3 The basic model for block maxima: The generalised extreme value distribution

[Some preamble here.]

3.1 History and theoretical motivation

Suppose that $X_1, X_2, \ldots, X_n$ is a sequence of independent and identically distributed (IID) random variables with common distribution function F. One way of characterising extremes is by considering the distribution of the maximum order statistic

$$M_n = \max\{X_1, X_2, \ldots, X_n\}. \tag{1}$$

This is trivial (in principle), since

$$\begin{aligned}
\Pr\{M_n \leq x\} &= \Pr\{X_1 \leq x, X_2 \leq x, \ldots, X_n \leq x\} \\
&= \Pr\{X_1 \leq x\} \times \Pr\{X_2 \leq x\} \times \cdots \times \Pr\{X_n \leq x\} \\
&= \{F(x)\}^n.
\end{aligned}$$

However, in practice the distribution function F is unknown. This leads to an approach based on asymptotic arguments – specifically, limiting distributions for M_n as $n \to \infty$.

3.1.1 The three extremal types and the Extremal Types Theorem

The obvious question now, is, what possible distributions might be considered candidates for the distribution for M_n as $n \to \infty$? It might also be convenient to find out if we can formulate this set of candidate distributions into a single class – say G – which is independent of F; if so, we can estimate the distribution of M_n using G, without any reference to F. Clearly, the limiting distribution of M_n is degenerate, since M_n converges with probability 1 to the upper endpoint of F. This is analogous to the sample mean $\bar{X}$ converging to the population mean μ with certainty in the Central Limit Theorem; here, the degenerate limit is prevented by allowing a linear rescaling, so that

$$\frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \rightarrow N(0, 1).$$

Similarly, we look for limiting distributions as $n \to \infty$ of normalised sequences $(M_n - b_n)/a_n$. The univariate classification (due to Fisher and Tippett in 1928 and formally proved by Gnedenko in 1943) is summarised by the 'extremal types Theorem'; before stating the result, it is important to define an equivalence class of distributions: F and F^* are *type equivalent* if there exists constants a and b such that $F^*(ax + b) = F(x)$, for all x.

Theorem 3.1 *(Extremal types theorem)*
If there exist sequences of constants $a_n > 0$ and b_n such that, as $n \to \infty$,

$$\Pr\{(M_n - b_n)/a_n \leq x\} \to G(x) \tag{2}$$

for some non–degenerate distribution G, then G is of the same type as one of the following distributions:

$$
\begin{aligned}
I : G(x) &= \exp\{-\exp(-x)\} \qquad -\infty < x < \infty; && (3)\\
II : G(x) &= \begin{cases} 0 & x \leq 0 \\ \exp(-x^{-\alpha}) & x > 0, \alpha > 0; \end{cases} && (4)\\
III : G(x) &= \begin{cases} \exp\{-(-x)^{\alpha}\} & x < 0, \alpha > 0 \\ 1 & x \geq 0. \end{cases} && (5)
\end{aligned}
$$

Conversely, each of these distributions, G, may appear as a limit for the distribution of $(M_n - b_n)/a_n$ and, in fact, does so when G itself is the distribution of X in equation (1).

The three types of distribution in Theorem 3.1 (*I*, *II* and *III*) have become known as the Gumbel, Fréchet and Weibull types (respectively), and are known collectively as the *extreme value distributions*. It should be noted that Theorem 3.1 does not *ensure* the existence of a non–degenerate limit for M_n; nor does it specify *which* of types *I*, *II* or *III* is applicable *if* a limit distribution *does* exist (i.e. in which *domain of attraction* the distribution of G lies). However, when such a distribution does exist, we find that, by analogy with the Central Limit Theorem, the limiting distribution of sample maxima follows one of the distributions given in Theorem 3.1 no matter what the parent distribution F. See Leadbetter *et al.* (1983) for a detailed exposition of the existence of limits and domains of attraction.

3.1.2 The Generalized Extreme Value Distribution

In practice, it is inconvenient to work with three separate classes of limiting distribution (as in Theorem 3.1). However, there exists a parameterisation which encompasses all three types of extreme value distribution. Von Mises (1954) and Jenkinson (1955) independently derived the *generalised extreme value distribution* (GEV), denoted $\mathcal{G}(\mu, \sigma, \xi)$, whose cumulative distribution function is given by

$$G(x; \mu, \sigma, \xi) = \exp\left\{ -\left[1 + \xi\left(\frac{x-\mu}{\sigma}\right)\right]_+^{-1/\xi} \right\}, \tag{6}$$

where $a_+ = \max(0, a)$. The situation where $\xi = 0$ is not defined in (6), but is taken as the limit as $\xi \rightarrow 0$, given by

$$G(x; \mu, \sigma) = \exp\left\{ -\exp\left(\frac{x-\mu}{\sigma}\right) \right\}. \tag{7}$$

The parameters μ $(-\infty < \mu < \infty)$, σ (> 0) and ξ $(-\infty < \xi < \infty)$ are location, scale and shape parameters respectively. The value of the shape parameter ξ differentiates between the three types of extreme value distribution in Theorem 3.1: $\xi = 0$, leading to (7), corresponds to the Gumbel distribution (type *I*); $\xi > 0$ and $\xi < 0$ correspond to the Fréchet (type *II*) and Weibull (type *III*) distributions respectively.

3.1.3 When should we model block maxima?

The GEV distribution often proves useful for modelling environmental extremes. A typical use of the GEV might be to aid the design of a sea wall or a tall building. We might want the sea wall to withstand the sea surge we can expect to occur, on average, once every fifty years, or we might want the building to be built sufficiently strong so that it can withstand wind gusts we might only expect to occur once every hundred years. In such situations, it is the extremes, rather than the average level of the process, that is of interest. Theorem 3.1 and the results in Section 3.1.2 provide a way of making inferences on the extremes of a process once a suitable set of maxima M_n have been extracted from our data. For example, we might have daily total rainfall accumulations recorded at a certain location over a fifty year period. To implement the fitting of the GEV, we would need to partition our series into blocks and then, from each block, select the maximum rainfall total. For convenience, and possibly to avoid non–stationarity issues, we might choose the calendar year as our block unit; thus, the extreme dataset to be analysed would consist of 50 annual maximum rainfall accumulations. Such an approach is often referred to as a 'block maxima' approach. Although this sounds fairly straightforward, choosing the block size can be difficult: if the block size is too large, there is likely to be large estimation variance owing to the relatively small number of block maxima generated, whereas a block size that is too small will give a poor approximation to the asymptotic model in (6), possibly leading to estimation bias.

It is often the case that, where interest was always going to be in the extremes of a particular process, that only the annual maxima – or indeed the maxima taken over some other block – have been recorded; in this case, block size may not be an issue and we can proceed to model our data with the GEV immediately. Conversely, there may be much data – perhaps we have hourly wind speed measurements, or indeed measurements that have been recorded continuously. In such cases, to avoid wasting precious data, we might choose to take maxima over shorter block lengths, perhaps obtaining a set of daily or monthly maxima. Doing so could, as previously mentioned, lead to issues of non–stationarity, and such issues will be covered in Chapter 5. In fact, we may even consider an altogether different approach for identifying and analysing extremes; one such approach using *threshold methods* will be considered in Chapter 4.

3.1.4 How is the GEV distribution used?

A typical application of the generalised extreme value distribution is to fit (6) to a series of block (often annual) maxima. Numerous ways of fitting this to a set of annual maxima have been considered, though numerical maximum likelihood estimation has become generally accepted as the most robust procedure (Hosking (1990) argues that using L–moments can provide a more robust estimative procedure in small samples; applications of L–moments can be found in Guttman *et al.* (1993) and Hosking and Wallis (1993)). A potential difficulty with the use of likelihood methods for the GEV concerns the regularity conditions that are required for the usual asymptotic properties associated with the maximum likelihood estimator to be valid. Such conditions are not satisfied by the GEV because the end–points of the GEV distribution are functions of the parameter values: $\mu = \sigma/\xi$ is an upper end–point of the distribution when $\xi < 0$, and a lower end–point when $\xi > 0$. This violation of the usual regularity conditions means that the standard asymptotic likelihood results are not automatically applicable. Smith (1985) found that, for $-1 < \xi \leq -0.5$, maximum likelihood estimators are obtainable but, in general, do not have all of the standard asymptotic properties: estimators are 'super–efficient' (i.e. have variances smaller than the Cramér–Rao lower bound) and are not asymptotically normal.

Further, he found that maximum likelihood estimators for $\xi \leq -1$ do not, in general, exist. However, when $\xi > -0.5$, though the classical regularity conditions are still not satisfied, the information matrix is finite and the classical asymptotic properties continue to hold.

The (relatively) recent explosion in simulation–based inference and the advent of Markov chain Monte Carlo algorithms have offered an alternative way to make inferences from the likelihood function, which many practitioners prefer. Although (in most environmental applications) instances of $\xi < -0.5$ are rare, they are not unheard of, and so here the Bayesian framework provides a viable inferential alternative to maximum likelihood estimation (see, for example, Coles and Powell, 1996). We consider Bayesian inference for extremes, and more generally, in Chapter 8; until then, we adopt standard likelihood techniques to demonstrate applications of extreme value theory.

Once the model in (6) has been fitted to our series of, say, annual maxima, and estimates of the location, scale and shape obtained ($\hat{\mu}$, $\hat{\sigma}$ and $\hat{\xi}$, respectively), estimates of extreme quantiles of these annual maxima can then be obtained by inversion of (6). For example, assuming we have annual maxima (block lengths of one year), the *100–year return level*, q_{100}, is the level that is exceeded, on average, once every 100 years, and is estimated by setting $1 - G(\hat{q}_{100}; \hat{\mu}, \hat{\sigma}, \hat{\xi}) = 1/100$ and solving for $\hat{q}_{100}$. It is trivial to show that the r–year return level, q_r, can be found as

$$q_r \;=\; \mu - \frac{\sigma}{\xi}\left[1 - \{-\log(1 - 1/r)\}^{-\xi}\right] \tag{8}$$

for $\xi \neq 0$, where $G(q_r; \mu, \sigma, \xi) = 1 - 1/r$. In the case of $\xi = 0$, inverting (7) gives

$$q_r \;=\; \mu - \sigma \log\left[-\log(1 - 1/r)\right]. \tag{9}$$

It is usually the return level estimate that is of most interest to practitioners – for example, civil engineers or planners may need to know how high to build a sea wall or how strong to make a building so that it withstands the once–in–a–hundred year sea surge or the once–in–fifty year wind gust. The delta method (reference?) provides a way of measuring the uncertainty about the estimate $\hat{q}_r$ (see Section 3.2.3) though, due to the (often severe) asymmetry of the likelihood surface for q_r, confidence intervals are often obtained via the method of profile likelihood (see Section 3.2.4).

3.2 Simple case study

In this Section we demonstrate a simple application of the GEV to a series of annual maxima, where we:

- define the log–likelihood function for the GEV in **R**;
- maximise this log–likelihood function using the `nlm` routine in **R**, to obtain fitted values for the GEV parameters;
- use **R** to obtain the expected information matrix, and then invert this to obtain the estimated variance–covariance matrix for $(\hat{\mu}, \hat{\sigma}, \hat{\xi})^T$;

- use the fitted values of the GEV parameters to estimate some return levels q_r, using the delta method to estimate the standard errors for these;
- use **R** to plot the profile log–likelihood for the GEV parameters and some return levels, and demonstrate the method of profile likelihood for obtaining more realistic confidence intervals (particularly for q_r);
- use **R** to check the goodness–of–fit of the GEV to our series of annual maxima.

The series we consider are the annual maxima of daily rainfall accumulations (in mm) at a location in southwest England, from 1914 to 1961 (inclusive). This set of annual maxima are given in Table 1, and can be manually typed into **R** using the following code; we will consider how to extract these maxima from the full set of daily observations in Section 3.3.1.

```
> anmax.rainfall<-c(44.5,43.2,38.1,39.1,32.3,25.4,33.0,32.5,48.5,34.3,...
```

A plot of these maxima is shown in Figure 1 below; such a plot can be produced in **R** using the command

```
> plot(anmax.rainfall~year)
```

where the vector `year` is a vector of years constructed using the command

```
> year<-seq(1914,1961,1)
```

The title of the graph was included using the argument `main=`, and the labels on the axes using the arguments `xlab=` and `ylab=` respectively. The argument `type=b` was also used to plot both points *and* lines. Each block has 365 observations (or 366 in leap years), which is reasonably large, and so we will fit model (6) to the 48 annual maxima.

44.5	43.2	38.1	39.1	32.3	25.4	33.0	32.5
48.5	34.3	45.7	35.3	76.7	34.0	86.6	48.5
59.4	53.3	30.5	42.7	83.3	59.2	37.3	67.3
38.4	47.0	38.1	72.4	40.9	47.0	36.3	85.3
35.6	55.9	44.2	45.2	51.6	59.4	47.8	55.4
42.4	40.1	36.6	47.0	48.8	51.3	39.4	45.7

Table 1: Annual maximum rainfall accumulations from a location in southwest England

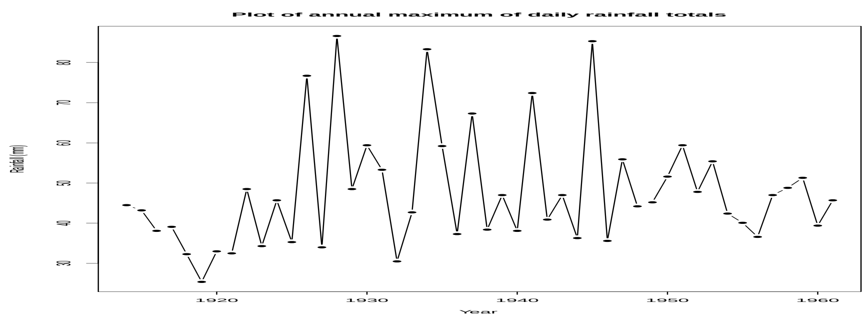

Figure 1: Annual maxima of daily rainfall totals at a location in south west England

3.2.1 Fitting the GEV

Assuming our 48 annual maxima $\boldsymbol{M} = (M_1, \ldots, M_{48})$ are independent, the likelihood for the GEV parameters is given by:

$$L(\mu, \sigma, \xi; \boldsymbol{M}) = \prod_{i=1}^{48} g(M_i; \mu, \sigma, \xi), \tag{10}$$

where g represents the density function for the GEV and is found, by differentiation of (6), to be

$$g(x; \mu, \sigma, \xi) = \frac{1}{\sigma}\left[1 + \xi\left(\frac{x-\mu}{\sigma}\right)\right]_+^{-(1/\xi+1)} \exp\left\{-\left[1 + \xi\left(\frac{x-\mu}{\sigma}\right)\right]_+^{-1/\xi}\right\}. \tag{11}$$

Substituting this into (10) for each annual daily rainfall maximum M_i, $i = 1, \ldots, 48$, gives:

$$L(\mu, \sigma, \xi; \boldsymbol{M}) = \prod_{i=1}^{48} \frac{1}{\sigma}\left[1 + \xi\left(\frac{M_i-\mu}{\sigma}\right)\right]_+^{-(1/\xi+1)} \exp\left\{-\left[1 + \xi\left(\frac{M_i-\mu}{\sigma}\right)\right]_+^{-1/\xi}\right\}.$$

Taking logs gives the log–likelihood $\ell(\mu, \sigma, \xi; \boldsymbol{M})$:

$$\begin{aligned}\ell(\mu, \sigma, \xi; \boldsymbol{M}) = & -48\log_e\sigma - \sum_{i=1}^{48}\left[1 + \xi\left(\frac{M_i-\mu}{\sigma}\right)\right]_+^{-1/\xi} \\ & -(1+1/\xi)\sum_{i=1}^{48}\log_e\left[1 + \xi\left(\frac{M_i-\mu}{\sigma}\right)\right]_+ .\end{aligned} \tag{12}$$

We leave the derivation of this as an exercise for the reader.

We define the GEV log–likelihood function in **R** in the following way:

```
> dataset<-anmax.rainfall #see note 1

> theta<-c(mean(dataset),sd(dataset),0.1) #see note 2

> gev.loglik<-function(theta)
   {
     mu<-theta[1]
     sigma<-theta[2]
     xi<-theta[3]
     m<-min((1+(xi*(dataset-mu)/sigma)))   #see note 3
     if(m<0.00001)return(as.double(1000000))    #see note 4
     if(sigma<0.00001)return(as.double(1000000))    #see note 5
     loglik<--length(dataset)*log(sigma)-sum((1+(xi*(dataset-mu)/sigma))^(-1/xi))
     -(1+1/xi)*sum(log(1+(xi*(dataset-mu)/sigma)))
     return(-loglik)    #see note 6
   }
```

Notes on the above code

1. This line stores our rainfall data in the vector `dataset` for use in the function `gev.loglik`.

2. We need to initialise the parameter vector $\theta = (\mu, \sigma, \xi)$ at some sensible values before we maximise the log–likelihood; here, we choose the mean of our data for μ, the standard deviation for σ, and 0.1 for ξ.

3. This line stores the component
$$\left[1 + \xi\left(\frac{M_i - \mu}{\sigma}\right)\right]_+$$
in the scalar `m`.

4. Recall that, if the component `m` is less than zero, we take it to be equal to zero, as $a_+ = \max(0, a)$. Thus, if for a particular combination of $\theta = (\mu, \sigma, \xi)$ we get `m<0.00001`, we return a huge value for the negative log–likelihood (1e+06) ensuring that this combination will almost surely never be the combination that minimises the negative log–likelihood (and so will almost surely never maximise the log–likelihood – see note 6).

5. This line performs a similar task for `sigma` as the code referred to in note 4 does for `m`. σ is the scale parameter and so can never be negative; thus, any attempt to minimise the negative log–likelihood using `sigma<0.00001` returns a huge negative log–likelihood (1e+06), again ensuring that this choice of σ can almost surely never minimise the negative log–likelihood (again, see note 6).

6. The **R** routine `nlm` is a minimisation routine that minimises a supplied function (here `gev.loglik`) using a Newton–type algorithm. Since the aim is to find estimates of the GEV parameters that *maximise* the log–likelihood, this will be equivalent to finding estimates of the GEV parameters that *minimise* the *negative* log–likelihood, which is why we ask the function `gev.loglik` to return `-loglik`.

We implement the `nlm` routine in the following way, giving the output shown underneath:

```
> nlm(gev.loglik,theta)

$minimum
[1] 188.0154

$estimate
[1] 40.7829629  9.7283652  0.1072363

$gradient
[1]  1.216790e-06 -5.638553e-07  8.782308e-06

$code
[1] 1

$iterations
[1] 19
```

Thus, our estimates of the GEV parameters are

$$\hat{\mu} = 40.78 \qquad \hat{\sigma} = 9.73 \qquad \hat{\xi} = 0.11$$

The *minimised negative* log–likelihood is 188.0154, giving a maximised log–likelihood at the above values for the GEV parameters of −188.0154. `gradient` gives the gradient at the estimated minimum of the function `gev.loglik`, `iterations` gives the number of iterations performed before the solution to the minimisation problem was found, and `code` is an integer which indicates why the optimisation process terminated (`1` indicates the relative gradient is close to zero and so the current iterate is probably the solution).

3.2.2 Standard errors

Using `nlm` to minimise the negative log–likelihood function is fine, but our estimates of the GEV parameters do not have any associated estimate of variability attached. However, using the argument `hessian` in the `nlm` routine returns the matrix of second–order partial derivatives (the *Hessian matrix*) of the function we are trying to minimise – in a statistical inference context, this would equate to the elements of the expected information matrix. In **R**, we type:

```
> a<-nlm(gev.loglik,theta,hessian=TRUE)
```

This will give the same output as before, but now the Hessian matrix will also be returned. Note we have also stored the output in `a`. To look at the output, we type:

```
> a
```

which gives the following:

```
$minimum
[1] 188.0154

$estimate
[1] 40.7829629  9.7283652  0.1072363

$gradient
[1]  1.216790e-06 -5.638553e-07  8.782308e-06

$hessian
           [,1]       [,2]      [,3]
[1,]  0.5686920 -0.3346753  2.106623
[2,] -0.3346753  0.9167153 -0.205903
[3,]  2.1066232 -0.2059030 94.085661

$code
[1] 1

$iterations
[1] 19
```

We now want the inverse of the expected information matrix (`hessian` above) to obtain the variance–covariance matrix for our GEV parameters. One way of doing this is to use the **R** command `solve`. This command solves, for X, a system of equations of the form

$$\mathbf{a}X = \mathbf{b},$$

where both $\mathbf{a}$ and $\mathbf{b}$ can be numeric or complex matrices, $\mathbf{a}$ corresponding to the coefficients of the linear system and $\mathbf{b}$ giving the right–hand–side of this system. In our example, $\mathbf{a}$ is the expected information matrix (`hessian`). If $\mathbf{b}$ is missing in the execution of `solve`, then the identity matrix is assumed and **R** will thus return the inverse of $\mathbf{a}$ – exactly what we want here! In **R**, this gives:

```
> solve(a$hessian)
            [,1]        [,2]        [,3]
[1,]  2.48370206  0.89470121 -0.05365326
[2,]  0.89470121  1.41368485 -0.01693899
[3,] -0.05365326 -0.01693899  0.01179286
```

Thus, the square roots of the diagonal elements of the above matrix will give the expected standard errors for the GEV parameters (μ, σ and ξ respectively). We could obtain these standard errors in the following way:

```
> varcovar<-solve(a$hessian)
> sqrt(diag(varcovar))
[1] 1.5759765 1.1889848 0.1085950
```

The first line of code stores the matrix shown above in the matrix `varcovar`; the second line of code finds the square roots of the diagonal elements of `varcovar`. The resulting expected standard errors are shown underneath. Thus, we now have the following inference for our annual maxima of daily rainfall totals, in terms of the GEV distribution:

$$\hat{\mu} = 40.78\ (1.58) \qquad \hat{\sigma} = 9.73\ (1.19) \qquad \hat{\xi} = 0.11\ (0.11)$$

From this, we can construct confidence intervals in the usual way ("Wald" confidence intervals). For example, standard symmetric 95% confidence intervals are found as: parameter estimate $\pm$ 1.96 $\times$ standard error, giving

$$(37.68, 43.88) \qquad (7.40, 12.06) \qquad \text{and} \qquad (-0.11, 0.33)$$

for μ, σ and ξ (respectively). Note that the confidence interval for ξ includes zero, suggesting the possibility of a Gumbel–type tail for our data.

