SUPPLEMENTARY MATERIALS FOR UNIVARIATE EXTREME VALUE MIXTURE MODELLING

CARL SCARROTT

1. Introduction to evmix Package

The broadest collection of the extreme value mixture models are implemented in the `evmix` package (Scarrott and Hu, 2015) for `R` (R Core Team, 2013a), which is available from `CRAN`. Other `R` packages that explicitly include a more limited selection of such models are `condmixt` (Carreau, 2012) and `CompLognormal` by Nadarajah (2013). See the main Chapter for a more discussion of these packages. This supplementary material will focus on demonstrating features of the `evmix` package.

The seed idea for the `evmix` package was formed in the review of threshold estimation tools in Scarrott and MacDonald (2012), which includes extreme value mixture models. Hu (2013a) produced the basis of the package as part of his Master's thesis. The thesis placed the majority of the existing extreme value mixture models into a standardised framework to compare their properties, which has been referred to as the "**standard model**" in the main Chapter. Some of the alternative formulations of extreme value mixture models were also considered. An extensive simulation study compared the performance of these mixture models, including specifically investigating the performance of the bulk or parameterised tail fractions and continuity constraints.

Hu (2013a) wrote a User Guide for the original version of the package (Hu, 2013b). The `evmix` package is subject to ongoing development and a more comprehensive description of the latest package features are given by Hu and Scarrott (2013). Both these documents are distributed as part of the package help files and provide fairly comprehensive coverage of the basic features of the package. There is no intention to repeat the information provided in these publications herein. The focus of this supplementary material is to provide additional insights about these models and guidance on using the package in practice, so mostly real data applications are considered below.

The following guide was written for version 0.2.5 of the `evmix` package. New features are continuously being added and modified. Extreme value mixture models and the inference scheme for them are a matter of ongoing research and as such functionality may vary in the future and no guarantee of backwards compatibility is provided.

2. Key Functions and Naming Conventions in the evmix Package

The nomenclature for the quartet of distribution functions for each mixture model uses the commonplace letter prefixes (`d`, `p`, `q` and `r`), postfixed with an abbreviated name derived from the bulk and tail components. For example, when a normal bulk model is combined with a GPD tail the postfix is `normgpd`, giving the quartet of distribution function names:

- `dnormgpd` - probability density function (pdf);
- `pnormgpd` - cumulative distribution function (cdf);
- `qnormgpd` - quantile (inverse cumulative distribution) function; and
- `rnormgpd` - random number generation.

The usual options given in the `stats` package (R Core Team, 2013b) quartet of distribution functions are provided for these mixture models (e.g., `lower.tail` in the `pnormgpd` function). A complete list of the implemented models and their abbreviated names are given in Section 3.

Maximum likelihood estimation (MLE) fitting functions (prefixed by `f` for 'f'itting) are provided in a standardized form, similar to the schema in the `evd` package (Stephenson, 2002) to which most researchers in the field will be familiar. The fitting functions have options for different maximum likelihood approaches for the threshold and other parameters:

- `useq = NULL` - MLE of complete likelihood of all parameters including the threshold (currently the default option, but not preferred option see Section 4.2 below);
- `useq` as a scalar and `fixedu = TRUE` - fixed threshold approach with threshold set to `useq`;
- `useq` as a scalar and `fixedu = FALSE` - use the `useq` as the initial value for the threshold in the optimisation of complete likelihood;
- `useq` as a vector and `fixedu = TRUE` - carry out a grid search over the thresholds in `useq` to find the value which maximises the profile likelihood, which is then fixed at that value and the conditional likelihood of the non-threshold parameters is used in the MLE; and
- `useq` as a vector and `fixedu = FALSE` - carry out a grid search over the thresholds in `useq` to find the value which maximises the profile likelihood, which is then used as the initial value for the threshold in the MLE over the complete likelihood.

The MLE fitting sub-functions use a similar nomenclature, for example:

- `fnormgpd` - fitting by maximum likelihood estimation;
- `lnormgpd` - log-likelihood function;
- `nlnormgpd` - negative log-likelihood function, designed for optimisation; and
- `proflunormgpd` and `nlunormgpd` - profile likelihood and conditional negative log-likelihood function for a given threshold.

The semiparametric bulk model with a mixture of gammas for the bulk model also has special functions for the expectation and maximisation steps in the EM algorithm (see `mgamma` and `mgammagpd`). The hybrid Pareto (see `hpd`) and dynamically weighted mixture models (see `dwm`) have no threshold parameter, so do not have corresponding profile likelihood functions.

2.1. **Exceptions to Naming Conventions.** The two-tailed mixture models with GPD for both the upper and lower tails (`gng`, `gkg` and `itmgng`) utilise a profile likelihood grid search over all combination pairs of both the lower and upper thresholds (`ulseq` and `urseq` respectively). As such, the profile likelihood (and underlying conditional negative log-likelihood) functions are applied over each combination pair.

The interval transitions models (`itmnormgpd`, `itmweibullgpd` and `itmgng`) transition between the bulk and tail models over a symmetric interval around the threshold. For these models, the profile likelihood grid search is over all combination pairs of both the thresholds (`useq`) and interval half-widths (`eseq`).

Scalar and vectorised parameter inputs for the quartet of distribution functions are provided for all the mixture models to allow for nonstationary parameters, except those with nonparametric bulk models which currently only allow scalar parameter inputs. However, parameter estimation for nonstationary modelling is currently unavailable, so all the fitting related functions only except scalar parameter inputs.

Each implemented mixture model has an additional variant which includes a constraint of continuity of the pdf at the threshold, currently achieved by constraining the GPD scale parameter. The function name nomenclature uses a further postfix of `con` for the constrained variant, for example:

- `dnormgpd` - normal bulk and GPD tail with no continuity constraint; and
- `dnormgpdcon` - normal bulk and GPD tail with single constraint of continuity of the pdf at the threshold.

The only exception is the hybrid Pareto (see `dhpd`) which has two constraints of continuity in the zeroth and first derivative of the pdf at the threshold, as discussed in the main chapter. The hybrid Pareto with only a single constraint of continuity of the pdf at the threshold uses the `con` postfix (see `dhpdcon`).

2.2. **Graphical Diagnostics.** In addition to the mixture model related functions various graphical diagnostic functions for assessing the model fit are provided. The usual probability, quantile, return level and density plots are provided for model fit diagnostics (see `evmix.diag`), similar to those implemented in the `ismev` package Stephenson (2014). These diagnostics are demonstrated in Section 4 below.

The following graphical diagnostics for threshold choice are currently provided:

- `tcplot` - threshold stability plots for GPD shape and modified scale parameters;
- `mrlplot` - mean residual life plot;
- `hillplot` - Hill plot and some of it's many variants (AltHill, SmooHill and AltSmooHill plots); and
- `pickandsplot` - Pickand's plot.

The maximum likelihood estimates of both GPD parameters and asymptotic Wald absed confidence intervals using the observed information matrix are output by the `tcplot` function. The `mrlplot` function outputs the sample mean excesses and central limit theorem based confidence intervals. The Hill and Pickand's shape parameter estimates and their corresponding confidence intervals are output by the latter functions.

3. Implemented Extreme Value Mixture Models

3.1. **Standard Mixture Models.** The current release 0.2.5 of the `evmix` package (Scarrott and Hu, 2015) implements the following parametric bulk extreme value mixture models with a GPD for the upper tail (for references see help files, or main Chapter):

- normal, see `help(normgpd)`;
- gamma, see `help(gammagpd)`;
- Weibull, see `help(weibullgpd)`;
- log-normal, see `help(lognormgpd)`; and
- beta, see `help(betagpd)`.

The beta distribution has support $x \in [0, 1]$, so when spliced with a GPD upper tail the threshold must be less than one. However, the beta-GPD can have a support with upper limit above one. The exponential bulk model of Teodorescu and Vernicu (2009) has not yet been explicitly implemented, as it is a special case of the gamma bulk model. Similarly, the Pareto tail models discussed in the main Chapter have not yet been explicitly implemented as they are a special case of the GPD tail.

The following semiparametric bulk model has been implemented:

- finite mixture of gammas with GPD for upper tail, see `help(mgammagpd)`.

The semi-parametric bulk model using a mixture of exponentials of Lee et al. (2012) is not explicitly implemented, as it is a special case of the mixture of gammas bulk model. The EM algorithm is used for the mixture of gammas bulk mixture model. Standalone functions for the mixture of gammas model, with no GPD spliced into the upper tail, have been provided for non-extreme value applications.

The following nonparametric bulk models with GPD upper tail have been implemented:

- standard kernel density estimator using a constant bandwidth (estimated using leave-one-out cross-validation likelihood) with a wide range of kernels, see `help(kdengpd)`; and
- boundary corrected kernel density estimator (using a bandwidth estimated using leave-one-out cross-validation likelihood) with a range of boundary correction approaches and kernel functions, see `help(bckdengpd)`.

The so-called simple boundary corrected kernel density estimator of Jones (1993) using a constant bandwidth was considered by MacDonald et al. (2013); MacDonald (2012), which is the preferred option of those provided. However, a wide range of other kernel and pseudo-kernel density estimators for bounded support have also been implemented, see Hu and Scarrott (2013) for details. Standalone functions for implementing the standard (`help(kden)`) and boundary corrected kernel density estimators (`help(bckden)`), with no extreme value tail model, have been provided for non-extreme value applications.

3.2. **Two-tailed Mixture Models.** All of the above mixture models have a GPD for the upper tail, so are referred to a one-tailed mixture models. The following two-tailed mixture models have been provided with a GPD for both the lower and upper tails, with bulk model of:

- normal, see `help(gng)`.
- standard kernel density estimator using a constant bandwidth (estimated using a leave one out cross-validation likelihood) with a wide range of kernels, see `help(gkg)`.

3.3. **Alternative Mixture Models.** The alternative extreme value mixture models which do not fit within the framework for the standard mixture models, that have been implemented are:

- dynamically weighted mixture model, see `help(dwm)`;
- interval transition mixture model, see variants listed below; and
- hybrid Pareto, see `help(hpd)`.

An alternative variant of the hybrid Pareto with only a single constraint of continuity of the pdf at the threshold is also provided, see `help(hpdcon)`. The many variants of the "mixture of hybrid Pareto's" has not been implemented, as these are available in the `condmixt` package (Carreau, 2012).

Various bulk models within the interval transition mixture model framework have been implemented, with model name prefixed with `itm`:

- Weibull and GPD for upper tail, see `help(itmweibullgpd)`;
- normal and GPD for upper tail, see `help(itmnormgpd)`; and
- normal and GPD for both the upper and lower tails, see `help(itmgng)`.

4. Examples

Examples of the usage of the `evmix` package are provided in the User Guide (Hu, 2013b) and (Hu and Scarrott, 2013). Here the focus is on demonstrating features of the extreme value mixture models and maximum likelihood inference approaches, which will be used to provide some justification of the advice provided in the main Chapter.

Firstly, Section 4.1 uses simulated data to demonstrate the expected behaviour when the bulk and tail models are appropriate descriptors for the population distribution. The Standard and Poor's and Dow Jones closing price returns are used in Sections 4.3.1 and 4.3.2 to demonstrate what features to expect when the bulk model is not entirely appropriate. These datasets will also be used to demonstrate the benefits of the parameterised tail fraction approach compared to the bulk model based tail fraction.

The poor performance of black box optimisers within the default inference approach of maximising the complete likelihood of all the parameters is demonstrated in Section 4.2. The benefits are demonstrated of the recommended approach of using a profile likelihood search for the threshold, which is then fixed in latter MLE inferences for the non-threshold parameters.

Numerous of the mixture models developed in the literature suffer from poor performance in wide application, due to ignoring the tail fraction scaling which usually scales the conditional GPD to make it an unconditional tail model. Section 4.4 demonstrates the poor performance and some of the model features associated with ignoring the tail fraction.

The Appendix includes further examples of using the `evmix` package to reproduce all of the figures in the main Chapter.

4.1. When the Bulk Model is Appropriate.

A sample of 1,000 standard normal variates are simulated to demonstrate features of these mixture models when the bulk model is appropriate, and the population tail is known to be in the domain of attraction of the GPD. The following code takes the simulated dataset and fits the extreme value mixture model with a normal bulk and GPD for the upper tail (denoted normal+GPD).

