Memo on Financial risk modelling and portfolio optimization with R

Chapter 7: Extreme value theory

Block maxima model for Siemens

`siemens` is Daily Log Returns on Siemens Share Price. These data are the daily log returns on Siemens share price from Tuesday 2nd January 1973 until Tuesday 23rd July 1996. The data are contained in a numeric vector. The dates of each observation are contained in a times attribute, which is an object of class "POSIXct" (see DateTimeClasses). Note that these data form an irregular time series because no trading takes place at the weekend. ###Data

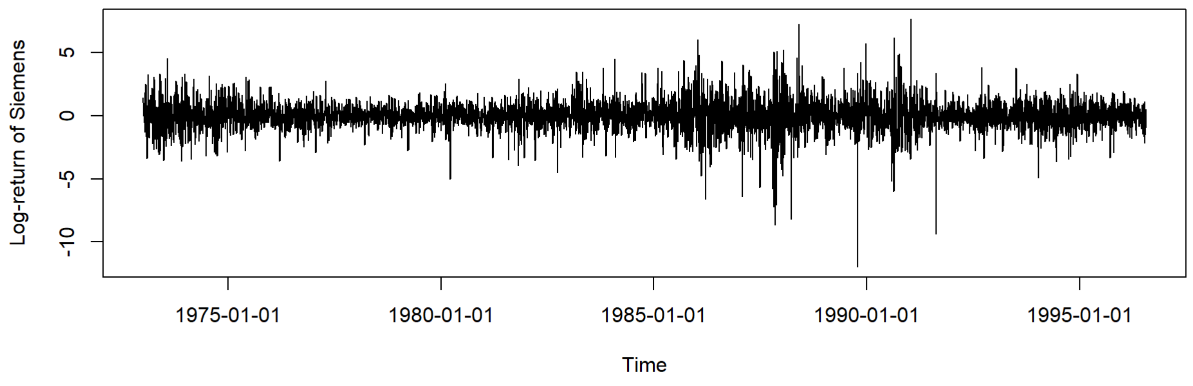
```
library(evir)
data(siemens)
head(siemens)
```

```
## [1]  0.014347448  0.010861972  0.007020857  0.001863933  0.000000000
## [6] -0.001397624
```

```
library(timeSeries)
```

```
## Loading required package: timeDate
```

```
SieDates <- as.character(format(as.POSIXct(attr(siemens, "times")), "%Y-%m-%d"))
SieRet <- timeSeries(siemens * 100, charvec = SieDates)
plot(SieRet, ylab="Log-return of Siemens")
```



```
png(filename="7_1_SiemensLogRet.png", width = 800, height = 400)
plot(SieRet, ylab="Log-return of Siemens")
dev.off()
```

```
## png
##   2
```

Losses

First, the daily returns are converted to positive loss figures expressed as percentages.

```
SieLoss <- -100.0 * siemens
head(SieLoss)
```

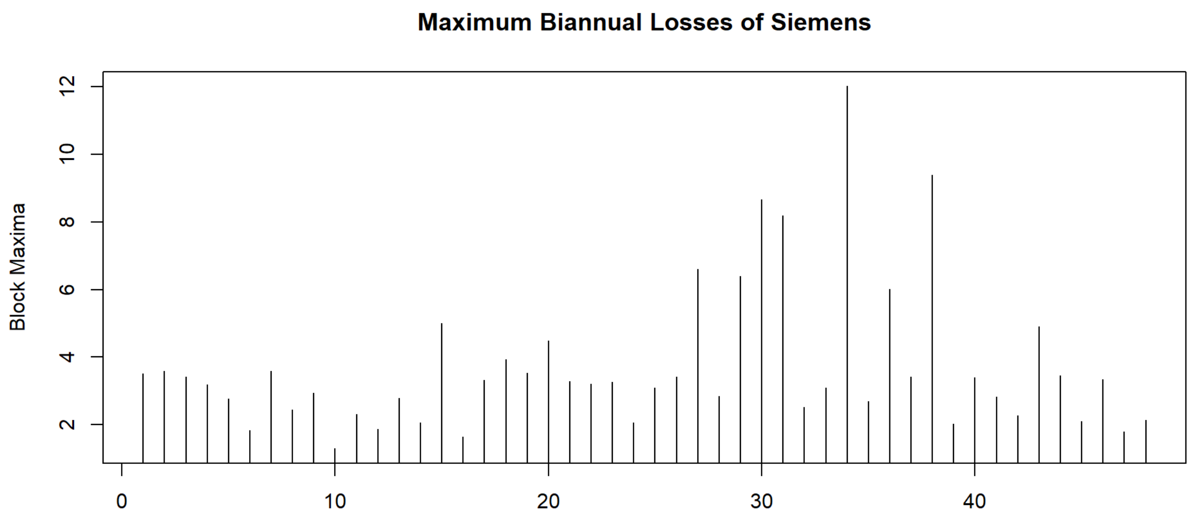
```
## [1] -1.4347448 -1.0861972 -0.7020857 -0.1863933  0.0000000  0.1397624
```

Fitting GEV to block maxima of Siemens

The generalized extreme value (GEV) distribution is a family of continuous probability distributions developed within extreme value theory to combine the Gumbel, Fréchet and Weibull families.

```
## package evir:
library(evir)
SieGEV <- gev(SieLoss, block = "semester")
```

```
plot(SieGEV$data, type = "h", xlab = "", pch = 19,
     ylab = "Block Maxima", main = "Maximum Biannual Losses of Siemens")
```



```
png(filename="7_2_BlockMaxima.png", width = 800, height = 400)
plot(SieGEV$data, type = "h", xlab = "", pch = 19,
     ylab = "Block Maxima", main = "Maximum Biannual Losses of Siemens")
dev.off()
```

```
## png
##   2
```

```
options(digits = 4)
## Estimates
SieGEV$par.ests
```

```
##     xi  sigma     mu
## 0.2867 1.0467 2.6997
```

```
## Standard error
SieGEV$par.ses
```

```
##     xi  sigma     mu
## 0.1171 0.1413 0.1698
```

```
## package ismev:
library(ismev)
```

```
## Loading required package: mgcv
```

