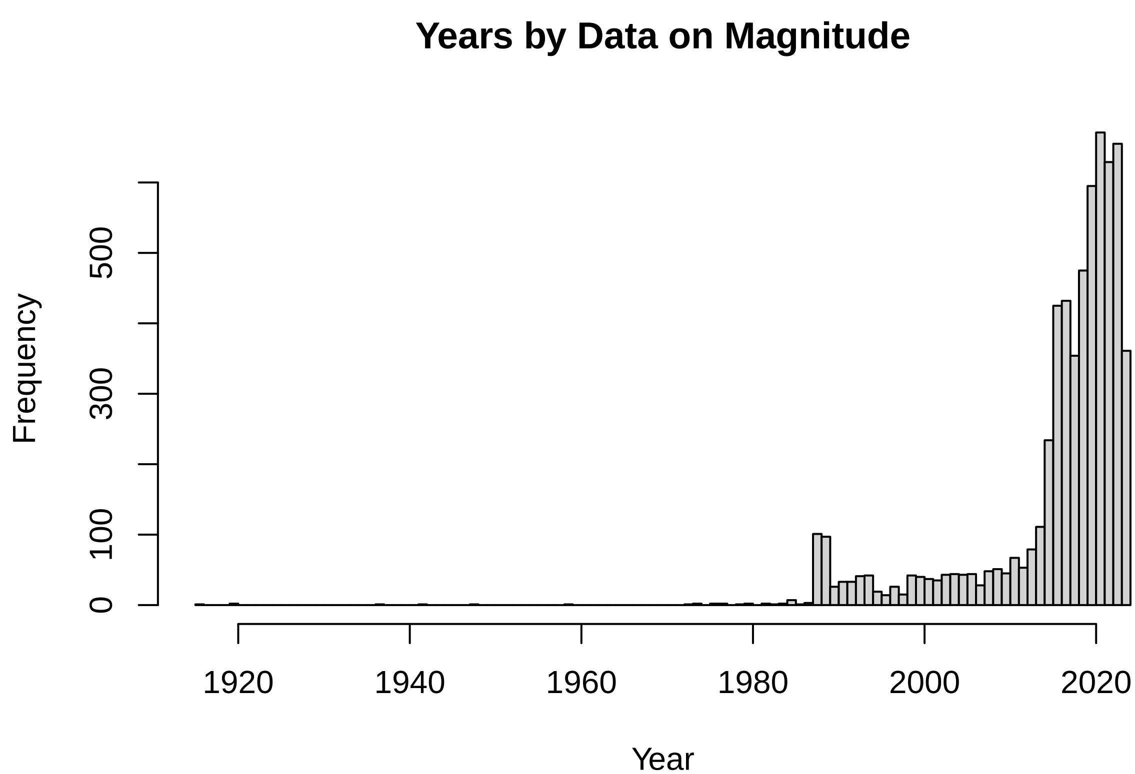
Earthquakes

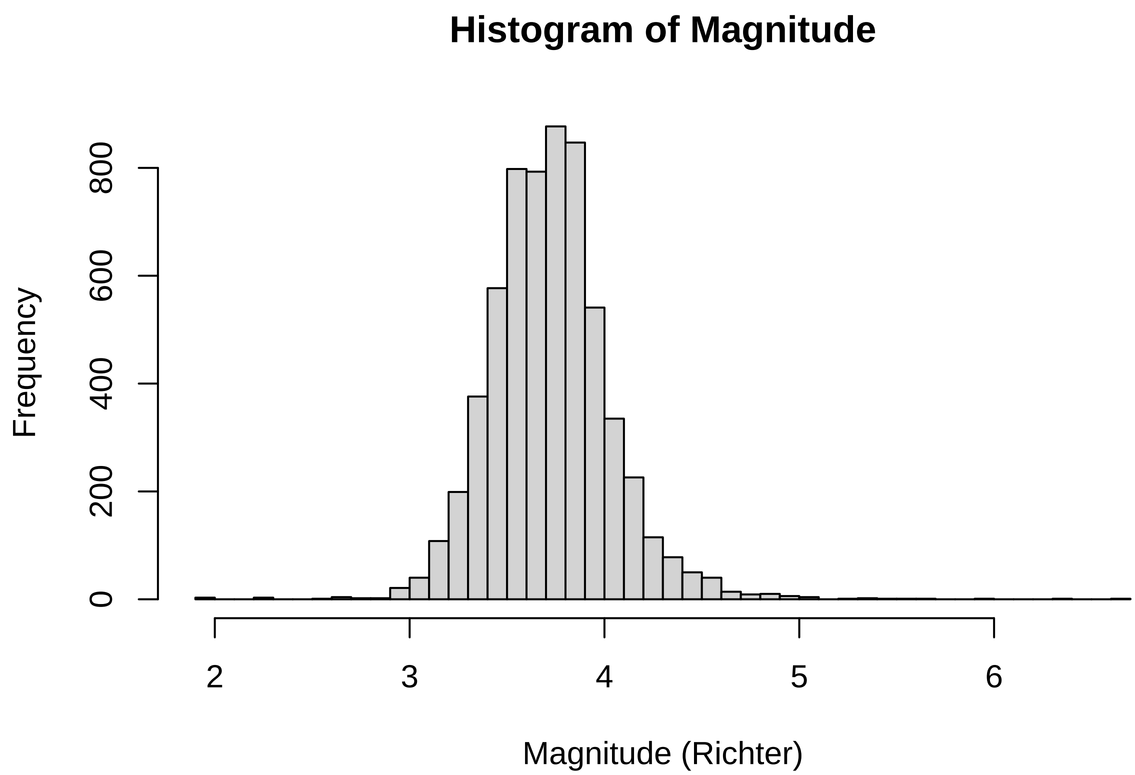
Marissa Kroeger, Luis Conde Meza, Edgar Moya, Adrian Bautista, advisor: Dr. Sergio Juarez