A grid search over thresholds $u = 0, 0.1, \ldots, 2.5$ is carried out using the profile likelihood. The threshold is then fixed at the threshold value that maximises the profile likelihood

(`fixedu = TRUE`). Variants of the normal+GPD with a bulk model based (`fit.bulk`) and parameterised tail fraction (`fit.para`) are both fitted.

```
library(evmix)

set.seed(1)
x = rnorm(1000)

# fit normal bulk with GPD upper tail (bulk model based tail fraction)
fit.bulk = fnormgpd(x, useq = seq(0, 2.5, 0.1), fixedu = TRUE)

# fit normal bulk with GPD upper tail (parameterised tail fraction)
fit.para = fnormgpd(x, phiu = FALSE, useq = seq(0, 2.5, 0.1), fixedu = TRUE)
```

The fitted density functions are plotted with the following code and the result is shown in Figure 1(a). The fitted density is close to the true population density, except due to the discontinuity just above the threshold. The bulk model based and parameterised tail fraction approaches have both given essentially the same parameter estimates in this case, see Table 1(a). In particular, the estimated thresholds ($u = 2.2$) are the same. Notice that in the current version (0.2.5) of the package that there is currently no uncertainty estimate for the threshold when the fixed threshold approach is taken.

(a) `set.seed(1)`

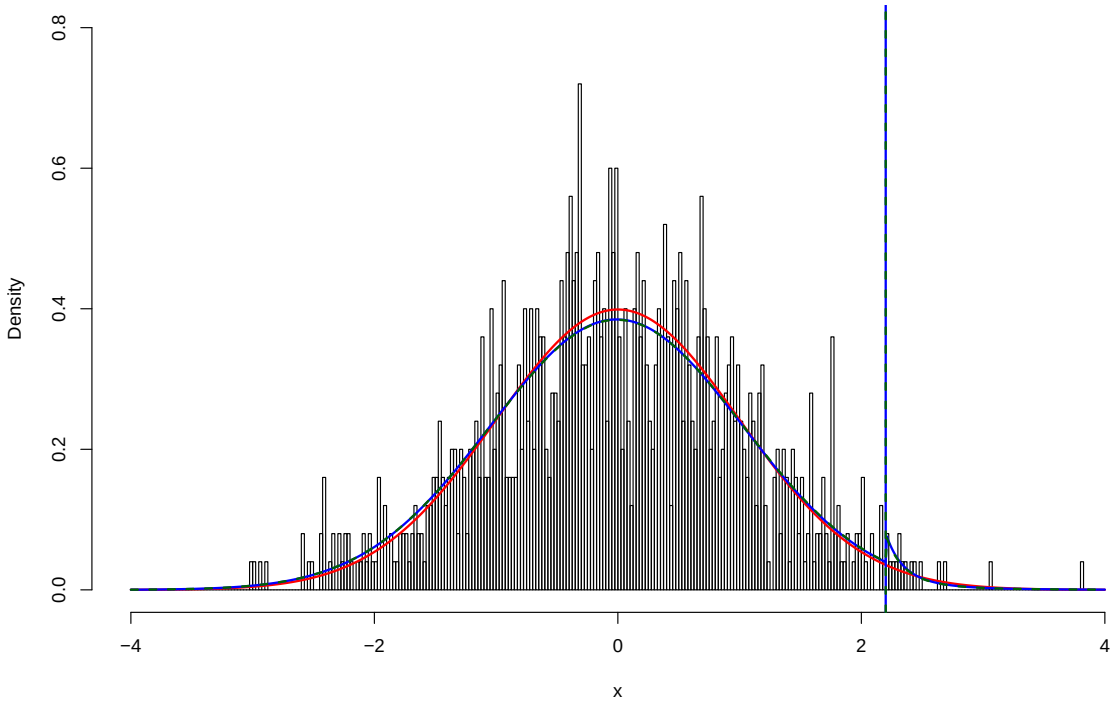
Tail Fraction	$\hat{\mu}$	$\hat{\sigma}$	$\hat{u}$	$\hat{\sigma_u}$	$\hat{\xi}$
Bulk	-0.011 (0.033)	1.036 (0.023)	2.2	0.203 (0.086)	0.350 (0.353)
Parameterised	-0.009 (0.036)	1.039 (0.028)	2.2	0.203 (0.086)	0.350 (0.353)

(b) `set.seed(7)`

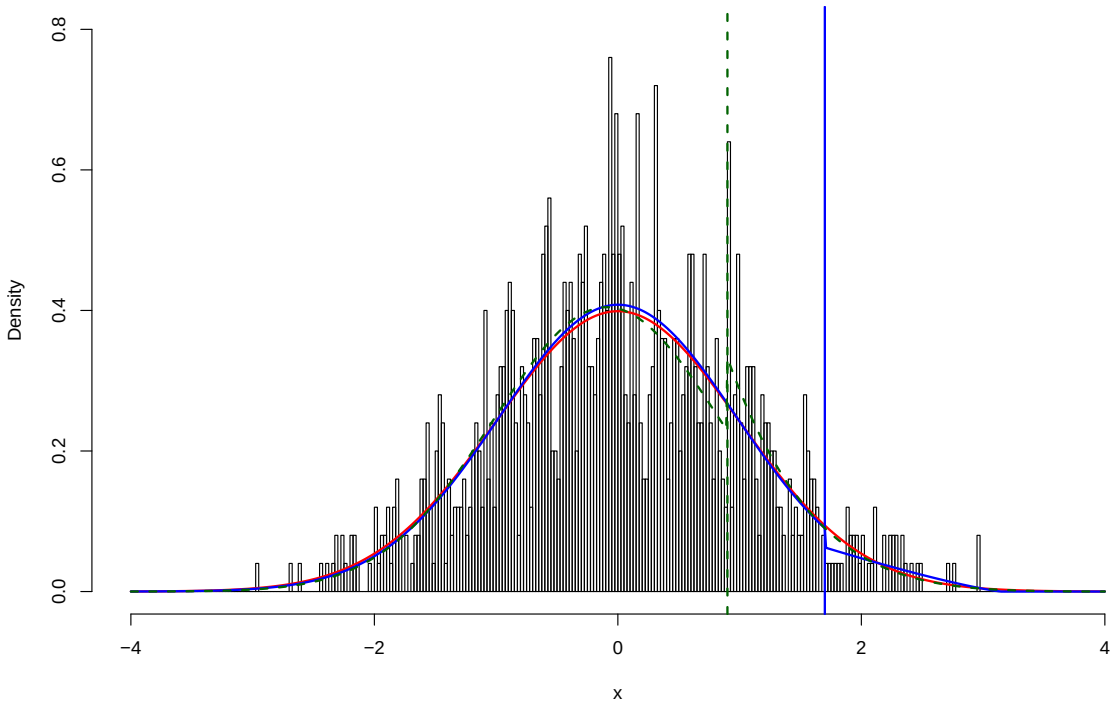
Tail Fraction	$\hat{\mu}$	$\hat{\sigma}$	$\hat{u}$	$\hat{\sigma_u}$	$\hat{\xi}$
Bulk	0.001 (0.031)	0.977 (0.023)	1.7	0.655 (0.131)	-0.451 (0.143)
Parameterised	-0.091 (0.056)	0.927 (0.039)	0.9	0.583 (0.062)	-0.152 (0.079)

TABLE 1. Estimated parameters (and standard error) for the normal+GPD model with seeds 1 and 7, with bulk model and parameterised tail fractions. The threshold is estimated using profile likelihood estimation but no standard is provided.

```
# density histogram
hist(x, freq = FALSE, breaks = seq(-4, 4, 0.025), ylim = c(0, 0.8))

# Overlay population normal distribution
x.to.plot = seq(-4, 4, 0.01)
lines(x.to.plot, dnorm(x.to.plot), col = "red", lwd = 2)

# Overlay fitted normal+GPD densities
fx.bulk = with(fit.bulk, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi))
lines(x.to.plot, fx.bulk, col = "blue", lwd = 2)
abline(v = fit.bulk$u, col = "blue", lwd = 2)

fx.para = with(fit.para, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi, phiu))
lines(x.to.plot, fx.para, col = "darkgreen", lty = 2, lwd = 2)
abline(v = fit.para$u, col = "darkgreen", lty = 2, lwd = 2)

with(fit.bulk, rbind(mle, se))
with(fit.para, rbind(mle, se))
```



(a) `set.seed(1)`



(b) `set.seed(7)`

FIGURE 1. Density histogram of 1,000 simulated standard normal variates overlaid with the true density function (red solid line), with different seeds. The fitted density function of the normal+GPD with bulk model based (blue solid line) and parameterised tail fraction (green dashed line).

Notice that the shape (and scale) parameter estimates of the GPD exhibit a large uncertainty, due to high threshold which leads to little tail observations. Model fit diagnostics are obtained for the bulk model based tail fraction with the following code. The default settings of the `evmix.diag` function set the axis scales to focus on only the upper tail regions of each diagnostic plot, as demonstrated by the first line. This default behaviour is overridden in the second line using the option `upperfocus = FALSE`, which allows each diagnostic plot to cover the full range of values. The diagnostic plots in Figure 2(a) indicate no issues with the upper tail fit and Figure 2(b) indicates an adequate fit over the observed range of support.

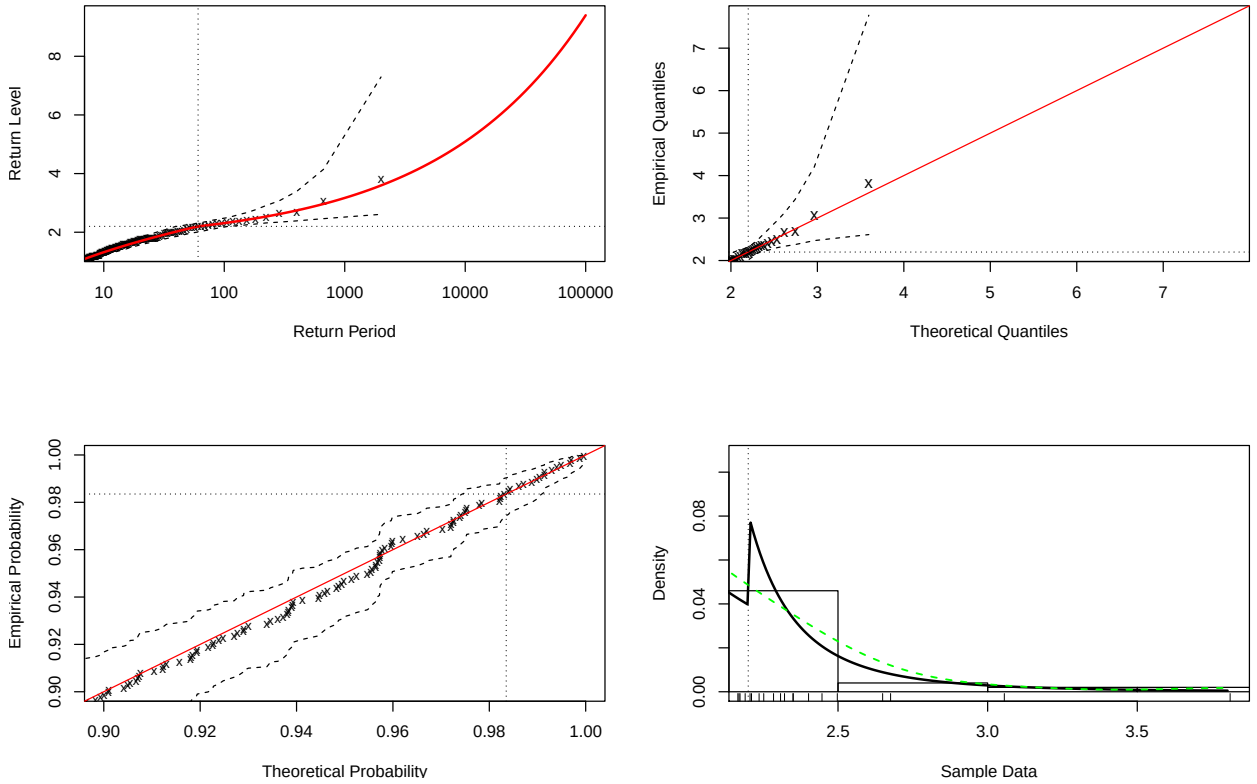
```
# Model fit diagnostics
evmix.diag(fit.bulk)
evmix.diag(fit.bulk, upperfocus = FALSE)
```

It is often the case that the estimated thresholds (and thus the other parameters) for the bulk model based and parameterised tail fraction approaches are different. This is most easily demonstrated from re-running the above simulation code with a different `seed()`. Figure 1(b) and Table 1(b) shows the resultant fitted density with a seed of 7. The two variants have very different estimated thresholds, but provide a similar fit to the upper tail over the range of support shown. However, the differing shape parameter estimates so will have rather different extrapolations. Notice that the uncertainty in the shape and scale parameter estimates is much lower, due to the lower threshold and therefore larger sample of tail observations.