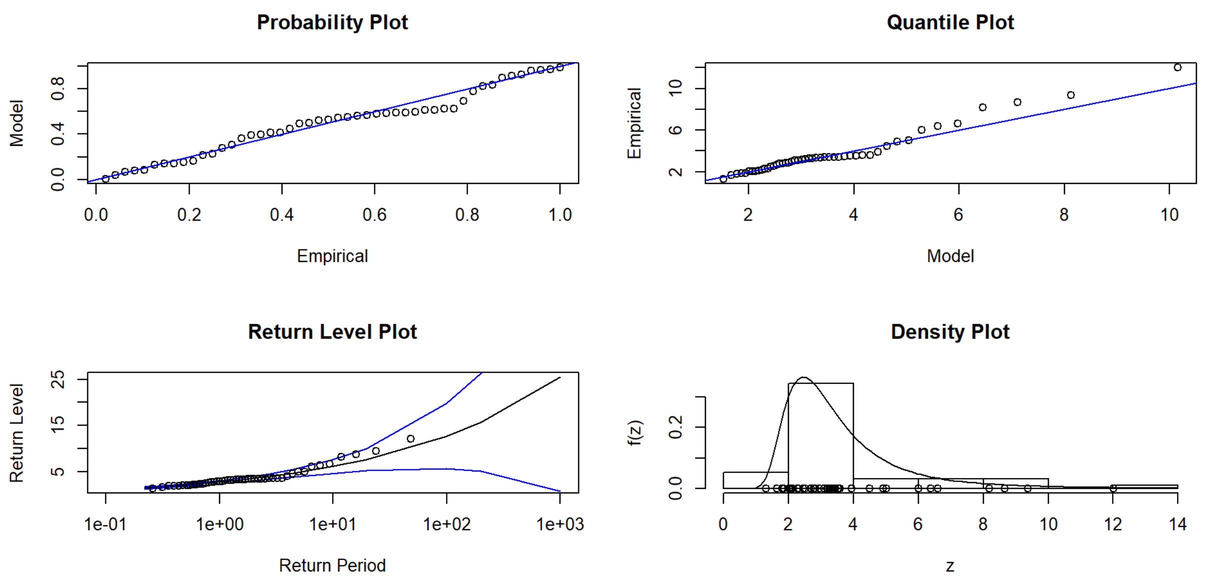
```
## Loading required package: nlme
```

```
## This is mgcv 1.8-23. For overview type 'help("mgcv-package")'.
```

```
SieGEV2 <- gev.fit(SieGEV$data)
```

```
## $conv
## [1] 0
##
## $nllh
## [1] 85.95
##
## $mle
## [1] 2.6997 1.0467 0.2867
##
## $se
## [1] 0.1698 0.1413 0.1171
```

```
gev.diag(SieGEV2)
```



```
png(filename="7_3_diagSimens.png", width = 800, height = 600)
gev.diag(SieGEV2)
dev.off()
```

```
## png
##   2
```

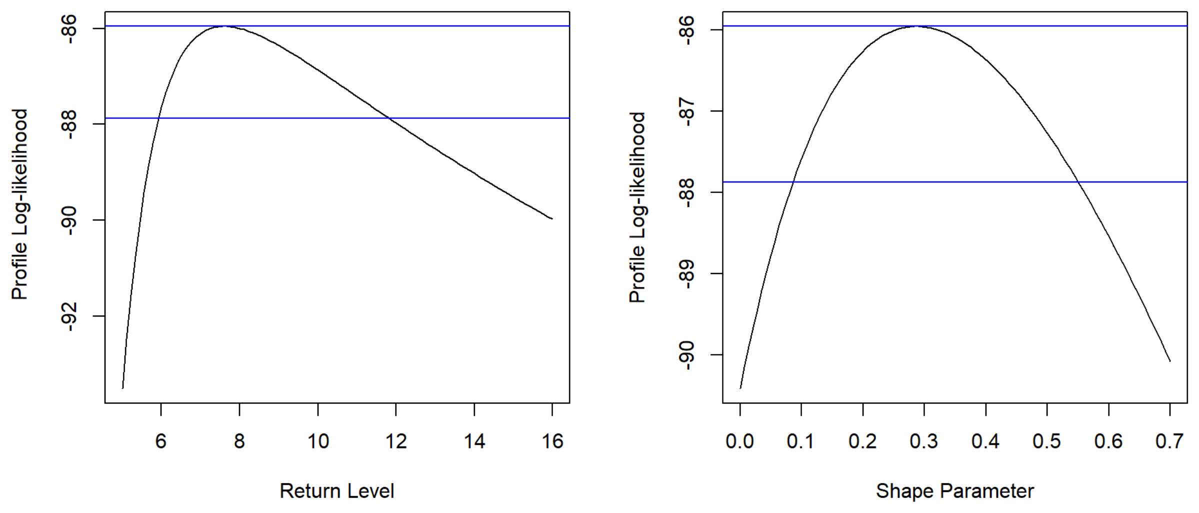
`gev.prof()` produces a return level plot for m years per block. `gevprofxi()` returns the profile with respect to the shape parameter only. Confidence bands can be superimposed on both kinds of graph.

```
par(mfrow = c(1, 2))
gev.prof(SieGEV2, m = 20, xlow = 5, xup = 16, conf = 0.95)
```

```
## If routine fails, try changing plotting interval
```

```
gev.profxi(SieGEV2, xlow = 0.0, xup = 0.7, conf = 0.95)
```

```
## If routine fails, try changing plotting interval
```



```
png(filename="7_4_likeli.png", width = 800, height = 400)
gev.prof(SieGEV2, m = 20, xlow = 5, xup = 16, conf = 0.95)
```

```
## If routine fails, try changing plotting interval
```

```
dev.off()
```

```
## png
##   2
```

```r
retrun.level.data <- function (z, m, xlow, xup, conf = 0.95, nint = 100)
{
    if (m <= 1)
        stop("`m' must be greater than one")
    cat("If routine fails, try changing plotting interval", fill = TRUE)
    p <- 1/m
    v <- numeric(nint)
    x <- seq(xlow, xup, length = nint)
    sol <- c(z$mle[2], z$mle[3])
    gev.plik <- function(a) {
        if (abs(a[2]) < 10^(-6)) {
            mu <- xp + a[1] * log(-log(1 - p))
            y <- (z$data - mu)/a[1]
            if (is.infinite(mu) || a[1] <= 0)
                l <- 10^6
            else l <- length(y) * log(a[1]) + sum(exp(-y)) +
                sum(y)
        }
        else {
            mu <- xp - a[1]/a[2] * ((-log(1 - p))^(-a[2]) - 1)
            y <- (z$data - mu)/a[1]
            y <- 1 + a[2] * y
            if (is.infinite(mu) || a[1] <= 0 || any(y <= 0))
                l <- 10^6
            else l <- length(y) * log(a[1]) + sum(y^(-1/a[2])) +
                sum(log(y)) * (1/a[2] + 1)
        }
        l
    }
    for (i in 1:nint) {
        xp <- x[i]
        opt <- optim(sol, gev.plik)
        sol <- opt$par
        v[i] <- opt$value
    }
    v <- data.frame(percentage=x, LogLik=v)
    return(v)
}

retrun.level.20 <- retrun.level.data(SieGEV2, m = 20, xlow = 5, xup = 16, conf = 0.
95)
```

```
## If routine fails, try changing plotting interval
```

```r
retrun.level.20$percentage[retrun.level.20$LogLik == min(retrun.level.20$LogLik)]
```

```
## [1] 7.556
```

```r
mLoss <- max(SieGEV$data)
mLoss
```

```
## [1] 12.01
```

```
mYears <- 1 / (1 - pgev(mLoss, mu=SieGEV2$mle[1], sigma=SieGEV2$mle[2], xi=SieGEV2
$mle[3])) / 2
mYears
```

```
## [1] 41.77
```