2025-07-08

```
df <- df[, c("Fecha", "Magnitud")] # select the columns of interest

# create a column for the year
df <- df %>%
  mutate(Year = year(Fecha))

# Filter out NAs
df <- df %>%
  filter(!is.na(Magnitud))

# View the number of valid entries per year to see when sufficient data is available
hist(df$Year, breaks = seq(min(df$Year) -1 , max(df$Year), by = 1), main = "Years by Data on Magnitude"
```



```r
df <- df %>%
 filter(Year >= 1988)

# View the magnitude distribution
hist(df$Magnitud, breaks = seq(min(df$Magnitud), max(df$Magnitud), by = .1), main = "Histogram of Magni
```



GEVD Fit

Extract Maxima

```r
years <- seq(1988,2024)
N <- length(years)

yearly_maxima <- vector("list", N)

# Extract maxima for each year
for(i in 1:N) {
 year <- df %>%
   filter(Year == years[i])

 index <- which.max(year$Magnitud)
```

```
 yearly_maxima[[i]] <- year[index,]
}

yearly_maxima = dplyr::bind_rows(yearly_maxima) # combine into a tibble

yearly_maxima
```

```
## # A tibble: 37 x 3
##    Fecha               Magnitud  Year
##    <dttm>                 <dbl> <dbl>
##  1 1988-05-16 00:00:00      4.8  1988
##  2 1989-02-21 00:00:00      4.6  1989
##  3 1990-02-10 00:00:00      4.9  1990
##  4 1991-07-25 00:00:00      5.3  1991
##  5 1992-06-04 00:00:00      4.7  1992
##  6 1993-04-05 00:00:00      4.8  1993
##  7 1994-11-10 00:00:00      4.5  1994
##  8 1995-01-03 00:00:00      4.4  1995
##  9 1996-06-03 00:00:00      4.9  1996
## 10 1997-09-06 00:00:00      4.7  1997
## # i 27 more rows
```

Fit Model

```
# Fitting the GEVD
fit <- gev.fit(yearly_maxima$Magnitud)
```

```
## $conv
## [1] 0
##
## $nllh
## [1] 14.50867
##
## $mle
## [1] 4.7130942 0.2637780 0.2552582
##
## $se
## [1] 0.04958329 0.04066797 0.14477193
```

```
summary(fit)
```

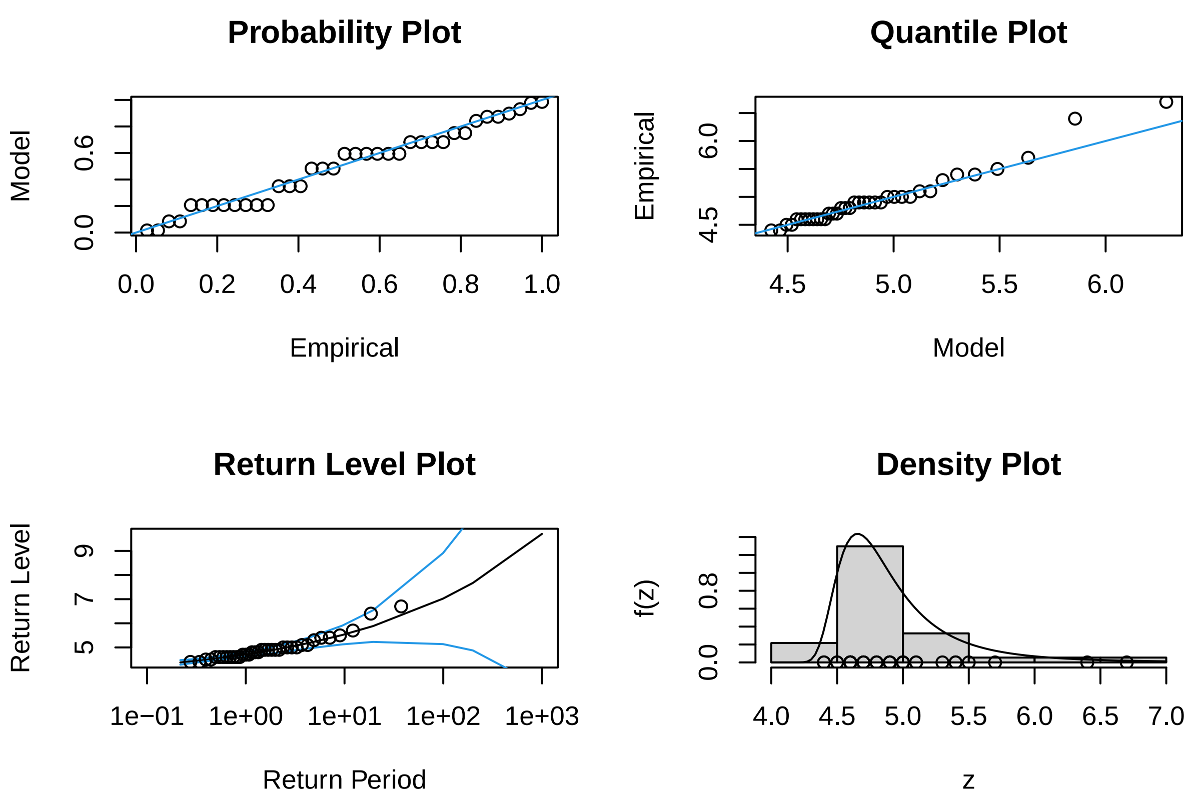
```
##       Length Class  Mode
## trans   1    -none- logical
## model   3    -none- list
## link    1    -none- character
## conv    1    -none- numeric
## nllh    1    -none- numeric
## data   37    -none- numeric
## mle     3    -none- numeric
## cov     9    -none- numeric
## se      3    -none- numeric
## vals  111    -none- numeric
```