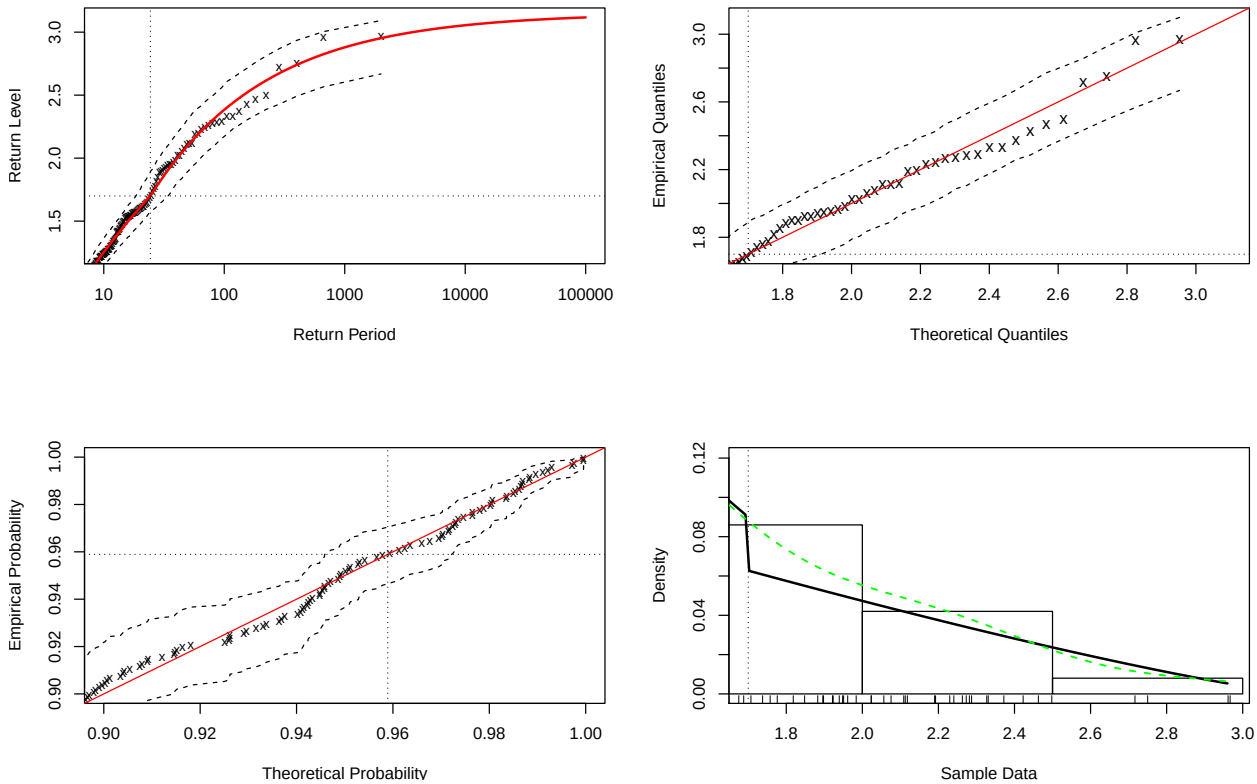
4.1.1. *Discussion of Correct Bulk Model Results.* The bulk model is the same as the population that the data are simulated from, so one would expect the bulk model based tail fraction should be a good estimator. The bulk model based tail fraction uses the estimated parameters for the bulk model which are based on the (usually) ample information below the threshold. Hu (2013a) showed via simulation from many different populations, that there is a small benefit from using the bulk model based tail fraction if the bulk model is known to be correct, or if there is sufficient data to demonstrate that it is providing a good fit. However, in general for extreme value applications the bulk distribution is rarely known and data is often limited. As the benefit of the bulk model based tail fraction is small, then it is safest to use the parameterised tail fraction approach. This advice is demonstrated via simulation by Hu (2013a) and briefly shown using a data example in Section 4.3.1 below.

If the bulk model is the same as the population then, in some sense, there is no need for a GPD tail model at all. As far as the author is aware, no-one has considered the relative benefits of using a the true population model versus an extreme value mixture model, with the population distribution as the bulk model and the usual GPD tail model, for estimation of tail quantities in finite samples. One of the motivating factors for using extreme value models is that they are designed to perform well for tail estimation. It is intuitive that the performance of either model will depend on the estimation approach and in particular it's influence function.

One would expect that for large samples that the true population model will perform best in general, as any asymptotic approximation error of the GPD will depreciate the performance of the extreme value mixture model (unless the tail is exactly a GPD). However, in the finite sample case the performance of either model for tail estimation is less obvious. In the above simulation the data are simulated from a normal population,



(a) `upperfocus = TRUE` (default)



(b) `upperfocus = FALSE`

FIGURE 2. Standard model fit diagnostics for the upper tail of the bulk model based tail fraction normal+GPD model fitted to the simulated data in Figure 1(a).

for which the upper tail is known to be relatively slow to converge to the exponential limit (Coles, 2001), so it is not entirely obvious as to whether using the true normal distribution as the model or normal+GPD mixture model will performance best for tail estimation purposes. In real life applications, the population distribution is rarely known, at which this issue becomes moot.

There also appears to have been no substantive study of the performance of hypothesis testing in deciding if a bulk only model is sufficient or if there is evidence for the need for the extreme value mixture model, with consideration of the resulting impacts on tail inference.

If the bulk model is the same as the population model, and the population is in the domain of attraction of the GPD, then a wide range of thresholds may be able to provide similar fitting performance. We should therefore expect a high variance in the threshold estimates and will often observe multiple modes in the (profile) likelihood, and presumably any other objective function used for estimation purposed. When using traditional graphical diagnostics for GPD threshold choice (see main Chapter) users can subjectively balance the benefits of reduced estimation variance by decreasing the threshold, or reduce the bias induced by the GPD tail approximation by increasing the threshold. The user aims to choose the lowest threshold for which the GPD provides a suitable fit. The author is unaware of any attempts to adapt standard inference approaches (like maximum likelihood estimation or method of moments) to select the lowest threshold that provides "close to" optimal performance, e.g. to choose the lowest threshold such that the likelihood is within some small ϵ of the maximum likelihood.

It is also worthwhile being aware that although the threshold potentially has an impact over entire range of support (i.e. both the bulk and tail fits), it is also a very "localised degree of freedom" so adds great flexibility in the fitted distribution around the threshold. This feature was highlighted by MacDonald et al. (2011). The consequence of this localised degree of freedom is that the threshold values chosen by the inference schemes are often associated with the largest deviations in sample density/distribution function due to natural sample variation. This feature also leads to the commonly observed multimodal objective functions (e.g. multiple modes in the likelihood or Bayesian posterior), for which examples are provided within the references given in the main Chapter.

The local degree of freedom applied by the threshold has strongly influenced the estimates in some of the examples presented in this supplementary material. For example, in Figure 1(b) the parameterised tail fraction based fit has a threshold (vertical dashed line) next to an unusually large change in the sample density. On the left of the threshold the density histogram estimator is relatively low (well below the red line) and to the right of the threshold it is relatively high. The local flexibility has allowed the bulk model to have density which is rather low just below the threshold and the GPD tail model rather high above the threshold, to match what is observed in the sample data. Similar deviations can be observed in the cumulative distribution function and quantile function about the threshold.

These sample density/distribution function variations are purely due to natural sample variation. A comparison of the sensitivity of alternative threshold estimation procedures and the thresholds selected by extreme value mixture models to these natural deviations has not been well studied. It is intuitive that traditional graphical diagnostics will also exhibit deviations around these locations (e.g. the MLE of the shape may change markedly

on either side of a location of high density), so users may be visually drawn to these locations when making subjective assessments for threshold choice.

Of course, the extent of this localised effect is reduced if smoothness constraints (e.g. continuity in zeroth or first) on the pdf at the threshold are used. However, these can lead to further lack of robustness of the tail fit to that the bulk, as information is passed between the bulk and tail model for parameter inference, as discussed in the main Chapter.

Obviously, when choosing the subset of the thresholds to apply the grid search over the profile likelihood it is important to try a physically relevant range, with a sufficient fine subdivision. In the ideal world one would use an numerical optimisation routine to find the threshold that maximises the profile likelihood. Unfortunately, due to the multiple modes that are often present in the profile likelihood (see main Chapter for discussion) standard optimisers often get stuck in a local mode and so often fail to find the global maximum. In applying the grid search of profile likelihood, one should always check if the estimated threshold is close to the boundaries in the support of grid of thresholds, as this could be an indication of the grid support not being wide enough or a problem with the model or it's fit to the sample data which is demonstrated in Section 4.3.1.

Rounding of the data can also create problems for (profile) likelihood maximisation. Data generating processes which lead to isolated sample data values with high frequency, relative to the surrounding density, are particularly problematic. A common issue demonstrated in Section 4.3.2 is such a bias in threshold choice due to excess zeroes. If the bulk model cannot capture the excess zeroes, then the threshold is often drawn to zero as the extra "local degree of freedom" enable the overall mixture model to capture the excess sample frequency around zero.

4.2. **Do Not Use Defaults!** Currently, the default approach within the fitting function (e.g. by `fnormgpd` used above) is to use MLE for all the parameters, including the threshold. In the main Chapter this objective function is described as the complete likelihood. Unfortunately, this is not ideal as the likelihood function is often multi-modal which is an expected and inherent feature of these models due to the threshold stability properties of the GPD, see the main Chapter for further details. The complete likelihood optimisation often gets stuck in a local mode close to the initial parameter values.

In particular, the multi-modality is often due to multiple different threshold providing suitable fit. As such the estimated threshold is often close to the initial value, which by default is the 90% sample quantile estimated using the `quantile()` function.

There are numerous approaches to overcome this problem. The obvious one is to use an optimisation scheme which can better cope with potentially multi-modal objective functions. In general, Hu (2013a) found the quasi-Newton "`BFGS`" performed well compared to the other inbuilt optimisation schemes within the `optim` function, namely the "`Nelder-Mead`" and conjugate gradient "`CG`" approaches. The `BFGS` optimisation scheme is therefore used by default throughout the `evmix` package.

However, all of the optimisation schemes (inlcuding `BFGS`) within the `optim` were found to frequently get stuck in local modes and so not always find the global maximum. The only other viable alternative within the `optim` function is the stochastic optimiser which uses simulated-annealling, "`SANN`", which performed better but was too slow for practical

purposes. Any of the optimisation schemes available in the `optim` function can be used by setting the `method` argument in the fitting function.

Another approach is to manually try multiple different starting values for the optimisation, using the `pvector` option.

The current default setting is a compromise and is certainly not ideal, so is a focus of future research. As all the code in the `evmix` library is user-visible, it is possible for you to adapt the fitting functions to use your favourite optimisation routine.

4.2.1. *Profile Likelihood Grid Search is Better.* All of the fitting functions include options to use grid search over the profile likelihood to find the threshold that maximises the complete likelihood. The details of which are given in Section 2 above. In summary, the user specifies a vector of thresholds to consider (`useq` for one tail, `ulseq` and `urseq` for lower and upper tail threshold for two tailed models) and the value which maximises the profile likelihood provides the threshold estimate. The user can then choose to either fix the threshold at the estimated value (`fixedu=TRUE`) or use the estimated value as the initial value for the maximisation of the complete likelihood (`fixedu=FALSE`).