Further inference from the model can be made using the profile log-likelihoods. Figure 7.4 shows these for a 10-year return level (left panel) and for the shape parameter (right panel). A daily loss as high as 7.6% would be observed once every 10 years. This point estimate falls within a 95% confidence level ranging from 6% to 11.75%. Hence, the maximum observed loss of 12.01% would not have been covered as a “once in every 10 years” event, but rather this loss would occur only once every 42 years or so. In the right panel of Figure 7.4, the profile log-likelihood for the shape parameter is shown with a 95% confidence band (the horizontal light gray lines) superimposed. As can clearly be seen, the confidence band is asymmetric and to the right for the point estimate of ξ???? = 0.287. A value of almost 0.6 would be covered by this confidence band.

```
## package fExtremes:
library(fExtremes)
```

```
## Loading required package: fBasics
```

```
## Loading required package: fGarch
```

```
##
## Attaching package: 'fExtremes'
```

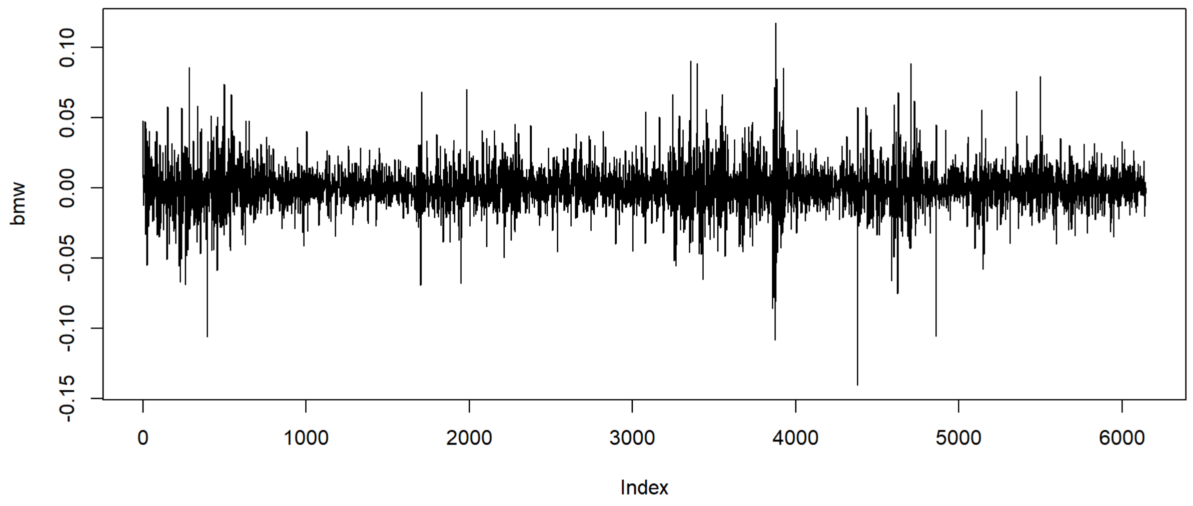
```
## The following objects are masked from 'package:evir':
##
##     dgev, dgpd, pgev, pgpd, qgev, qgpd, rgev, rgpd
```

```
SieGEV3 <-  gevFit(SieGEV$data, type = "pwm")
SieGEV3
```

```
##
## Title:
##  GEV Parameter Estimation
##
## Call:
##  gevFit(x = SieGEV$data, type = "pwm")
##
## Estimation Type:
##   gev pwm
##
## Estimated Parameters:
##     xi     mu   beta
## 0.3185 2.6746 1.0008
##
## Description
##   Thu Apr 26 02:19:29 2018
```

r-block maxima for BMW

```
library(evir)
library(ismev)
## Order statistics
data(bmw)
plot(bmw, type="l")
```

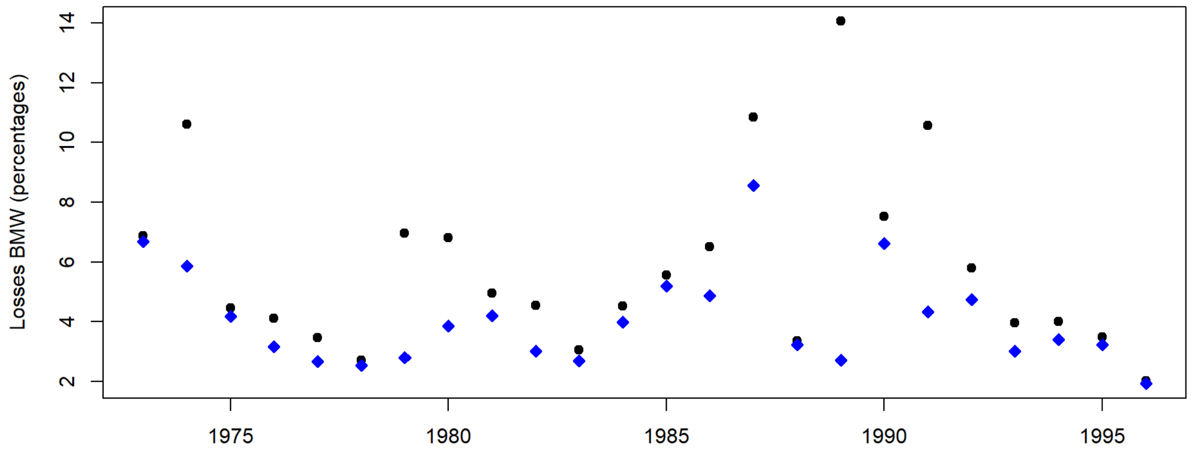


```
png(filename="7_5_bmwRet.png", width = 800, height = 400)
plot(bmw, type="l")
dev.off()
```

```
## png
##   2
```

```
BmwLoss <- -1.0 * bmw * 100
Years <- format(attr(BmwLoss, "time"), "%Y")
attr(BmwLoss, "years") <- Years
Yearu <- unique(Years)
idx <- 1:length(Yearu)
r <- 2
BmwOrder <- t(sapply(idx, function(x)
              head(sort(BmwLoss[attr(BmwLoss, "years") == Yearu[x]], decreasing = T
RUE), r)))
rownames(BmwOrder) <- Yearu
colnames(BmwOrder) <- paste("r", 1:r, sep = "")
```

```
## Plot of order data
plot(Yearu, BmwOrder[, 1], col = "black", ylim = range(BmwOrder),
     ylab = "Losses BMW (percentages)", xlab = "",
     pch = 21, bg = "black")
points(Yearu, BmwOrder[, 2], col = "blue", pch = 23, bg = "blue")
```



