MS.R

Alice Li

Tue Jul 04 13:57:52 2017

```r
rm(list=ls())

#a)
#MS
MS<-read.csv("MS.csv")
x_array <- rep(0,nrow(MS)-1)

for (i in 1:nrow(MS)-1){
  x_array[i] = -100*log(MS$Adj.Close[i+1]/MS$Adj.Close[i])
}


blocksize <- 26
blocks <- floor(length(x_array)/blocksize)

max_array <- rep(0,blocks)

for (i in 1:blocks){
  start_ind <- (i - 1)*26
  end_ind  <- start_ind + 25
  max_array[i] = max(x_array[start_ind:end_ind])
}

hist(max_array,
     main="Histogram of MSMax",
     xlab="MSMax",
     breaks = 15)




#b)
library(qrmdata) # for the S&P 500 data
```
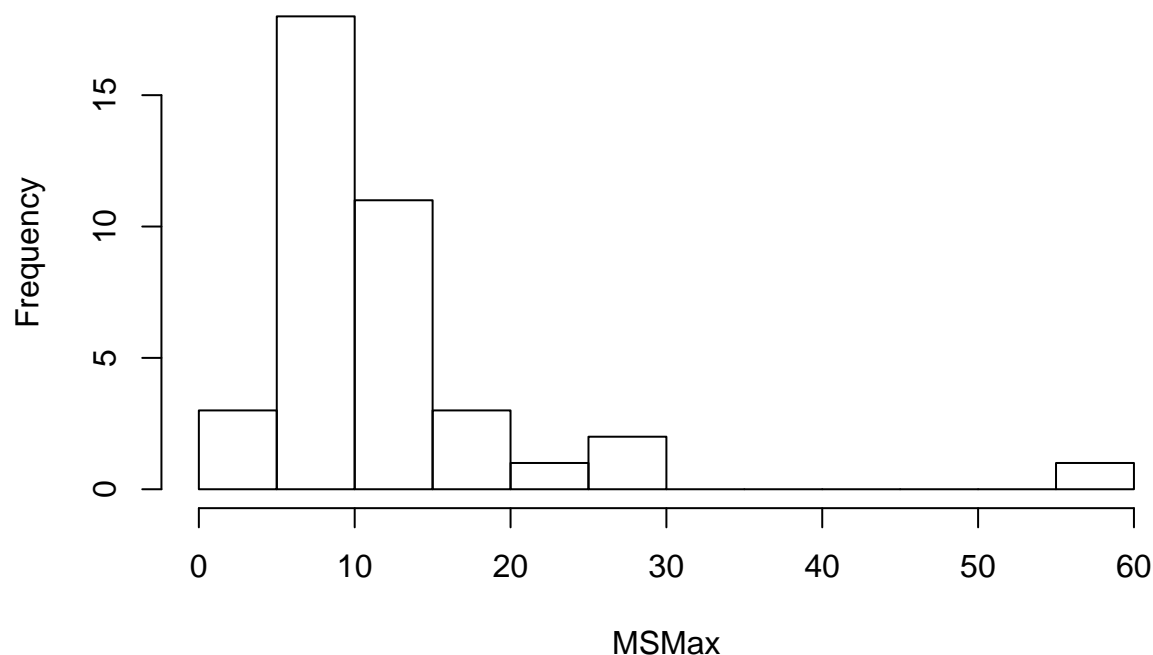
Histogram of MSMax


```r
library(xts) # for functions around time series objects
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(QRM) # for fit.GEV() and fit.GPD()
```

```
## Loading required package: gsl
```

```
## Loading required package: Matrix
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: numDeriv
```

```
## Loading required package: timeSeries
```