Diagnostics

```r
V = fit$cov
round(V,5)
```

```
##          [,1]     [,2]     [,3]
## [1,]  0.00246  0.00123 -0.00225
## [2,]  0.00123  0.00165 -0.00032
## [3,] -0.00225 -0.00032  0.02096
```

```r
gev.diag(fit)
```



Predict Return Values

```r
# Return values

mu.hat <- fit$mle[1]
sigma.hat <- fit$mle[2]
xi.hat <- fit$mle[3]
V <- fit$cov
V
```

```
##               [,1]          [,2]          [,3]
## [1,]  0.002458502  0.0012309046 -0.0022528109
## [2,]  0.001230905  0.0016538840 -0.0003242836
## [3,] -0.002252811 -0.0003242836  0.0209589108
```

```
ps <- c(1/10,1/20,1/30,1/40,1/50,1/60,1/70,1/80,1/90,1/100)
Level <- 0
Linf <- 0
Lsup <- 0

for(i in 1:length(ps)){
  p <- ps[i]

  # quantile function
  return = mu.hat - (sigma.hat/xi.hat)*(1- ((-log(1-p))^(-xi.hat)))

  # delta method to estimate the SD
  nabla <- c(1,
  (-1/xi.hat)*(1-(-log(1-p))^(-xi.hat)),
  (sigma.hat/xi.hat^2)*((1-(-log(1-p))^(-xi.hat)))
  -(sigma.hat/xi.hat)*((-log(1-p))^(-xi.hat))*(log(-log(1-p))))

  sd <-sqrt(as.vector(nabla)%*%V%*%as.vector(t(nabla)))
  li <- return - 2*sd
  ls <- return + 2*sd

  Level[i] <- return
  Linf[i] <- li
  Lsup[i] <- ls
}

data.frame(Period = 1/ps, Level = round(Level,2),
Linf <- round(Linf,2), Lsup = round(Lsup,2), dif = round(Lsup-Linf,2))
```