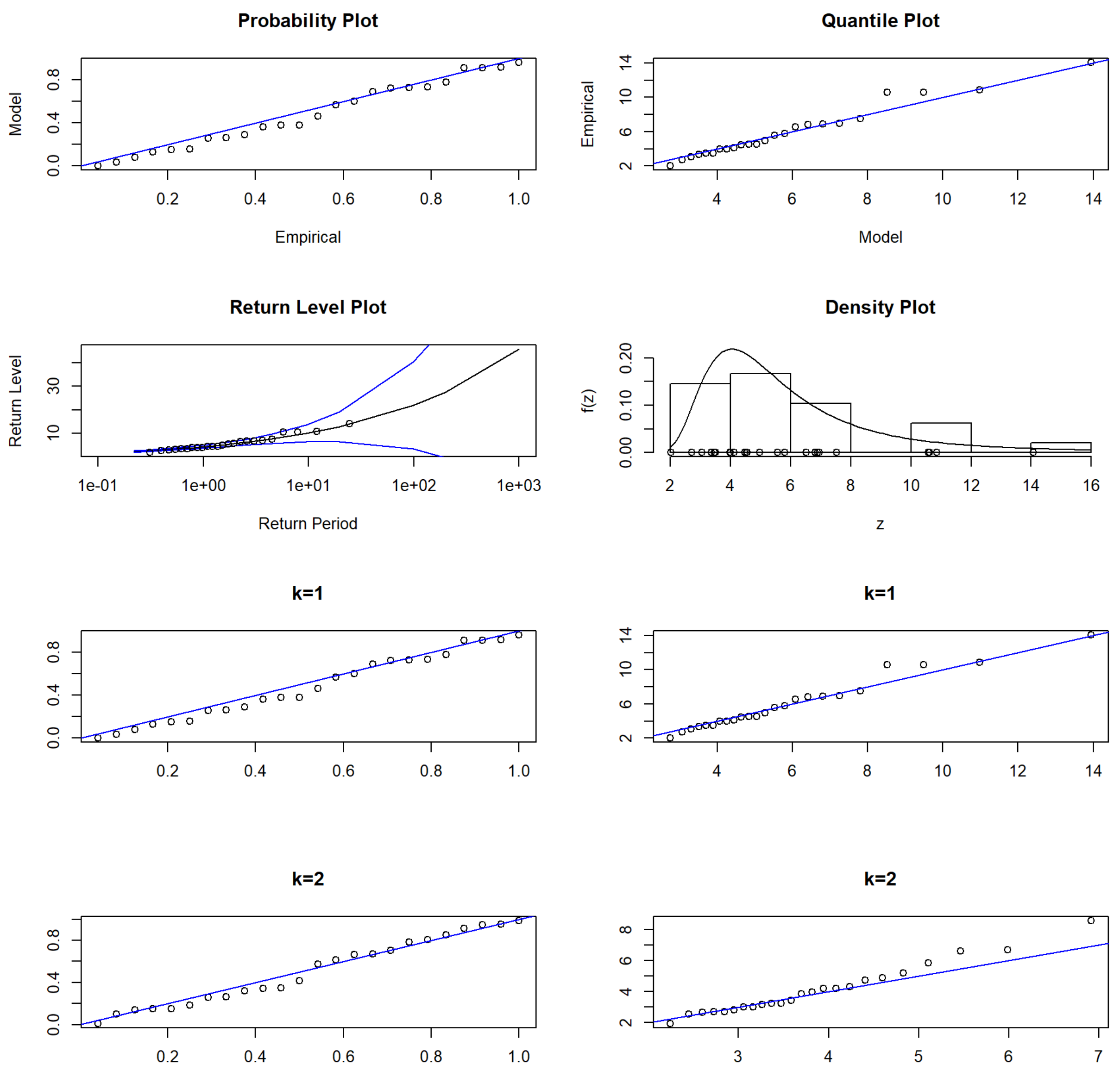
```
png(filename="7_6_rPlot.png", width = 800, height = 400)
plot(Yearu, BmwOrder[, 1], col = "black", ylim = range(BmwOrder),
     ylab = "Losses BMW (percentages)", xlab = "",
     pch = 21, bg = "black")
points(Yearu, BmwOrder[, 2], col = "blue", pch = 23, bg = "blue")
dev.off()
```

```
## png
##   2
```

```
## Fit and diagnostics
BmwOrderFit <- rlarg.fit(BmwOrder)
```

```
## $conv
## [1] 0
##
## $nllh
## [1] 77.76
##
## $mle
## [1] 4.4798 1.7525 0.3035
##
## $se
## [1] 0.3411 0.2892 0.1587
```

```
rlarg.diag(BmwOrderFit)
```

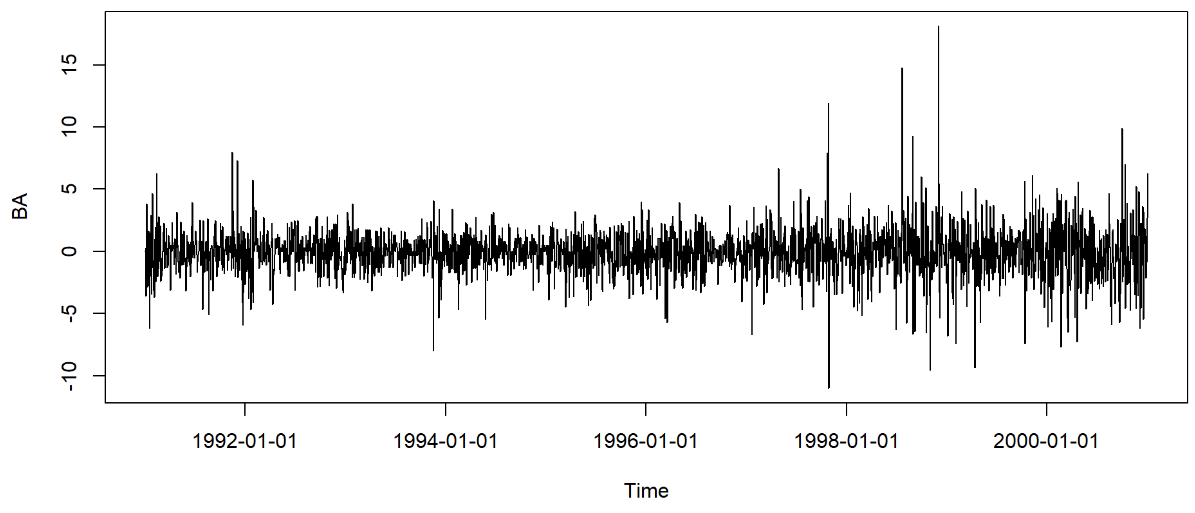


```
png(filename="7_7_diagbmw.png", width = 800, height = 400)
par(mfrow=c(2,4))
rlarg.diag(BmwOrderFit)
dev.off()
```

```
## png
##   2
```

POT method for Boeing

```
library(fBasics)
library(fExtremes)
## Data handling
data(DowJones30)
DJ <- timeSeries(DowJones30[, -1], charvec = as.character(DowJones30[, 1]))
BALoss <- -1.0 * returns(DJ[, "BA"], percentage = TRUE, trim = TRUE)
plot(BALoss)
```

