Müller (1979), where the kernels are truncated at the boundary and renormalised to unity:

$$K_p(z) = \frac{1}{a_0(x_i/\lambda)} K(z). \tag{15}$$

  Note that correction factor $a_0(x_i/\lambda)$ is evaluated at the distance of the kernel centre to the boundary x_i/λ, so that each kernel is renormalised to unity.

- `bcmethod="reflect"` applies the reflection method of Schuster (1985) which is equivalent to the dataset being supplemented by the same dataset negated. This method implicitly assumes $f'(0) = 0$, so can cause extra artifacts at the boundary.

- `bcmethod="logtrans"` applies KDE on the log-scale and then back-transforms (with explicit normalisation) following Marron and Ruppert (1994), which is the approach implemented in the **ks** package (Duong 2013). As the KDE is applied on the log scale, the effective bandwidth on the original scale is non-constant. An offset is needed to avoid $\log(0)$ for datapoints on the boundary, which is user specified by the `offset` input.

All of the above methods can use any of the kernel functions described in the previous section. The user chooses the kernel function with the `kernel` input which defaults to using the Gaussian kernel. Note that for the Gaussian kernel with unbounded support the **evmix** package applies this correction over $x \in [0, 5\lambda]$, or the equivalent range after back transformation in the log-transformation method.

The following boundary correction methods do not use kernels in their usual sense, but are commonly used and have similar features. The first two of these pseudo-kernel density estimators allow for an upper and lower boundary on the support, which has been implemented using a lower boundary of zero and upper boundary specifed by the `xmax` input.

- `bcmethod="beta1"` and `"beta2"` uses the beta and modified beta kernels of Chen (1999) specified by their $\hat{f}_1(x)$ and $\hat{f}_2(x)$ respectively. Renormalisation is required to obtain a proper density.

- `bcmethod="copula"` uses the bivariate normal copula based kernels of Jones and Henderson (2007). In this case the bandwidth is defined as $\lambda = 1 - \rho^2$ where ρ is the usual correlation coefficients so is limited to $(0, 1)$.

- `bcmethod="gamma1"` and `"gamma2"` uses the gamma and modified gamma kernels of Chen (2000) specified by their $\hat{f}_1(x)$ on page 472 and $\hat{f}_2(x)$ on page 473 respectively. Renormalisation is required to obtain a proper density.

Maximum cross-validation likelihood estimation for the bandwidth is implemented for all the BCKDE methods. The only difference between this estimation method for the BCKDE, versus the KDE above, is that in the latter an initial value for the bandwidth is provided (using Silverman's rule of thumb in `bw.nrd0` function) but is not provided for BCKDE. Not all the implemented BCKDE approaches have a rule of thumb type bandwidth estimator, so the user must provide an initial guess. If the initial guess is inappropriate then the `fbckden` MLE fitting function will automatically try values upto 32 times bigger (or smaller) than that provided, which in most cases will permit the optimisation routine to get past the initial step.

8.1. Example of boundary corrected kernel density estimation

The following code and result in Figure 11 gives an example of estimating the bandwidth using maximum cross-validation likelihood estimation implementing the reflection method. The reflection and cut & normalisation approaches are the most computationally efficient of the implemented boundary correction methods, as no renormalisation or non-negative correction is required. The standard kernel density estimator is severely biased near the boundary. The reflection approach has better performance, but still exhibits some bias. The simple boundary correction method performs much better, even when using the same bandwidth estimate.

In general, we find that bandwidth estimates from the computationally efficient reflection or cut & normalisation methods, generally provides reliable estimates when used in the simple or renormalisation boundary correction methods. This sub-optimal approach avoids the computational burden associated with the renormalisation (and/or non-negative correction).