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

```
##
## Attaching package: 'timeSeries'
```

```
## The following object is masked from 'package:zoo':
##
##     time<-
```

```
##
## Attaching package: 'QRM'

## The following object is masked from 'package:base':
##
##     lbeta
```

```
library(qrmtools) # for returns() <- be RYre to load "qrmtools" last
```

```
##
## Attaching package: 'qrmtools'

## The following objects are masked from 'package:QRM':
##
##     dGEV, dGPD, pGEV, pGPD, qGEV, qGPD, rGEV, rGPD

## The following object is masked from 'package:timeSeries':
##
##     returns
```

```
fit.week <- fit.GEV(max_array)
(xi.week <- fit.week$par.ests[["xi"]])
```

```
## [1] 0.296817
```

```
(mu.week <- fit.week$par.ests[["mu"]])
```

```
## [1] 7.903999
```

```
(sig.week <- fit.week$par.ests[["sigma"]])
```

```
## [1] 3.927648
```

```
fit.week
```

```
## $par.ests
##       xi       mu    sigma
## 0.296817 7.903999 3.927648
##
## $par.ses
##        xi        mu     sigma
## 0.1197650 0.6978555 0.5840945
##
## $varcov
##              [,1]        [,2]        [,3]
## [1,]   0.014343665 -0.01793308 0.009842084
## [2,]  -0.017933083  0.48700226 0.248894205
## [3,]   0.009842084  0.24889421 0.341166353
##
## $converged
## [1] TRUE
##
## $llmax
## [1] -121.668
```