3.2.3 Return level inference

As discussed in Section 3.1.4, of most interest to practitioners are estimates of *return levels*, i.e. levels which are exceeded, on average, once every r years. Since our data occur yearly anyway (we have annual maxima of daily rainfall totals, and so we have one observation per year), a point estimate for q_r can be obtained using equation (8), substituting μ, σ and ξ with our maximum likelihood estimators found by minimisation of the negative log–likelihood (see previous Section). In fact, doing so will give the maximum likelihood estimates of q_r due to the invariance property of MLEs. Table 2 overleaf gives the MLEs for the 10, 50, 100 and 1000 year return levels obtained via equation (8).

	$\hat{q}_{10}$	$\hat{q}_{50}$	$\hat{q}_{100}$	$\hat{q}_{1000}$
Point estimate	65.54	87.92	98.64	140.34

Table 2: Some estimated return levels for the rainfall data (units are millimetres)

The **R** function below was written to find these estimates, essentially just using **R** like a calculator:

```
gev.ret.level<-function(mu,sigma,xi,period)
{
ret.level<-mu-(sigma/xi)*(1-(-log(1-(1/period)))^(-xi))
ret.level
}
```

Then, for example, the 100–year return level is found by typing:

```
> gev.ret.level(a$est[1],a$est[2],a$est[3],100)
[1] 98.63595
```

As with our inference for the GEV parameters, it is preferable to quote estimates of return levels with their estimated standard errors. Since q_r is a function of the GEV parameters μ, σ and ξ, we can use the *delta method* to find the approximate variance of $\hat{q}_r$. Specifically,

$$\mathrm{Var}(\hat{q}_r) \approx \nabla q_r^T V \nabla q_r,$$

where V is the variance–covariance matrix of $(\hat{\mu}, \hat{\sigma}, \hat{\xi})^T$ and

$$\begin{aligned} \nabla q_r^T &= \left[\frac{\partial q_r}{\partial \mu}, \frac{\partial q_r}{\partial \sigma}, \frac{\partial q_r}{\partial \xi}\right] \\ &= \left[1, -\xi^{-1}(1 - y_r^{-\xi}), \sigma\xi^{-2}(1 - y_r^{-\xi}) - \sigma\xi^{-1}y_r^{-\xi}\log y_r\right], \end{aligned}$$

where $y_r = -\log(1 - 1/r)$, evaluated at $(\hat{\mu}, \hat{\sigma}, \hat{\xi})$.

Recall that V is stored in the vector `varcovar` in **R**. We can define ∇z_{10}, for example, in **R** as

```
> y10<--log(1-(1/10))
> del<-matrix(ncol=1,nrow=3)
> del[1,1]<-1
> del[2,1]<--((a$est[3])^(-1))*(1-(y10^(-a$est[3])))
> del[3,1]<-((a$est[2])*((a$est[3])^(-2))*(1-((y10)^(-a$est[3]))))
-((a$est[2])*((a$est[3])^(-1))*((y10)^(-(a$est[3])))*log(y10))
> del.transpose<-t(del)
```

Then the **R** command for matrix multiplication – `%*%` – can be used to obtain an estimate of the standard error for $\hat{q}_{10}$ in the following way:

```
> sqrt(del.transpose%*%varcovar%*%del)
         [,1]
[1,] 4.526629
```

	$\hat{q}_{10}$	$\hat{q}_{50}$	$\hat{q}_{100}$	$\hat{q}_{1000}$
MLE (e.s.e.)	65.54 (4.53)	87.92 (11.48)	98.64 (16.22)	140.34 (41.83)

Table 3: Some estimated return levels for the rainfall data, with associated estimated standard errors (units are millimetres)

Estimated standard errors for other return levels can be obtained in a similar way. For example, for the standard error for the 50–year return level, we would replace

```
> y10<--log(1-(1/10))
```

with

```
> y50<--log(1-(1/50))
```

and then `y10` would be replaced with `y50` throughout. Table 3 shows the estimated return levels from Table 2, but now with standard errors in parentheses. Caution is required in the interpretation of return level inferences. For instance, estimates of these return levels assume the model is correct. Now for the 10 year return level this can be checked; however, we only have 48 years worth of data, so estimates of the other three return levels shown in Table 3 extrapolate beyond the range of the observed data, and this extrapolation is based on unverifiable assumptions. This is reflected in the increasing standard errors as we move further into the tails of the GEV, with a very large standard error (41.83) for the 1000–year return level estimate. The normal distribution also provides a poor approximation to the distribution of the MLE for q_r, meaning the usual Wald confidence intervals can be misleading. For example, for the 100–year return level we get (66.85, 130.43), which will not properly capture the (often severe) asymmetry of the likelihood surface often encountered for return levels. Indeed, better approximations are usually obtained from the appropriate *profile* likelihood function (see Section 3.2.4).

Obviously, it would be preferable to combine the code in the current Section with that in Sections 3.2.1 and 3.2.2 to create a single function that estimates the GEV parameters and a desired return level, with associated standard standard errors, in a single sweep. An example of this is shown in the function `gevfit` below:

```
gevfit<-function(data,r)
  {
    theta<-c(mean(data),sd(data),0.1)
    gev.loglik<-function(theta)
      {
        mu<-theta[1]
        sigma<-theta[2]
        xi<-theta[3]
        m<-min((1+(xi*(data-mu)/sigma)))
        if(m<0.00001)return(as.double(1000000))
        if(sigma<0.00001)return(as.double(1000000))
        loglik<--length(data)*log(sigma)-sum((1+(xi*(data-mu)/sigma))^
        (-1/xi))-(1+1/xi)*sum(log(1+(xi*(data-mu)/sigma)))
        return(-loglik)
      }
```

```
    a<-nlm(gev.loglik,theta,hessian=TRUE)
    gev<-a$est
    varcovar<-solve(a$hessian)
    ese<-sqrt(diag(varcovar))
    ret.level<-gev[1]-(gev[2]/gev[3])*(1-(-log(1-(1/r)))^(-gev[3]))
    yr<--log(1-(1/r))
    del<-matrix(ncol=1,nrow=3)
    del[1,1]<-1
    del[2,1]<--((gev[3])^(-1))*(1-(yr^(-gev[3])))
    del[3,1]<-((gev[2])*((gev[3])^(-2))*(1-((yr)^(-gev[3]))))-((gev[2])*
    ((gev[3])^(-1))*((yr)^(-(gev[3])))*log(yr))
    del.transpose<-t(del)
    ese.ret.level<-sqrt(del.transpose%*%varcovar%*%del)
    ese.ret.level<-ese.ret.level[1,1]
    return(gev,ese,ret.level,ese.ret.level)
  }
```

To execute this for the annual maximum rainfall dataset, and to estimate the 100–year return level with its standard error, we would type:

```
> gevfit(dataset,100)
```

which would give the following output:

```
$gev
[1] 40.7829629  9.7283652  0.1072363

$ese
[1] 1.5759765 1.1889848 0.1085950

$ret.level
[1] 98.63595

$ese.ret.level
[1] 16.21965
```

3.2.4 Profile likelihood

In Section 3.2.3 we obtained estimated standard errors for the GEV parameters via classical likelihood theory; the delta method was then employed to obtain estimated standard errors for return levels. Maximum likelihood estimates, and their associated standard errors, were then used to obtain symmetrical confidence intervals assuming the limiting quadratic behaviour of the likelihood surface near the maximum (parameter estimate $\pm$ 1.96$\times$standard error for a 95% confidence interval, for example). An alternative, and often more accurate, method for making inferences on a particular parameter can be found using the *profile log–likelihood*. Formally, the log–likelihood for θ can be written as $\ell(\theta_i, \theta_{-i})$, where θ_{-i} corresponds to all components of θ excluding θ_i. The profile log–likelihood for θ_i is defined as

$$\ell_p(\theta_i) \quad = \quad \max_{\theta_{-i}} \ell(\theta_i, \theta_{-i}).$$

Thus, for each value of θ_i the profile log–likelihood is the maximised log–likelihood with respect to all other components of θ.

For example, to find the profile log–likelihood for ξ in our rainfall maxima case study, we fix $\xi = \xi_0$ and maximise (12) with respect to the remaining parameters μ and σ, repeating this for a range of values of ξ_0. In **R**, the following code creates a vector `xi.0` that contains values of ξ_0 from -0.3 up to 0.6, in steps of 0.001:

```
> xi.0<-seq(-0.3,0.6,0.001)
```

We then define the profile log–likelihood function for ξ as:

```
> theta2<-c(mean(dataset),sd(dataset))
> prof.loglik<-function(theta2)
  {
    mu<-theta2[1]
    sigma<-theta2[2]
    m<-min((1+(xi*(dataset-mu)/sigma)))
    if(m<0.00001)return(as.double(1000000))
    if(sigma<0.00001)return(as.double(1000000))
    if(xi==0)
      {
        loglik<--length(dataset)*log(sigma)-sum((dataset-mu)/sigma)-
        sum(exp(-((dataset-mu)/sigma)))
      }
  else{
    loglik<--length(dataset)*log(sigma)-sum((1+(xi*(dataset-mu)/sigma))^
    (-1/xi))-(1+1/xi)*sum(log(1+(xi*(dataset-mu)/sigma)))
  }
    loglik
    return(-loglik)
  }
```

Notice that care has been taken to replace the GEV log–likelihood with the Gumbel limit when $\xi = 0$. We then minimise the negative log–likelihood (and so maximise the log–likelihood itself) provided by the function `prof.loglik` for each of $\xi = \xi_0$, and plot the resulting negated value against the corresponding value of ξ_0 to obtain the profile of the log–likelihood surface viewed from the ξ–axis. In **R**, we type

```
> prof.plot<-vector('numeric',length(xi.0))
```

to create a vector in which we will store the maximised log–likelihood value for each value of ξ_0. Then the following code

```
> for(i in 1:length(prof.plot))
    {
     xi<-xi.0[i]
     a<-nlm(prof.loglik,theta2)
     prof.plot[i]<--(a$minimum)
    }
> plot(prof.plot~xi.0,main='Profile log-likelihood for xi',xlab='xi',
  ylab='Profile log-likelihood',type='l')
```

maximises the log–likelihood for each ξ_0 (with respect to the other parameters μ and σ), populates the vector `prof.plot` with the maximised log–likelihood values, and then plots these values against the corresponding value of ξ_0 used. Figure 2 below shows the resulting plot. The code

```
> abline(h=-188.0154)
> abline(v=0.1072363)
```

superimpose the (solid) horizontal line at the maximised log–likelihood (minimised negative log–likelihood), and the (solid) vertical line at the MLE for ξ, respectively.

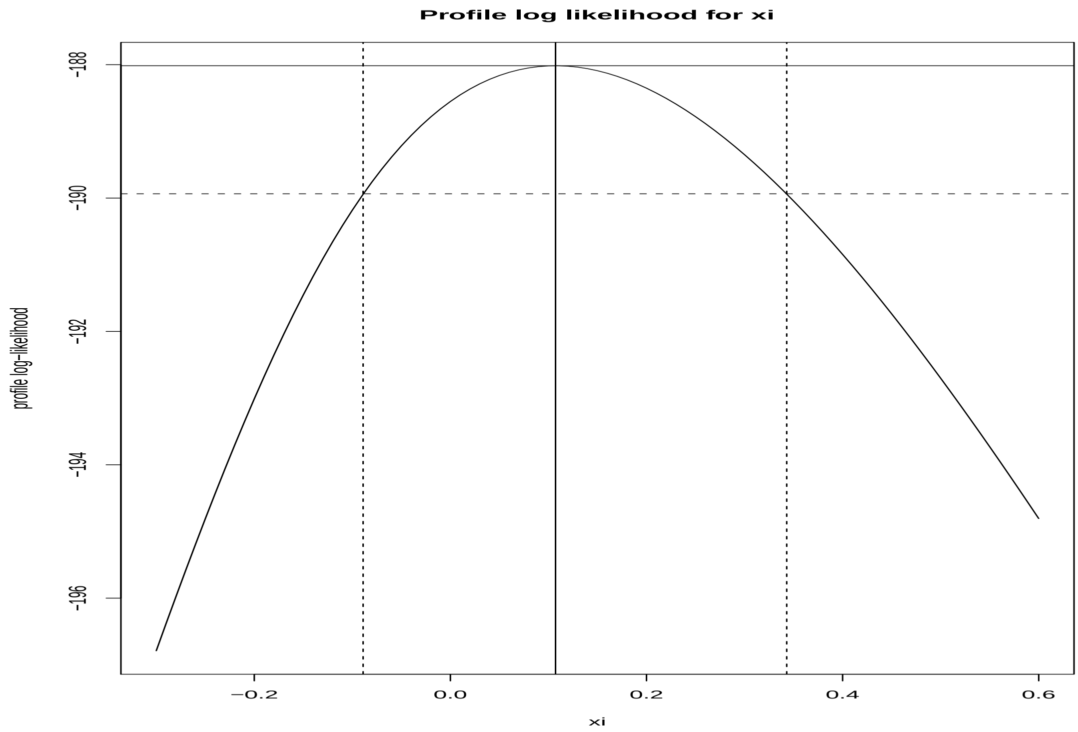

Figure 2: Profile log–likelihood for ξ in the rainfall example

In this example, we are partitioning the GEV parameter vector $\theta = (\mu, \sigma, \xi)$ into two components $(\theta^{(1)}, \theta^{(2)})$, where $\theta^{(1)} = \xi$ and $\theta^{(2)} = (\mu, \sigma)$, and the profile log–likelihood is now defined as

$$\ell_p(\theta^{(1)}) \quad = \quad \max_{\theta^{(2)}}(\theta^{(1)}, \theta^{(2)}).$$

The following result leads to a procedure for making inferences on the maximum likelihood estimator of $\theta^{(1)}$.

Result

Let $x_1, \ldots, x_n$ be independent realisations from a distribution within a parametric family $\mathcal{F}$, and let $\hat{\theta}_0$ be the maximum likelihood estimator of the d–dimensional model parameter $\theta_0 = (\theta^{(1)}, \theta^{(2)})$, where $\theta^{(1)}$ is a k–dimensional subset of θ_0. Then, under suitable regularity conditions, for large n

$$D_p(\theta^{(1)}) \quad = \quad 2\left\{\ell(\hat{\theta}_0) - \ell_p(\theta^{(1)})\right\} \sim \chi^2_k.$$

Thus, for our single component $\theta^{(1)} = \xi$, the set of values C_α for which $\{\xi : D_p(\xi) \leq c_\alpha\}$ provides a $(1-\alpha)$ confidence interval for ξ, where c_α is the $(1-\alpha)$ quantile of the χ^2_1 distribution.

We apply this result to the profile log–likelihood for ξ; a cut–off point equal to $\frac{1}{2} \times \chi^2_1(0.05)$ is shown in Figure 2 by the horizontal broken line, and was superimposed using the code:

```
> abline(h=-188.0154-0.5*qchisq(0.95,1),lty=2)
```

The points of intersection between this line and the profile log–likelihood for ξ define the set of values $C_{0.05}$ giving a more appropriate 95% confidence interval for ξ than obtained using the usual Wald approach. These points of intersection are also shown on Figure 2 (broken vertical lines), and give a confidence interval for ξ based on the profile likelihood approach of $(-0.089, 0.343)$. Recall that the corresponding Wald interval is $(-0.11, 0.33)$; a little left–shifted relative to the more appropriate interval based on the profile likelihood, but only slightly different.

It is for return level inference that the profile likelihood is most useful. Suppose our interest lies in estimation of the 100–year return level. We can simply re–parameterise the GEV model so that q_{100} is one of the model parameters; the profile log–likelihood can then be obtained via maximisation with respect to the remaining parameters in the usual way. For example, transposition of Equation (9) to make μ the subject gives

$$\mu \;=\; q_r + \frac{\sigma}{\xi}\left[1 - \{-\log(1-1/r)\}^{-\xi}\right]. \qquad (13)$$

We can then replace μ with (13) in the expression for the log–likelihood (12) to obtain $\ell(q_r, \sigma, \xi)$ – the log–likelihood for $\theta = (q_r, \sigma, \xi)$. Inferences for q_r based on the profile log–likelihood can then be made in exactly the same way we made inferences for ξ. For example, shown in Figure 3 is the profile of the log–likelihood surface viewed from the q_{100}–axis – i.e. the profile log–likelihood for the 100–year return level. Adopting the same result to form the 95% confidence interval as used for ξ gives (78.85, 159.73) for q_{100} (shown by the broken lines in Figure 3). A symmetric confidence interval for q_{100} was constructed in the usual way in Section 3.2.3 and was found to be (66.85, 130.43) – quite different from the more realistic interval obtained via the profile log–likelihood. Engineers often design to the upper–endpoint of a 95% confidence interval for such return level estimates; doing so based on a symmetric Wald interval could result in substantial under–protection (130mm instead of 160mm).

3.2.5 Model diagnostics

We usually fit a statistical model to data to draw conclusions about some aspect of the population from which the data were obtained. The more accurate the fitted model is, the more reliable these conclusions are likely to be. Since inferences are sensitive to the accuracy of the fitted model, it is important that we check that the model fits well. Ideally, we would like to check that our model describes well the variations in the wider population, but this is usually not possible unless there are other sources of data for us to judge the model,against. Thus, it is common to assess the goodness–of–fit of a particular model using the data that were used to estimate it in the first place.

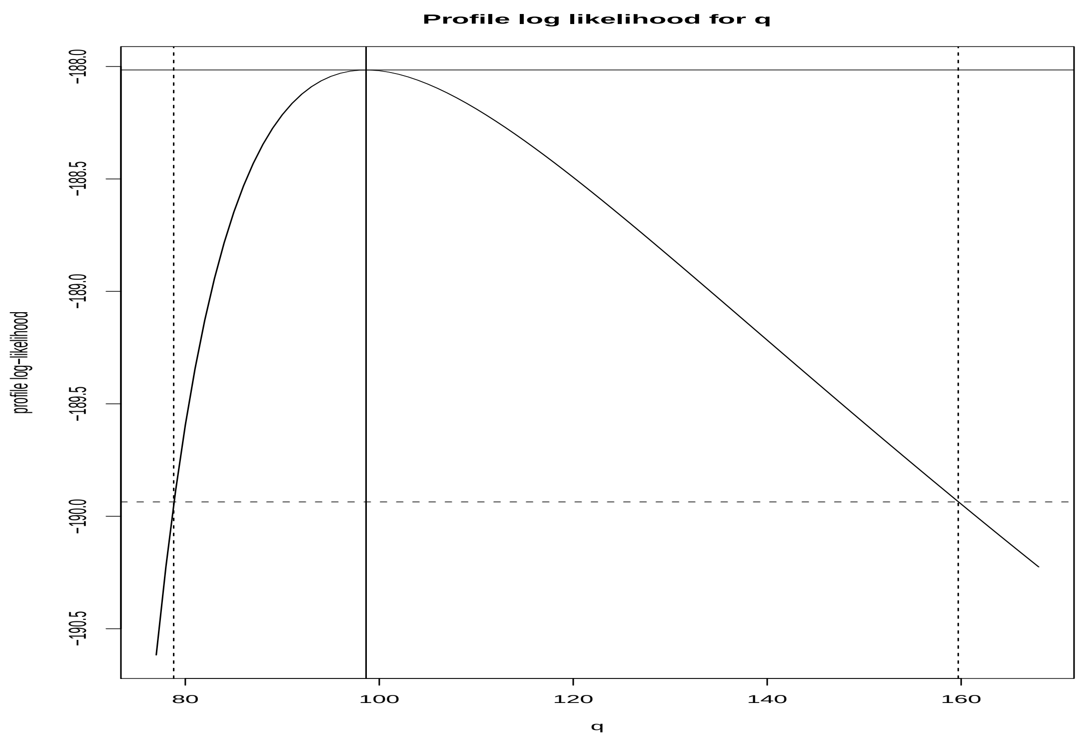

Figure 3: Profile log–likelihood for q_{100} in the rainfall example

A *probability plot* is a plot of the points

$$\left\{\left(\hat{F}\left(x_{(i)}\right), \frac{i}{n+1}\right) : i = 1, \ldots, n\right\},$$

where $x_{(i)}, i = 1, \ldots, n$ is an ordered sample of independent observations and $\hat{F}$ is a candidate model for the true probability distribution F. The quantity $i/n+1$ corresponds to the *empirical distribution function* evaluated at $x_{(i)}$. If $\hat{F}$ is a reasonable model for the true distribution, the points in the probability plot will lie close to the unit diagonal.