The following code applies the reflection method (`bcmethod="reflect"`) for estimating the bandwidth using cross-validation likelihood estimation (`fbckden` function) for 500 variates from a gamma($\alpha = 1, \beta = 2$) distribution:

```
R> set.seed(0)
R> x = rgamma(500, shape = 1, scale = 2)
R> fit = fbckden(x, linit = 1, bcmethod = "reflect")
```

This estimated bandwidth is then used in the, reflection, simple and renormalisation boundary correction approaches:

```
R> xx = seq(-1, 10, 0.01)
R> hist(x, 20, freq = FALSE, ylim = c(0, 0.6))
R> rug(x)
R> lines(xx, dgamma(xx, shape = 1, scale = 2), col = "black")

R> with(fit, lines(xx, dbckden(xx, x, lambda, bcmethod = "reflect"), col = "red"))
R> with(fit, lines(xx, dbckden(xx, x, lambda, bcmethod = "simple"), col = "blue"))
R> with(fit, lines(xx, dbckden(xx, x, lambda, bcmethod = "renorm"), col = "goldenrod"))

R> lines(density(x), lty = 2, col = "green")
R> legend("topright", c("True Density", "BC KDE using reflect",
+   "BC KDE using simple", "BC KDE using renorm", "KDE Using density"),
+   lty = c(1, 1, 1, 1, 2), col = c("black", "red", "blue", "goldenrod", "green"))
```

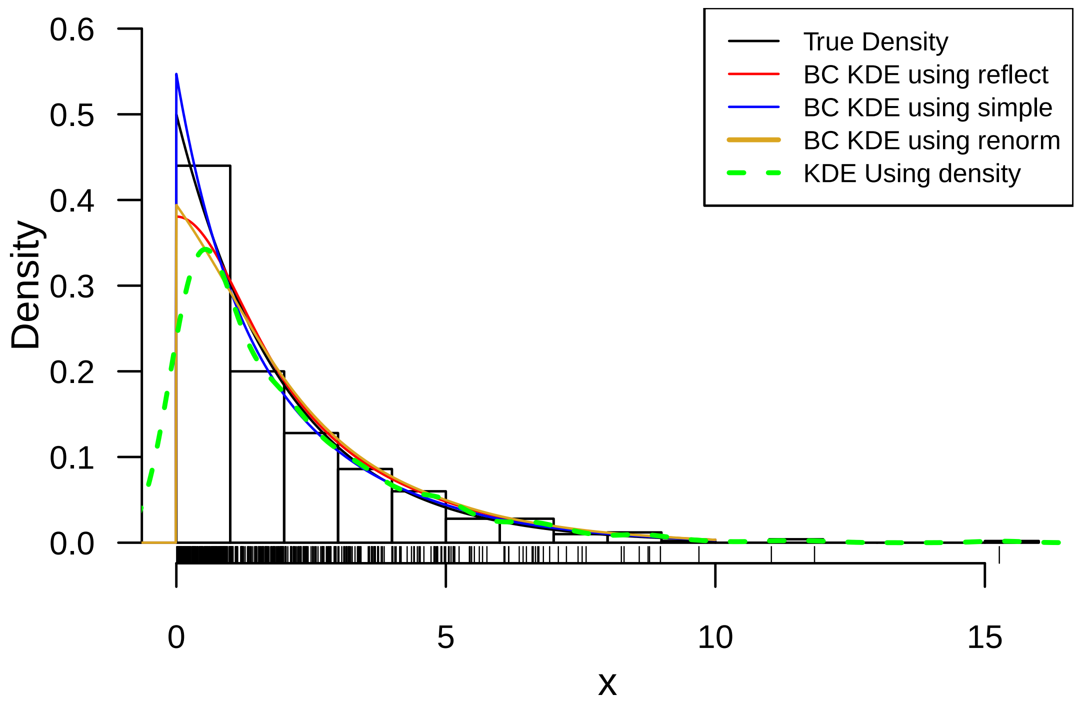


Figure 11: Example of fitted KDE for simulated data from gamma(1, 2) using reflection method in red. The simple and renormalisation boundary correction methods are shown in blue and goldenrod, with bandwidth from reflection method. Comparison to poor performance of standard KDE in `density` function using default bandwidth estimator in green.