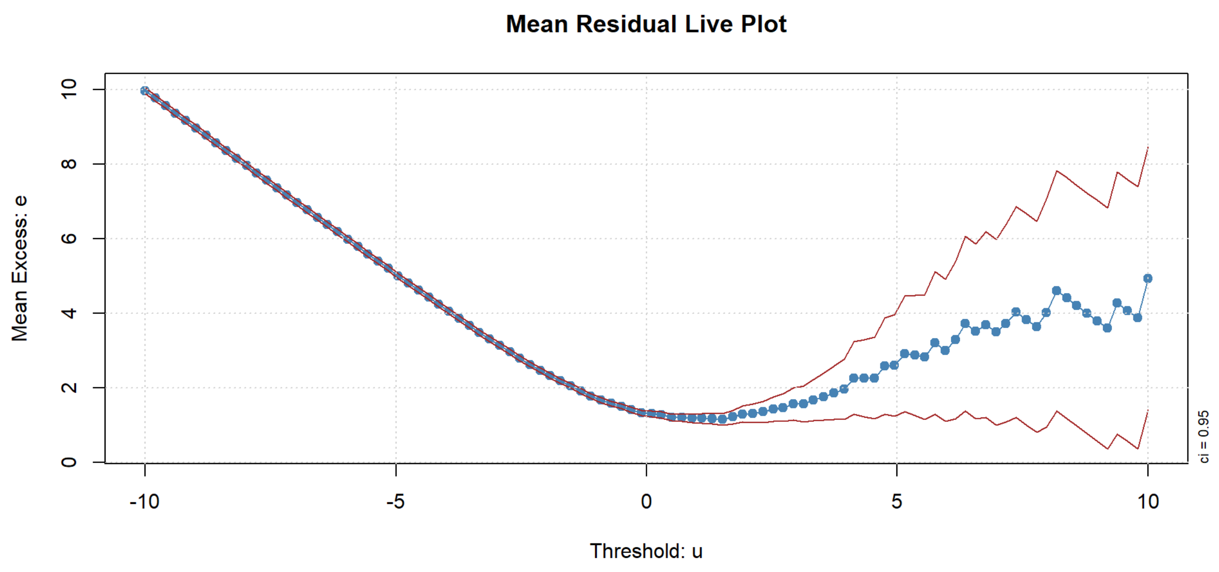


```
png(filename="7_8_BARet.png", width = 800, height = 400)
plot(BALoss)
dev.off()
```

```
## png
##   2
```

MRL-plot

```
mrlPlot(BALoss, umin = -10, umax = 10, labels = T)
```



```
png(filename="7_9_mrlPlot.png", width = 800, height = 400)
mrlPlot(BALoss, umin = -10, umax = 10, labels = T)
dev.off()
```

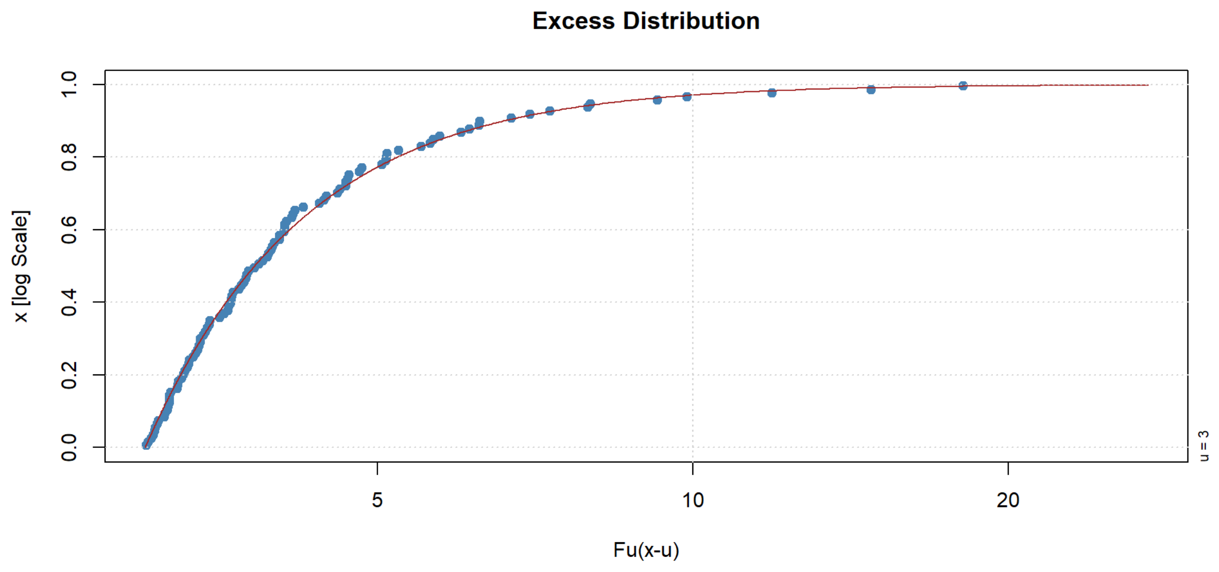
```
## png
##   2
```

```
findThreshold(BALoss, n = floor(0.041*length(as.vector(BALoss))), doplot = F)
```

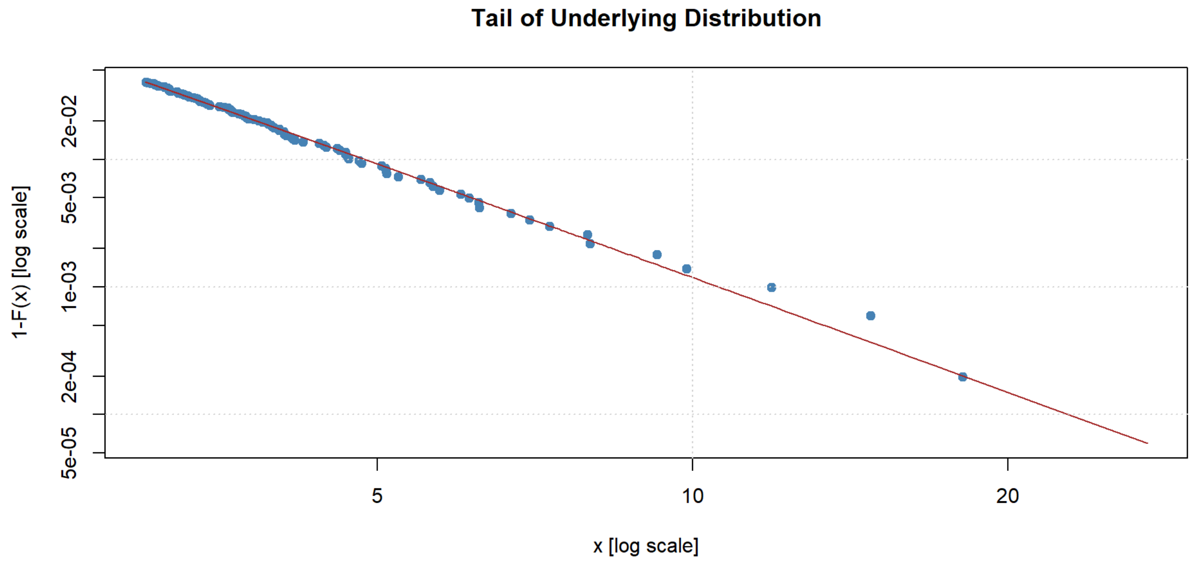
```
## n=103
## 2.984
```

```
### GPD
BAFit <- gpdFit(BALoss, u = 3)
summary(BAFit)
```