The current best practice for MLE of the mixture mdoel parameters, is that one should be use the profile likelihood estimation of the threshold, which is then fixed at the value so that the maximum likelihood estimation is over all the non-threshold parameters conditional on the threshold. Unfortunately, it is not straightforward to make this the default option in all the fitting functions within the package, as for some of the mixture models it is not obvious how to prescribe the vector of thresholds which would need to be considered by default. This is part of ongoing developments of the package.

```
set.seed(7)
x = rnorm(1000)

# fit normal bulk with GPD upper tail using profile likelihood
fit.bulk = fnormgpd(x, useq = seq(0, 2.5, 0.1), fixedu = TRUE)

# fit normal bulk with GPD upper tail using complete likelihood (default)
fit.comp = fnormgpd(x)

# density histogram
hist(x, freq = FALSE, breaks = seq(-4, 4, 0.025), ylim = c(0, 0.8))

# Overlay population normal distribution
x.to.plot = seq(-4, 4, 0.01)
lines(x.to.plot, dnorm(x.to.plot), col = "red", lwd = 2)

# Overlay fitted normal+GPD densities
fx.bulk = with(fit.bulk, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi))
lines(x.to.plot, fx.bulk, col = "blue", lwd = 2)
abline(v = fit.bulk$u, col = "blue", lwd = 2)

# initial value is 90% sample quantile
abline(v = quantile(x, 0.9), col = "red", lwd = 2)

fx.comp = with(fit.comp, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi, phiu))
lines(x.to.plot, fx.comp, col = "darkgreen", lty = 2, lwd = 2)
abline(v = fit.comp$u, col = "darkgreen", lty = 2, lwd = 2)

-c(fit.bulk$nllh, fit.comp$nllh)
```

As an example, the above code reuses one of the simulation examples above and compares the results to those achieved when refit using the (default) complete likelihood approach. The resultant fitted density function is shown in Figure 3. The initial value for the threshold parameter in the complete likelihood approach is the 90% sample quantile which is 1.24 (vertical red line), with the final maximum likelihood estimate of 1.31 (vertical green line) which has barely changed. The complete likelihood approach's log-likelihood (given by the final line of code) is -1398.565 as compared to that obtained when the threshold is fixed at the value that maximises the profile likelihood of -1398.524. So the likelihood itself has not much. However, the fit from the complete likelihood is clearly worse than from profile likelihood. If one repeats this exercise with the `set.seed(1)` as used above, the threshold moves even less from it's initial value.

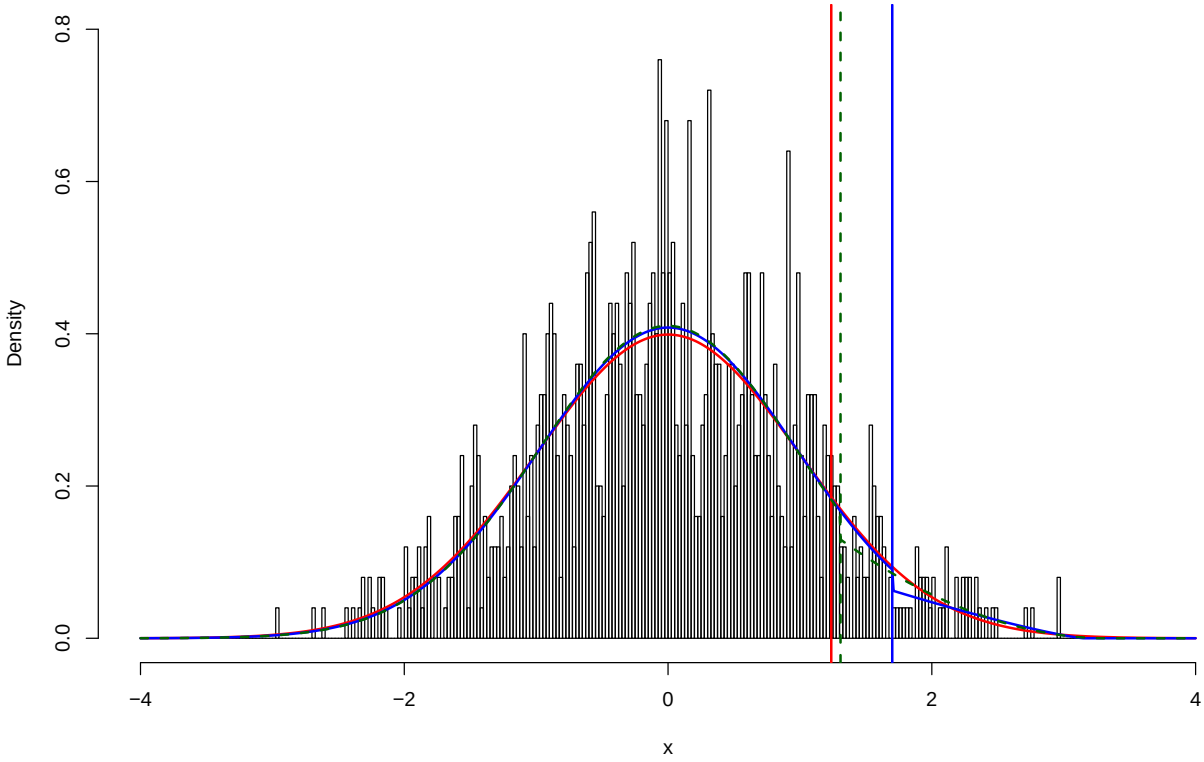


FIGURE 3. Density histogram of 1,000 simulated standard normal variates overlaid with the true density function (red solid line). The fitted density function of the normal+GPD with bulk model based tail fraction (blue solid line) using the profile likelihood estimate of the threshold and complete likelihood (green dashed line). Initial value for the threshold (90% sample quantile) for complete likelihood as (vertical red line).

4.3. **When the Bulk Model is Not Appropriate.**

4.3.1. *Standard and Poor Returns.* The classic extreme value dataset of Standard and Poor daily log returns `spto87` from the `evir` package neatly demonstrates some of the issues that arise when using an inappropriate bulk model. The time series is shown in Figure 4. These exhibit the expected features of financial returns, in particular the volatility clustering, but these complexities are ignored in the following.

```
data(spto87, package = "evir")

# time series plot
plot(attr(spto87, "times"), spto87, type = "l",
     xlab = "Date", ylab = "Daily log Returns",
     main = "Standard and Poor Daily Closing log Returns from evir Package")
```

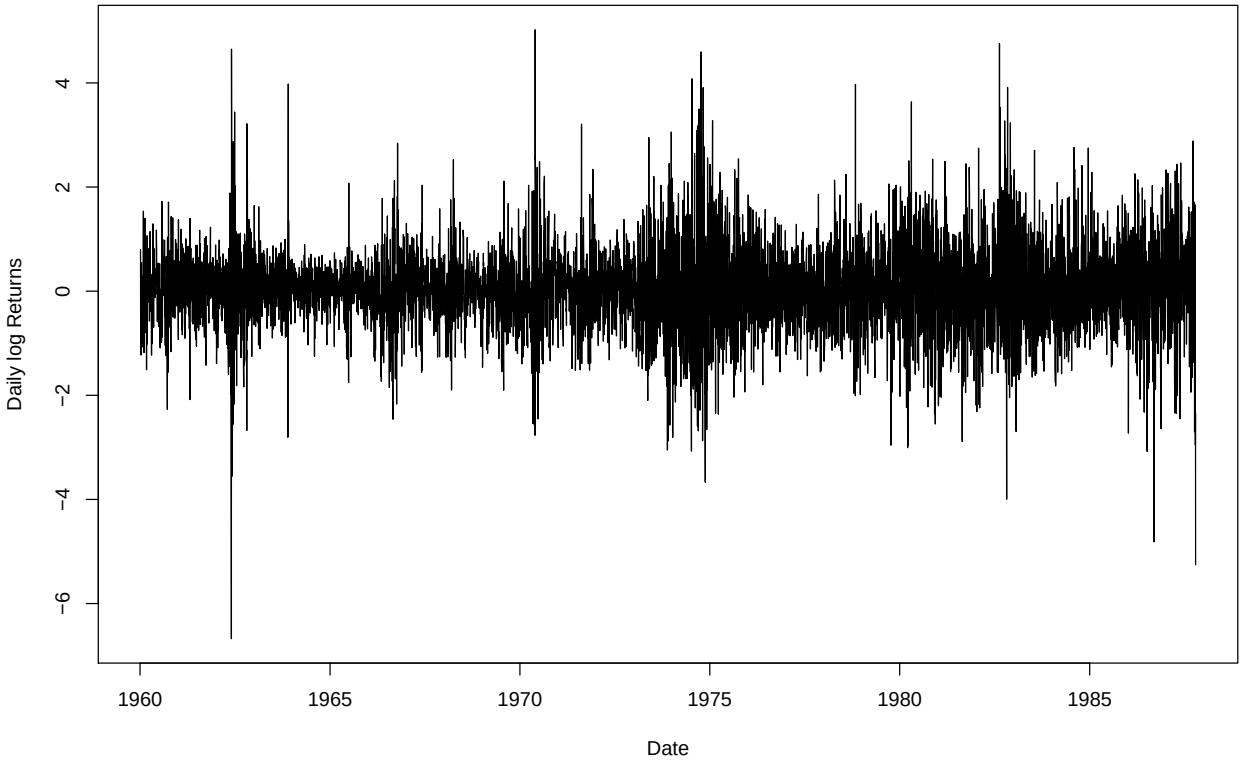


FIGURE 4. Time series plot of the Standard and Poor's daily log returns dataset `spto87` from the `evir` package.

The following code fits a normal distribution as well as the the normal+GPD mixture model, with both specifications of the tail fraction as used with the simulated data in Section 4.1. The normal parameters are estimated by the usual unbiased estimators via the `mean(spto87) = 0.025` and `sd(spto87) = 0.805` functions. The normal distribution is a very poor fit to this data as shown by the red line in Figure 4, as the tails are much heavier than those a normal.

As the bulk model is not a good model for the data (including lower tail), the normal+GPD is also a poor fit. Notice that the bulk model based (blue solid line) and parameterised tail fraction (green dashed line) provide very different fits, particularly below the threshold.

As expected the bulk model based tail fraction has a similar fit to the bulk to that of the normal distribution (blue line close to red line), below the threshold. The estimated normal bulk parameters are $\hat{\mu} = 0.032$ and $\hat{\sigma} = 0.782$ which are similar to the usually unbiased estimators of the normal distribution parameters given above. There is no-rescaling of the bulk model density when the bulk model based tail fraction is used, which explains why it has barely changed.

```
# fit normal bulk with GPD upper tail (bulk model based tail fraction)
fit.bulk = fnormgpd(spto87, useq = seq(0.2, 2, 0.1), fixedu = TRUE)

# fit normal bulk with GPD upper and lower tails (parameterised tail fraction)
fit.para = fnormgpd(spto87, phiu = FALSE, useq = seq(0.2, 2, 0.1), fixedu = TRUE)

# density histogram
hist(spto87, freq = FALSE, breaks = seq(-7, 7, 0.05), ylim = c(0, 0.8),
     xlab = "Daily Closing Returns", main = "Normal+GPD Upper Tail")

# Overlay normal distribution with unbiased parameter estimates
x.to.plot = seq(-7, 7, 0.01)
lines(x.to.plot, dnorm(x.to.plot, mean(spto87), sd(spto87)), col = "red", lwd = 2)

# Overlay fitted normal+GPD densities
fx.bulk = with(fit.bulk, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi))
lines(x.to.plot, fx.bulk, col = "blue", lwd = 2)
abline(v = fit.bulk$u, col = "blue", lwd = 2)

fx.para = with(fit.para, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi, phiu))
lines(x.to.plot, fx.para, col = "darkgreen", lty = 2, lwd = 2)
abline(v = fit.para$u, col = "darkgreen", lty = 2, lwd = 2)
```