A *quantile–quantile plot* (q–q plot, or just quantile plot) is a plot of the points

$$\left\{\left(\hat{F}^{-1}\left(\frac{i}{n+1}\right), x_{(i)}\right) : i = 1, \ldots, n\right\}.$$

The quantity $\hat{F}^{-1}(i/(n+1))$ gives a model–based estimate of the $i/(n+1)$ quantile provided by the candidate distribution $\hat{F}$ whilst $x_{(i)}$ itself provides an empirical estimate of this quantile. Again, a well–fitting model would provide points on this plot lying close to the unit diagonal.

In **R**, the code

```
> ordered<-sort(dataset)
```

stores the ordered annual rainfall maxima $x_{(i)}, i = 1, \ldots, 48$, in the vector `ordered`. For each point in our ordered sample, the following code then stores the empirical distribution function defined by $i/(n+1)$ in the vector `empirical`:

```
> empirical<-vector(''numeric'',length(ordered))
> for(i in 1:length(empirical))
         {
    empirical[i]<-i/(length(dataset)+1)
         }
```

The function GEV.DF defines the distribution function for the GEV, as provided by equations (6) and (7):

```
> GEV.DF<-function(data,mu,sigma,xi)
  {
    if(xi==0)
      {
        GEV<-exp(-exp(-((data-mu)/sigma)))
      }
    else
      {
        GEV<-exp(-(1+xi*((data-mu)/sigma))^(-1/xi))
      }
    GEV
  }
```

Then the following code stores a model–based estimate of the distribution function, evaluated at each point in the ordered sample ordered, in the vector model (recall that maximum likelihood estimates of the GEV parameters were stored in the object a previously):

```
> model<-vector("numeric",length(dataset))
> for(i in 1:length(model))
  {
    model[i]<-GEV.DF(ordered[i],a$est[1],a$est[2],a$est[3])
  }
```

Plotting model against empirical produces the corresponding probability plot for the set of rainfall annual maxima; this can be done using the code:

```
> plot(model~empirical,main='Probability plot')
> abline(0,1)
```

where the code abline(0,1) superimposes the line with intercept 0 and gradient 1 (i.e. the unit diagonal). The resulting plot is shown in Figure 4.

In a similar fashion, we can produce a quantile plot using the following code in **R**:

```
> model.quantile<-vector("numeric",length(dataset))
> GEV.INV<-function(data,mu,sigma,xi)
  {
    if(xi==0)
      {
        INV<-mu-sigma*log(-log(1-data))
      }
```

```
    else
      {
        INV<-mu+(sigma/xi)*(((-log(data))^(-xi))-1)
      }
    INV
  }
> for(i in 1:length(model.quantile))
    {
      model.quantile[i]<-GEV.INV(empirical[i],a$est[1],a$est[2],a$est[3])
    }
> plot(model.quantile~ordered,main='Quantile plot')
> abline(0,1)
```

The function `GEV.INV` computes the inverse of the GEV distribution at each of the points in `data`; we evaluate this for each value in `empirical` to find $\hat{F}(i/(n+1))$ and store the results in the vector `model.quantile`. We then plot `model.quantile` against `ordered` to obtain the quantile plot, again superimposing the line of equality using `abline`. The resulting plot is shown in Figure 5. Both plots show a reasonable fit of the annual rainfall maxima to the GEV model, with points lying fairly close to the unit diagonal.

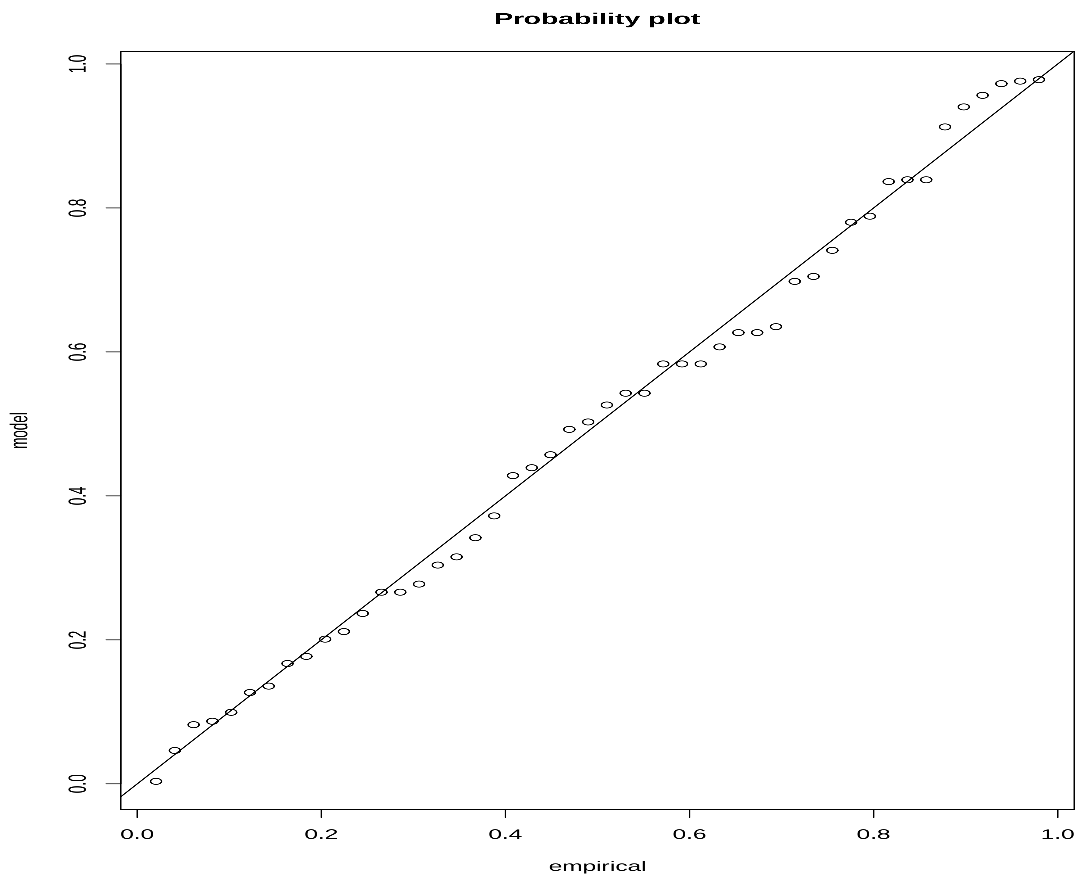

Figure 4: Probability plot for the GEV in the annual maximum rainfall example

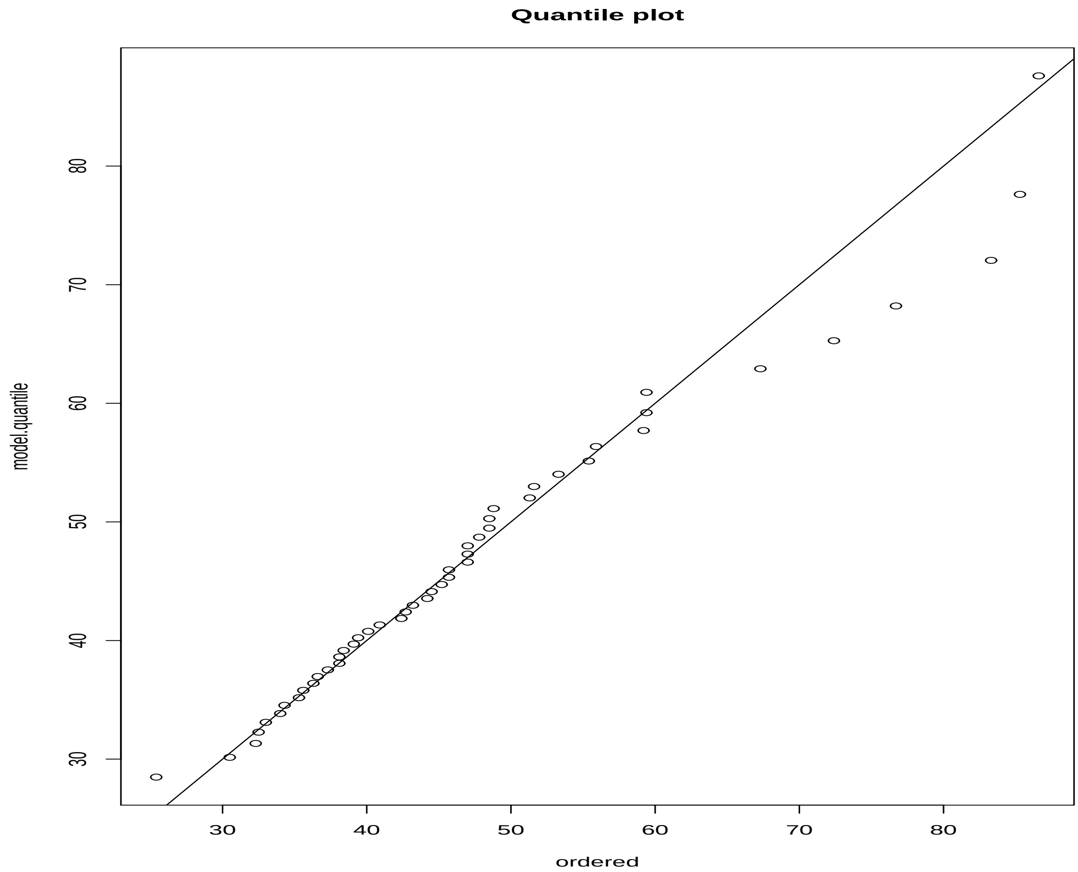

Figure 5: Quantile plot for the GEV in the annual maximum rainfall example

3.2.6 Using the `ismev` package in R

The `ismev` package provides an add–on suite of functions for **R** users to support the examples and computations in *An Introduction to Statistical Modeling of Extreme Values* (Coles, 2001); the acronym of this book gives the package its name. Once installed, `ismev` gives the user access to the datasets used in Coles (2001), and the functions supplied by this package allow the user to reproduce the examples used in the book as well as perform rudimentary extreme value analyses on other datasets. All routines in the `ismev` package assume the data (where necessary) have been pre–processed so that, for example, the user has a set of block maxima ready to analyse. We shall give some thought to the practicalities of obtaining a set of block maxima and dealing with incomplete datasets in the next Section of this book; for now, we assume any necessary data pre–processing has taken place.

Installing the `ismev` package requires a version of **R** no older than **R** 1.50. Provided you have **R** 1.50, or something more recent, installed on your computer, you can load `ismev` using the command:

```
> library(ismev)
```

Doing so will give you access to the following functions which can be used to perform a basic analysis of annual maxima in **R**: `gev.fit`, `gev.prof`, `gev.profxi` and `gev.diag`. The command gev.fit provides a maximum likelihood fitting routine for the GEV that also allows generalised linear modelling of each parameter (we will return to this in Chapter 5). For example, with our set of annual maximum rainfall stored in dataset, typing

```
> gev.fit(dataset)
```

gives the following output:

```
$conv
[1] 0

$nllh
[1] 188.0154

$mle
[1] 40.784484  9.727975  0.107147

$se
[1] 1.5759871 1.1882402 0.1085371
```

If the convergence code, conv, takes the value zero, then successful convergence has been achieved, as is the case in our example above. nllh gives the value of the minimised negative log–likelihood, which has been found to be 188.0154 here – exactly as it was when we minimised the negative log–likelihood in Section 3.2.1 using the nlm routine. The output also shows the maximum likelihood estimators, with their standard errors, giving (to two decimal places):

$$\hat{\mu} = 40.78\ (1.58) \qquad \hat{\sigma} = 9.73\ (1.19) \qquad \hat{\xi} = 0.11\ (0.11)$$

Again, this is exactly what we obtained when fitting the GEV 'from first principles' in Sections 3.2.1 and 3.2.2.

The function gev.profxi produces the profile log–likelihood for the shape parameter ξ; this can be implemented by typing gev.profxi(z,xlow,xup,conf=0.95), where z is an object returned by gev.fit, xlow and xup are the lower and upper values at which to evaluate the profile log–likelihood, and conf evaluates the appropriate cut–off point from the χ_1^2 distribution for the confidence interval required (this is then superimposed on the profile log–likelihood plot). For example, to obtain the profile log–likelihood plot for ξ in our rainfall data, and to construct a 95% confidence interval from this, we might type:

```
> a<-gev.fit(dataset)
> gev.profxi(a,-0.3,0.6,conf=0.95)
```

Doing so gives the plot shown in Figure 6, which is almost identical to the plot produced in Section 3.2.4. Lower and upper bounds for the 95% confidence interval for ξ can then be found at the intersection of the lower horizontal line and the profile log–likelihood. Confidence intervals using the profile log–likelihood can also be obtained for any return level of interest using the function gev.prof. For example, the following code:

```
> gev.prof(a,75,180,conf=0.95)
```

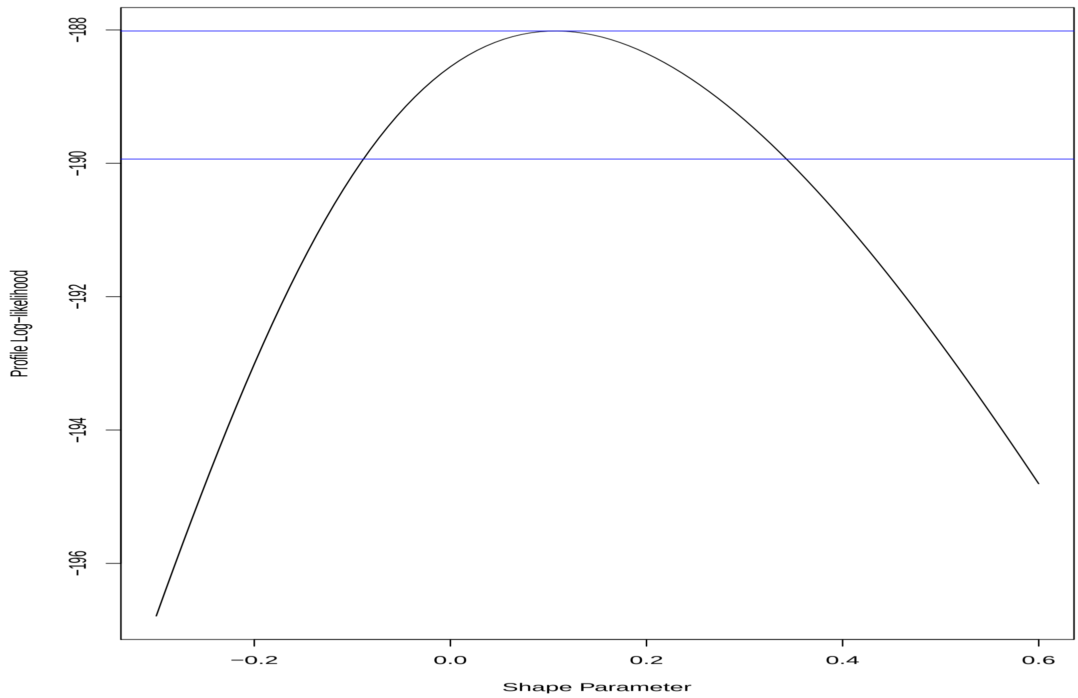

Figure 6: Profile log–likelihood for ξ in the rainfall example, constructed using the `gev.profxi` function in `ismev`

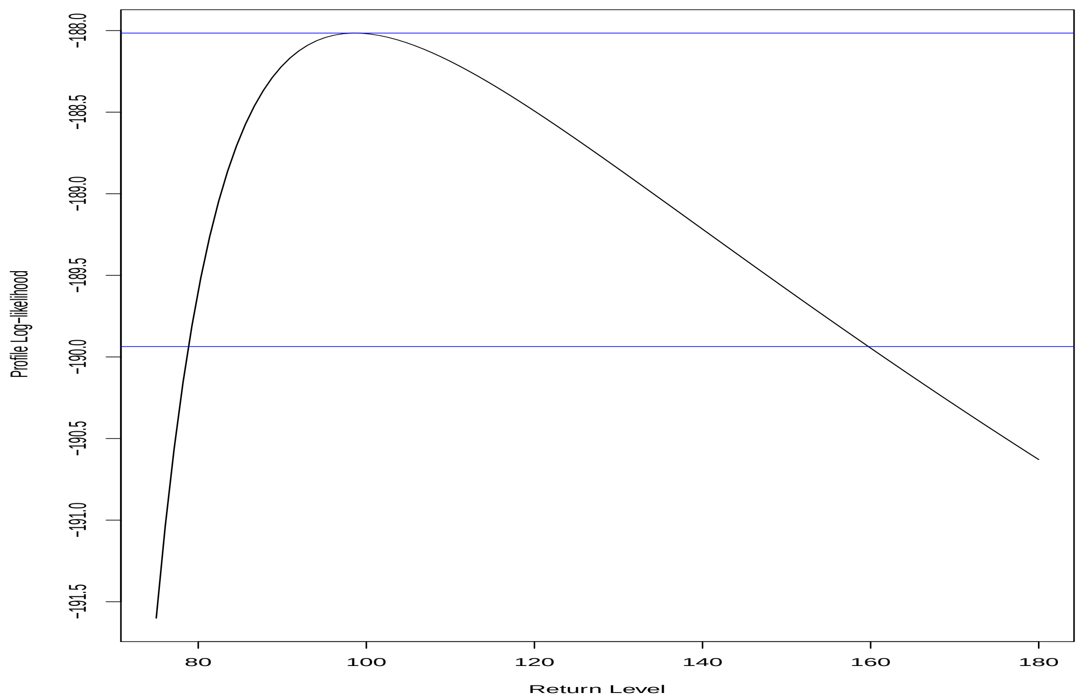

Figure 7: Profile log–likelihood for q_{100} in the rainfall example, constructed using the `gev.prof` function in `ismev`

produces the profile log–likelihood for the 100–year return level (see Figure 7). Again, we can obtain the 95% confidence interval in the usual way. To check the goodness–of–fit of the GEV we can use the `ismev` function `gev.diag`. Recall that the output of the fitting procedure is stored in the object `a`. Then, typing

```
> gev.diag(a)
```

gives the plots shown in Figure 8. The two plots in the top row of this panel are equivalent to the probability plot (left–hand–side) and quantile plot (right–hand–side) we constructed in Section 3.2.5; note the switching of axes in the quantile plot produced by `gev.diag`. However, `gev.diag` also produces a *return level plot* (bottom left) and a *density plot* (bottom right). One problem with probability plots, particularly for extreme value models, is that both the model–based and empirical estimates of probability are bound to converge to 1 as the ordered value ($x_{(i)}$) increases, and it is the goodness–of–fit of the model at such high levels that is of most interest here (in practice, lack–of–fit at high levels of the data are usually not detected by the probability plot). The quantile plot gets around this issue. Indeed, in our rainfall example a potential problem at high levels of the rainfall process is detected by the quantile plot (Figure 5 and Figure 8 (top right)) but *not* detected by the probability plot (Figure 4 and Figure 8 (top left)).

The return level plot is even more informative than the quantile plot for assessing the adequacy of an extreme value model. Here, q_r is now plotted against $\log(-\log(1-1/r))$, giving a linear plot when $\xi = 0$, a convex plot when $\xi < 0$ (with an asymptotic limit at $\mu = \sigma/\xi$ as $1/r \to 0$) and a concave plot when $\xi > 0$ (with no finite bound). This choice of scale also compresses the tail of the distribution, drawing attention to the effect of extrapolation far beyond the range of the data observed. To account for sampling error, confidence intervals have been superimposed on the return level plot for our data in Figure 8, as have empirical estimates taken from the data themselves. Provided the empirical estimates do not show any substantial departure from the model–based estimates (given by the line) – and fall within the 95% confidence bands to allow for sampling error – we can assume the model chosen is adequate. Indeed, there appears to be little cause for concern in the return level plot shown in Figure 8.