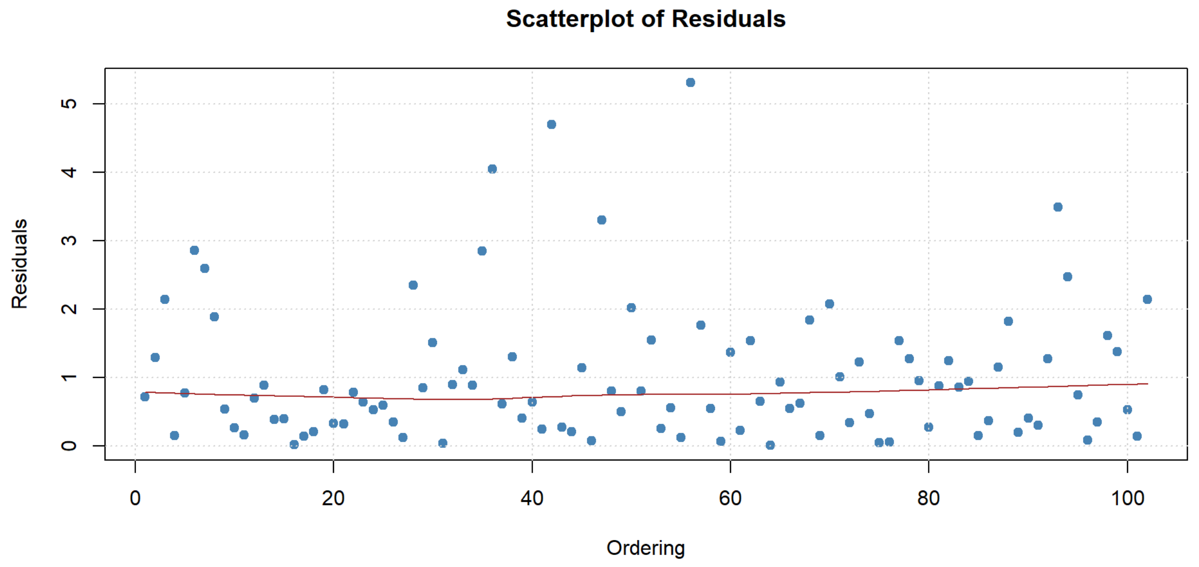
```
##
## Title:
##  GPD Parameter Estimation
##
## Call:
## gpdFit(x = BALoss, u = 3)
##
## Estimation Type:
##   gpd mle
##
## Estimated Parameters:
##     xi   beta
## 0.3307 1.0467
##
## Standard Deviations:
##     xi   beta
## 0.1280 0.1658
##
## Log-Likelihood Value:
##   140.4
##
## Type of Convergence:
##   0
```



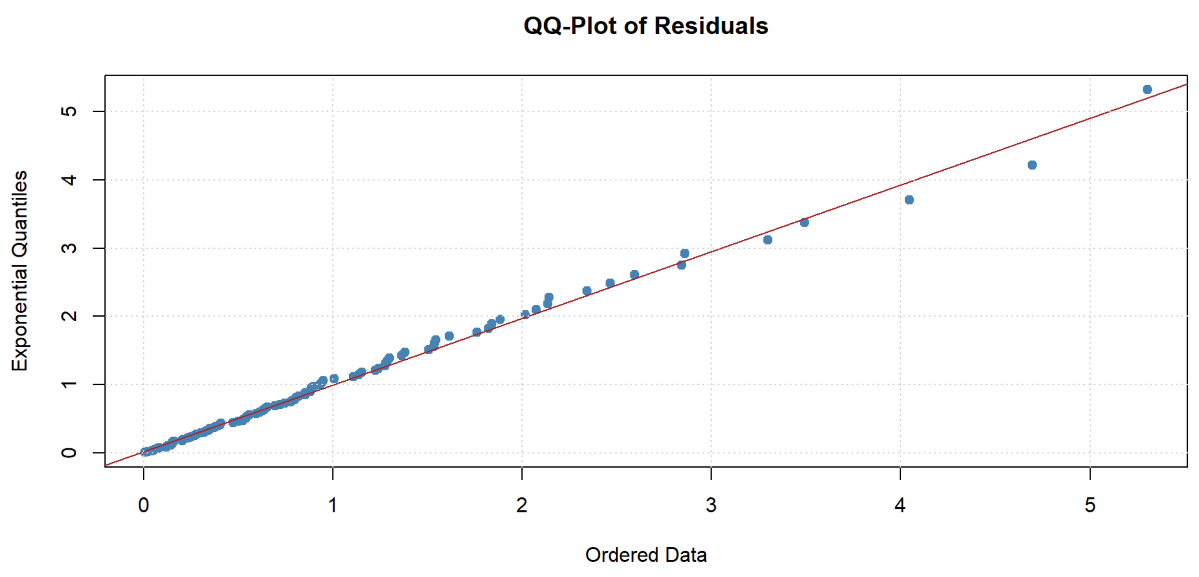
Excess Distribution
x [log Scale]
0.0
0.2
0.4
0.6
0.8
1.0
5
10
20
Fu(x-u)
u = 3



Tail of Underlying Distribution
1-F(x) [log scale]
2e-02
5e-03
1e-03
2e-04
5e-05
5
10
20
x [log scale]



Scatterplot of Residuals
Residuals
0
1
2
3
4
5
0
20
40
60
80
100
Ordering



```
##
## Description
##   Thu Apr 26 02:19:32 2018 by user: s7794
```

```
## Diagnostic plots
# plot(BAFit)
png(filename="7_10_diagBA.png", width = 800, height = 400)
par(mfrow=c(2,2))
summary(BAFit)
```

```
##
## Title:
##  GPD Parameter Estimation
##
## Call:
## gpdFit(x = BALoss, u = 3)
##
## Estimation Type:
##   gpd mle
##
## Estimated Parameters:
##     xi    beta
## 0.3307 1.0467
##
## Standard Deviations:
##     xi    beta
## 0.1280 0.1658
##
## Log-Likelihood Value:
##   140.4
##
## Type of Convergence:
##   0
```

```
##
## Description
##   Thu Apr 26 02:19:32 2018 by user: s7794
```

```
dev.off()
```

```
## png
##   2
```

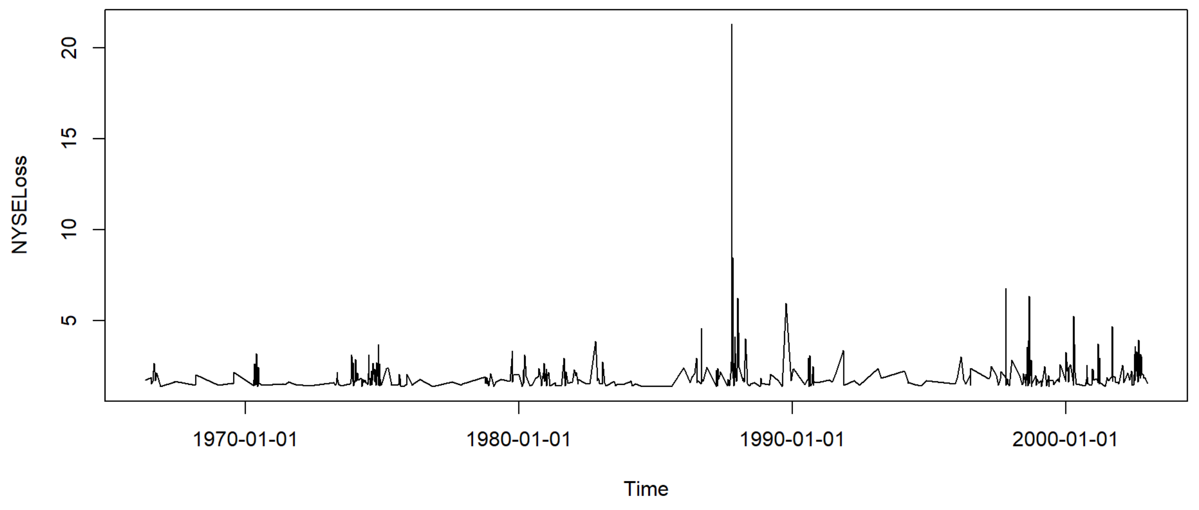
Risk measures

```
gpdRiskMeasures(BAFit, prob = c(0.95, 0.99, 0.995))
```