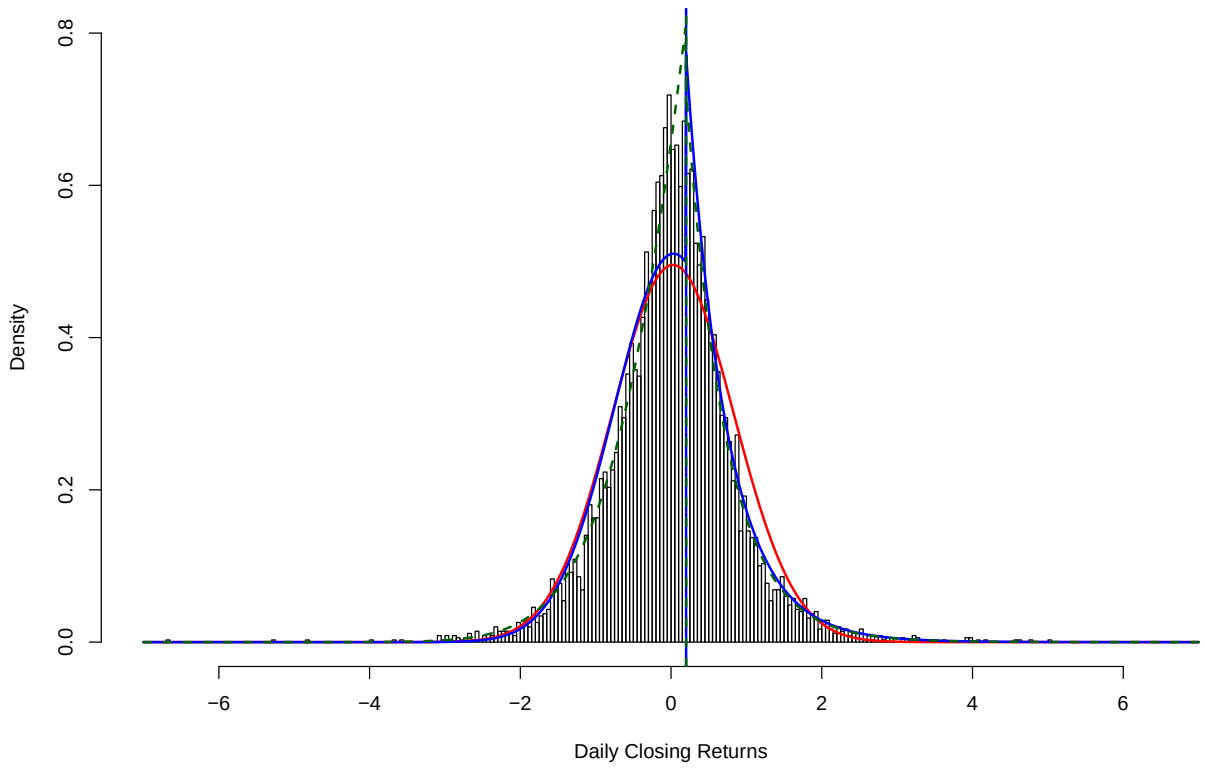


FIGURE 5. Density histogram of Standard and Poor's daily log returns dataset `spto87` from the `evir` package. A fitted normal distribution is shown by (red solid line). The fitted density function of the normal+GPD with bulk model based (blue solid line) and parameterised tail fraction (green dashed line).

In contrast, the parameterised tail fraction approach rescales the bulk density, see first line of equation 5 in the main Chapter. As such an extra degree of freedom is provided,

which effects both the bulk and tail model fit. In this example, the parameterised tail fraction provides a much better fit below the selected threshold. Given the tail fraction is not influenced by the poor bulk model fit, we would also expect the tail performance would be better than the bulk model based tail fraction.

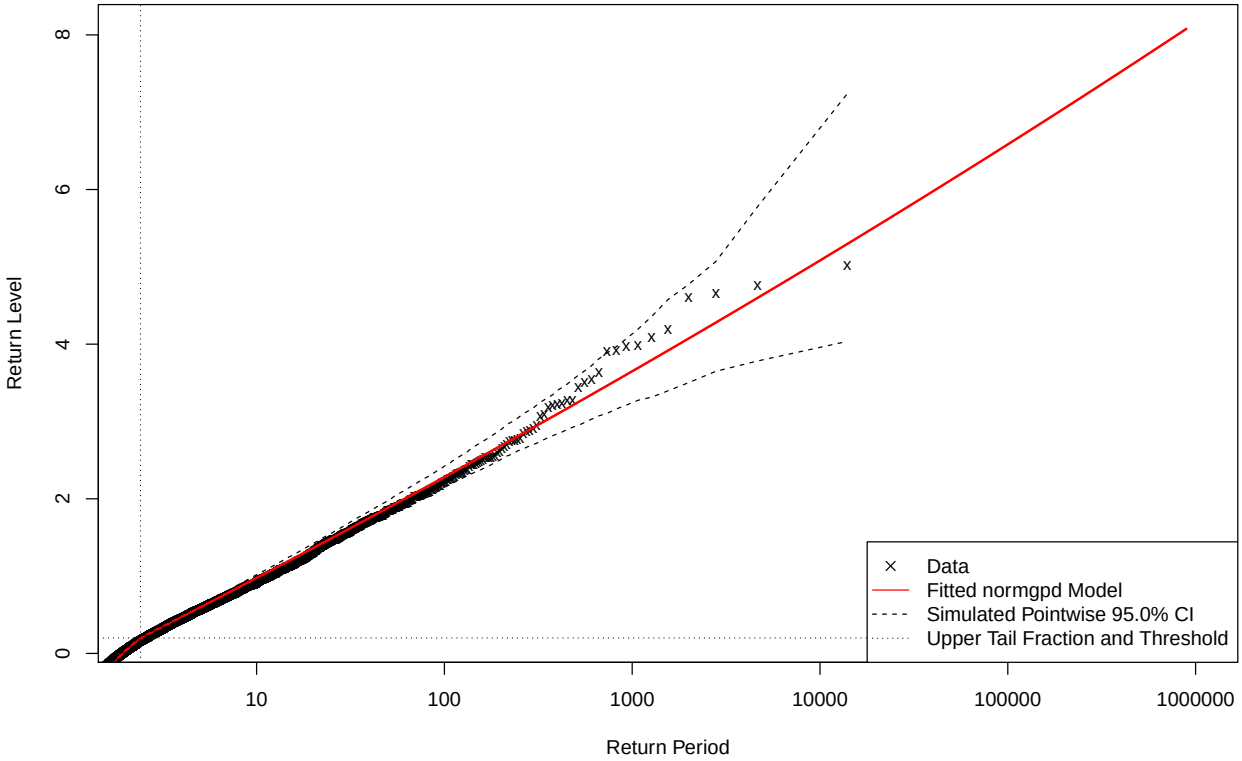
The following code produces return level plots for the two model fits for comparison of the tail performance. Both the bulk model based and parameterised tails fraction approaches produce similar fits for the highest return periods. However, nearer the threshold (horizontal dashed lines) the fit is much better for the parameterised tail fraction approach (datapoints closer to fitted red line) than for the bulk model based tail fraction. The extra tail fraction parameter is essentially able to overcome the deficiencies of the bulk model which cannot provide a good tail fraction estimate, to get the model "back on track" above the threshold.

```
par(mfrow = c(2, 1), mar = c(3, 3, 1, 1) + 0.1, mgp = c(2, 1, 0))
rlplot(fit.bulk)
rlplot(fit.para)
```

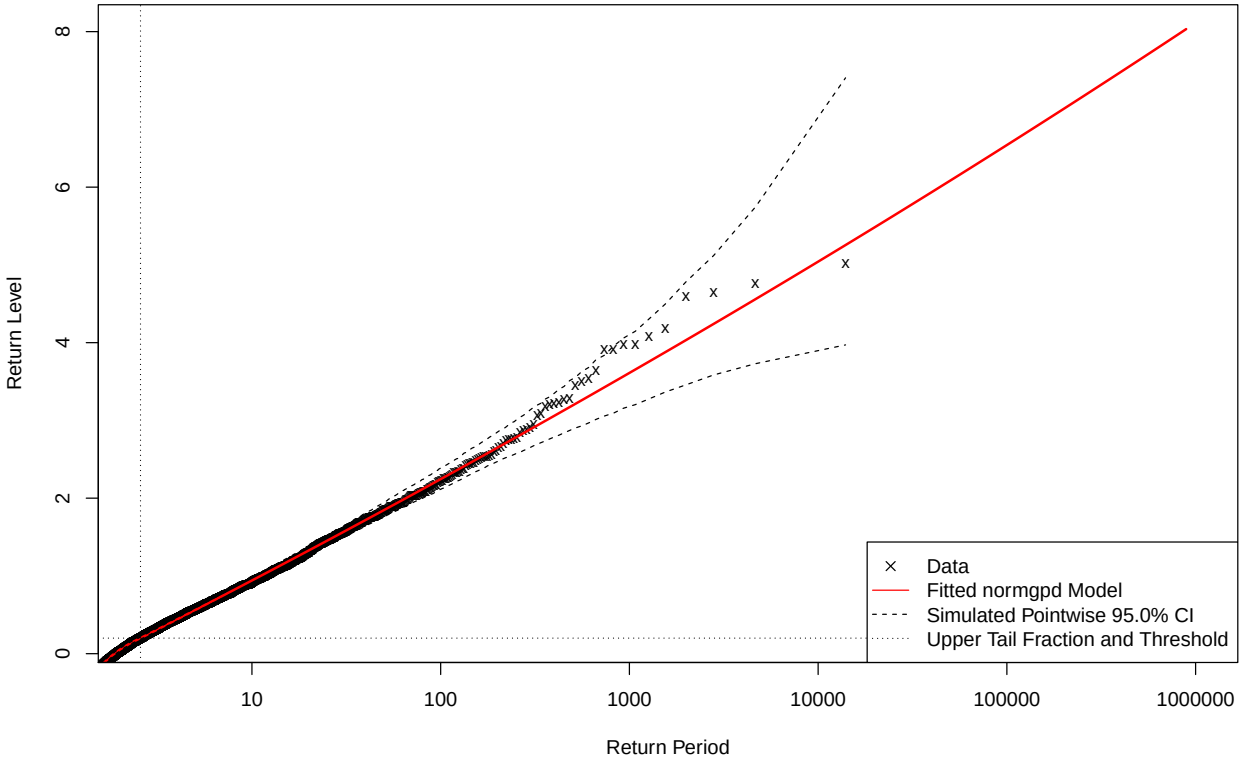
Another feature of the poor performance to note here is that the threshold is far lower (closer to the mode) than one would expect. In fact, the estimated thresholds are on the lower bound of those considered in the profile likelihood grid search. This unexpected result is induced by the poor performance of the bulk model. The threshold is biased downwards to allow the GPD to overcome as much of the poor fit as it is capable. Such a result is a common indicator of a poor fit (a warning message is provided by the fitting functions) and in particular a poor bulk model, so users should check the model performance and re-evaluate their choice of model.

For this dataset, a better model is obtained by splicing a GPD for both tails, as it is the heavy (upper and lower) tailed nature of the data that results in the normal distribution providing a poor fit. Example code for fitting the GPD+Normal+GPD mixture model is provide below, with the fitted density in Figure 7. The fit is much better for both tails and the bulk density between the two thresholds. Hence, the threshold parameter are bias towards the main mode and so the GPD fits are performing well.

```
# fit normal bulk with GPD upper and lower tails (bulk model based tail fraction)
fit.gng = fgng(spto87, ulseq = seq(-2, -0.2, 0.1), urseq = seq(0.2, 2, 0.1), fixedu = TRUE)

# density histogram
hist(spto87, freq = FALSE, breaks = seq(-7, 7, 0.05), ylim = c(0, 0.8),
     xlab = "Daily Closing Returns", main = "Normal+GPD Upper Tail")

# Overlay fitted GNG density
fx.gng = with(fit.gng, dgng(x.to.plot, nmean, nsd, ul, sigmaul, xil, phiul = TRUE,
                                                ur, sigmaur, xir, phiur = TRUE))
lines(x.to.plot, fx.gng, lwd = 2)
abline(v = c(fit.gng$ul, fit.gng$ur), lwd = 2)

# model fit diagnostics
evmix.diag(fit.gng)
```



(a) Bulk model based tail fraction



(b) Parameterised tail fraction

FIGURE 6. Return level plot for normal+GPD model fits with bulk model based tail fraction (upper plot) and parameterised tail fraction (lower plot).