The density plot shown in Figure 8 (bottom right) shows the probability density function for the GEV superimposed onto a histogram of the set of annual maxima. Although such a plot is generally less informative than the other plots described (mainly because the form of a histogram can change substantially with the size of the class intervals used), we can see from our density plot that the data are in reasonably close agreement with what we'd expect from the model.

3.3 Practical considerations

3.3.1 Obtaining maxima from a set of data

The example in Section 3.2 assumed that we already have the data in the form of annual maxima. Sometimes this might be the case, and so we can use **R** to fit the GEV to our dataset directly. However, it might be the case that we have daily, or maybe even hourly, measurements, and before we implement an extreme value analysis in **R** we must first obtain this set of maxima. Suppose we have daily measurements on a particular

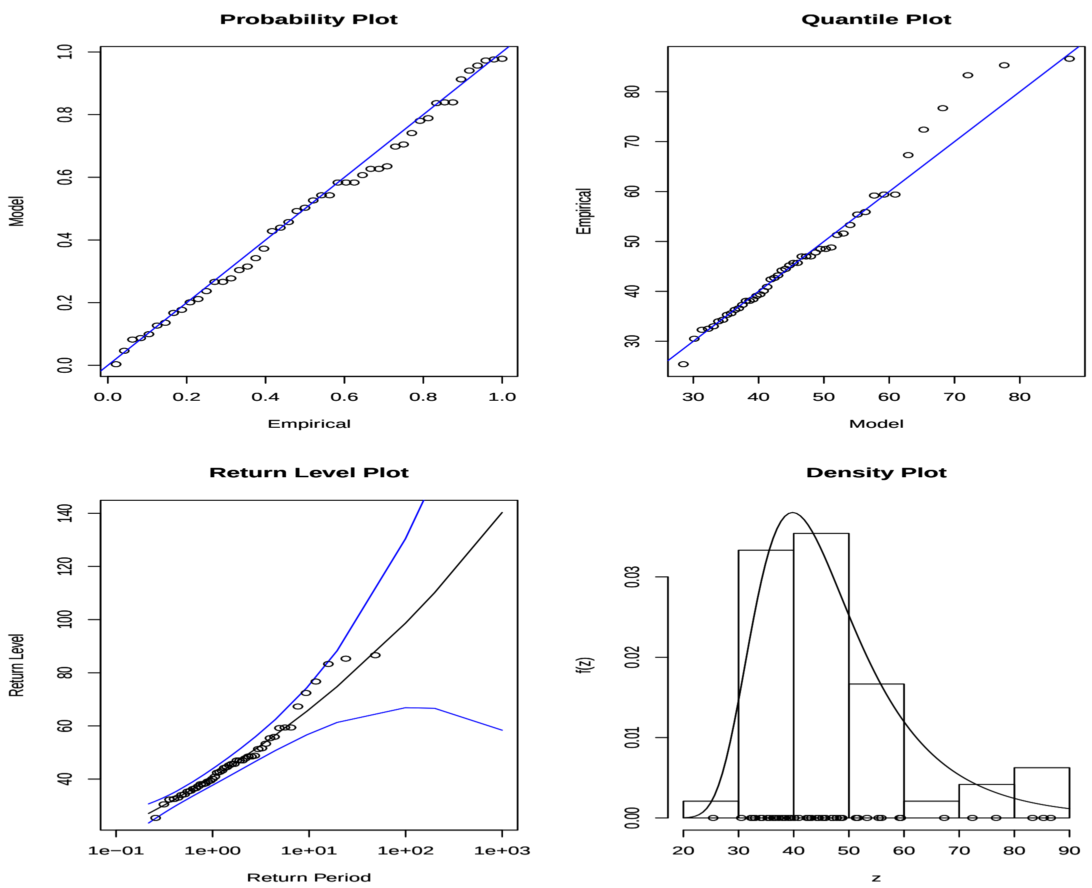

Figure 8: Diagnostic plots created using the function `gev.diag` to check the goodness–of–fit of the GEV model in the rainfall example

variable over a period of seven years – say 2003–2009 inclusive – and we want to perform an extreme value analysis on the set of annual maxima. We can set up a vector containing the years by typing:

```
> year<-seq(2003,2009,1)
```

If we use calendar years as our blocks, we should ensure that we take into account leap years. The following code creates a vector the same length as `year` called `ind`; this will indicate whether or not the corresponding year is a leap year by taking the value 366 for leap years, and 365 elsewhere (a leap year has 366 days, a non–leap year 365):

```
> year.over.four<-year/4
> rounded<-round(year.over.four)
> ind<-vector("numeric",length(year))
> for(i in 1:length(ind))
  {
    if(rounded[i]==year.over.four[i])
      {
        ind[i]<-366
      }
    else{
```

```
      ind[i]<-365
    }
   }
```

The first line in the above code divides the year by four; the second line then rounds this answer. For leap years only, `year.over.four` will be identical to the corresponding value in `rounded`, and this is used in the for loop to give `ind` a value of 366 (or 365 for non–leap years). When obtaining the set of annual maxima, we want to find the maximum value between `lower` and `upper` in each block: for example, for our first year of data, `lower` would be 1 and `upper` 365 (since 2003 is not a leap year), for the second year of data, `lower` would be 366 and `upper` 731 (since 2004 *is* a leap year). We can compute `lower` and `upper` for each year using the following code:

```
> upper<-vector("numeric",length(ind))
> for(i in 1:length(upper))
  {
     upper[i]<-sum(ind[1:i])
   }
> lower<-vector("numeric",length(ind))
> lower[1]<-1
> for(i in 2:length(lower))
  {
    lower[i]<-upper[i-1]+1
  }
```

Now typing

```
> cbind(year,ind,lower,upper)
```

gives

```
     year ind lower upper
[1,] 2003 365     1   365
[2,] 2004 366   366   731
[3,] 2005 365   732  1096
[4,] 2006 365  1097  1461
[5,] 2007 365  1462  1826
[6,] 2008 366  1827  2192
[7,] 2009 365  2193  2557
```

Suppose our daily measurements observed over the seven years are stored in the vector `mydata`; then the following code would extract the set of annual maxima:

```
> anmax<-vector("numeric",length(year))
> for(i in 1:length(anmax))
   {
    anmax[i]<-max(mydata[lower[i]:upper[i]])
   }
```

Of course, the above code could be incorporated into a function where the start year and end year are arguments. For example:

```
> extract<-function(data,start.year,end.year)
  {
    year<-seq(start.year,end.year,1)
    year.over.four<-year/4
    rounded<-round(year.over.four)
    ind<-vector("numeric",length(year))
    for(i in 1:length(ind))
      {
        if(rounded[i]==year.over.four[i])
          {
            ind[i]<-366
          }
        else{
          ind[i]<-365
        }
      }
    upper<-vector("numeric",length(ind))
    for(i in 1:length(upper))
      {
        upper[i]<-sum(ind[1:i])
      }
    lower<-vector("numeric",length(ind))
    lower[1]<-1
    for(i in 2:length(lower))
      {
        lower[i]<-upper[i-1]+1
      }
    anmax<-vector("numeric",length(year))
    for(i in 1:length(anmax))
      {
        anmax[i]<-max(data[lower[i]:upper[i]])
      }
    return(cbind(year,ind,lower,upper),anmax)
  }
```

More generally, the following function would extract a set of block maxima from our dataset with a specified (constant) block length of k observations:

```
> maxima<-function(data,k)
  {
    blockmax<-vector("numeric",length(data)/k)
    for(i in 1:length(blockmax))
      {
        blockmax[i]<-max(data[((k*i)-(k-1)):(k*i)])
      }
    blockmax
  }
```

This function assumes that the block length used divides the data exactly into k equally sized blocks. For example, we might have 300 measurements taken over time stored in `mydata2`, and we might want to extract the maximum value for every 30 observations (i.e. we are using a block length of $k = 30$). Then typing

```
> blockmax.mydata2<-maxima(mydata2,30)
```

would store the set of 10 block maxima in the vector `blockmax.mydata2`. Then we could proceed with an extreme value analysis by typing

```
> gev.fit(blockmax.mydata2)
```

Similar approaches to data pre–processing can be taken to extract, for example, monthly maxima or weekly maxima, or block maxima from data observed more/less frequently than every hour.

The series of 48 annual maxima used to illustrate the use of the GEV in Section 3.2 was extracted from a set of daily rainfall measurements. The entire dataset is available from the `ismev` package by typing

```
> data(rain)
```

Typing

```
> help(rain)
```

gives the following display in **R**:

```
Daily Rainfall Accumulations in South-West England

Description:

     A numeric vector containing daily rainfall accumulations at a
     location in south-west England over the period 1914 to 1962.

Usage:

     data(rain)

Format:

     A vector containing 17531 observations.

Source:

     Coles, S. G. and Tawn, J. A. (1996) Modelling extremes of the
     areal rainfall process. _Journal of the Royal Statistical Society,
     B_ *53*, 329-347.

References:

     Coles, S. G. (2001) _An Introduction to Statistical Modelling of
     Extreme Values._ London: Springer.
```

Notice that the description tells us we have daily rainfall accumulations from 1914–1962; this would suggest there are 49 years of daily measurements, not 48 as we suggested in Section 3.2. However, we are also told that there are 17,531 observations. Now there are 12 leap years between 1914 and 1962 (and thus 37 non–leap years), giving $(12 \times 366) + (37 \times 365) = 17,897$ daily observations, *not* the 17,531 as we are told. In fact, $17,897 - 17,531 = 366$, suggesting there *could* be a whole year of missing observations, that year being a leap year. Another possibility is that the data have been recorded over the period 1914–1961, instead of 1914–1962, though 1962 was not a leap year and so this would not account for the difference of 366 observations. Both the source (Coles and Tawn, 1996) and reference (Coles, 2001) texts do not shed any light on this, and so we choose to assume that we have continuously collected daily measurements from 1914–1961, terminating our final year one observation early. This requires a slight modification of the function `extract`; the last element of the vector `upper` *would be* 17,532, but the length of our dataset is only 17,531. Inserting the following line of code:

```
> upper[length(upper)]<-length(data)
```

straight after the loop `for(i in 1:length(upper)){...}` would make this adjustment for our dataset whilst still keeping the `extract` function generic enough to work for *any* set of daily observations. Now we type:

```
> extract(rain,1914,1961)
```

which gives the following (edited) output:

```
      year ind lower upper
 [1,] 1914 365     1   365
 [2,] 1915 365   366   730
 [3,] 1916 366   731  1096
 [4,] 1917 365  1097  1461
 [5,] 1918 365  1462  1826
 [6,] 1919 365  1827  2191
 [7,] 1920 366  2192  2557
 [8,] 1921 365  2558  2922
  .      .   .     .     .
  .      .   .     .     .
  .      .   .     .     .
[47,] 1960 366 16802 17167
[48,] 1961 365 17168 17531

$anmax
 [1] 44.5 43.2 38.1 39.1 32.3 25.4 33.0 32.5 48.5 34.3 45.7 35.3 76.7 34.0 86.6
[16] 48.5 59.4 53.3 30.5 42.7 83.3 59.2 37.3 67.3 38.4 47.0 38.1 72.4 40.9 47.0
[31] 36.3 85.3 35.6 55.9 44.2 45.2 51.6 59.4 47.8 55.4 42.4 40.1 36.6 47.0 48.8
[46] 51.3 39.4 45.7
```

Notice that the observations listed in `anmax` are exactly the set of maxima we examined in Section 3.2 (see Table 1). To store these in the vector `anmax.rainfall`, as we used initially in Section 3.2, we could type:

```
> b<-extract(rain,1914,1961)
> anmax.rainfall<-b$anmax
```

Now that the annual maxima have been extracted, and stored in the vector `anmax.rainfall`, we can proceed with an extreme value analysis using, for example, `gev.fit` (as we did in Section 3.2.6), or by using our own functions written 'from first principles' (as we did in Sections 3.2.1–3.2.5).

3.3.2 Incomplete data sets and missing values

Something here.

3.3.3 Models for minima

Let $\bar{M}_n = \min\{X_1, \ldots, X_n\}$. If we can assume the X_i are independent and identically distributed, we can apply similar arguments to $\bar{M}_n$ as we applied to M_n in Section 3.1.

Theorem 3.2
If there exist sequences of constants $a_n > 0$ and b_n such that, as $n \to \infty$,

$$\Pr\left\{(\bar{M}_n - b_n)/a_n \leq x\right\} \quad \to \quad \bar{G}(x)$$

for some non–degenerate distribution $\bar{G}$, then $\bar{G}$ is a member of the GEV family of distributions for minima:

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

where $a_+ = \max(0, a)$, $-\infty < \bar{\mu} < \infty$, $\sigma > 0$ and $-\infty < \xi < \infty$. This result can be useful where we are interested in modelling extremely small, rather than extremely large, observations (e.g. annual minimum air temperatures). Alternatively, we could negate our set of block minima and then model the corresponding set of maxima, giving identical maximum likelihood estimates of the GEV parameters but for the sign correction $\hat{\bar{\mu}} = -\hat{\mu}$.

3.4 Generalisation to the r–largest order statistics

Given a sequence of IID random variables $X_1, X_2, \ldots$ we have shown how the generalised extreme value distribution (expression 6) can be used to model the set of normalised annual maxima. This approach is highly inefficient since all but the maximum in each year (or block) are discarded – other observations which could be considered extreme are simply thrown away because they are not as extreme as the maximum value in that year. A generalisation of the result in expression (6) attempts to overcome this, by incorporating the largest r order statistics from each year, where r is any positive integer. If we denote the r largest order statistics in an IID sample by $M^{(1)} \geq M^{(2)} \geq \ldots \geq M^{(r)}$, for $r \geq 1$, then the technique here is to obtain the limiting joint distribution of

$$\left(\frac{M_n^{(1)} - b_n}{a_n}, \frac{M_n^{(2)} - b_n}{a_n}, \ldots, \frac{M_n^{(r)} - b_n}{a_n}\right).$$

It can be shown that the complete class of limiting non–degenerate joint distributions is in fact given by the probability density function

$$
\begin{aligned}
f(x_1, x_2, \ldots, x_r; \mu, \sigma, \xi) \quad = \quad & \sigma^{-r}\exp\left\{ - \left[1 + \xi \left(\frac{x^{(r)} - \mu}{\sigma} \right) \right]_+^{-1/\xi} \right. \\
& \left. - \left(1 + \frac{1}{\xi} \right) \sum_{j=1}^{r} \log \left[1 + \xi \left(\frac{x^{(j)} - \mu}{\sigma} \right) \right]_+ \right\} \qquad (14)
\end{aligned}
$$

for $j = 1, \ldots, r$. As before, the case $\xi = 0$ is taken as the limit as $\xi \to 0$ in (14), to give

$$
\begin{aligned}
f(x_1, x_2, \ldots, x_r; \mu, \sigma, \xi) \quad = \quad & \sigma^{-r}\exp\left\{ - \exp \left[- \left(\frac{x^{(r)} - \mu}{\sigma} \right) \right] \right. \\
& \left. - \sum_{j=1}^{r} \left(\frac{x^{(j)} - \mu}{\sigma} \right) \right\}.
\end{aligned}
$$

Obviously, the case where $r = 1$ is equivalent to the annual maxima approach, for which the GEV holds as the limiting distribution. The increased precision over the traditional annual maxima approach (due to more extremes being incorporated into the analysis) has obvious appeal; however, Smith (1986) shows that as r increases, the rate of convergence to the limiting distribution decreases rapidly, and so the number of order statistics to include must be considered carefully. Such methods must also take account of serial correlation, and are vulnerable to the effects of seasonal variation. Papers in the Journal of Hydrology by Smith (1986) and Tawn (1988b) illustrate the use of r–largest methods;in Section 3.5.6, we apply the technique to the 10 largest sea levels observed each year in Venice.

3.5 Case studies

3.5.1 Wind speed

3.5.2 Rainfall

3.5.3 Air pollution

3.5.4 Material strengths

3.5.5 Financial data

3.5.6 Sea level data

3.6 Further reading

4 The basic model for threshold exceedances: the Generalised Pareto distribution

4.1 History and theoretical motivation

Generalisation of the classical annual maxima approach for modelling extreme values to the r–largest order statistics method was discussed in Section 3.4, the main advantage being the inclusion of more extreme data in the analysis, leading to more precise inferences on the extremal behaviour of the process under study. However, only the r largest values within each year (or block) are used, and any other extremes discarded. In the present chapter, we discuss an approach which aims to include *all* extreme values in the analysis, extreme in the sense that they exceed some pre–determined high level, or *threshold*.

Threshold methods developed rapidly during the 1980s, culminating in the Davison and Smith (1990) paper which applied these techniques to an environmental data set that displayed short–term serial dependence and seasonal variation (see Chapter 5 for more detail on such modelling issues). Since then, threshold methods have become the standard tool for many practitioners involved in modelling extreme values. Relative to the traditional annual maxima and r–largest approaches, threshold methods attempt to maximise efficiency by using *all* extreme values in their analysis; however, as we shall discuss in Chapter 5 (and Section 4.1.2 below), the fact that we use *all* extremes can itself create problems (though, as Davison and Smith (1990) illustrate, pragmatic solutions to these problems can be found).

4.1.1 The generalised Pareto distribution

Ignoring, for now, the practical implications of using *all* our extreme data, again consider a sequence of IID random variables $X_1, X_2, \ldots, X_n$ with common distribution function F. Then for a sufficiently large threshold u, the distribution of $(X - u)$, conditional on $X > u$, is approximately

$$G(y; \bar{\sigma}, \xi) \quad = \quad 1 - \left(1 + \frac{\xi y}{\bar{\sigma}}\right)_+^{-1/\xi}, \tag{15}$$

where $\bar{\sigma}$ (> 0) and ξ $(-\infty < \xi < \infty)$ are scale and shape parameters respectively. The shape parameter ξ in the GPD takes exactly the same value as that for the corresponding GEV distribution; the scale parameter $\bar{\sigma}$ is equal to $\sigma + \xi(u - \mu)$, where σ and μ are the scale and location parameters (respectively) in the corresponding GEV distribution (see expression 6). Specifically, G is defined on $0 < y < \infty$ if $\xi > 0$, and $0 < y < -\bar{\sigma}/\xi$ if $\xi \leq 0$. The case $\xi = 0$ is interpreted as the limit $\xi \to 0$, and is the exponential distribution with rate $1/\bar{\sigma}$. This is known as the *generalised Pareto distribution*, or GPD. The GPD is a limiting distribution for excesses over thresholds if, and only if, the parent distribution lies in the domain of attraction of one of the three extreme value distributions (see Theorem 3.1). However, since the limiting distribution of sample maxima follows one of the distributions given in Theorem 3.1 no matter what the parent distribution, the GPD is the *only* non–degenerate limiting distribution for excesses over thresholds of IID sequences. Until this point we have used the notation $\bar{\sigma}$ to denote the scale parameter for the GPD, so as to distinguish it from the corresponding parameter of the GEV distribution. For notational convenience we now drop this distinction, using σ to denote the scale parameter within either family.

The GPD yields several important properties. One first of these, known as the 'threshold stability property', is that if $(X - u_0)$ follows a generalised Pareto distribution (conditional on $X > u_0$), then $(X - u)$ also follows a generalised Pareto distribution for any $u > u_0$. In other words, once a suitably high enough threshold has been found such that the GPD may be assumed a valid model for excesses over that threshold, then the GPD holds for excesses over any higher threshold too. Another property unique to the GPD is that if $N \sim$ Poisson, and $X_1, \ldots, X_N$ are IID random variables following a GPD, then $\max\{X_1, \ldots, X_N\}$ has the GEV distribution. As will be demonstrated, the threshold stability property can be exploited in graphical procedures for threshold selection and assessing the fit of the GPD. The second property suggests that, if the exceedances of u occur as a Poisson process with threshold excesses which are IID and generalised Pareto distributed, then the maximum value over any block size has a generalised extreme value distribution. Thus, if we assume extreme events occur over time as a Poisson process, models which fit the GEV to sets of block maxima are consistent with models which fit the GPD to sets of threshold excesses.

As with fitting the GEV, most practitioners use numerical maximum likelihood estimation to fit (15) to a set of threshold excesses. For $-1 < \xi \leq -0.5$, maximum likelihood estimators exist (in large enough samples), though in general (as before) do not possess all of the standard asymptotic properties; when $\xi \leq -1$, maximum likelihood estimators do not, in general, exist. For $\xi > -0.5$, maximum likelihood estimators are asymptotically normal and efficient (Smith, 1985). Luckily, in most environmental applications, values of $\xi \leq -0.5$ are rare, but do occur from time to time. Again, the Bayesian methodology provides a framework within which this problem can be avoided. However, in a Bayesian setting, use of the GPD(σ, ξ) model may be restrictive since the scale parameter σ is dependent on the choice of threshold level u; an uninformative prior for σ then becomes informative at higher thresholds. To overcome this, the GPD can be reparameterised with scale and shape parameters $\tilde{\sigma}$ and ξ (respectively, ξ remaining unchanged), where $\tilde{\sigma} = \sigma - \xi u$. Under this parameterisation, both parameters are threshold–independent. As demonstrated in Chapter 3 for the GEV, estimates of extreme quantiles can be obtained through inversion of (15).