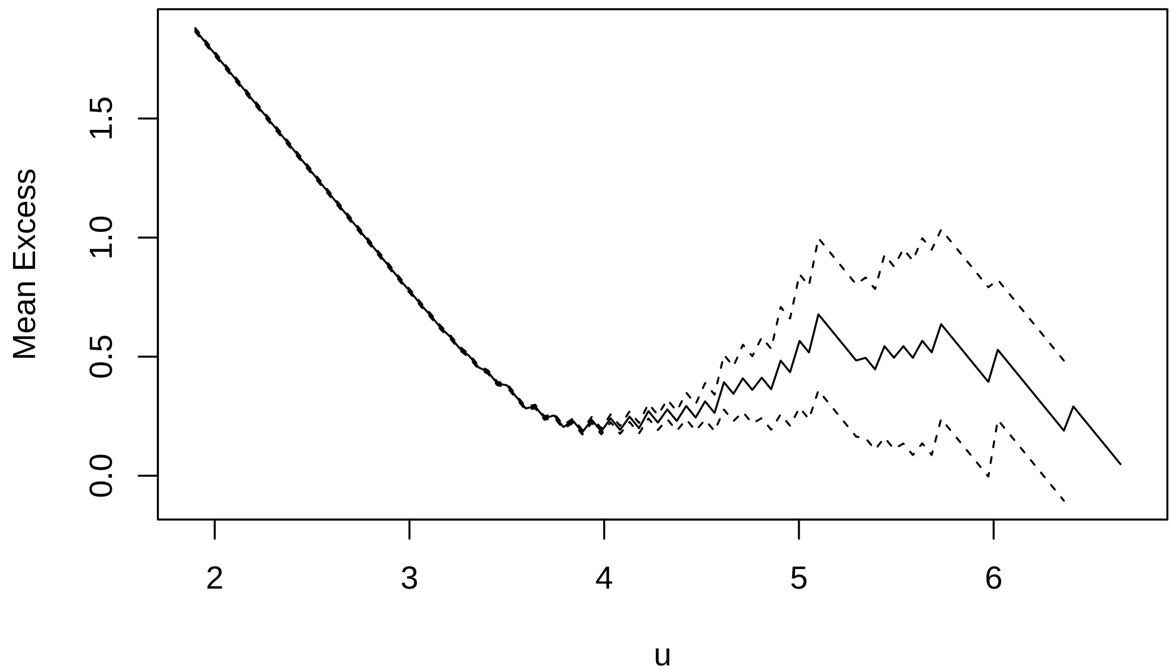
```
##    Period Level Linf....round.Linf..2. Lsup  dif
## 1      10  5.52                   5.12 5.91 0.79
## 2      20  5.89                   5.22 6.55 1.33
## 3      30  6.13                   5.24 7.02 1.78
## 4      40  6.32                   5.24 7.41 2.17
## 5      50  6.48                   5.22 7.73 2.51
## 6      60  6.61                   5.20 8.02 2.82
## 7      70  6.73                   5.18 8.28 3.10
## 8      80  6.84                   5.15 8.52 3.37
## 9      90  6.93                   5.13 8.74 3.61
## 10    100  7.02                   5.10 8.95 3.85
```

GPD Model

Selecting A Threshold

Mean Life Excess Approach:

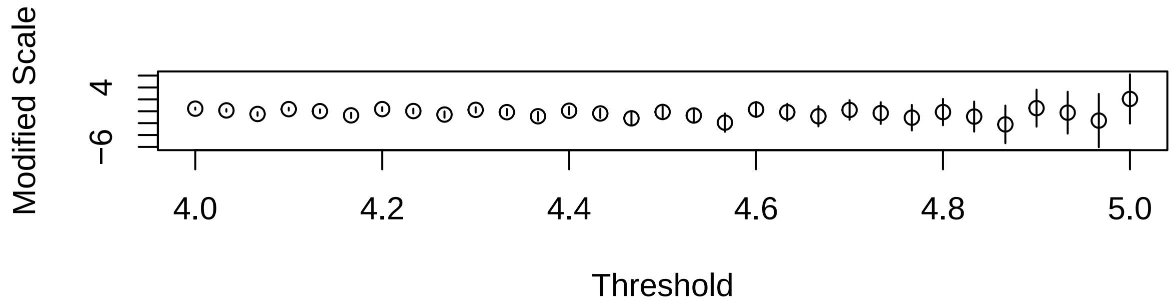
```r
mrl.plot(df$Magnitud, umin = min(df$Magnitud), umax = max(df$Magnitud), conf = .95, nint = 100)
```

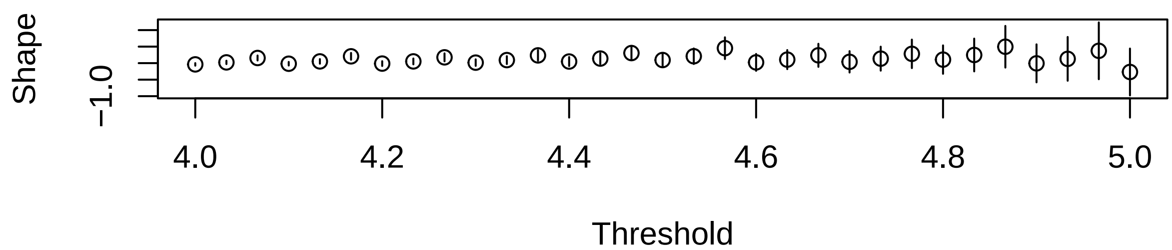


One approach is to choose u such that for x > u, the MRL plot is approximately linear. This point seems to be around 5.8, which is too extreme of a quantile for our data to create an appropriate model, since this will cause biased predictions.

Stability Plot Approach (discussed in Modelling Extremal Values)

```r
gpd.fitrange(df$Magnitud, 4, 5, nint = 31)
```