```
#c)
q.week <- qGEV(ppoints(length(max_array)), xi.week, mu.week, sig.week) # contruct theoretical quantiles
qqplot(q.week, as.vector(max_array), xlab = "Theoretical Quantile (Fitted GEV)",
       ylab = "Empirical Quantile") # compare weekly maxima with theoretical quantiles
abline(0,1) # 45 degree line
```
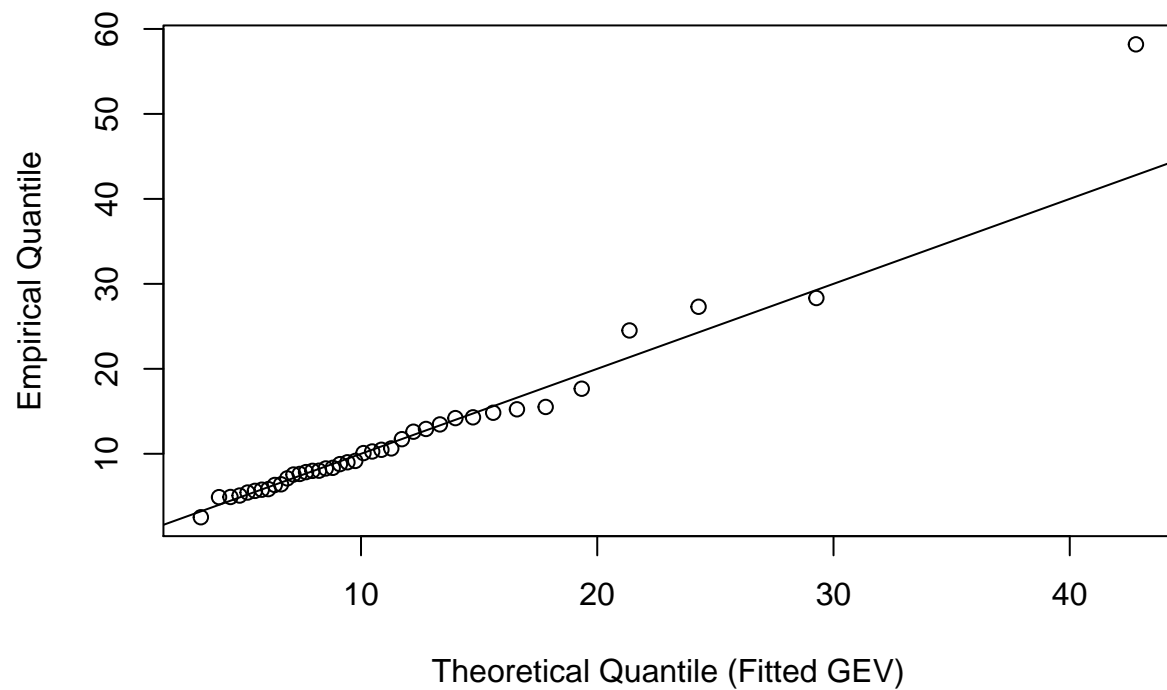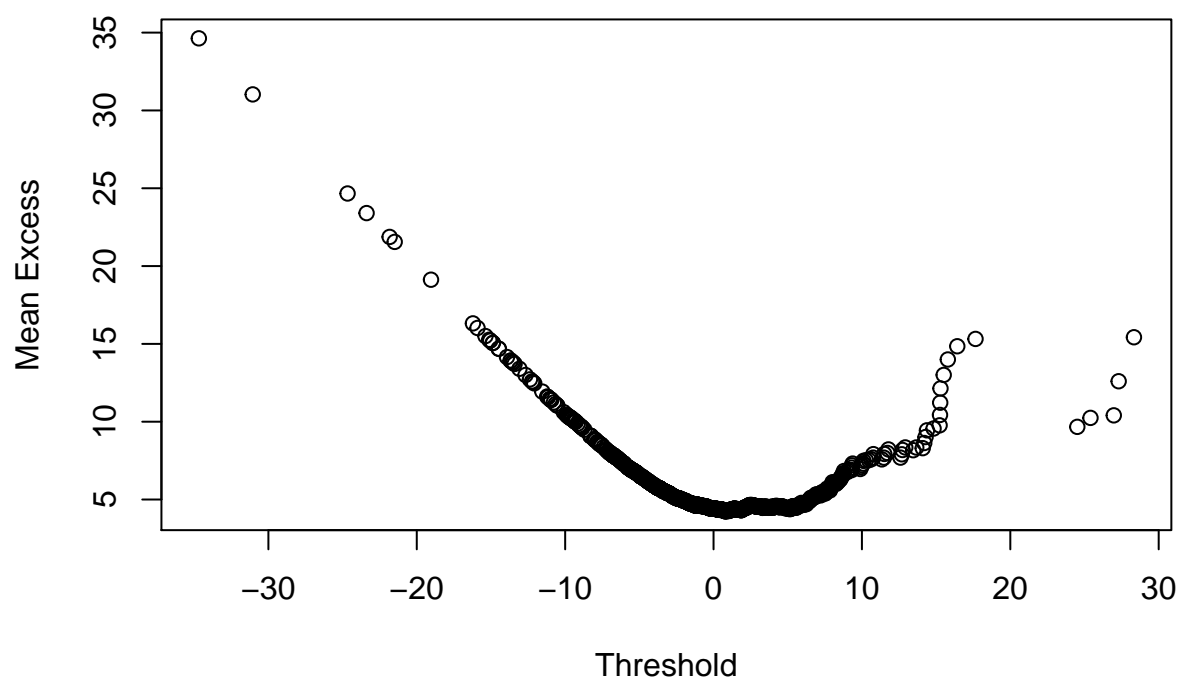
```
#d)
llmax1 <- fit.week$llmax
h.func<-function(M){1/sig.week*exp(-(M-mu.week)/sig.week - exp(-(M-mu.week)/sig.week))}
llmax0 <- sum(log(h.func(max_array)))
D<- 2*(llmax1 - llmax0)
D
```

```
## [1] 13.18707
```

```
1-pchisq(D,1)
```

```
## [1] 0.0002818879
```

```
#e)
MEplot(x_array, main = "Sample Mean-Excess Plot")
```
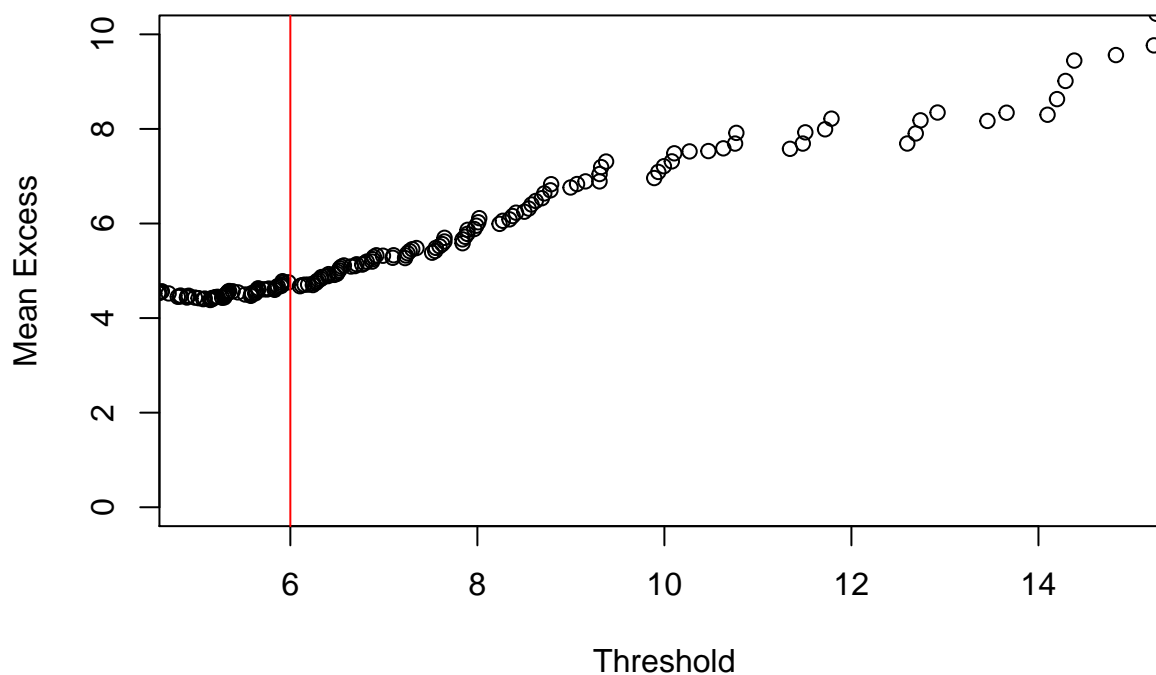
Sample Mean−Excess Plot


```r
quantile(x_array,0.9)
```

```
##      90%
## 6.500111
```

```r
MEplot(x_array, main = "Sample Mean-Excess Plot", xlim = c(5,15), ylim = c(0,10)) #zoomin
#pick d1 = 6
d1 <- 6
abline(v = d1, col = 2)
```