4.1.2 When should we model threshold exceedances?

Since extremes are – by their very nature – scarce, any modelling approach that increases precision has obvious appeal. The r–largest approach attempts to increase precision through the inclusion of more data, but could still be considered rather wasteful. The aim of threshold methods is to maximise precision by analysing *all* extremes. However, we might not have access to the entire dataset; it might be the case that we only have the set of derived block maxima – in which case the most appropriate method of analysis would be to fit the GEV to our data directly.

As we shall discuss in Section 4.3 of this Chapter, and Chapter 5, there might also be various modelling issues to address arising as a direct consequence of using threshold exceedances, not least the problem of short term temporal dependence. Shown in Figure

9 is the autocorrelation function, and partial autocorrelation function, for the entire series of daily rainfall measurements discussed in Chapter 3 (recall that we analysed the set of rainfall annual maxima in Chapter 3). These plots were produced using the commands:

```
> acf(rain)
> pacf(rain)
```

where, as in Chapter 3, the series of daily rainfall measurements are stored in the vector `rain`.

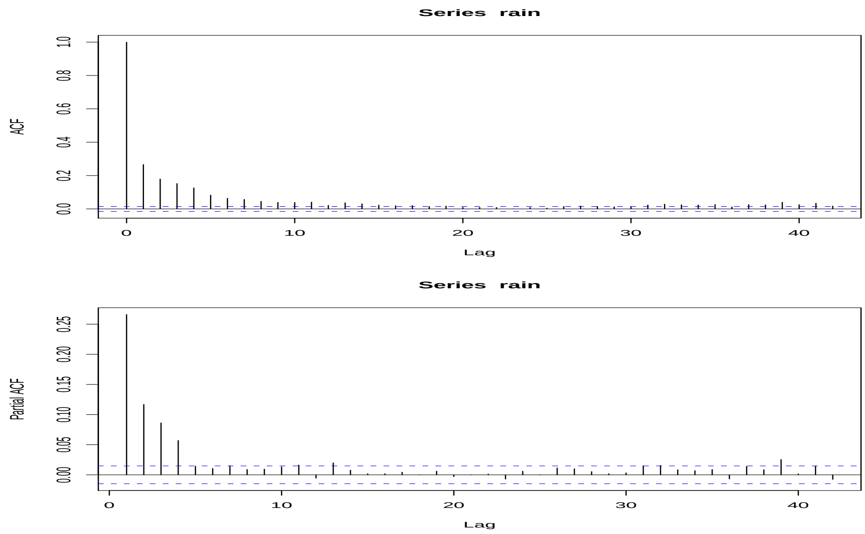

Figure 9: Autocorrelation function, and partial autocorrelation function, for the rainfall data

Both plots in Figure 9 indicate the presence of short term autocorrelation in the series, particularly the plot of partial autocorrelations which eliminates the effect of intermediate autocorrelations. For example, a process that is truly first–order Markov might have significant autocorrelations at lag 2 and beyond owing wholly to the temporal effects induced by the strength of dependence between successive observations. However, the partial autocorrelation function would provide a better indication of the order of dependence, showing only the first partial autocorrelation as significant. For the rainfall data, we see significance in the partial autocorrelation function up to lag 4, suggesting genuine short–term temporal dependence. The GPD in the form it is given in equation (15) assumes our series is IID, which is clearly not the case here. Though (as we shall see in Chapter 5) various techniques have been developed to circumvent the problem of temporal dependence, it is not always obvious how to implement these and parameter estimates can be sensitive to the technique chosen. When temporal dependence and other modelling issues might arise as a direct consequence of using all threshold exceedances, it might be considered preferable to work with a set of block maxima (or perhaps a set of "cluster maxima" – see Section 4.3.3).

4.1.3 How is the GPD used?

Once we have identified our set of threshold exceedances, a typical application would fit the model in (15) via maximum likelihood (perhaps) to obtain estimates of the scale and shape parameters σ and ξ (the problems associated with maximum likelihood for particular values of ξ, for example, are discussed in Section 3.1.4 and also apply here). We can then use our estimates of σ and ξ to obtain estimates of return levels by inversion of (15). For example, suppose that a GPD with parameters σ and ξ is a suitable model for exceedances of a threshold u by a variable X, i.e. for $x > u$,

$$\Pr(X > x | X > u) \quad = \quad \left[1 + \xi\left(\frac{x-u}{\sigma}\right)\right]_+^{-1/\xi}.$$

Then

$$\Pr(X > x) \quad = \quad \lambda_u \left[1 + \xi\left(\frac{x-u}{\sigma}\right)\right]_+^{-1/\xi},$$

where $\lambda_u = \Pr(X > u)$. Hence, the level x_t that is exceeded on average once every t observations is the solution of

$$\lambda_u \left[1 + \xi\left(\frac{x_t-u}{\sigma}\right)\right]_+^{-1/\xi} \quad = \quad \frac{1}{t}.$$

Rearranging, we get

$$x_t \quad = \quad u + \frac{\sigma}{\xi}[(t\lambda_u)^\xi - 1],$$

provided t is sufficiently large to ensure that $x_t > u$, and $\xi \neq 0$. If $\xi = 0$, we have the exponential case, and so

$$x_t \quad = \quad u + \sigma \log(t\lambda_u),$$

again provided t is sufficiently large. By construction, x_t is the t–observation return level; however, it is often more convenient to give return levels on an annual scale, so that the r*–year* return level is the level expected to be exceeded once every r *years*. If there are n_y observations per year, this corresponds to the t–observation return level with $t = r \times n_y$. Hence, the r–year return level q_r is defined by

$$q_r \quad = \quad u + \frac{\sigma}{\xi}\left[(rn_y\lambda_u)^\xi - 1\right], \tag{16}$$

unless $\xi = 0$, in which case

$$q_r \quad = \quad u + \sigma \log(rn_y\lambda_u). \tag{17}$$

We can then estimate the r–year return level q_r by substituting our estimates of σ and ξ (say $\hat{\sigma}$ and $\hat{\xi}$) into equation 16 (or equation 17 when $\xi = 0$); λ_u can be estimated empirically as the proportion of threshold exceedances, and u is the threshold chosen to identify extremes (see Section ?.??). The usual approach for estimating standard errors for the GPD parameters can be used (i.e. inversion of the expected information matrix) and, as in Chapter 3, the delta method can be used to obtain estimated standard errors for return levels. We can also use profile likelihood to estimate more appropriate confidence intervals for return levels.

4.2 Simple case study

In this Section, we demonstrate a simple application of the GPD to a set of threshold exceedances. We illustrate this by using a set of threshold exceedances from the full rainfall series used in Chapter 3. Recall that this can be loaded into **R** by first installing the ismev package:

```
> library(ismev)
```

and then typing:

```
> data(rain)
```

Typing

```
> help(rain)
```

gives a description of the rainfall series. In this Section, we will

- exploit the threshold stability property of the GPD to produce a graphical tool for identifying a suitable threshold u;
- maximise the log–likelihood function for the GPD in **R**;
- use **R** to obtain the expected information matrix, and then invert this to obtain the estimated variance–covariance matrix for $(\hat{\lambda}_u, \hat{\sigma}, \hat{\xi})^T$;
- use the fitted values of the GPD parameters to estimate some return levels q_r, along with standard errors for these;
- use **R** to plot the profile log–likelihood for some return levels and obtain profile likelihood confidence intervals;
- use **R** to check the goodness–of–fit of the GPD to our series of threshold exceedances,

4.2.1 Identifying a suitable threshold: the mean residual life plot

In a mean residual life (MRL) plot we make use of the fact that if the GPD is the correct model for all exceedances x_i above some high threshold u_0, then the *mean excess*, i.e. the mean value of $(x_i - u)$, plotted against $u > u_0$, should give a linear plot. This is because $E[X_i - u_0]$ is a linear function of $u : u > u_0$. By producing such a plot for values of u starting at zero, we can select reasonable candidate values for u_0. In **R**, the following code sets up a vector of possible thresholds, starting at zero and going up to the maximum value in our dataset in steps of 0.1:

```
> u<-seq(0,max(rain),0.1)
```

The vector `x` will now be set up to take the corresponding values for the mean excess over each value in `u`:

```
> x<-vector('numeric', length(u))
```

Then the following code computes the mean excess for each value in `u` and stores it in `x`:

```
> for(i in 1:length(x))
  {
    threshold.exceedances<-rain[rain>u[i]]
    x[i]<-mean(threshold.exceedances-u[i])
  }
```

The MRL plot is then produced using the following code, giving the plot in Figure 10:

```
> plot(x~u,type='l', main='MRL plot',ylab='mean excess')
```

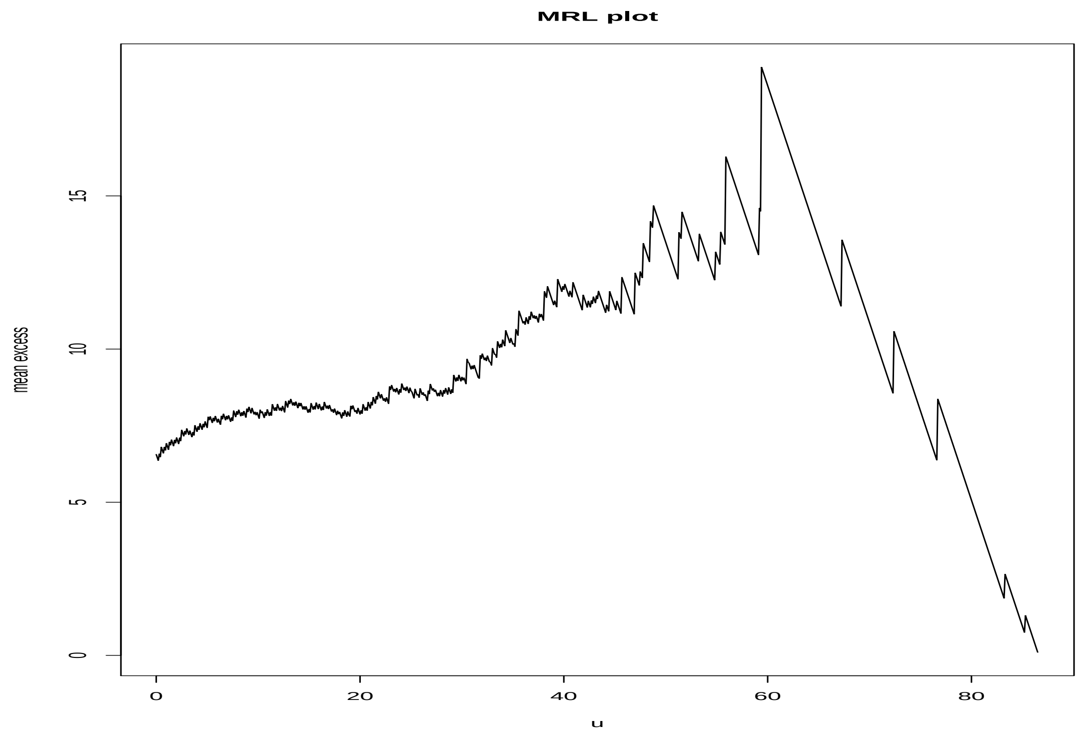

Figure 10: Mean residual life plot for the rainfall data

Though interpretation of these plots can be subjective, linearity in Figure 10 might be suggested at about $u_0 = 30$mm (information in the far right–hand–side of these plots is unreliable; here, variability is high due to the limited amount of data above such high thresholds). Using $u_0 = 30$ as our threshold for identifying extremes, we can then obtain our set of threshold exceedances for modelling with the generalised Pareto distribution:

```
> above.threshold<-rain[rain>30]
> threshold.exceedances<-above.threshold-30
```

We can look at our set of threshold exceedances by typing:

```
> threshold.exceedances
  [1]  1.8  2.5  1.8 14.5  0.5 13.2  5.6  8.1  2.0  1.8  3.0  9.1  0.5  1.8  2.3
 [16]  3.0  0.5  2.5 18.5  5.3 10.6  0.5  4.3  2.8  0.5 15.7  1.8  3.5  3.5  1.8
 [31]  4.8  5.3  7.8 46.7  2.3  4.0  3.8  6.6  0.5 15.7 56.6  5.6 17.8 17.5  4.3
 [46] 18.5  0.7 13.4 29.4  5.1 23.3  3.5  0.5  0.2 10.9 12.7 53.3 24.9 29.2  1.8
```

```
 [61]   7.3  2.5  4.0 37.3  1.2  0.2  6.1  6.8  8.4  1.0  3.3 17.0  2.0  3.0  8.1
 [76]   0.5 42.4  4.3  7.1  3.0 10.9  9.9 17.0  6.3  0.5  0.5 25.9  1.8 21.3 55.3
 [91]  11.9  0.5  3.0  5.6 25.9 14.2  8.1  4.3  1.8  2.0  1.8  5.6 15.2  0.5  9.4
[106]   0.2 14.5  1.8  3.8 21.6  5.3 29.4  3.5  5.3  0.5  6.8 17.8 12.9  7.6 25.4
[121]   5.3 12.4  3.0  3.0 10.1  4.8  8.1  9.4  4.0  5.6  4.3  3.5  1.0  6.6  6.3
[136]   8.4  8.1 17.0  1.0  0.5  1.2  5.6 18.8 11.9  1.7  1.2 21.3  3.5  7.6  9.4
[151]   9.4 15.7
```

Thus, we have identified 152 observations as being extreme.

4.2.2 Fitting the GPD

The GPD log–likelihood function can be derived in the same way that the log–likelihood for the GEV was derived in Section 3.2.1; this is left as an exercise for the reader, but can be shown to be:

$$\ell(\sigma,\xi;\boldsymbol{y}) \quad = \quad -152\log\sigma - (1+1/\xi)\sum_{i=1}^{152}\log_e\left(1+\frac{\xi y_i}{\sigma}\right)_+, \tag{18}$$

where $\boldsymbol{y} = (y_1,\ldots,y_{152})$ are the set of exceedances above threshold $u_0 = 30$. For the case $\xi = 0$, interpreted as $\xi \to 0$, we have the log–likelihood for an exponential distribution with rate $1/\sigma$. We thus define the GPD log–likelihood in **R** in the following way:

```
> dataset<-threshold.exceedances

> theta<-c(sd(dataset),0.1)

> gpd.loglik<-function(theta)
  {
    sigma<-theta[1]
    xi<-theta[2]
    m<-min(1+(dataset*(xi/sigma)))
    if(m<0.00001)return(as.double(1000000))
    if(sigma<0.00001)return(as.double(1000000))
    loglik<--length(dataset)*log(sigma)-sum(log(1+(dataset*(xi/sigma)))*(1/xi+1))
    return(-loglik)
  }
```

The reader is referred to the corresponding code for the log–likelihood function for the GEV (Section 3.2.1) at this point for any clarification of the above code. We can now use the **R** routine `nlm` to minimise the negative log–likelihood, as we did for the GEV:

```
> nlm(gpd.loglik,theta,hessian=TRUE)
$minimum
[1] 485.0937

$estimate
[1] 7.4402433 0.1845009

$gradient
```

```
[1] -2.101763e-05  8.327561e-05

$hessian
          [,1]      [,2]
[1,]  2.000085  12.79144
[2,] 12.791440 179.34888

$code
[1] 1

$iterations
[1] 12
```

As in Chapter 3, we could then obtain estimated standard errors for the GPD parameters:

```
> a<-nlm(gpd.loglik,theta,hessian=TRUE)
> varcovar<-solve(a$hessian)
> sqrt(diag(varcovar))
[1] 0.9588034 0.1012522
```

An estimate of the threshold exceedance rate, λ_u, can be obtained by typing:

```
> lambda<-length(threshold.exceedances)/length(rain)
[1] 0.008670355
```

Since the number of exceedances of $u_0 = 30$ follows a binomial distribution $\text{Bin}(N, \lambda_u)$, where N is the total number of observations in our series, then

$$\text{Var}(\hat{\lambda}_u) \approx \hat{\lambda}_u(1 - \hat{\lambda}_u)/N,$$

giving an estimated standard error for the threshold exceedance rate of

$$\sqrt{0.0087(1 - 0.0087)/17531} \approx 0.0007.$$

Thus, in summary, we have:

$$\hat{\sigma} = 7.44\ (0.96) \qquad \hat{\xi} = 0.18\ (0.10) \qquad \hat{\lambda}_u = 0.0087\ (0.0007),$$

giving the following symmetric confidence intervals for σ, ξ and λ_u respectively:

$$(5.56, 9.32) \qquad (-0.016, 0.376) \qquad (0.007, 0.010)$$

Note the similarity in estimates of ξ in this analysis (0.18) and the GEV analysis in Chapter 3 (0.11), with the confidence intervals from both analyses including zero (both suggesting the possibility of a Gumbel–type tail). Also notice that, using the earlier notation for the scale parameter of the GPD, and denoting by $\hat{\mu}_{\text{GEV}}$, $\hat{\sigma}_{\text{GEV}}$ and $\hat{\xi}_{\text{GEV}}$ the GEV parameter estimates from Chapter 3, we have

$$\begin{aligned}\hat{\tilde{\sigma}} &= \hat{\sigma}_{\text{GEV}} + \hat{\xi}_{\text{GEV}}(u_0 - \hat{\mu}_{\text{GEV}}) \\ &= 9.73 + 0.11 \times (30 - 40.78) \\ &= 8.54,\end{aligned}$$

which is not significantly different from our estimate of the scale parameter in the analysis of threshold exceedances.

4.2.3 Threshold choice revisited: parameter stability plots

If the GPD with shape parameter ξ and scale parameter σ_{u_0} is the correct model for excesses over u_0, then, owing to the threshold stability property, for any threshold $u > u_0$, the excesses will be GPD with shape parameter ξ and scale parameter

$$\sigma_u = \sigma_{u_0} + \xi(u - u_0).$$

If we now use a modified version of the scale parameter,

$$\sigma^* = \sigma_u - \xi u,$$

we can see that both σ^* and ξ should be constant over thresholds greater than u_0 if we model excesses $x_i - u$ for $u > u_0$ using the GPD. This provides us with a further tool for assessing our original choice of threshold u_0: we can refit the GPD for a range of thresholds upwards of u_0 and investigate the stability of our estimates of ξ and σ^*. In **R**, we now let `u` take values between the threshold indicated by the mean residual life plot ($u_0 = 30$) and some values upwards of this (we arbitrarily choose a maximum value of 50):

```
> u<-seq(30,50,1)
```

Now the following code sets up vectors `sigma.star`, `xi`, and `s.e.sigma.star` and `s.e.xi`, to take the MLEs for σ^* and ξ and and the corresponding estimated standard errors (respectively) for each threshold in `u`:

```
> sigma.star<-vector('numeric',length(u))
> xi<-vector('numeric',length(u))
> s.e.sigma.star<-vector('numeric',length(u))
> s.e.xi<-vector('numeric',length(u))
```

The following loop identifies the set of threshold excesses for each value in `u` and then fits the GPD, storing each estimated value of σ^* and ξ in `sigma.star` and `xi` (respectively):

```
> for (i in 1:length(u))
  {
    above.thresh<-rain[rain>u[i]]
    threshold.exceed<-above.thresh-u[i]
    dataset<-threshold.exceed
    b<-nlm(gpd.loglik,theta,hessian=TRUE)
    xi[i]<-b$estimate[2]
    sigma.star[i]<-b$estimate[1]-xi[i]*u[i]
```

Our parameter stability plots will show our estimated values for σ^* and ξ for each threshold in `u`, along with the associated confidence intervals constructed using the estimated standard errors. The estimated standard error for ξ is obtained directly from the variance–covariance matrix, which itself is found using the code `solve(a$hessian)` within the loop above. However, the standard error for the modified scale parameter σ^* requires use of the delta method (see Section 3.2.3), where

$$\mathrm{Var}(\sigma^*) \approx \nabla\sigma^{*T} V \nabla\sigma^*$$

and

$$\nabla\sigma^{*T} = \left[\frac{\partial\sigma^*}{\partial\sigma_u}, \frac{\partial\sigma^*}{\partial\xi}\right] = [1, -u].$$

The code shown below continues the loop from above, storing the estimated standard error for ξ at each threshold in `s.e.xi`; the delta method is also applied to obtain the estimated standard error for each σ^*, and this is stored in `s.e.sigma.star`.

```
    varcov<-solve(a$hessian)
    s.e.xi[i]<-sqrt(varcov[2,2])
    del<-matrix(ncol=1,nrow=2)
    del[1,1]<-1
    del[2,1]<--u[i]
    del.transpose<-t(del)
    s.e.sigma.star[i]<-sqrt(del.transpose%*%varcov%*%del)
  }
```

Then the lower and upper 95% confidence bounds can be obtained for σ^* and ξ:

```
> lower.sigma.star<-sigma.star-(1.96*s.e.sigma.star)
> upper.sigma.star<-sigma.star+(1.96*s.e.sigma.star)
> lower.xi<-xi-(1.96*s.e.xi)
> upper.xi<-xi+(1.96*s.e.xi)
```

Finally, a plot can be produced (Figure 11) showing each estimated value of σ^* and ξ against u, with their associated 95% confidence intervals, using the following code:

```
> par(mfrow=c(2,1))
> plot(sigma.star~u,type='b',ylim=c(-50,170),lwd=2,ylab='sigma.star')
> lines(lower.sigma.star~u,type='l',lty=2)
> lines(upper.sigma.star~u,type='l',lty=2)
> plot(xi~u,type='b',ylim=c(-2,1),lwd=2,ylab='xi')
> lines(lower.xi~u,type='l',lty=2)
> lines(upper.xi~u,type='l',lty=2)
```

The values in `ylim` determine the plotting range; for the parameter stability plot for σ^*, for example, typing `min(lower.sigma.star)` and `max(upper.sigma.star)` suggest a plotting range between about –50 and 170 would be appropriate. The argument `lwd=2` controls the weight of the plotting line, and in this example we have set this equal to 2 so that the maximum likelihood estimates will appear bold relative to the lines that represent the confidence bounds. Figure 11 shows much variation in the far right–hand–side of the plots, owing to few observations being available for analysis above such high thresholds. However, overall we seem to have reasonable 'stability' in our plots for all thresholds above 30, which suggests we can be reassured about our original choice of $u_0 = 30$.