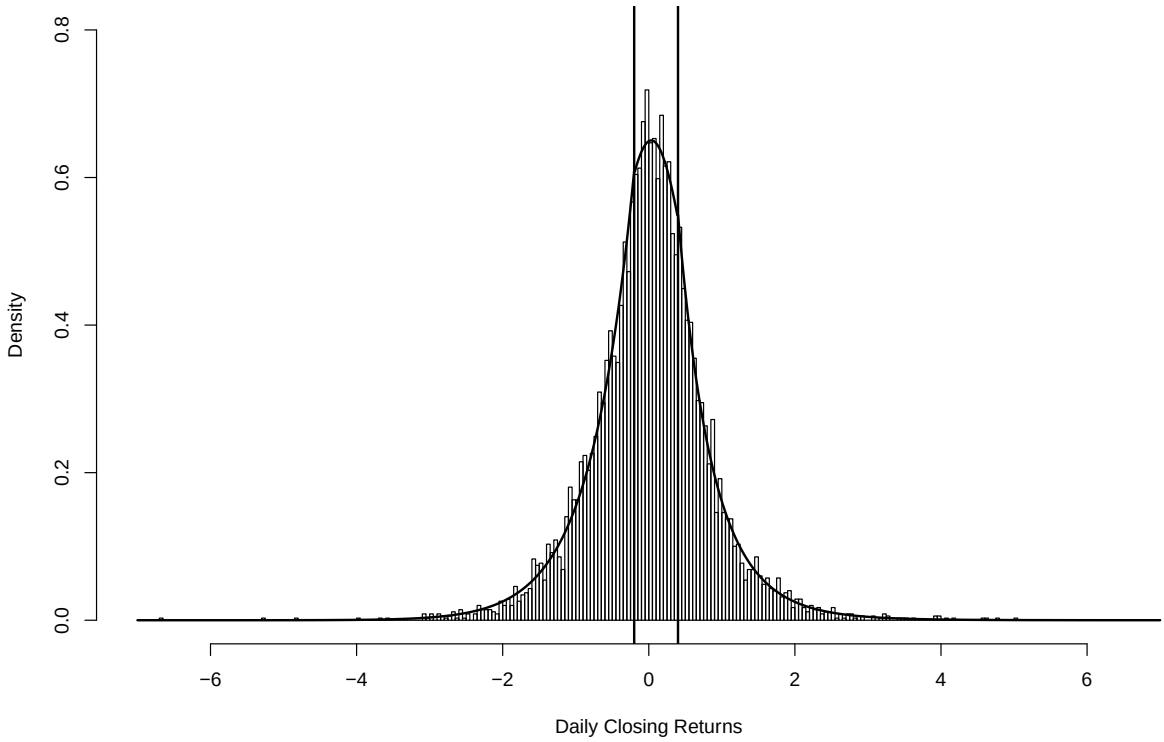


FIGURE 7. Density histogram of Standard and Poor returns from 1960 to 1987 from `evir` package. The black solid line is the fitted density function of the GPD+normal+GPD with bulk model based tail fraction. The vertical lines are the estimated thresholds

4.3.2. *Dow Jones Returns.* The Dow Jones data `dowjones` available in the `ismev` package provides another good example of the challenges associated with fitting extreme value mixture models. The following code plots the time series data.

```
data(dowjones, package = "ismev")

par(mfrow = c(2, 1))
# time series plot
plot(dowjones$Date, dowjones$Index, type = "l",
     xlab = "Date", ylab = "Daily Closing Price",
     main = "Dow Jones Daily Closing Price from ismev Package")

returns = log(dowjones$Index[-1]) - log(dowjones$Index[-length(dowjones$Index)])

plot(dowjones$Date[-1], returns, type = "l",
     xlab = "Date", ylab = "Daily log Returns",
     main = "Dow Jones Daily Closing log Returns from ismev Package")
```

Example code for fitting the normal+GPD mixture models with bulk model based and parameterised tail fraction approaches is provided below. Similar comments about the poor bulk and tail model fit to that observed for the Standard and Poor data in Section 4.3.1 also apply to this data, so will not be repeated here. The normal+GPD with parameterised tail fraction has a threshold on the boundary of those considered in the profile likelihood, which is indicative of similar deficiencies. The reason for the deficiency with this dataset is that there is not only heavier tails than expected for a normal population, but also there is an excess of zero and near-zero log returns.

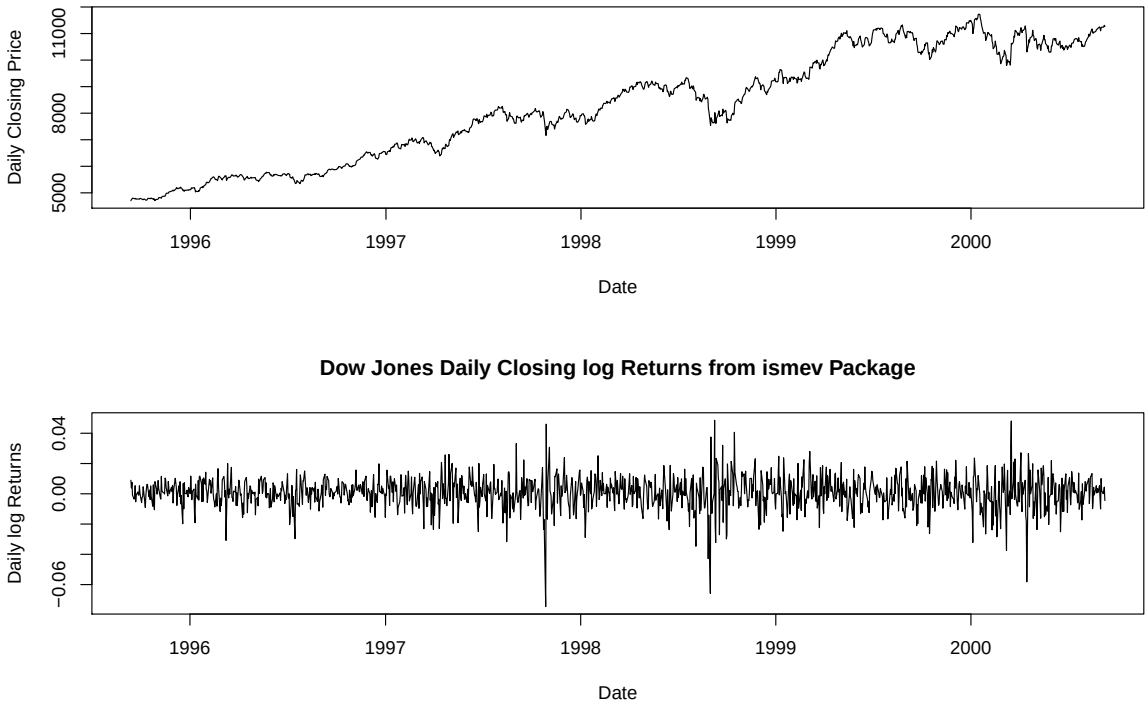


FIGURE 8. Time series plot of the Dow Jones daily closing price (upper plot) and log returns (lower plot) dataset `dowjones` from the `ismev` package.

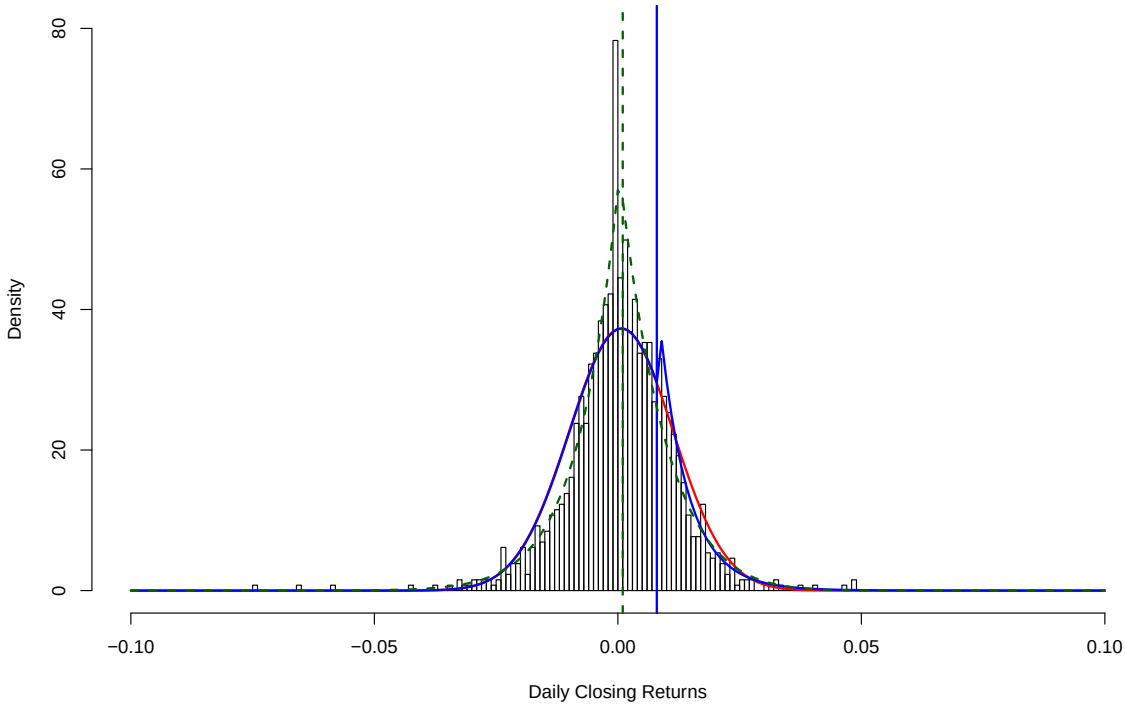


FIGURE 9. Density histogram of Dow Jones daily log returns dataset `dowjones` from the `ismev` package. A fitted normal distribution is shown by (red solid line). The fitted density function of the normal+GPD with bulk model based (blue solid line) and parameterised tail fraction (green dashed line).

```
data(dowjones, package = "ismev")

par(mfrow = c(2, 1))
# time series plot
plot(dowjones$Date, dowjones$Index, type = "l",
     xlab = "Date", ylab = "Daily Closing Price",
     main = "Dow Jones Daily Closing Price from ismev Package")

returns = log(dowjones$Index[-1]) - log(dowjones$Index[-length(dowjones$Index)])

plot(dowjones$Date[-1], returns, type = "l",
     xlab = "Date", ylab = "Daily log Returns",
     main = "Dow Jones Daily Closing log Returns from ismev Package")

# fit normal bulk with GPD upper tail (bulk model based tail fraction)
fit.bulk = fnormgpd(returns, useq = seq(0, 0.02, 0.001), fixedu = TRUE)

# fit normal bulk with GPD upper and lower tails (parameterised tail fraction)
fit.para = fnormgpd(returns, phiu = FALSE, useq = seq(0, 0.02, 0.001), fixedu = TRUE)

# density histogram
par(mfrow = c(1, 1))
hist(returns, freq = FALSE, breaks = seq(-0.1, 0.1, 0.001), ylim = c(0, 80),
     xlab = "Daily Closing Returns", main = "Normal+GPD Upper Tail")

# Overlay normal distribution with unbiased parameter estimates
x.to.plot = seq(-0.1, 0.1, 0.001)
lines(x.to.plot, dnorm(x.to.plot, mean(returns), sd(returns)), col = "red", lwd = 2)

# Overlay fitted normal+GPD densities
fx.bulk = with(fit.bulk, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi))
lines(x.to.plot, fx.bulk, col = "blue", lwd = 2)
abline(v = fit.bulk$u, col = "blue", lwd = 2)

fx.para = with(fit.para, dnormgpd(x.to.plot, nmean, nsd, u, sigmau, xi, phiu))
lines(x.to.plot, fx.para, col = "darkgreen", lty = 2, lwd = 2)
abline(v = fit.para$u, col = "darkgreen", lty = 2, lwd = 2)
```