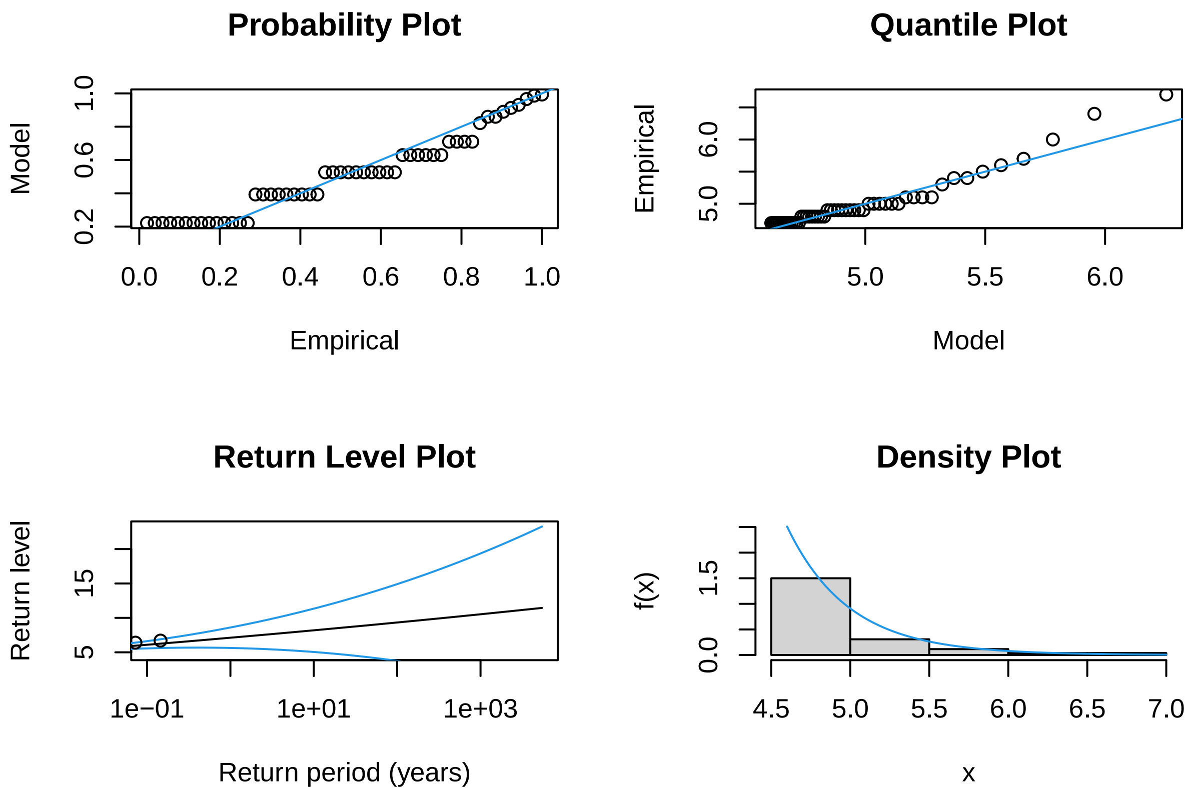


Since the estimates are fairly constant after 4.6, so we should consider u > 4.6 (taking into account that higher values will have higher bias)

```
U <- seq(4.6, 4.9, by = .1)


for (u in U){
  excedances <- df[df$Magnitud > u, ]

  fit.gpd <- gpd.fit(excedances$Magnitud,u)

  gpd.diag(fit.gpd)
}
```

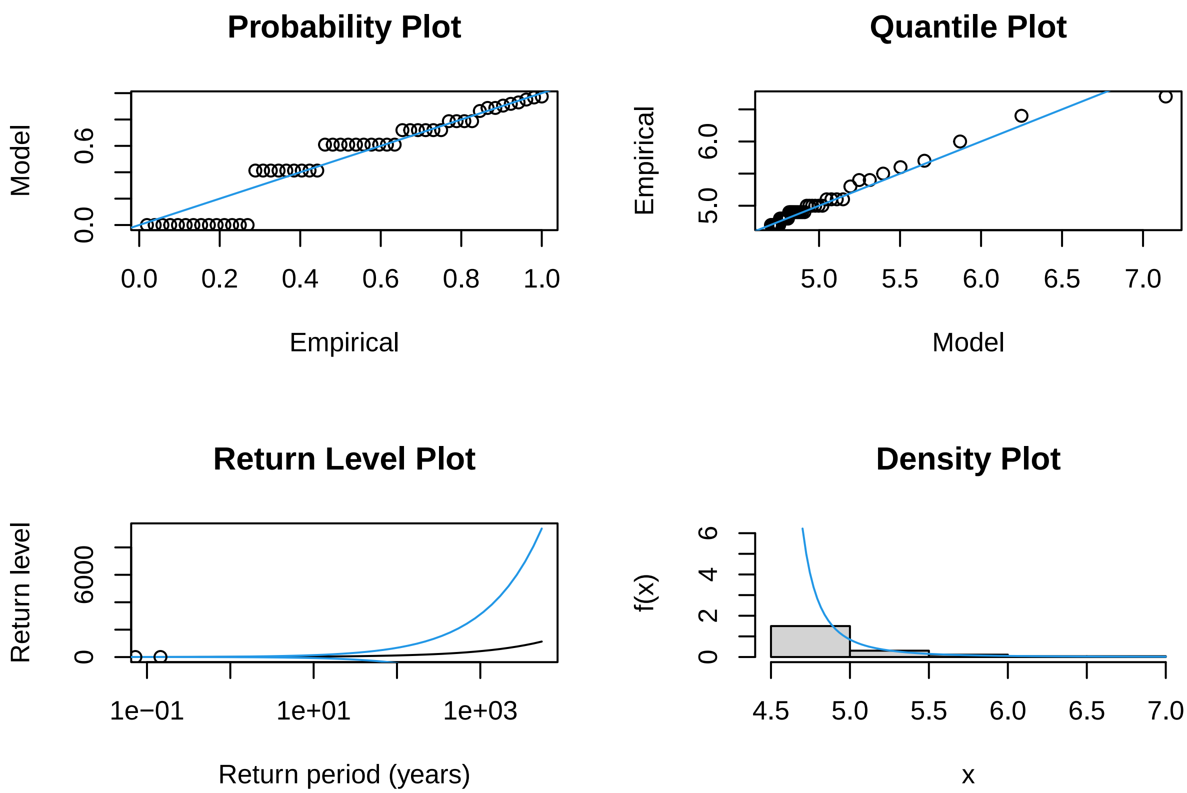
```
## $threshold
## [1] 4.6
##
## $nexc
## [1] 52
##
## $conv
## [1] 0
##
## $nllh
## [1] 5.3275
##
## $mle
```

```
## [1] 0.39839121 0.02274065
##
## $rate
## [1] 1
##
## $se
## [1] 0.07578603 0.13025635
```