4.2.4 Return level inference

We can obtain point estimates of any return level by substituting our estimates of the GPD parameters into equation (16). For example, an estimate of the 100–year return level q_{100} can be found as

$$\begin{aligned}\hat{q}_{100} &= 30 + \frac{7.44}{0.18}\left[(100 \times 365.25 \times 0.0087)^{0.18} - 1\right] \\ &= 105.26\end{aligned}$$

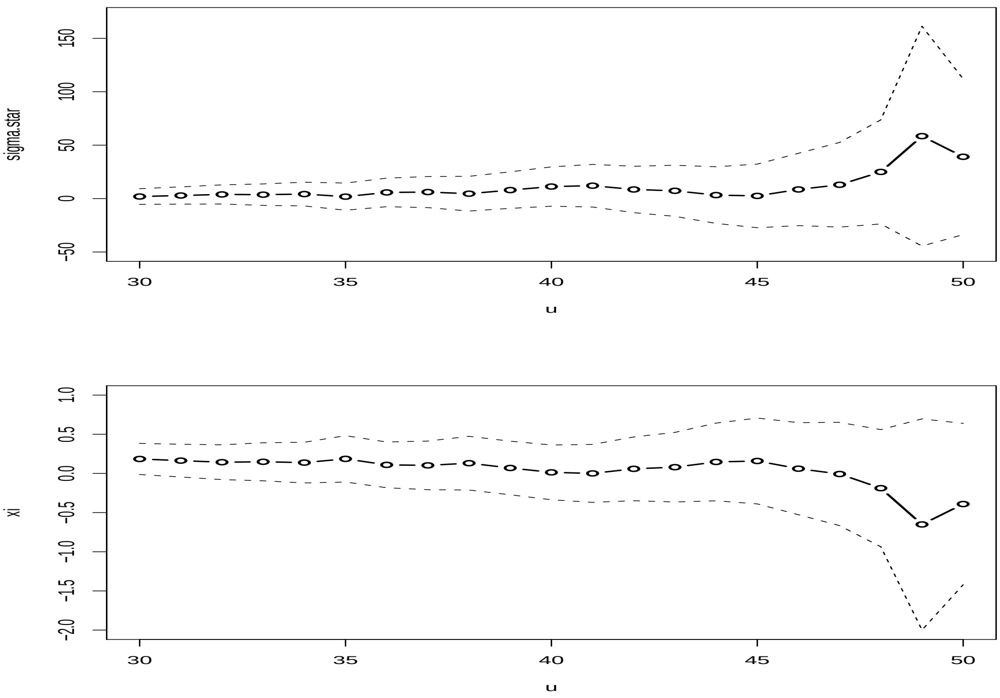

Figure 11: Parameter stability plots for σ^* and ξ for the rainfall data

where, to account for leap years, 365.25 is used as the number of daily observations per year. The delta method can, once again, be used to obtain estimated standard errors for such return levels. We should, however, also include uncertainty in our estimate of λ_u in the calculation (since q_r is a function of λ_u). Recall that, by the delta method,

$$\mathrm{Var}(\hat{q}_r) \quad \approx \quad \nabla q_r^T V \nabla q_r.$$

Here, V is now the variance–covariance matrix of the triple $(\hat{\lambda}_u, \hat{\sigma}, \hat{\xi})^T$; in our rainfall example, this is

$$V \quad = \quad \begin{pmatrix} 0.0007^2 & & \\ 0 & 0.96^2 & \\ 0 & -0.0656 & 0.10^2 \end{pmatrix};$$

the value –0.0656 is the estimated covariance between $\hat{\sigma}$ and $\hat{\xi}$, and is found in the variance–covariance matrix `varcovar`. We now write a new variance–covariance matrix in **R** to include the variance of $\hat{\lambda}_u$; we call this `varcovar2`:

```
> varcovar2<-matrix(ncol=3,nrow=3)
> varcovar2[]<-0
> varcovar2[1,1]<-0.0007^2
> varcovar2[2,2]<-0.96^2
> varcovar2[3,3]<-0.1^2
> varcovar2[2,3]<--0.0656
> varcovar2[3,2]<-0.0656
```

	$\hat{q}_{10}$	$\hat{q}_{50}$	$\hat{q}_{100}$	$\hat{q}_{1000}$
MLE (e.s.e.)	65.96 (5.14)	92.34 (14.09)	105.26 (20.09)	168.10 (58.33)

Table 4: Some estimated return levels for the rainfall data using the threshold excess model. Associated standard errors are in parentheses (units are millimetres)

As before,

$$\nabla q_r^T = \left[\frac{\partial q_r}{\partial \lambda_u}, \frac{\partial q_r}{\partial \sigma}, \frac{\partial q_r}{\partial \xi}\right];$$

this can be shown to give

$$\begin{aligned}\nabla q_r^T = & \left[\sigma (r n_y)^{\xi} \lambda_u^{\xi-1}, \xi^{-1}\left\{(r n_y \lambda_u)^{\xi} - 1\right\},\right. \\ & \left. -\sigma\xi^{-2}\left\{(r n_y \lambda_u) - 1\right\} + \sigma\xi^{-1}(r n_y \lambda_u)^{\xi} \log(r n_y \lambda_u)\right],\end{aligned}$$

which we evaluate at $(\hat{\lambda}_u, \hat{\sigma}, \hat{\xi})$. In **R**, the following code computes the estimated standard error for the 100–year return level q_{100}:

```
> r<-100
> ny<-365.25
> del<-matrix(ncol=1,nrow=3)
> sigma<-a$estimate[1]
> xi<-a$estimate[2]
> lambda<-length(threshold.exceedances)/length(rain)
> del[1,1]<-sigma*(r*ny)^(xi)*lambda^(xi-1)
> del[2,1]<-xi^(-1)*((r*ny*lambda)^(xi)-1)
> del[3,1]<--sigma*(xi^(-2))*((r*ny*lambda)^(xi)-1)
+sigma*xi^(-1)*(r*ny*lambda)^(xi)*log(r*ny*lambda)
> del.transpose<-t(del)
> sqrt(del.transpose%*%varcovar2%*%del)
         [,1]
[1,] 20.09450
```

Thus, an estimate of the 100–year return level q_{100}, with its standard error in parentheses, is: 105.26 (20.45) from which we could obtain a confidence interval. Obviously, we could replace `r<-100` in the above code with `r<-10`, `r<-50` or `r<-1000`, for example, to obtain the estimated standard error for the 10, 50 and 1000 year return levels (respectively). In fact, Table 4 shows exactly these results.

As we did for the GEV in Section 3.2.3 we could, of course, combine the above code with that used to obtain the GPD MLEs and estimated standard errors to write a function that does everything in a single sweep; i.e., given the `data`, a suitable `threshold`, the number of observations per year `ny` and a specified return `period` r, our function could return $(\hat{\lambda}_u, \hat{\sigma}, \hat{\xi})$, with the estimated standard errors for these, as well as the r–year return level q_r along with its estimated standard error. An example of this is shown overleaf in the function `gpdfit`.

```
gpdfit<-function(data,threshold,ny,r)
  {
     above.threshold<-data[data>threshold]
     thresh.exceed<-above.threshold-threshold

     lambda<-length(thresh.exceed)/length(data)
     s.e.lambda<-sqrt(lambda*(1-lambda)/length(data))

     theta<-c(sd(thresh.exceed),0.1)
     gpd.loglik<-function(theta)
       {
         sigma<-theta[1]
         xi<-theta[2]
         m<-min(1+(thresh.exceed*(xi/sigma)))
         if(m<0.0001)return(as.double(1000000))
         if(sigma<0.0001)return(as.double(1000000))
         loglik<--length(thresh.exceed)*log(sigma)
          -sum(log(1+(thresh.exceed*(xi/sigma)))*(1/xi+1))
         return(-loglik)
       }
     a<-nlm(gpd.loglik,theta,hessian=TRUE)
     gpd<-c(lambda,a$est)

     varcovar<-solve(a$hessian)
     ese<-c(s.e.lambda,sqrt(diag(varcovar)))

     ret.level<-threshold+(a$est[1]/a$est[2])*(((r*ny)*lambda)^(a$est[2])-1)

     varcovar2<-matrix(ncol=3,nrow=3)
     varcovar2[]<-0
     varcovar2[1,1]<-ese[1]^(2)
     varcovar2[2,2]<-ese[2]^(2)
     varcovar2[3,3]<-ese[3]^2
     varcovar2[2,3]<-varcovar[2,1]
     varcovar2[3,2]<-varcovar[2,1]
     del<-matrix(ncol=1,nrow=3)

     del[1,1]<-(a$est[1])*(r*ny)^((a$est[2]))*lambda^((a$est[2])-1)
     del[2,1]<-(a$est[2])^(-1)*((r*ny*lambda)^((a$est[2]))-1)
     del[3,1]<--(a$est[1])*((a$est[2])^(-2))*((r*ny*lambda)^((a$est[2]))-1)
      +(a$est[1])*(a$est[2])^(-1)*(r*ny*lambda)^((a$est[2]))*log(r*ny*lambda)
     del.transpose<-t(del)
     ese.ret.level<-sqrt(del.transpose%*%varcovar2%*%del)
     ese.ret.level<-ese.ret.level[1,1]

     return(gpd,ese,ret.level,ese.ret.level)
   }
```

To execute this code for the threshold exceedances in the rainfall dataset, and to estimate the 100–year return level, we would type:

```
> gpdfit(rain,30,365.25,100)
```

which gives the following output:

```
$gpd
[1] 0.008670355 7.440243310 0.184500851

$ese
[1] 0.0007002033 0.9588034461 0.1012521934

$ret.level
[1] 106.3430

$ese.ret.level
[1] 20.85760
```

The three elements returned in `gpd` are the MLEs for λ_u, σ and ξ, respectively; underneath, in `ese`, we have the corresponding estimated standard errors. `ret.level` is the estimated 100–year return level, whose estimated standard error is stored in `ese.ret.level`. Notice that here, we have $\hat{q}_{100} = 106.34$ (20.86), whereas in Table 4 we have $\hat{q}_{100} = 105.26$ (20.09). These discrepancies can be attributed to rounding error – the function `gpdfit` uses the full values for $\hat{\sigma}$ and $\hat{\xi}$, for example, from `a$est[1]` and `a$est[2]` (respectively), instead of $\hat{\sigma} = 7.44$ and $\hat{\xi} = 0.18$ to two decimal places. Similarly, the full values of the estimated standard errors of the GPD parameters (and λ_u) are used instead of 0.96 and 0.10 (and 0.0007 for $\hat{\lambda}_u$), respectively.

4.2.5 Profile likelihood

As previously discussed in Section 3.2.4, using the estimated standard error to produce a symmetric confidence interval for a parameter can sometimes be misleading, particularly when the parameter of interest is a return level; we often observe severe positive skew in the likelihood surface when viewed from the q_r–axis, and so using the profile likelihood for q_r usually offers a more appropriate method for obtaining confidence intervals. We again demonstrate the calculation of a 95% profile confidence interval for q_r in **R**, but this time using the threshold exceedance model.

Suppose our interest lies in the 100–year return level q_{100}. Firstly, we re–parameterise the GPD so that q_{100} is one of the parameters. We can do this by rearranging equation (16) to make σ the subject, which gives

$$\sigma = \frac{\xi(q_{100} - u)}{(100 n_y \lambda_u)^{\xi} - 1}. \tag{19}$$

We then replace σ with (19) in the expression for the log–likelihood (18) to obtain $\ell(q_{100}, \xi)$. Fixing $q_{100} = q_{100,0}$ and maximising $\ell(q_{100}, \xi)$, for a range of values for $q_{100,0}$, gives the profile log–likelihood for q_{100}. For example, we know from the previous section that $\hat{q}_{100} = 106.34$; thus in **R**, we define the vector `q100.0` to take values between (arbitrarily) 80 and 190:

```
> q100.0<-seq(80,190,0.01)
```

We then define the profile log–likelihood function as:

```
> theta2<-sd(dataset)
> prof.loglik<-function(theta2)
  {
    xi<-theta2
    sigma<-xi*(q100-30)/((100*365.25*0.008670355)^(xi)-1)
    m<-min(1+(dataset*(xi/sigma)))
    if(m<0.0001)return(as.double(1000000))
    if(sigma<0.0001)return(as.double(1000000))
    if(xi==0)
      {
        loglik<-length(dataset)*log(1/sigma)-(1/sigma)*(sum(dataset))
      }
    else{
       loglik<--length(dataset)*log(sigma)
         -sum(log(1+(dataset*(xi/sigma)))*(1/xi+1))
    }
      loglik
      return(-loglik)
    }
```

As before, care has been taken to replace the GPD log–likelihood with the exponential limit when $\xi = 0$. Notice the parameter vector (called `theta2` here) now only has one element – ξ; this is because we are going to minimise the negative log–likelihood (and so maximse the log–likelihood itself) with respect to ξ whilst keeping q_{100} fixed at $q_{100,0}$ (and hence keeping σ fixed via equation (19)). We do this for each value in `q100.0`, and then plot the maximised likelihood value against the corresponding `q100.0` value. The following code:

```
> prof.plot<-vector('numeric',length(q100.0))
```

creates a vector in which we will store the maximised log–likelihood value for each value of $q_{100,0}$. Then the following code

```
> for(i in 1:length(prof.plot))
    {
     q100<-q100.0[i]
     a<-nlm(prof.loglik,theta2)
     prof.plot[i]<--(a$minimum)
    }
> plot(prof.plot~q100.0), main='Profile log--likelihood for q100',xlab='q100',
  ylab='Profile log-likelihood',type='l')
```

produces the plot shown in figure 12. The code

```
> abline(h=-485.0937)
> abline(v=106.34)
```

superimposes a horizontal line at the maximised log–likelihood (see the output in Section 4.2.2) and a vertical line at the MLE for q_{100}. We can then apply the result from Section 3.2.4 to superimpose a horizontal broken line giving a cut–off point equal to $\frac{1}{2} \times \chi_1^2(0.05)$, from which we can deduce the 95% profile confidence interval for q_{100}:

```
> abline(h=-485.0937-0.5*qchisq(0.95,1),lty=2)
```

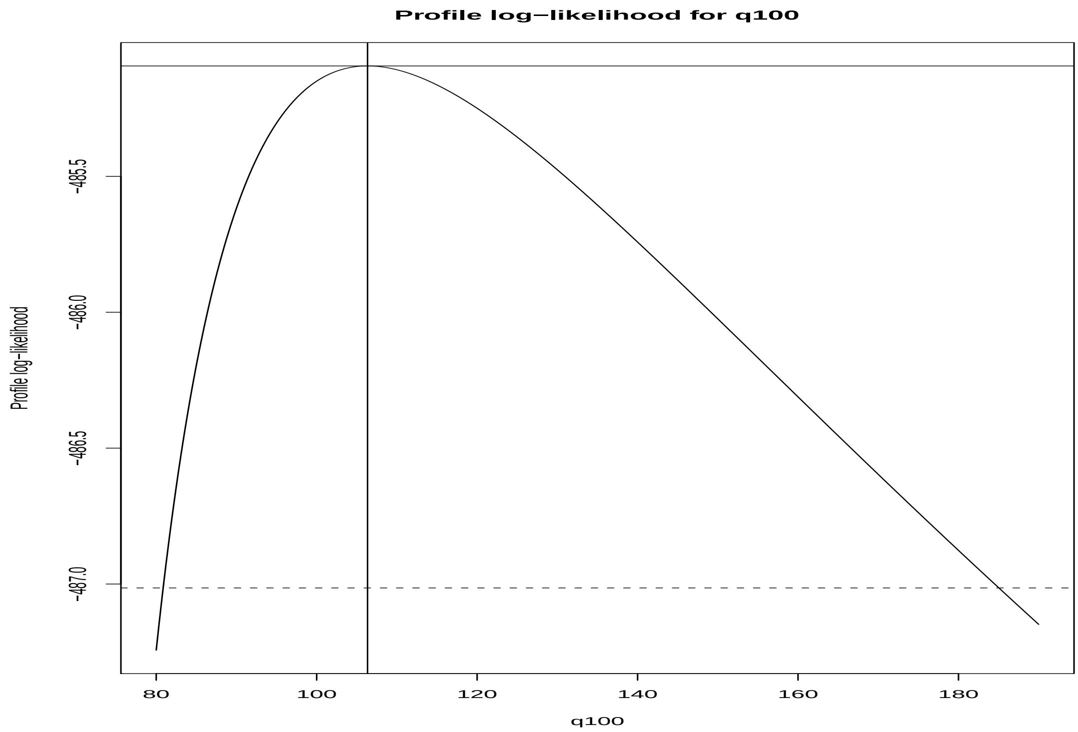

Figure 12: Profile log–likelihood for q_{100} in the rainfall example, using the threshold excess model

The horizontal dotted line intersects with the profile log–likelihood for q_{100} at 80.86 and 185.03, giving the 95% confidence interval (80.86, 185.03) millimetres. A symmetric confidence interval using the MLE and estimated standard error obtained via the delta method is $106.343 \pm 1.96 \times 20.8576$, i.e. (65.46, 147.22) millimetres; given the positive skew observed in the plot in Figure 12, the upper bound provided by the symmetric confidence interval is far too low. Constructing a sea wall, for example, to with a height as specified by the upper endpoint of the standard 95% confidence interval could provide substantial under–protection.

4.2.6 Model diagnostics

Recall from Section 3.2.5 that a probability plot and a quantile–quantile plot can be used to check the goodness–of–fit of our chosen model to the data – in this case, the GPD as a model for threshold exceedances. Recall that a probability plot is a plot of the points

$$\left\{ \left(\hat{F}\left(x_{(i)}\right), \frac{i}{n+1} \right) : i = 1, \ldots, n \right\},$$

and a quantile–quantile plot is a plot shows the pairs of points:

$$\left\{\left(\hat{F}^{-1}\left(\frac{i}{n+1}\right), x_{(i)}\right) : i = 1, \ldots, n\right\}.$$

If our threshold exceedances are stored in the vector `dataset`, then

```
> ordered<-sort(dataset)
```

stores the ordered threshold exceedances $x_{(i)}$, $i = 1, \ldots, 152$ in the vector `ordered`. For each point in this ordered sample, the following then stores the empirical distribution function defined by $i/(n+1)$ in the vector `empirical`:

```
> empirical<-vector('numeric',length(ordered))
> for (i in 1:length(empirical))
          {
           empirical[i]<-i/(length(dataset)+1)
          }
```

The function `GPD.DF` defines the distribution function for the GPD, as given by equation (15); recall that, for the case $\xi = 0$, we have the exponential distribution with rate $1/\sigma$).

```
> GPD.DF<-function(data,sigma,xi)
  {
    if(xi==0)
      {
        GPD<-1-exp(-data/sigma)
      }
    else
      {
        GPD<-1-(1+(xi*data)/sigma)^(-1/xi)
      }
    GPD
   }
```

Then, as we did for the GEV in Section 3.2.5, we store a model–based estimate of the distribution function, evaluated at each point in the ordered sample `ordered`, in `model`:

```
> model<-vector('numeric',length(dataset))
> for(i in 1:length(model))
  {
    model[i]<-GPD.DF(ordered[i],a$est[1],a$est[2])
  }
```

Then plotting `model` against `empirical` produces the probability plot for the set of rainfall threshold exceedances; this is shown in Figure 13, with the line of equality superimposed.