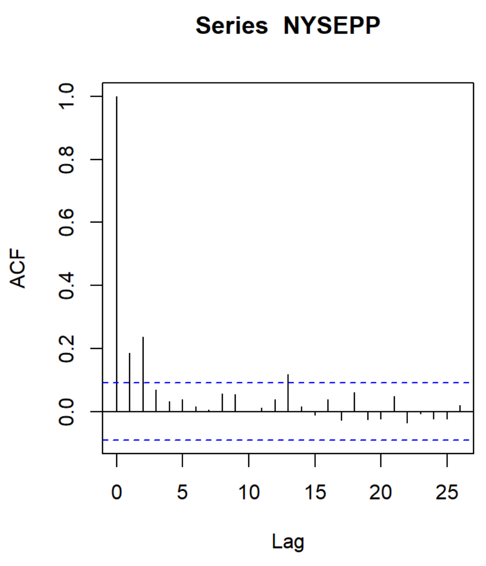
```
##       p quantile shortfall
## 1 0.950    2.783     4.240
## 2 0.990    4.855     7.336
## 3 0.995    6.149     9.268
```

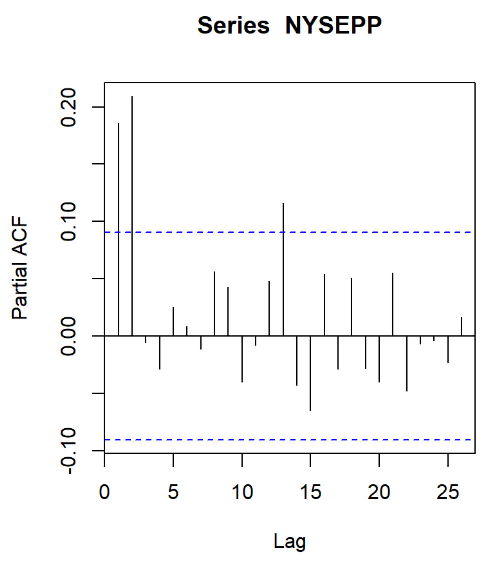
```
library(fExtremes)
library(fBasics)
data(nyse)
NYSELevel <- timeSeries(nyse[, 2], charvec = as.character(nyse[, 1]))
NYSELoss <- na.omit(-1.0 * diff(log(NYSELevel)) * 100)
colnames(NYSELoss) <- "NYSELoss"
```

```
## Point process data
NYSEPP <- pointProcess(x = NYSELoss, u = quantile(NYSELoss, 0.95))
plot(NYSEPP)
```



```
par(mfrow=c(1,2), mar=c(5,5,5,5)); acf(NYSEPP); pacf(NYSEPP)
```



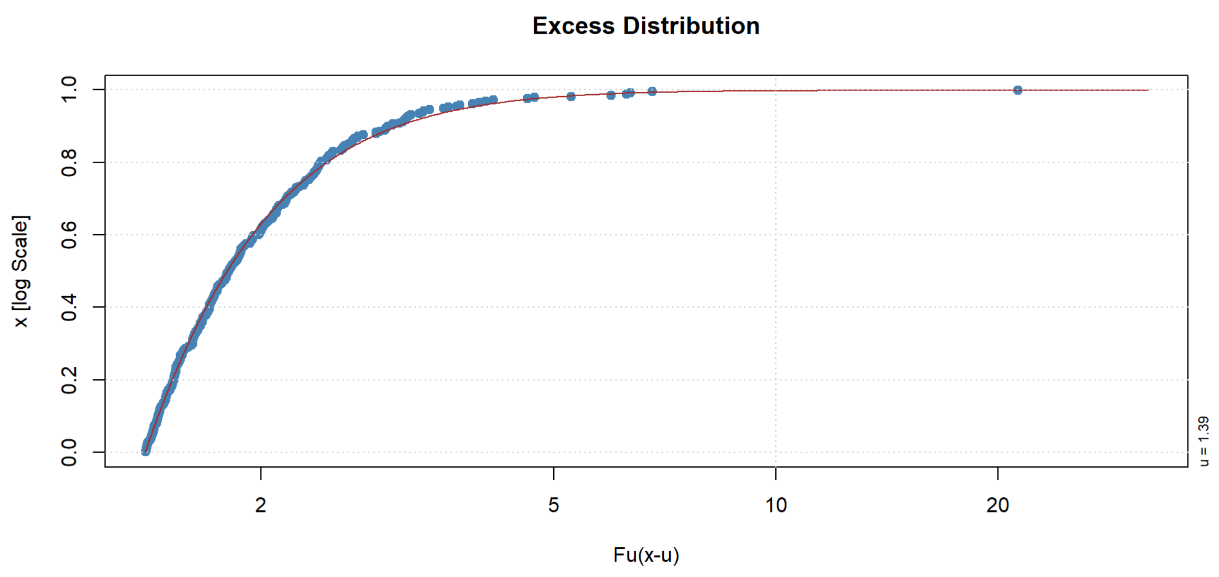


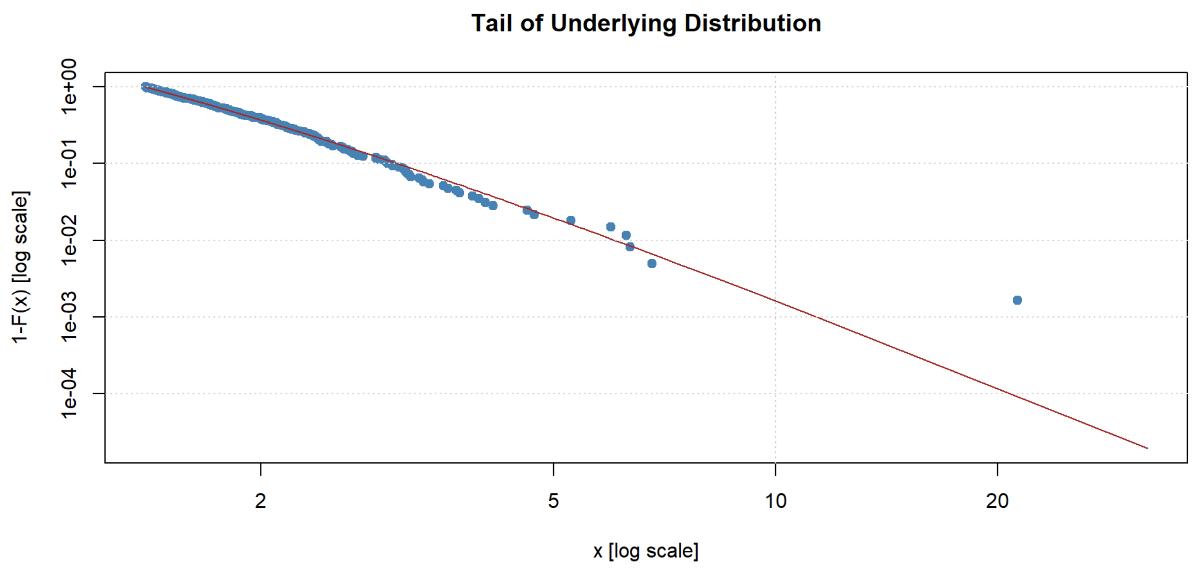
```
## Declustering
DC05 <- deCluster(x = NYSEPP, run = 5, doplot = FALSE)
DC10 <- deCluster(x = NYSEPP, run = 10, doplot = FALSE)
DC20 <- deCluster(x = NYSEPP, run = 20, doplot = FALSE)
DC40 <- deCluster(x = NYSEPP, run = 40, doplot = FALSE)
DC60 <- deCluster(x = NYSEPP, run = 60, doplot = FALSE)
DC120 <- deCluster(x = NYSEPP, run = 120, doplot = FALSE)
```