9. Nonparametric extreme value mixture models

The kernel density estimators (including the boundary correction approaches) can be used to describe the bulk component of an extreme value mixture model. MacDonald *et al.* (2011) considered the use of a standard KDE as described in Section 7 for the bulk, combined with a GPD for the upper tail. They also considered an extension to allow a GPD for the upper and lower tails. The **evmix** package extends their work to permit a variety of kernels rather than just the Gaussian. Further, we have also extended these to permit a constraint of continuity in the density at the threshold(s). The key function names are:

- `dkdengpd`, `pkdengpd`, `qkdengpd`, `rkdengpd`, `lkdengpd`, `nlkdengpd`, and `fkdengpd` for mixture model with kernel density estimator for bulk and GPD for upper tail;
- `dgkg`, `pgkg`, `qgkg`, `rgkg`, `lgkg`, `nlgkg`, and `fgkg` for mixture model with kernel density estimator for bulk and GPD for both upper and lower tails;

Similarly, the boundary corrected kernel density estimators can also be used to describe the bulk distribution in an extreme value mixture model as considered by MacDonald *et al.* (2013). They only considered the simple boundary correction method of Jones (1993) using a Gaussian kernel, with limited exploration of other BCKDE's by MacDonald (2012). The **evmix** package implements all the BCKDE's in Section 8 for the bulk distribution, with a GPD for the upper tail in the following functions:

- `dbckdengpd`, `pbckdengpd`, `qbckdengpd`, `rbckdengpd`, `lbckdengpd`, `nlbckdengpd`, and `fbckdengpd`.

All the above nonparametric mixture models have been extended to versions with a constraint of continuous density at the threshold, with the function names appended with "`con`".

9.1. Example of KDE based extreme value mixture model

The following code provides an example of applying the two tailed extreme value mixture model, with a middle bulk component kernel density estimator using the default Gaussian kernel and GPD for both upper and lower tails (using `fgkg` function). The simulated data in this example are from a heavy tailed Student-$t(\nu = 2)$ distribution.

```
R> set.seed(1)
R> x = rt(500, df = 2)
R> fit.gkg = fgkg(x)
```

The bulk model based tail fraction is used. The resultant fit should be compared to the standard KDE as the blue line in Figure 10.

```
R> xx = seq(-10, 10, 0.01)
R> hist(x, 200, freq = FALSE, xlim = c(-10, 10), ylim = c(0, 0.55))
R> rug(x)
R> lines(xx, dt(xx, df = 2), col = "black")
R> with(fit.gkg, lines(xx, dgkg(xx, x, lambda, ul, sigmaul, xil, phiul,
+    ur, sigmaur, xir, phiur), col = "red"))
R> abline(v = c(fit.gkg$ul, fit.gkg$ur), lty = 2, col = "red")
```

```
R> fit.kden = fkden(x, linit = 1)
R> with(fit.kden, lines(xx, dkden(xx, x, lambda), col = "blue"))
R> abline(v = fit.kden$u, lty = 2, col = "blue")
R> legend("topright", c("True Density", "KDE for bulk, two GPD tails", "KDE only"),
+    bg = "white", lty = 1, col = c("black", "red", "blue"))
```

The standard KDE fit is poor due to the positive bias in the cross-validation likelihood estimator for the bandwidth due to the heavy tails, as discussed in Section 7. The extreme value mixture model which uses the GPD tail approximation ensure the bandwidth is robust to the heavy tails, as shown by the red line in Figure 12.