For the quantile plot:

```
> model.quantile<-vector('numeric',length(dataset))
> GPD.INV<-function(data,sigma,xi)
  {
```

```
    if(xi==0)
      {
        INV<--sigma*log(1-data)
      }
    else
      {
        INV<-(sigma/xi)*((1-data)^(-xi)-1)
      }
    INV
  }
> for(i in 1:length(model.quantile))
    {
      model.quantile[i]<-GPD.INV(empirical[i],a$est[1],a$est[2])
    }
```

The function `GDP.INV` computes the inverse of the GPD distribution function at each of the points in the vector `data`; we evaluate this for each value in `empirical` to find $\hat{F}(i/(n+1))$ and store the results in the vector `model.quantile`. Plotting `model.quantile` against `ordered` produces the quantile plot shown in Figure 14, again with the line of equality superimposed to visually assess the goodness–of–fit. As can be seen from both figures 13 and 14, the GPD seems to provide a more than adequate fit to the set of rainfall threshold exceedances, with most points lying on or close to the line of equality; however, as in the GEV analysis of annual maxima, we notice a potential problem at high levels of the rainfall process detected by the quantile plot.

4.2.7 Using the `ismev` package

As in Chapter 3, we now illustrate fitting the GPD to the set of threshold exceedances using the functions provided by the `ismev` package (see Section 3.2.6). Recall that, provided you have a version of **R** no older than **R** 1.50, you can install the `ismev` packages by typing:

```
> library(ismev)
```

The function `gpd.fit` fits the GPD to a set of threshold exceedances; however, no data pre–processing is necessary. You simply supply the function with the full data set, and your threshold. We know from the previous Section that a threshold of about 30mm seemed adequate for the rainfall data, and we used a mean residual life plot to find this value. The `ismev` function `mrl.plot` produces the plot that we constructed from first principles in Section 4.2.1 (see Figure 10). Typing

```
> mrl.plot(rain)
```

produces the plot shown in figure 15. Notice that this plot is the same as that produced earlier, but also has the corresponding 95% bounds superimposed.

As before, we see that this plot appears linear above a value of about 30mm (though these plots can be open to interpretation). Remember that, since there is a limited amount of data available above very high thresholds, variability is high in the right–hand–side of these plots). We can now fit the GPD to threshold exceedances above u_0 = 30mm by typing

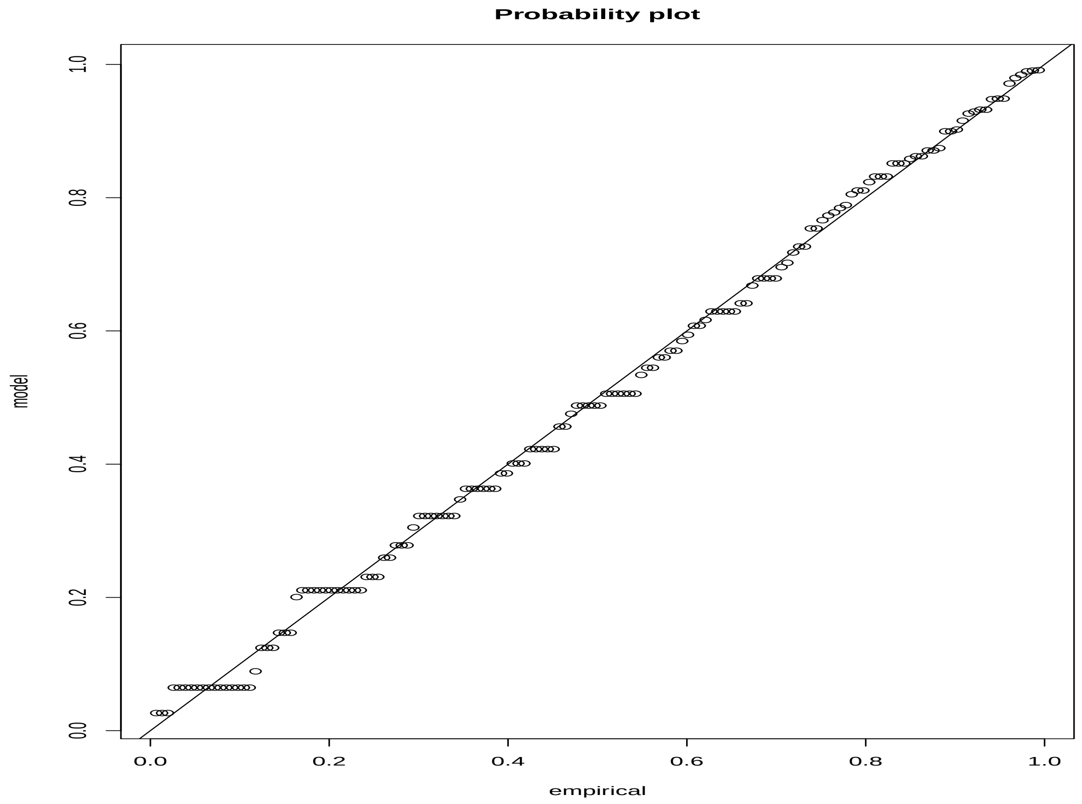

Figure 13: Probability plot for the GPD in the rainfall threshold exceedances example

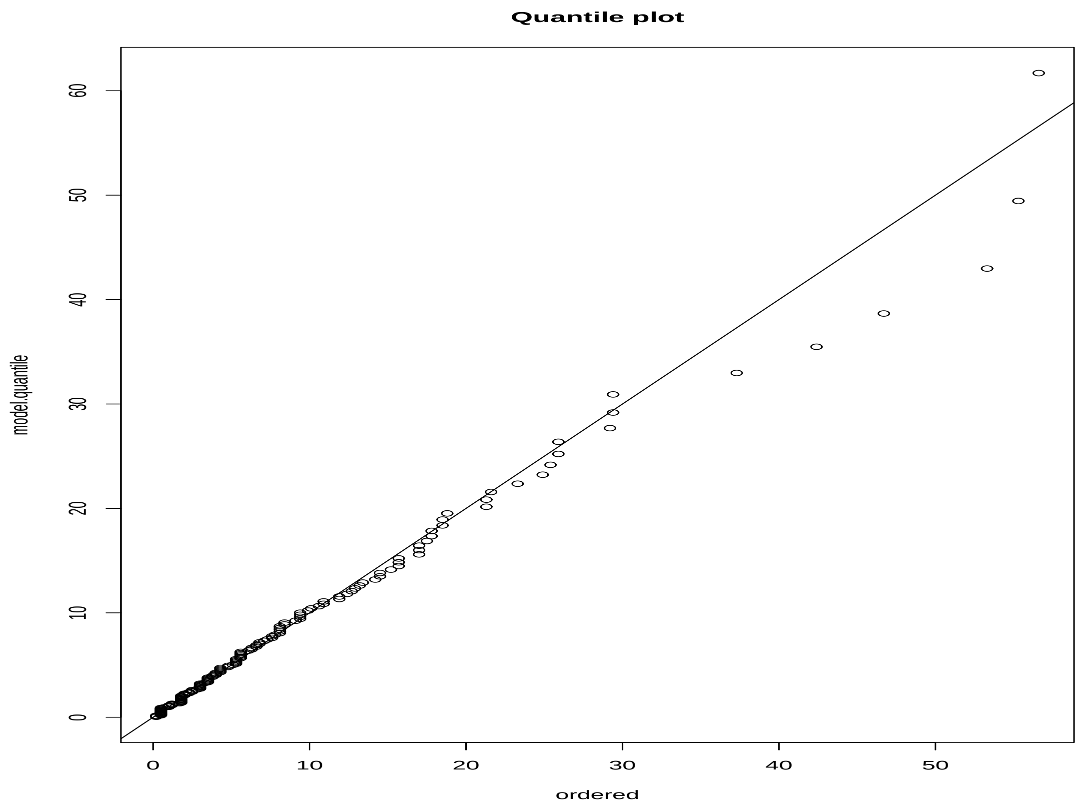

Figure 14: Quantile plot for the GPD in the rainfall threshold exceedances example

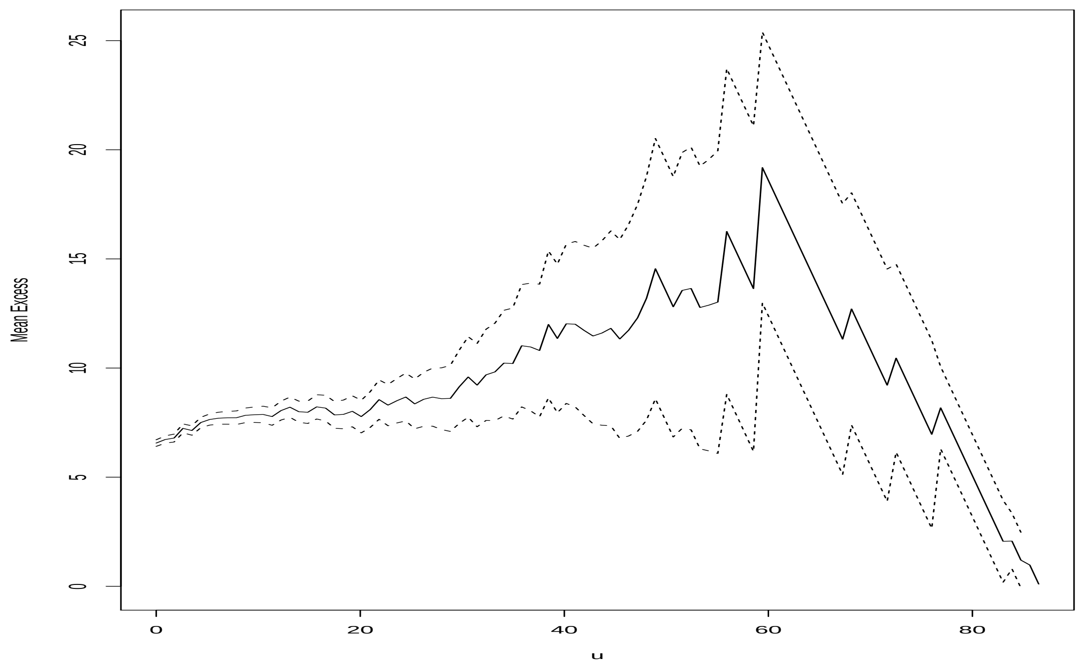

Figure 15: Mean residual life plot for the rainfall data produced suing the `mrl.plot` function in `ismev`

```
gpd.fit(rain,30)
```

which gives the following output:

```
$threshold
[1] 30

$nexc
[1] 152

$conv
[1] 0

$nllh
[1] 485.0937

$mle
[1] 7.4406505 0.1843329

$rate
[1] 0.008670355

$se
[1] 0.958432 0.101151
```

As before, the MLEs for σ and ξ (provided by `$mle`) are 7.44 and 0.18, respectively. The standard errors are stored in `$se` and are 0.96 and 0.18 for σ and ξ (again, as we found before). The **R** output also gives an estimate of the threshold exceedance rate $\hat{\lambda}_u = 0.0087$. `$nexc` gives the number of exceedances above the specified threshold (152 values exceed the threshold 30), and `$nllh` gives the value of the negative log–likelihood. As with the `gev.fit` function used in Chapter 3, `$conv` is a convergence code for the minimisation routine; a zero (as shown here) indicates successful convergence. A more detailed explanation of this output can be obtained by typing `help(gpd.fit)`. Just as the functions `gev.profxi` and `gev.prof` produced profile log–likelihood plots for the shape parameter ξ and a specified return level (respectively), the function `gpd.profxi` and `gpd.prof` can be used to produce profile log–likelihood curves for ξ and a specified return level using the threshold excess model. For example, if we store the output of the fitting routine `gpd.fit` in the object `a`, i.e.:

```
> a<-gpd.fit(rain,30)
```

then the following code produces a profile log–likelihood plot for ξ between the values –0.1 and 0.7 (the resulting plot is shown in figure 16); `conf=0.95` gives the cut–off point that can be used to construct a 95% confidence interval for ξ, though of course this can be changed to provide cut–off points for other confidence intervals. From figure 16, we can read off a 95% confidence interval for ξ as $(0.007, 0.416)$mm.

```
> gpd.profxi(a,-0.1,0.7, conf=0.95)
```

Figure 12, which shows the profile log–likelihood for the 100–year return level using the threshold exceedance model, can be replicated by typing:

```
> gpd.prof(a,100,80,190,conf=0.95)
```

where `100` is the return period, and `80,190` provides the plotting range for the profile log–likelihood. This plot can be seen in figure 17, and, of course, gives rise to the same confidence interval for q_{100} as we obtained in Section 4.2.5: (80.86, 185.03)mm. Confidence intervals for other return levels can be obtained in the same way, by changing the return period in the `gpd.prof` function, though the plotting range will also have to be changed accordingly.

To check the goodness–of–fit of the GPD for our threshold exceedance above 30mm, we can construct probability plots and quantile plots using the function `gpd.diag`. For example,

```
> gpd.diag(a)
```

where the object `a` stores the results from the fitting routine `gpd.fit`, gives the diagnostic plots shown in figure 18. As before with the `gev.diag` function in Section 3.2.6, we get a 2×2 panel of plots: the top row replicates the probability plot and quantile plot we produced from first principles in Section 4.2.6 (figures 13 and 14 respectively), whilst the bottom row shows the return level plot and a plot of the empirical density with the GPD density overlaid. All four plots seem to indicate that the GPD provides quite a good fit to excesses above a threshold of 30mm.

We can also use the `gpd.fitrange` function in `ismev` to reproduce the "parameter stability plots" shown in figure 11. Recall that these plots can aid in the threshold selection process: once a candidate threshold u_0 has been identified in the mean residual life plot (figures 10

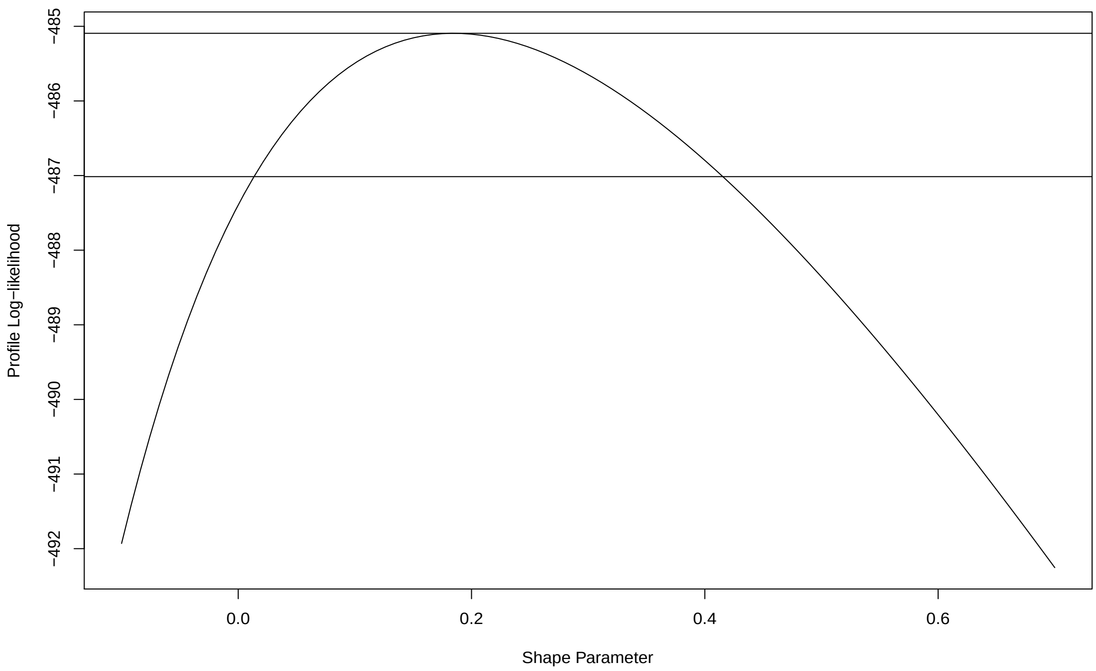

Figure 16: Profile log–likelihood for ξ in the rainfall example, constructed using the `gpd.profxi` function in `ismev`

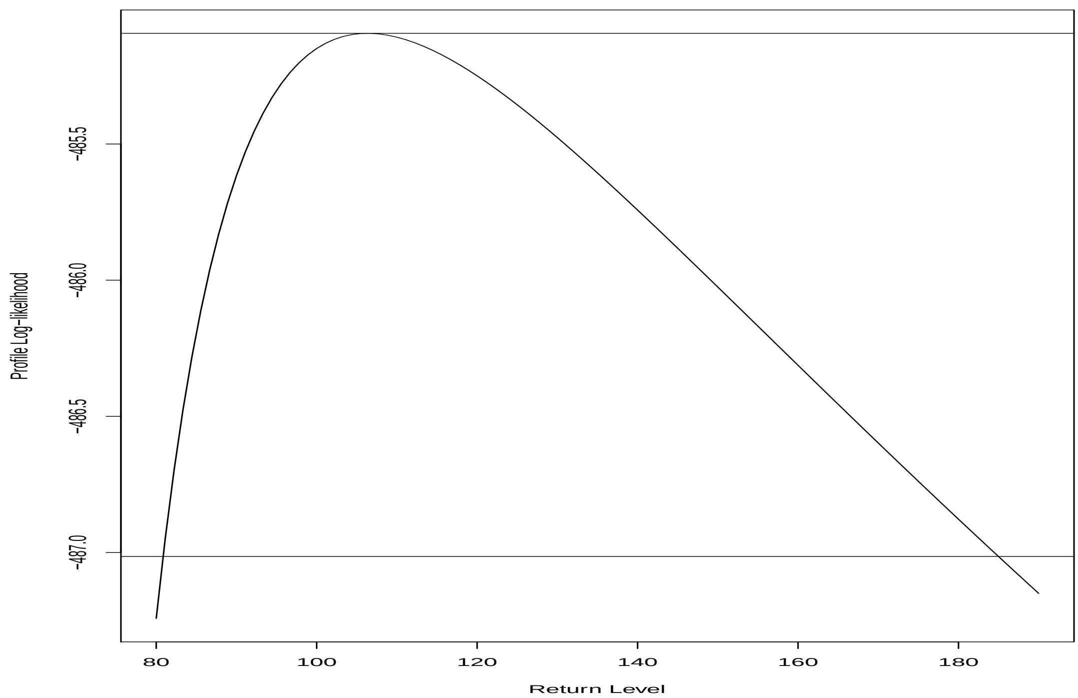

Figure 17: Profile log–likelihood for q_{100} in the rainfall example, constructed using the `gpd.prof` function in `ismev`

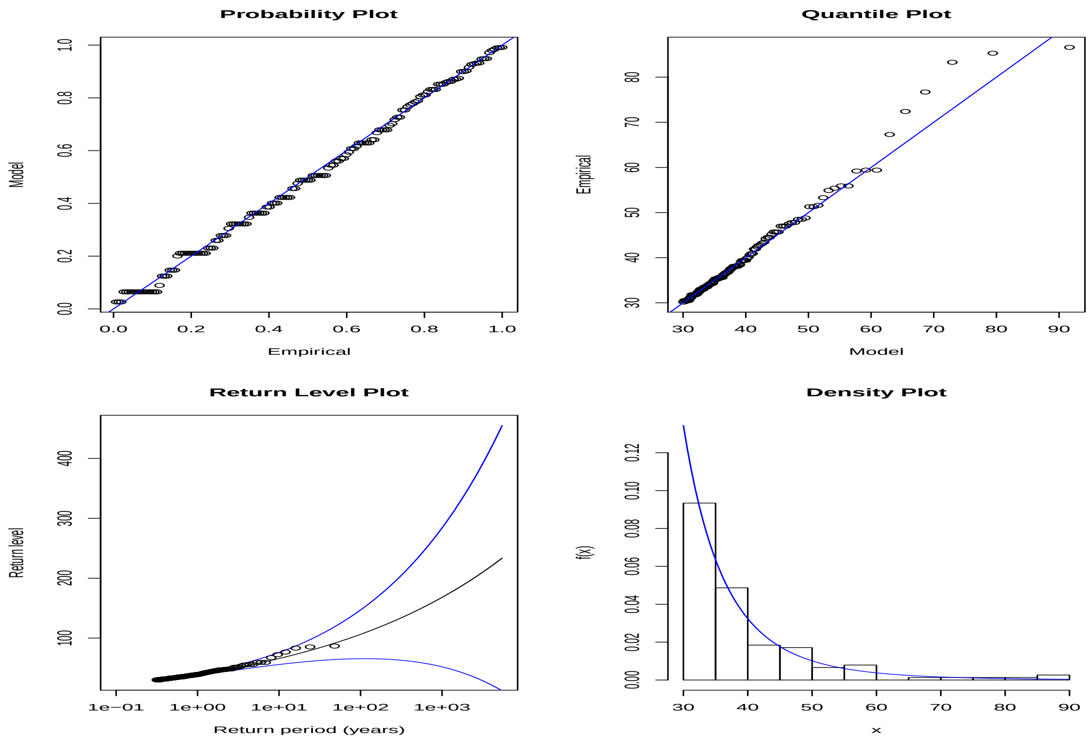

Figure 18: Diagnostic plots created using the function `gpd.diag` to check the goodness–of–fit of the GPD model in the rainfall example

and 15), then, owing to the threshold stability property of the GPD, we should observe stability in estimates of σ^* and ξ for all thresholds $u > u_0$ if the GPD is appropriate for exceedances over u_0. In **R**, typing

```
> gpd.fitrange(rain,umin=30,umax=50)
```

reproduces plots similar to those shown in figure 11, again with the associated c95% confidence bands. The default procedure fits the GPD at 10 equally–spaced points between u = `umin` and u = `umax`; the optional argument `nint` can be used to change this. For example, the code

```
> gpd.fitrange(rain,umin=30,umax=50,nint=50)
```

fits the GPD at 50 equally–spaced values of u between 30 and 50 (inclusive); this plot is shown in figure 19.