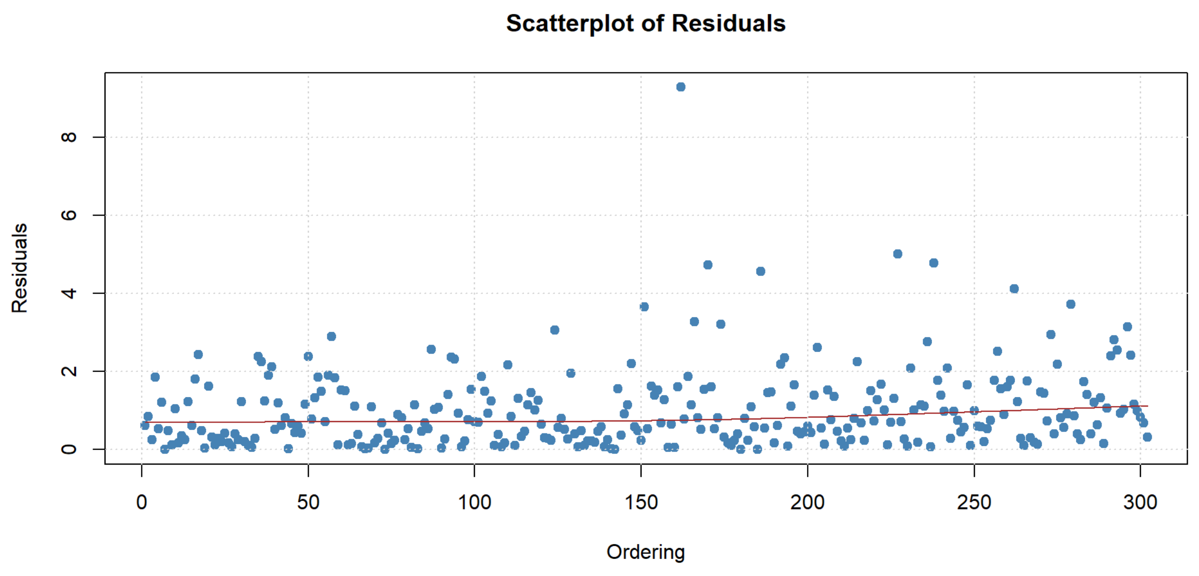
```
## Fit of declustered data
DC05Fit <- gpdFit(DC05, u = min(DC05))
DC10Fit <- gpdFit(DC10, u = min(DC10))
DC20Fit <- gpdFit(DC20, u = min(DC20))
DC40Fit <- gpdFit(DC40, u = min(DC40))
DC60Fit <- gpdFit(DC60, u = min(DC60))
DC120Fit <- gpdFit(DC120, u = min(DC40))
```

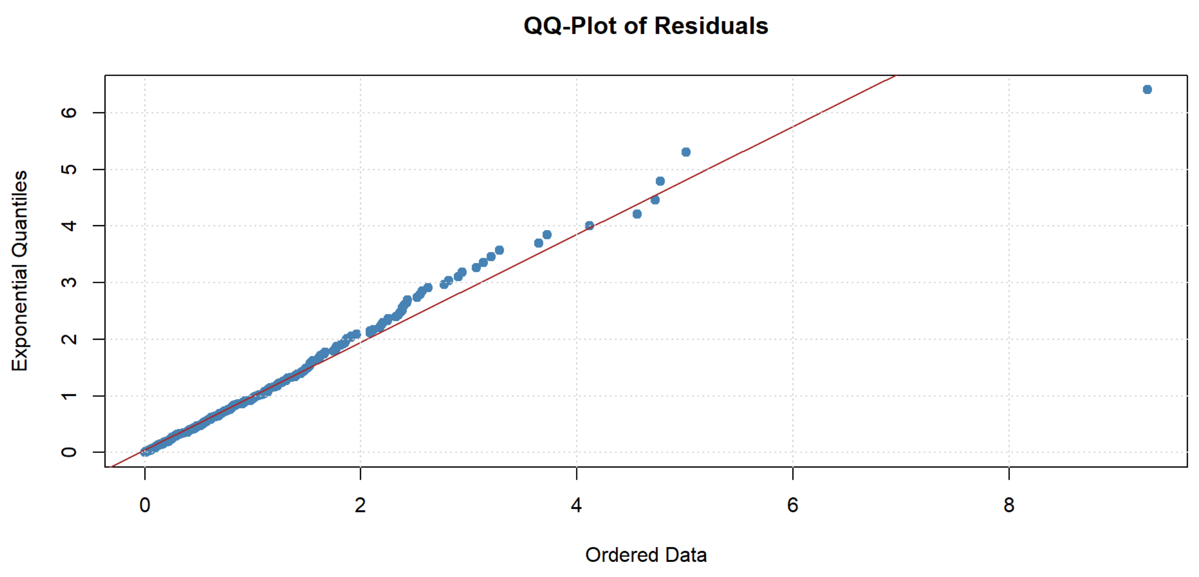
```
summary(DC05Fit)
```

```
##
## Title:
##  GPD Parameter Estimation
##
## Call:
## gpdFit(x = DC05, u = min(DC05))
##
## Estimation Type:
##   gpd mle
##
## Estimated Parameters:
##     xi   beta
## 0.2516 0.5364
##
## Standard Deviations:
##      xi    beta
## 0.06571 0.04620
##
## Log-Likelihood Value:
##   189.9
##
## Type of Convergence:
##   0
```


Excess Distribution
x [log Scale]
0.0
0.2
0.4
0.6
0.8
1.0
2
5
10
20
Fu(x-u)
u = 1.39


Tail of Underlying Distribution
1-F(x) [log scale]
1e+00
1e-01
1e-02
1e-03
1e-04
2
5
10
20
x [log scale]


Scatterplot of Residuals
Residuals
0
2
4
6
8
0
50
100
150
200
250
300
Ordering



```
##
## Description
##   Thu Apr 26 02:19:40 2018 by user: s7794
```