Sample Mean–Excess Plot


```
#f)
fitGPD <- fit.GPD(x_array, threshold = d1)

xi2 <- fitGPD$par.ests[["xi"]]
beta2 <- fitGPD$par.ests[["beta"]]
lmax1 <- fitGPD$ll.max

#g)
g.func<-function(X){1/beta2 * exp(-X/beta2)}
lmax0 <- sum(log(g.func(x_array[x_array>d1] - d1)))
lmax1
```

```
## [1] -292.5931
```

```
lmax0
```

```
## [1] -315.0335
```

```
D2<- 2*(lmax1 - lmax0)
1-pchisq(D2,1)
```

```
## [1] 2.09408e-11
```

```
#h)
Sxd <- length(x_array[x_array>d1])/length(x_array)
Q99<- d1 + beta2/xi2 * ((Sxd / (1 - 0.99))^xi2 - 1)
Q995<- d1 + beta2/xi2 * ((Sxd / (1 - 0.995))^xi2 - 1)
Q99
```

```
## [1] 18.08309
```

Q995

```
## [1] 23.98553
```

```r
CTE99 <- (Q99 + beta2 - xi2*d1)/(1-xi2)
CTE995 <- (Q995 + beta2 - xi2*d1)/(1-xi2)
CTE99
```

```
## [1] 29.89928
```

CTE995

```
## [1] 39.22618
```

```r
#empirical
Q99e <- quantile(x_array,0.99)
Q995e <- quantile(x_array,0.995)
Q99e
```

```
##      99%
## 15.74282
```

Q995e

```
##    99.5%
## 26.84011
```

```r
CTE99e <-mean(x_array[x_array > Q99e])
CTE995e <- mean(x_array[x_array > Q995e])

CTE99e
```

```
## [1] 28.51581
```

CTE995e

```
## [1] 35.64741
```