The excess zeroes are not well captured by the bulk model, so even the two-tailed GPD+Normal+GPD fitted with the following code struggles to provide a good fit as shown in Figure 10. It may provide an adequate fit for application, but it is clearly not ideal. The obvious solution is to consider a bulk model which allows for excess zero (and near-zero) returns, but no such model has yet been considered in the literature on extreme value mixture models.

```
# fit normal bulk with GPD upper and lower tails (bulk model based tail fraction)
fit.gng = fgng(returns, ulseq = seq(-0.02, 0, 0.001), urseq = seq(0, 0.02, 0.001), fixedu = TRUE
    )

# density histogram
hist(returns, freq = FALSE, breaks = seq(-0.1, 0.1, 0.001), ylim = c(0, 80),
     xlab = "Daily Closing Returns", main = "Normal+GPD Upper Tail")

# Overlay fitted GNG density
fx.gng = with(fit.gng, dgng(x.to.plot, nmean, nsd, ul, sigmaul, xil, phiul = phiul,
                                                ur, sigmaur, xir, phiur = phiur))
lines(x.to.plot, fx.gng, col = "darksalmon", lwd = 2)
abline(v = c(fit.gng$ul, fit.gng$ur), col = "darksalmon", lwd = 2)
```

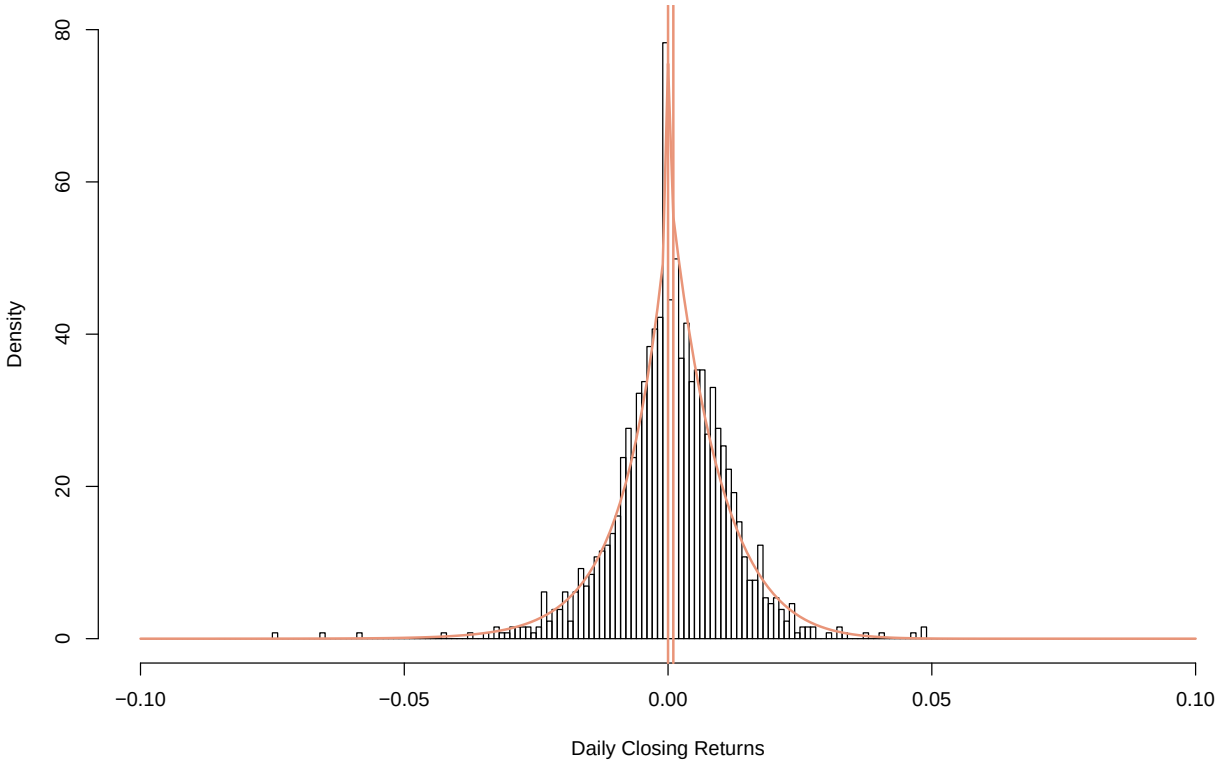


FIGURE 10. Density histogram of Dow Jones returns from `evir` package. The black solid line is the fitted density function of the GPD+normal+GPD with bulk model based tail fraction. The vertical lines are the estimated thresholds

4.4. **Don't Forget the Tail Fraction!** The alternative forms of extreme value mixture models which do not include the tail fraction scaling show the weak performance for general application, as discussed in Section 3.2 and 3.3 of the main Chapter. Essentially, these models treat the GPD as an unconditional model, which is used (almost) directly to capture the tail. In this Section, the hybrid PAreto is used as an example of the issues that arise when the tail fraction is ignored.

The following code compares the following models:

- hybrid Pareto (no tail fraction scaling), which includes constraints of continuity in the zeroth and first derivative;
- hybrid Pareto with single continuity constraint (no tail fraction); and
- normal+GPD with bulk model based tail fraction with single continuity constraint.

The only difference between the latter two models is that the former excludes the tail fraction scaling of the GPD (although the density it is renormalised to unity) and the latter included the tail fraction scaling (so does not require renormalisation to unity). Comparison of the latter two models therefore allows us to directly compare the effect of ignoring the tail fraction scaling of the GPD tail.

The hybrid Pareto has no explicit threshold parameter as this is constrained based on the model parameters, to satisfy the two continuity constraints. However, the implied

threshold can be calculated based on the parameter constraints given in Carreau and Bengio (2009). The implied threshold for the hybrid Pareto is included on the plot as vertical lines, similar to the explicit threshold of the normal+GPD.

```
set.seed(123)
x = rnorm(1000)

# Hybrid Pareto provides reasonable fit for some asymmetric heavy upper tailed distributions
# but not for cases such as the normal distribution
fit.hpd = fhpd(x, std.err = FALSE)

# HPD with only one constraint, with profile likelihood grid search over thresholds
fit.hpdcon = fhpdcon(x, useq = seq(-1, 3, 0.1), fixedu = TRUE)

# Notice that if tail fraction is included a better fit is obtained
fit.bulk = fnormgpdcon(x, useq = seq(-1, 3, 0.1), fixedu = TRUE)

# density histogram
hist(x, breaks = 100, freq = FALSE, xlim = c(-4, 4), main="")

# Overlay population normal distribution
x.to.plot = seq(-4, 4, 0.001)
lines(x.to.plot, dnorm(x.to.plot), col = "red", lwd = 2)

# Overlay fitted hybrid Pareto density
fx.hpd = with(fit.hpd, dhpd(x.to.plot, nmean, nsd, xi))
lines(x.to.plot, fx.hpd, col = "blue", lwd = 2)
abline(v = fit.hpd$u, col = "blue", lwd = 2) # implied threshold

# Overlay fitted hybrid Pareto with single continuity constraint density
fx.hpdcon = with(fit.hpdcon, dhpdcon(x.to.plot, nmean, nsd, u, xi))
lines(x.to.plot, fx.hpdcon, col = "black", lwd = 2)
abline(v = fit.hpdcon$u, col = "black", lwd = 2) # implied threshold

# Overlay fitted normal+GPD densities
fx.bulk = with(fit.bulk, dnormgpdcon(x.to.plot, nmean, nsd, u, xi))
lines(x.to.plot, fx.bulk, col = "green", lwd = 2)
abline(v = fit.bulk$u, col = "green", lwd = 2)

legend("topright", c("Standard Normal", "Hybrid Pareto (HPD)",
                     "HPD - Continuous Only ", "Normal+GPD Continuous"),
     lty = rep(1, 4), lwd = rep(2, 4),
     col = c("red", "blue", "black", "green"))
```

The domination of the GPD in the hybrid Pareto variants, due to the lack of tail fraction scaling of the GPD is evident, and the poor fit is very clear in Figure 12. The implied threshold for the hybrid Pareto is biased down to allow the GPD to dominate the density, with 63% of the density above the threshold in this case. The following code demonstrates that the upper tail fraction is just the reciprocal of the normalisation constant $\gamma = 1 + \Phi(\frac{u-\mu}{\sigma})$ needed to ensure the hybrid Pareto is proper:

```
# normalisation constant to unity
hpd.gamma = with(fit.hpd, 1 + pnorm(u, nmean, nsd))

# Upper tail fraction is reciprocal of gamma
c(1/hpd.gamma, fit.hpd$phiu, with(fit.hpd, phpd(u, nmean, nsd, xi, lower = FALSE)))
```

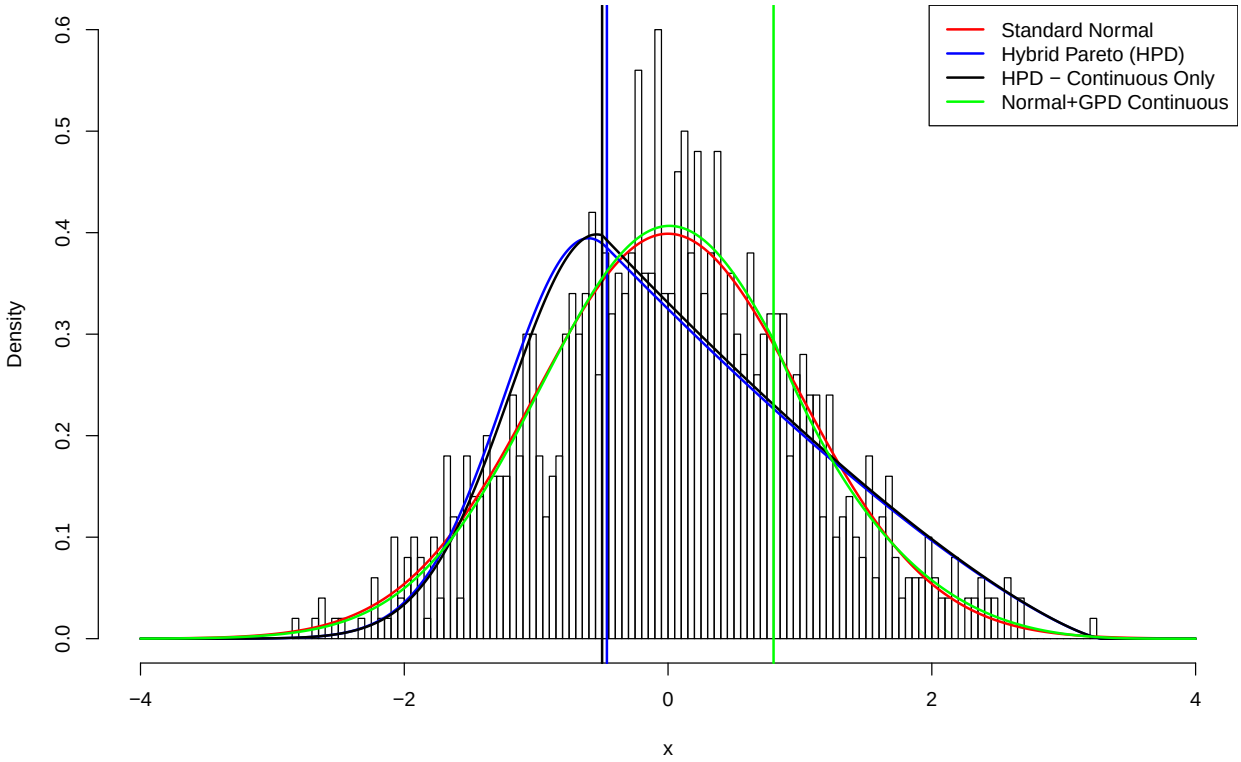


FIGURE 11. Density histogram of 1,000 simulated standard normal variates overlaid with the true density function (red solid line). The fitted density function of the hybrid Pareto (blue line) normal+GPD, hybrid Pareto with single continuity constraint (black line) and normal+GPD with bulk model based fraction (green line).