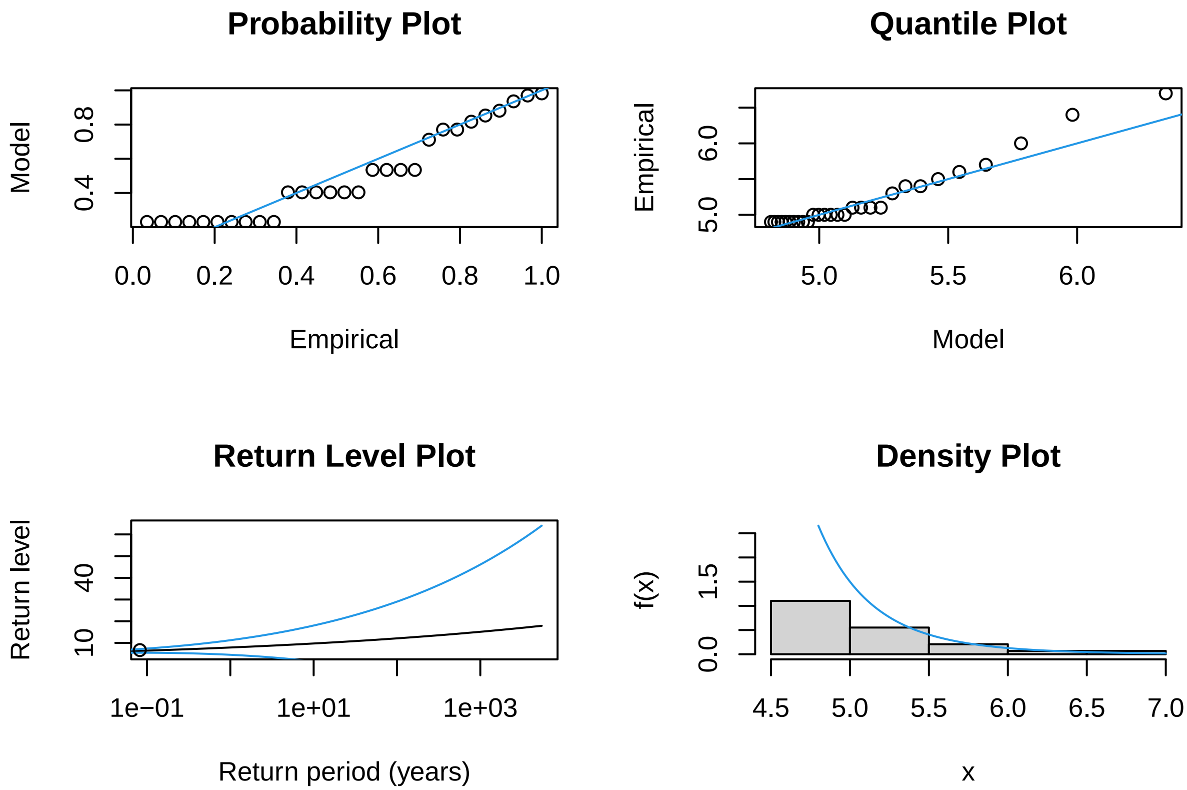



```
## $threshold
## [1] 4.7
##
## $nexc
## [1] 52
##
## $conv
## [1] 0
##
## $nllh
## [1] -13.34575
##
## $mle
## [1] 0.1606296 0.5720462
##
## $rate
## [1] 1
```