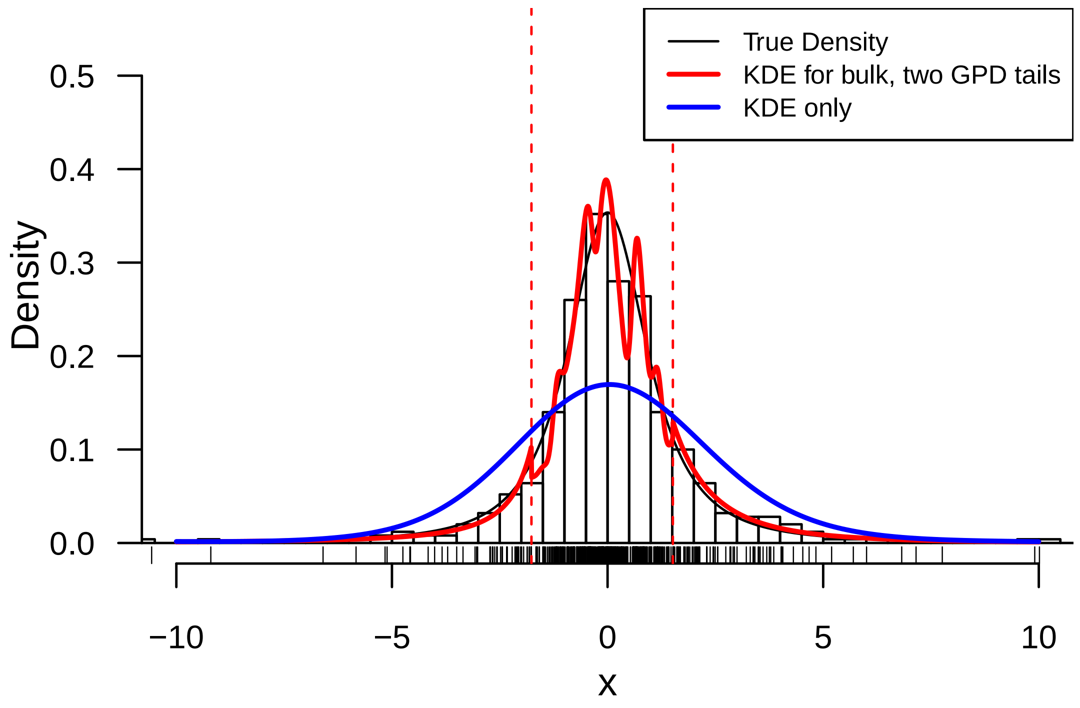


Figure 12: Example of two tailed KDE with Gaussian KDE for bulk fitted to 500 simulated data from Student-t($\nu = 2$). Comparison to `density` function using default bandwidth estimator. The blue and red lines show the fits for versions of the model with and without constraint of continuity at both thresholds, respectively. Compare to Figure 9 to see robustness of KDE bandwidth estimator when the heavy tails are described by the GPD and thus ignored in cross-validation likelihood for bandwidth.

10. Some parting advice on mixture models

Hu (2013) created the first version of this package as part of his Masters thesis along with a detailed simulation study comparing the performance of many of the aforementioned extreme value mixture models for approximating the tails of a wide variety of population distribution for samples of size 1000 and 5000. The performance in estimation of the 90, 95, 99 and 99.9% quantiles (and 0.1, 1, 5 and 10% for lower tail in two tailed models) is evaluated using the root mean squared error (RMSE). A simple 95% bootstrap percentile interval using 100,000 resamples of the RMSE is used to explore evidence for the difference in model performance.

The broad conclusions from this study were:

- the estimates of very high quantiles (e.g., 99.9% for samples of size 1000 and 5000) are very similar across most sensible extreme value mixture models, whereas the lower quantiles (e.g., 90-99%) can be very sensitive to the choice of bulk model;
- the bulk model based specification of the tail fraction $\phi_u = 1 - H(u|\boldsymbol{\theta})$ should be used if the bulk model is known to be the same as the population distribution, or the sample size is sufficient to adequately assess that the bulk model is a very good fit. In this way, the tail fraction estimate benefits from the usual ample data in the bulk compared to the limited tail data;
- conversely, in the more usual situation of an unknown population distribution or where the fit of bulk model cannot be adequately assessed it is better to use the parameterised tail fraction approach as the tail estimates are more robust to the bulk model fit;
- the constraint of continuity at the threshold provides little, if any, benefit for estimation of the highest quantiles. There is some evidence of a benefit for lower quantiles, near the threshold when the bulk and tail models both provide a very good fit;
- however, this continuity constraint has a detrimental effect if either the bulk or tail model do not provide a good fit, as the tail estimation is less robust to the bulk fit and vice-versa. Hence, it is generally safest not to include a constraint of continuity at the threshold, especially in the usual situation when the population distribution is unknown and the sample size is insufficient for the bulk and tail fits to be evaluated;
- the usual advice for choosing between parametric versus semi/nonparametric estimators hold for extreme value mixture models. For example, if the bulk model is the same as the population distribution or is a good approximation then a parametric extreme value mixture model provides the best tail estimator. However, in the more usual situation of an unknown population distribution the nonparametric extreme value mixture models provide similar overall level of performance in terms of tail estimation; and
- the lack of tail fraction scaling in the hybrid Pareto leads to poor performance, even for some asymmetric heavy tailed distributions, due to the domination of the GPD tail over the bulk model. This problem is resolved by inclusion of the tail fraction scaling.