Similar results are obtained using the `condmixt` package, which is demonstrated with the following code. Be careful to `detach(package:condmixt)` and `detach(package:evd)` after running this example, to prevent any confusion caused by overlapping functions between the `condmixt`, `evd` and `evmix` libraries.

```
# Reproduce HPD fit using condmixt package
library(condmixt)
params.init = hpareto.mme(x)

fit.condmixt = hpareto.fit(params.init, x)

# density histogram
hist(x, breaks = 100, freq = FALSE, xlim = c(-4, 4), main="")

# Overlay fitted hybrid Pareto density
lines(x.to.plot, fx.hpd, col = "blue", lwd = 2)
abline(v = fit.hpd$u, col = "blue", lwd = 2) # implied threshold

fx.condmixt = dhpareto(x.to.plot, fit.condmixt[1], fit.condmixt[2], fit.condmixt[3],
                       trunc = FALSE)
lines(x.to.plot, fx.condmixt, col = "red", lwd = 2, lty = 2)

legend("topright", c("condmixt", "evmix"), lty = c(2, 1), lwd = rep(2, 2),
       col = c("red", "blue"))
```

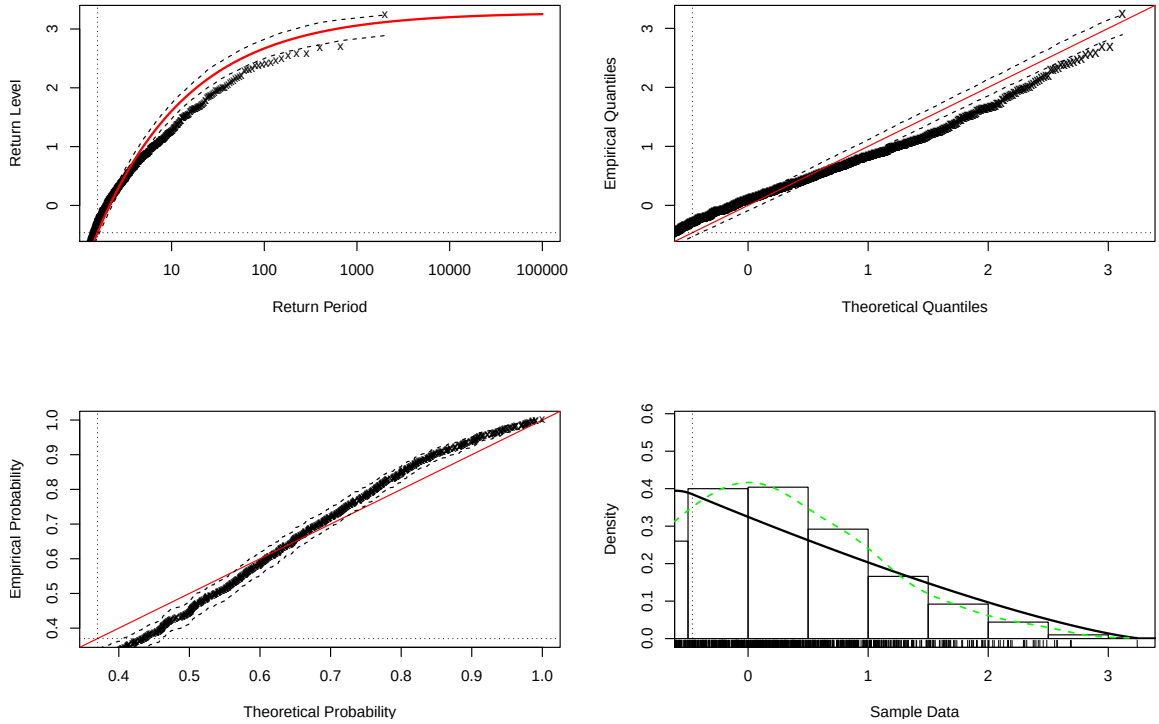


FIGURE 12. Standard model fit diagnostics for the hybrid Pareto fitted to the simulated data in Figure 11.

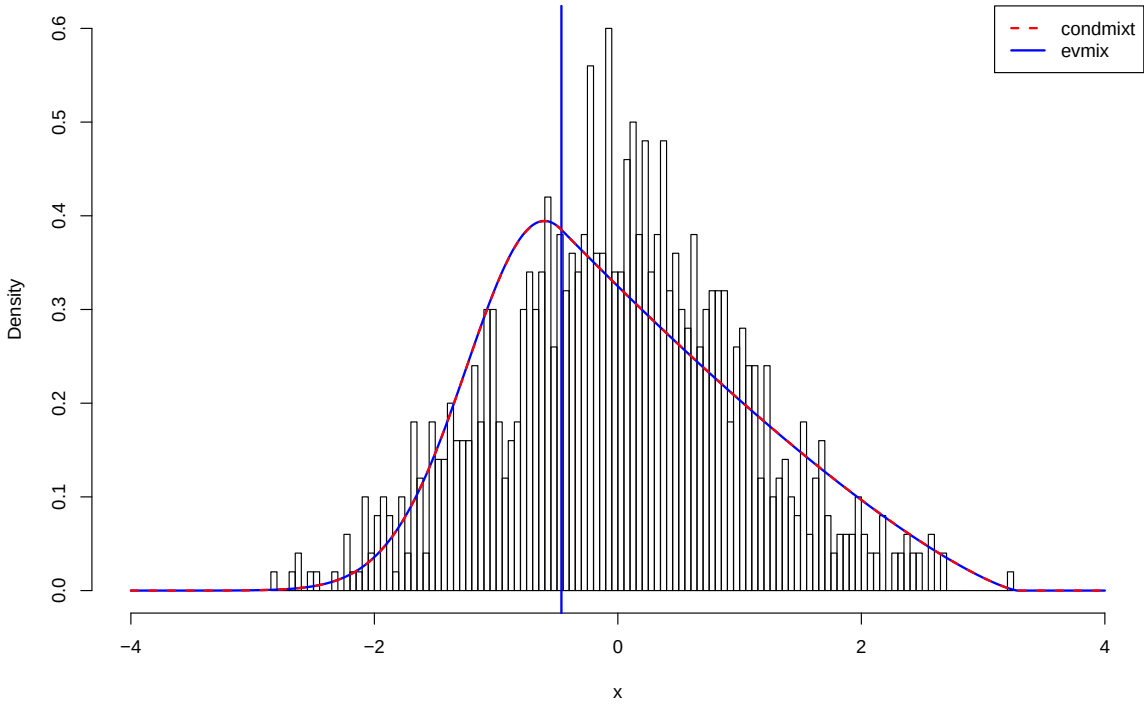


FIGURE 13. Same as Figure 11, comparing the hybrid Pareto fit from the `condmixt` (red dashed line) and `evmix` package (blue solid line).

Appendix - Reproduce Figures in Main Chapter

The figures in the main Chapter are reproducible with the following code.

Figure 1:

```
# Figure 1

library(evmix)

par(mfrow = c(1, 2))
x = seq(-5, 5, 0.001)
y = dnormgpd(x, u = 1.5, sigmau = 0.4)
plot(x, y, xlab = "x", ylab = "Density f(x)", type = "l",
     cex.lab = 2, cex.axis = 1.5)
abline(v = 1.5, lty = 2)
text(-4,0.25,"bulk model",cex=2);arrows(-4,0.22,-1,0.1)
text(3,0.25,"tail model",cex=2);arrows(2.5,0.22,2,0.02)
mtext("u",1,0.5,at=1.5,cex=2)

y = pnormgpd(x, u = 1.5, sigmau = 0.4)
plot(x, y, xlab = "x", ylab = "Distribution F(x)", type = "l",
     cex.lab = 2, cex.axis = 1.5)
abline(v = 1.5, lty = 2)
mtext("u",1,0.5,at=1.5,cex=2)
```

Figure 2:

```
# Figure 2

set.seed(123)
x = rnorm(1000)
xx = seq(-4, 4, 0.01)
y = dnorm(xx)

# Hybrid Pareto provides reasonable fit for some asymmetric heavy upper tailed distributions,
# but not for cases such as the normal distribution
fithpd = fhpd(x, std.err = FALSE)

# HPD with only one constraint
fithpdcon = fhpdcon(x, useq=seq(-1,3,0.1), fixedu=TRUE, std.err = FALSE)

# Notice that if tail fraction is included a better fit is obtained
fit = fnormgpdcon(x, useq=seq(-1,3,0.1), fixedu=TRUE, std.err = FALSE)

par(mfrow = c(1, 1))
hist(x, breaks = 100, freq = FALSE, xlim = c(-4, 4), main="")
lines(xx, y)

with(fithpd, lines(xx, dhpd(xx, nmean, nsd, xi), lwd = 2, lty = 1))
abline(v = fithpd$u, lwd = 2, lty = 1)

with(fithpdcon, lines(xx, dhpdcon(xx, nmean, nsd, u, xi), lwd = 2, lty = 2))
abline(v = fithpdcon$u, lwd = 2, lty = 2)

with(fit, lines(xx, dnormgpdcon(xx, nmean, nsd, u, xi), lwd = 2, lty = 3))
abline(v = fit$u, lwd = 2, lty = 3)

legend("topright", c("Standard Normal", "Hybrid Pareto (HPD)",
                     "HPD - Continuous Only ", "Normal+GPD Continuous"),
       lty = c(1, 1:3), lwd = c(1, rep(2, 3)))
```

Figure 3:

```
# Figure 3

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), lty = 2, lwd = 2)
lines(xx, dweibull(xx, shape = 2, scale = 1/gamma(1.5)), lty = 3, lwd = 2)
lines(xx, pcauchy(xx, location = 1, scale = 1), lty = 1)
legend('topright', c('DWM', 'Weibull', 'GPD', "Cauchy TF"),
       lty = c(1, 3, 2, 1), lwd = c(rep(2,3), 1), bg = "white")
```

References

Carreau, J. (2012), *condmixt: Conditional Density Estimation with Neural Network Conditional Mixtures*, R package version 1.0.

Carreau, J. and Bengio, Y. (2009), "A hybrid Pareto model for asymmetric fat-tailed data: The univariate case," *Extremes*, 12, 53–76.

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

Hu, Y. (2013a), "Extreme value mixture modeling with simulation study and applications in finance and insurance," MSc thesis, University of Canterbury, New Zealand, `http://ir.canterbury.ac.nz/simple-search?query=extreme&submit=Go`.

— (2013b), "User's guide for `evmix` package in `R`," Available from `http://www.math.canterbury.ac.nz/~c.scarrott/evmix`.

Hu, Y. and Scarrott, C. J. (2013), "evmix: An R package for extreme value mixture modelling, threshold estimation and boundary corrected kernel density estimation," *Submitted*, available from `http://www.math.canterbury.ac.nz/~c.scarrott/evmix`.

Jones, M. C. (1993), "Simple Boundary Correction for Kernel Density Estimation," *Statistics and Computing*, 3, 135–146.

Lee, D., Li, W. K., and Wong, T. S. T. (2012), "Modeling insurance claims via a mixture exponential model combined with peaks-over-threshold approach," *Insurance: Mathematics and Economic*, 51, 538–550.

MacDonald, A. (2012), "Extreme Value Mixture Modelling with Medical and Industrial Applications," PhD thesis, University of Canterbury, New Zealand.

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

MacDonald, A., Scarrott, C. J., and Lee, D. S. (2013), "Boundary correction, consistency and robustness of kernel densities using extreme value theory," *Submitted*, available from: http://www.math.canterbury.ac.nz/∼c.scarrott.

Nadarajah, S. (2013), *CompLognormal: Functions for actuarial scientists*, R package version 3.0.

R Core Team (2013a), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

— (2013b), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

Scarrott, C. J. and Hu, Y. (2015), "evmix 0.2.5: Extreme Value Mixture Modelling, Threshold Estimation and Boundary Corrected Kernel Density Estimation," Available on CRAN.

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

Stephenson, A. G. (2002), "evd: Extreme Value Distributions," *R News*, 2, 31–32.

— (2014), *ismev: An Introduction to Statistical Modeling of Extreme Values*, r package version 1.40.

Teodorescu, S. and Vernicu, R. (2009), "Some composite exponential-Pareto models for actuarial prediction," *Romanian Journal of Economic Forecasting*, 4, 82–100.

School of Mathematics and Statistics, University of Canterbury, New Zealand

E-mail address: carl.scarrott@@canterbury.ac.nz