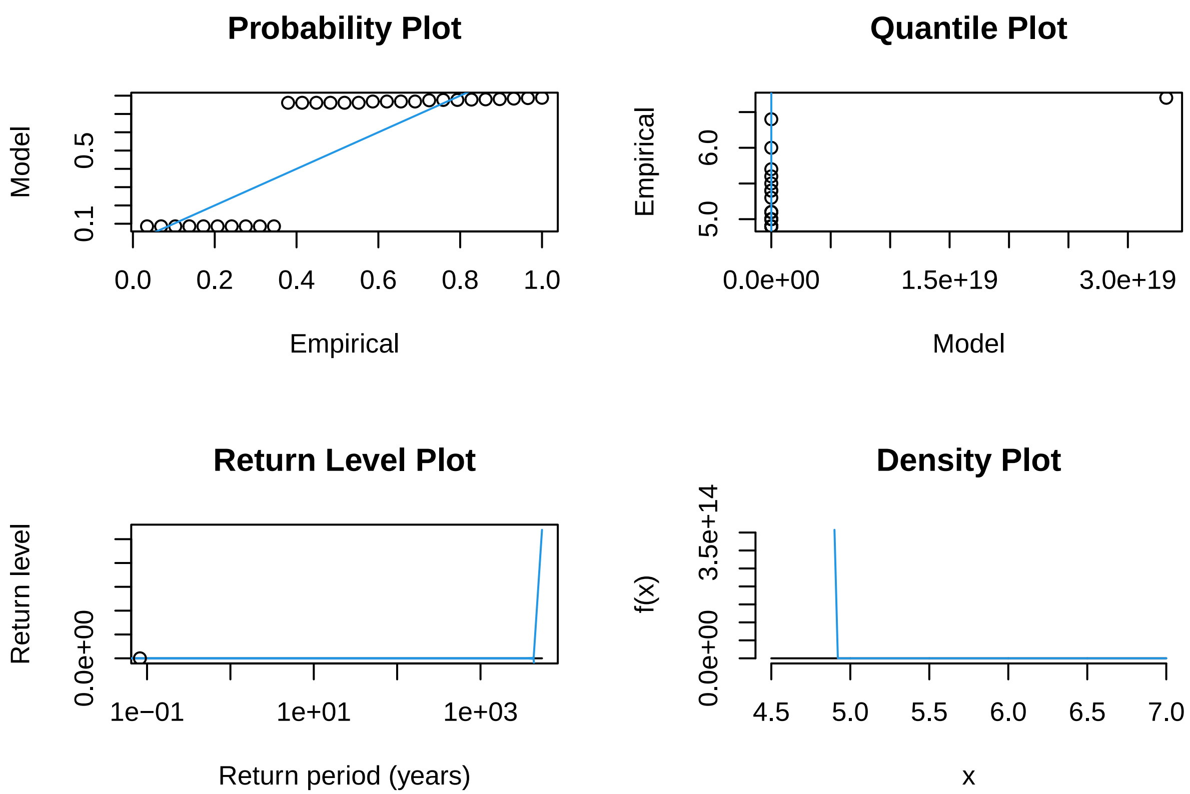
```
##
## $se
## [1] 0.0533273 0.3119333
```




```
## $threshold
## [1] 4.8
##
## $nexc
## [1] 29
##
## $conv
## [1] 0
##
## $nllh
## [1] 3.754375
##
## $mle
## [1] 0.3761995 0.1070975
##
## $rate
## [1] 1
##
## $se
## [1] 0.1078499 0.2193129
```