There are some limitations in this simulation study which are under further investigation, but are not likely to strongly change these conclusions as they are rather intuitive. Firstly, the simulations were mainly carried out using likelihood inference using the BFGS optimiser, with the initial value for the threshold set as the 90% quantile (or 10% quantile for lower tail). As mentioned above, the likelihood for these mixture models frequently has many local modes so this is a sub-optimal choice. Profile likelihood estimation of the threshold can be used to ameliorate this issue, see Section 4.2 above. Hu (2013) compared some of the likelihood based simulation results to those using Bayesian inference with posterior sampling using Markov chain Monte Carlo, which does not suffer the challenges associated with optimisation in the presence of local modes, with the same broad conclusions reached.

The use of the RMSE as the measure of performance ignores the potential for overfitting, as there is no penalty for the degrees of freedom used in the model fit. This provides a slight advantage to the parameterised tail fraction approach and versions of the extreme value mixture models with no constraints of continuity in the density at the thresholds. The nonparametric and semiparametric extreme value mixture models also benefit from extra degrees of the freedom. However, this performance gain is likely to be small as the degrees of freedom are expended on the bulk component of the distribution, not on the tail(s) where the performance is being evaluated.

10.1. Obtaining the package and guides

The latest package is released under the GPL-3 licence and 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`. Further resources are available on the package website `http://www.math.canterbury.ac.nz/~c.scarrott/evmix`

Acknowledgments

Some of the code for this library were originally written for MATLAB by Anna MacDonald and Xin Zhao and were translated and adapted for R. Clement Lee and Emma Eastoe receive thanks for many helpful discussions about these models and feedback on the package. In particular, they recommended the implementation of profile likelihood estimation of the threshold, which can sometimes greatly improve the model performance. The authors (Alec Stephenson, Janet Heffernan and Eric Gilleland) of the **evd** and **ismev** packages are also thanked as some aspects of the base code for the GPD and graphical diagnostic functions are based around functions in their libraries.

References

Behrens CN, Lopes HF, Gamerman D (2004). "Bayesian Analysis of Extreme Events with Threshold Estimation." *Statistical Modelling*, **4**(3), 227–244.

Carreau J (2012). ***condmixt**: Conditional Density Estimation with Neural Network Conditional Mixtures*. R package version 1.0, URL `http://CRAN.R-project.org/package=condmixt`.

Carreau J, Bengio Y (2009). "A Hybrid Pareto Model for Asymmetric Fat-tailed Data: The Univariate Case." *Extremes*, **12**(1), 53–76.

Chen SX (1999). "Beta kernel estimators for density functions." *Computational Statistics and Data Analysis*, **31**(2), 131–145.

Chen SX (2000). "Probability Density Function Estimation Using Gamma Kernels." *Annals of the Institute of Statistical Mathematics*, **52**, 471–480.

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

do Nascimento FF, Gamerman D, Lopes HF (2012). “A Semiparametric Bayesian Approach to Extreme Value Estimation.” *Statistics and Computing*, **22**, 661–675.

Duin RPW (1976). “On the Choice of Smoothing Parameters for Parzen Estimators of Probability Density Functions.” *IEEE Transactions on Computing*, **C - 25**(11), 1175–1179.

Duong T (2013). ***ks**: Kernel Smoothing.* R package version 1.8.12, URL http://CRAN.R-project.org/package=ks.

Frigessi A, Haug O, Rue H (2003). “A Dynamic Mixture Model for Unsupervised Tail Estimation Without Threshold Selection.” *Extremes*, **5**(3), 219–235.

Gasser T, Müller HG (1979). “Kernel Estimation of Regression Functions.” In T Gasser, M Rosenblatt (eds.), *Smoothing Techniques for Curve Estimation*, pp. 23–68. Springer-Verlag, Berlin.

Gilleland E, Katz RW (2005). “Extremes Toolkit (**extRemes**): Weather and Climate Applications of Extreme Value Statistics.” *Technical report*, NCAR: Weather and Climate Impact Assessment Science Program.

Gilleland E, Katz RW (2011). “New Software to Analyze How Extremes Change Over Time.” *Eos*, **92**(2), 13–14.