4.3 Practical considerations: temporal dependence

The asymptotic results for the GEV introduced in Chapter 3, and for the GPD in the current Chapter, have assumed the underlying processes independent and identically distributed (IID). They also assume this process is stationary. In practice, extreme value data – particularly environmental time series – exhibit some form of departure from this ideal. The most common forms are:

— Local temporal dependence, where successive values of the time series are dependent, but values farther apart are independent (to a good approximation);

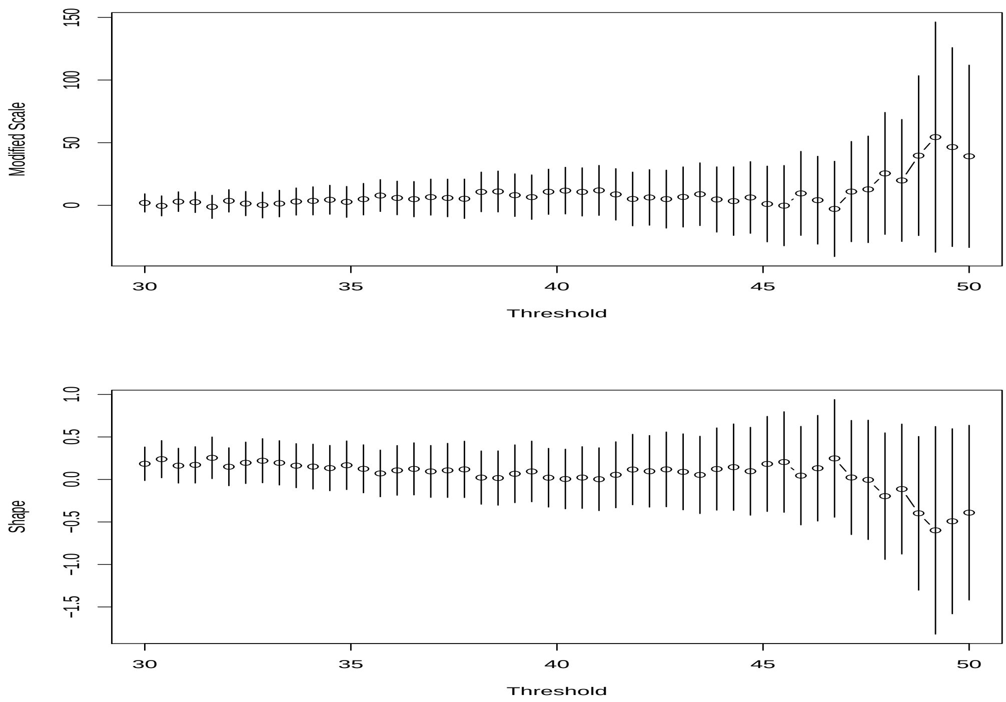

Figure 19: Parameter stability plots for σ^* and ξ for the rainfall data, constructed using the `gpd.fitrange` function in `ismev`

— Long term trends, where the underlying distribution changes gradually over time;

— Seasonal variation, where the underlying distribution changes periodically through time.

These departures can be handled through a combination of extending both the theory and the modelling. However, although a wide range of theoretical models for non–stationarity have been studied, only in a few cases have these been used for statistical modelling; the results have generally been too specific to be of use in modelling data for which the form of non–stationarity is unknown. Over the last decade or so, it has been more usual for practitioners to employ statistical procedures which allow the existing results to be applied. In Chapter 5 we will consider some of these in detail. For now, we will focus our attention on procedures which deal with short term temporal dependence.

For the types of data to which extreme value models are commonly applied, temporal independence is usually an unrealistic assumption. In particular, extreme conditions often persist over several consecutive observations, bringing into question the appropriateness of models such as the GEV. A detailed investigation of this requires mathematical treatment at a level of sophistication beyond the scope of this book; however, the general ideas are not difficult and the main result offers a simple, practical, interpretation. For the remainder of this Section on dependent sequences, we shall assume that our process is *stationary*, corresponding to a series whose variables may be mutually dependent, but whose stochastic properties are homogeneous throughout time.

Dependence in stationary sequences can take many different forms. With practical applications in mind, it is common to assume a condition that limits the extent of dependence to short–range temporal dependence so that, for example, events X_i and X_j, both of which are extreme, are independent provided time points i and j are far enough apart. Indeed, many stationary sequences satisfy this property. By excluding the possibility of long–range dependence in this way, we focus our attention on dependence at a much shorter range. Effects of such short–range dependence, it turns out, can be quantified within the standard extreme value limits discussed in Chapter 1 for the GEV and the current Chapter for the GPD.

4.3.1 Maxima of stationary sequences

The book by Leadbetter *et al.* (1983) considers, in great detail, properties of extremes of dependent processes. A key result often used is 'Leadbetter's $D(u_n)$ condition', which ensures that long–range dependence is sufficiently weak so as not to affect the asymptotics of an extreme value analysis. This condition is stated more formally in the Definition below.

Definition (Leadbetter's $D(u_n)$ condition)
A stationary series $X_1, X_2, \ldots$ is said to satisfy the $D(u_n)$ condition if, for all $i_1 < \ldots < i_p < j_1 < \ldots < j_q$ with $j_1 - i_p > l$,

$$\begin{aligned} \Big| &\Pr\left\{X_{i_1} \le u_n, \ldots, X_{i_p} \le u_n, X_{j_1} \le u_n, \ldots, X_{j_q} \le u_n\right\} \\ &-\Pr\left\{X_{i_1} \le u_n, \ldots, X_{i_p} \le u_n\right\} \Pr\left\{X_{j_1} \le u_n, \ldots, X_{j_q} \le u_n\right\} \Big| \le \alpha(n,l), \end{aligned} \tag{20}$$

where $\alpha(n,l) \to 0$ for some sequence l_n such that $l_n/n \to 0$ as $n \to \infty$.

For sequences of independent variables, the difference in probabilities in the above expression is exactly zero for *any* sequence u_n. More generally, we will require that the $D(u_n)$ condition holds only for a specific sequence of thresholds u_n that increases with n. For such a sequence, the $D(u_n)$ condition ensures that, for sets of variables that are far enough apart, the difference in probabilities expressed in (20), while not zero, is sufficiently close to zero to have no effect on the limit laws for extremes.

Theorem
Let $\tilde{X}_1, \tilde{X}_2, \ldots$ be a stationary series satisfying Leadbetter's $D(u_n)$ condition, and let $\tilde{M}_n = \max\{\tilde{X}_1, \ldots, \tilde{X}_n\}$. Now let $X_1, X_2, \ldots$ be an *independent* series with X having the same distribution as $\tilde{X}$, and let $M_n = \max\{X_1, \ldots, X_n\}$. Then if M_n has a non–degenerate limit law given by $\Pr\{(M_n - b_n)/a_n \le x\} \to G(x)$, it follows that

$$\Pr\left\{(\tilde{M}_n - b_n)/a_n \le x\right\} \to G^\theta(x) \tag{21}$$

for some $0 \le \theta \le 1$.

The parameter θ is known as the *extremal index*, and quantifies the extent of extremal dependence: $\theta = 1$ for a completely independent process, and $\theta \to 0$ with increasing levels of (extremal) dependence. Since G in the above theorem is necessarily an extreme value distribution, and due to the *max–stability* property (see Leadbetter *et al.*, 1983), then the distribution of maxima in processes displaying short–range temporal dependence (characterised by the extremal index θ) is also a GEV distribution; the powering of the limit distribution by θ only affects the location and scale parameters of this distribution.

The above theorem implies that if maxima of a stationary series converge – which, from Chapter 3, we know they will do – then, provided an appropriate $D(u_n)$ condition is satisfied, the limit distribution is related to the limit distribution of an independent series. The effect of dependence, as seen in expression (21), is just a replacement of G as the limit distribution with G^θ. In fact, if G corresponds to the GEV distribution with parameters (μ, σ, ξ), then

$$\begin{aligned} G^\theta(z) &= \exp\left\{-\left[1+\xi\left(\frac{z-\mu}{\sigma}\right)\right]^{-1/\xi}\right\}^\theta \\ &= \exp\left\{-\left[1+\xi\left(\frac{z-\mu^*}{\sigma^*}\right)\right]^{-1/\xi}\right\}, \end{aligned}$$

where $\mu^* = \mu - \frac{\sigma}{\xi}\left(1-\theta^{-\xi}\right)$ and $\sigma^* = \sigma\theta^\xi$. Thus, if the (approximate) distribution of M_n is GEV with parameters (μ, σ, ξ), then the (approximate) distribution of $\tilde{M}_n$ is GEV with parameters (μ^*, σ^*, ξ).

4.3.2 Modelling block maxima

Provided long–range dependence is weak, we can proceed to model block maxima from series with short–range extremal dependence as outlined in Chapter 3, since the distribution of block maxima falls within the same family of distributions as would be appropriate if the series were truly independent. This is fantastic news! Short–range temporal dependence is a much more plausible assumption than complete independence, and our modelling approach is still valid! However, the main difference – excluding the change in parameters from (μ, σ, ξ) to (μ^*, σ^*, ξ) – is that our implied n (the number we are taking the maxima over) is now effectively reduced due to the dependence, so convergence of maxima to the limit distribution will be slower. And shouldn't we be using threshold methods anyway, which use information on *all* extremes and not just those that are the maximum within their block?

4.3.3 Modelling threshold exceedances

Though the modelling procedure for fitting the GEV to a set of annual maxima is unchanged for series which display short–term temporal dependence, some revision is needed of the threshold exceedance approach. If all threshold exceedances are used in our analysis, and the GPD fitted to the set of threshold excesses, the likelihoods we use will be incorrect since they assume independence of sample observations. In practice, several techniques have been developed to circumvent this problem, including:

1. filtering out an (approximately) independent set of threshold exceedances

2. fitting the GPD to *all* exceedances, ignoring dependence, but then appropriately adjusting the inference to take into account the reduction in information

3. Explicitly modelling the temporal dependence in the process

Though the first approach above is by far the most widely–used, research has recently focussed on the relative merits of the other two approaches. The third approach makes use of multivariate extreme value theory, and so we shall re–visit this idea in more detail in Chapter 6. For now, let us consider the first two approaches, which we will call *removing* dependence and *ignoring* dependence, respectively.

Removing dependence between threshold exceedances: declustering
Figure 20 shows a series of 3–hourly measurements of sea–surge heights at Newlyn, a coastal town in the southwest of England, collected over a three year period. The sea–surge is the meteorologically induced non–tidal component of the still–water level of the sea. The practical motivation for the study of such data is that structural failure — probably a sea–wall in this case — is likely under the condition of extreme surges. Also shown in Figure 20 is a plot of the time series against the lag 1 time series. These data are available via the `ismev` package by typing:

```
> data(wavesurge)
```

Typing `wavesurge[1:10,]` shows the first ten rows of this dataset:

```
> wavesurge[1:10,]
> wavesurge[1:10,]
   wave  surge
1  1.50 -0.009
2  1.83 -0.053
3  2.44 -0.024
4  1.68  0.000
5  1.49  0.079
6  1.20  0.068
7  1.35 -0.009
8  1.15 -0.003
9  1.20  0.011
10 1.07  0.024
```

The data represented in figure 20 has been constructed using the second column of this dataset; we will come back to the `wave` heights in Chapter 6. To isolate the surge measurements in order to reproduce the graphs shown in figure 20, type:

```
> surge<-wavesurge[,2]
```

the plots in figure 20 can then be re–produced using the **R** commands `plot(ts...)` for the time series plot, `hist` for the histogram and `plot` for the scatter plot of each value against the next.

A natural way of modelling extremes of such time series is to use the Generalised Pareto Distribution (GPD) as a model for excesses over a high threshold. Figure 20 shows the presence of substantial temporal dependence in the sequence of three–hourly surges. The most commonly adopted approach to circumvent the problems caused by such temporal dependence is to employ a declustering scheme to filter out a set of approximately

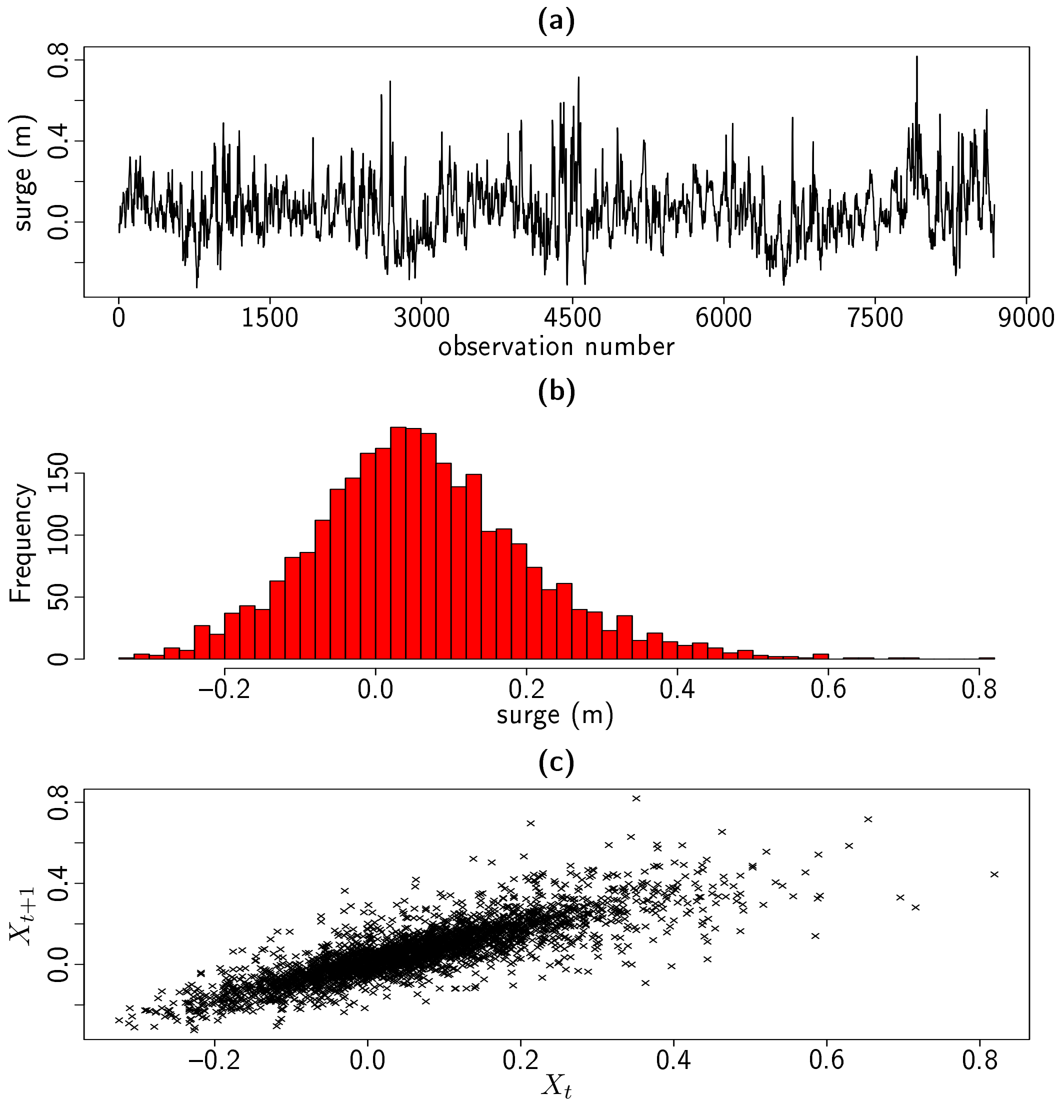

Figure 20: Newlyn sea–surge data: (a) Time series plot; (b) histogram; (c) plot of the time series against the series at lag 1.

independent threshold excesses. One method, which is often considered to be the most 'natural' way of identifying 'clusters' of extremes, is *runs declustering.* This is how it works:

1. Choose an auxiliary 'declustering parameter' (which we call κ)
2. A cluster of threshold excesses is then deemed to have terminated as soon as at least κ consecutive observations fall below the threshold
3. Go through the entire series identifying clusters in this way
4. The maximum (or 'peak') observation from each cluster is then extracted, and the GPD fitted to the set of cluster peak excesses.

This approach is often referred to as the *peaks over threshold* approach (POT, Davison and Smith, 1990) and is widely accepted as the main pragmatic approach for dealing with clustered extremes. Although this approach is quite easy to implement, there are issues surrounding the choice of κ; if

- κ is too small, the cluster peaks will not be far enough apart to safely assume independence
- κ is too large, there will be too few cluster exceedances on which to form our inference

It has also been shown that parameter estimates can be sensitive to the choice of κ. In this example, we use a separation interval of 60 hours (and so $\kappa = 20$) following the example of Coles and Tawn (1991), which should be large enough to safely assume independence between successively identified clusters allowing for wave propagation time. We used a mean residual life plot (see Section 4.2.1) to identify a suitably high threshold (0.3m). The following **R** code can be used to implement a runs declustering scheme with $\kappa = 20$ observations:

```
> cluster20<-function(data,threshold)
   {
     x<-list()
     z<-list()
     j<-1
{
for(i in (21):length(data))
         {
  if(data[i-20]>threshold & data[i-19]<=threshold
  & data[i-18]<=threshold & data[i-17]<=threshold
  & data[i-16]<=threshold & data[i-15]<=threshold
  & data[i-14]<=threshold & data[i-13]<=threshold
  & data[i-12]<=threshold & data[i-11]<=threshold
  & data[i-10]<=threshold & data[i-9]<=threshold
  & data[i-8]<=threshold & data[i-7]<=threshold
  & data[i-6]<=threshold & data[i-5]<=threshold
  & data[i-4]<=threshold & data[i-3]<=threshold
  & data[i-2]<=threshold & data[i-1]<=threshold
```

```
  & data[i]<=threshold)
              {
       x<-max(data[j:i])
               ifelse(i !=length(data), j<-i+1, NA)
               z<-c(z,x)
              }
         }
       }
    z
  }
```

Notice that this function starts work at observation 21 in the series `data`. It then looks back over the last 20 observations; if the first in this group of 20 exceeded the threshold, but the following 20 (present value included) lie sub–threshold, then a cluster of threshold exceedances is deemed to have terminated. The maximum value between counter `j` and texttti then goes into the list `x`. Notice that this *must* be the cluster maximum, since all values between `i-19` and `i` are at least less than `i-20` since they are below the threshold, and so the routine will scan back and select the maximum value in the cluster of exceedances before observation `i-19`. Implementing this gives:

```
> as.numeric(cluster20(surge,0.3))
 [1] 0.326 0.390 0.490 0.451 0.328 0.417 0.326 0.365 0.339 0.302 0.629 0.696
[13] 0.323 0.445 0.376 0.307 0.438 0.503 0.716 0.363 0.465 0.405 0.487 0.323
[25] 0.517 0.397 0.819 0.533 0.444 0.458 0.556
```

We need to use the command `as.numeric` since the function `cluster20` returns the set of cluster maxima in a list; `as.numeric` coerces this list into a vector of usable numbers. Thus, we have 31 cluster maxima to work with. We can then fit to our set of cluster peak excesses by typing:

```
> cluster.peaks<-as.numeric(cluster20(surge,0.3))
> gpd.fit(cluster.peaks,0.3)
$threshold
[1] 0.3

$nexc
[1] 31

$conv
[1] 0

$nllh
[1] -28.53091

$mle
[1]  0.1845773 -0.2307157

$rate
```

```
[1] 1

$se
[1] 0.04607956 0.17830126
```

This gives $\hat{\sigma} = 0.185$ (0.046) and $\hat{\xi} = -0.231$ (0.178), where estimated standard errors are shown in parentheses.

4.4 Case studies