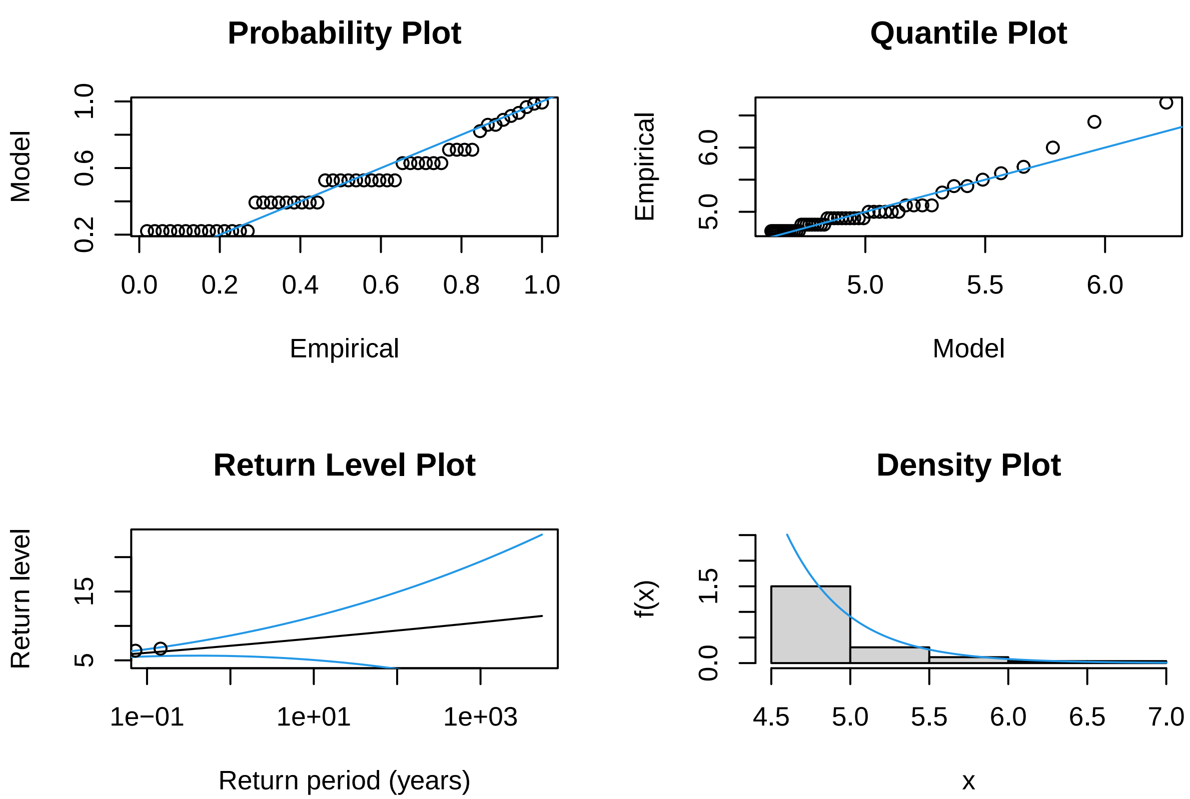

```
## $threshold
## [1] 4.9
##
## $nexc
## [1] 29
##
## $conv
## [1] 0
##
## $nllh
## [1] -246.7325
##
## $mle
## [1] 2.796977e-15 2.400284e+01
##
## $rate
## [1] 1
##
## $se
## [1] 1.999530e-06 4.556775e+00
```




```r
# Based on these diagnostics
u <- 4.6
excedances <- df[df$Magnitud > u, ]
fit.gpd <- gpd.fit(excedances$Magnitud,u)
```

```
## $threshold
## [1] 4.6
##
## $nexc
## [1] 52
##
## $conv
## [1] 0
##
## $nllh
## [1] 5.3275
##
## $mle
## [1] 0.39839121 0.02274065
##
## $rate
## [1] 1
##
## $se
## [1] 0.07578603 0.13025635
```

```r
gpd.diag(fit.gpd)
```




```r
V <- fit.gpd$cov
V
```

```
##              [,1]         [,2]
## [1,]  0.005743522 -0.006581596
## [2,] -0.006581596  0.016966717
```

```r
# Return levels
pi.hat <- fit.gpd$nexc/length(df$Magnitud) # proportion of excesses
beta.hat <- fit.gpd$mle[1]
xi.hat <- fit.gpd$mle[2]

# addint our proportion to the covariance matrix
V <- matrix(0,nrow=3,ncol=3)
V[1,1] <- pi.hat*(1-pi.hat)/length(df$Magnitud)
V[2,2] <- fit.gpd$cov[1,1]
V[3,3] <- fit.gpd$cov[2,2]
V[2,3] <- fit.gpd$cov[1,2]
V[3,2] <- fit.gpd$cov[1,2]
V
```

```
##              [,1]         [,2]         [,3]
```

```
## [1,] 1.391005e-06  0.000000000  0.000000000
## [2,] 0.000000e+00  0.005743522 -0.006581596
## [3,] 0.000000e+00 -0.006581596  0.016966717
```

```r
N <- seq(10,100,10)

ms <- N*365

Level <- 0
Linf <- 0
Lsup <- 0
for(i in 1:length(ms)){
  m <- ms[i]

  # quantile function
  return <- u + (beta.hat/xi.hat)*((m*pi.hat)^(xi.hat)-1)

  # using the delta method to estimate the SD
  nabla <- c(
    beta.hat*(m^xi.hat)*(pi.hat^(xi.hat-1)),

    (1/xi.hat)*(((m*pi.hat)^(xi.hat))-1),

    -(beta.hat/(xi.hat^2))*(((m*pi.hat)^(xi.hat))-1)+
    (beta.hat/xi.hat)*(m*pi.hat)^(xi.hat)*log(m*pi.hat))

  sd <- sqrt(as.vector(nabla)%*%V%*%as.vector(t(nabla)))
  li <- return - 2*sd
  ls <- return + 2*sd

  Level[i] <- return
  Linf[i] <- li
  Lsup[i] <- ls
}

data.frame(Period = N, Level = round(Level,2),
Linf <- round(Linf,2), Lsup = round(Lsup,2), dif = round(Lsup-Linf,2))
```

```
##    Period Level Linf....round.Linf..2. Lsup  dif
## 1      10  6.03                   5.52 6.53 1.01
## 2      20  6.33                   5.61 7.04 1.43
## 3      30  6.50                   5.64 7.37 1.73
## 4      40  6.63                   5.65 7.61 1.96
## 5      50  6.73                   5.65 7.81 2.16
## 6      60  6.81                   5.65 7.98 2.33
## 7      70  6.88                   5.64 8.13 2.49
## 8      80  6.94                   5.63 8.26 2.63
## 9      90  7.00                   5.62 8.37 2.75
## 10    100  7.04                   5.61 8.48 2.87
```