Habbema JDF, Hermans J, van den Broek K (1974). “A Stepwise Discriminant Analysis Program Using Density Estimation.” In G Bruckmann (ed.), *Proceedings of COMPSTAT 1974*, pp. 101–110. Physica-Verlag: Vienna.

Hu Y (2013). *Extreme Value Mixture Modeling with Simulation Study and Applications in Finance and Insurance.* MSc thesis, University of Canterbury, New Zealand. URL http://ir.canterbury.ac.nz/simple-search?query=extreme&submit=Go.

Hu Y, Scarrott CJ (2013). “evmix: Extreme Value Mixture Modelling, Threshold Estimation and Boundary Corrected Kernel Density Estimation.” Available on CRAN, URL http://www.math.canterbury.ac.nz/~c.scarrott/evmix.

Jones MC (1993). “Simple Boundary Correction for Kernel Density Estimation.” *Statistics and Computing*, **3**, 135–146.

Jones MC, Foster PJ (1996). “A Simple Nonnegative Boundary Correction Method for Kernel Density Estimation.” *Statistica Sinica*, **6**, 1005–1013.

Jones MC, Henderson DA (2007). “Miscellanea Kernel-type Density Estimation On the Unit Interval.” *Biometrika*, **94**(4), 977–984.

Lee D, Li WK, Wong TST (2012). “Modeling Insurance Claims Via a Mixture Exponential Model Combined with Peaks-over-threshold Approach.” *Insurance: Mathematics and Economic*, **51**(3), 538–550.

MacDonald A (2012). *Extreme Value Mixture Modelling with Medical and Industrial Applications.* PhD thesis, University of Canterbury, New Zealand. URL http://ir.canterbury.ac.nz/simple-search?query=extreme&submit=Go.

MacDonald A, Scarrott CJ, Lee D, Darlow B, Reale M, Russell G (2011). "A Flexible Extreme Value Mixture Model." *Computational Statistics and Data Analysis*, **55**(6), 2137–2157.

MacDonald A, Scarrott CJ, Lee DS (2013). "Boundary Correction, Consistency and Robustness of Kernel Densities Using Extreme Value Theory." *Submitted.* Available from: http://www.math.canterbury.ac.nz/∼c.scarrott.

Marron JS, Ruppert D (1994). "Transformations to Reduce Boundary Bias in Kernel Density Estimation." *Journal of the Royal Statistical Society* B, **56**(4), 653–671.

Mendes B, Lopes HF (2004). "Data Driven Estimates for Mixtures." *Computational Statistics and Data Analysis*, **47**(3), 583–598.

Papastathopoulos I, Tawn JA (2013). "Extended Generalised Pareto Models for Tail Estimation." *Journal of Statistical Planning and Inference*, **143**(1), 131–143.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL `http://www.R-project.org/`.

Scarrott CJ, MacDonald A (2012). "A Review of Extreme Value Threshold Estimation and Uncertainty Quantification." *REVSTAT: Statistical Journal*, **10**(1), 33–60.

Schuster EF (1985). "Incorporating Support Constraints Into Nonparametric Estimators of Densities." *Communications in Statistics - Theory and Methods*, **14**(5), 1123–1136.

Solari S, Losada MA (2004). "A Unified Statistical Model for Hydrological Variables Including the Selection of Threshold for the Peak Over Threshold Method." *Water Resources Research*, **48**(W10541).

Stephenson AG (2002). "**evd**: Extreme Value Distributions." *R News*, **2**(2), 31–32. URL `http://CRAN.R-project.org/doc/Rnews/`.

Tancredi A, Anderson CW, O'Hagan A (2006). "Accouting for Threshold Uncertainty in Extreme Value Estimation." *Extremes*, **9**(2), 87–106.

Zhao X, Scarrott CJ, Reale M, Oxley L (2010). "Extreme Value Modelling for Forecasting the Market Crisis." *Applied Financial Economics*, **20**(1), 63–72.

Affiliation:

Carl Scarrott
School of Mathematics and Statistics
University of Canterbury
Christchurch 8140, New Zealand
E-mail: `carl.scarrott@canterbury.ac.nz`
URL: `http://www.math.canterbury.ac.nz/~c.scarrott`