Statistical Methods for Extremal Events an Application on Danish Reinsurance Dataset

Mehrab Atighi

2025-01-10

Contents

Introduction **2**

Introduction **2**
Dataset Overview . . . 3
Loading Libraries . . . 3
Importing Data . . . 3

Exploratory Data Analysis for Extremes **4**
Summary Statistics of Dataset . . . 4
TimeSeries Plot of Data . . . 4
Histogram Plot of Data . . . 5
Histogram of Transformation data (ln) . . . 6
Mean Excess Plot of Data . . . 7
Key Observations: . . . 10
Quantile-Quantile Plot of Data with Exponential Distribution . . . 10
Quantile-Quantile Plot of Data with Pareto Distribution . . . 13
Gumbels Method of Exceedances . . . 15
The Return Period . . . 15
Records as an Exploratory Tool . . . 16
The Ratio of Maximum and Sum . . . 17

Parameter Estimation for the Generalised Extreme Value Distribution (GEV) **19**
introduction . . . 19
Maximum Likelihood Estimation . . . 19
for all data . . . 19

for Monthly maxima data(BlockMaximum) 21
Method of Probability Weighted Moments 22
introduction 22
for all data 22
for Monthly maxima data(BlockMaximum) 23

Estimating Under Maximum Domain of Attraction **24**
Pickands's Estimator 24
Fitting the GPD 25
Introduction 25
Maximum Likelihood Estimation 25

Conclusion **27**

Introduction

Introduction

Extreme Value Theory (EVT) is a vital field in statistics that focuses on modeling and analyzing the tails of probability distributions, particularly to understand the behavior of extreme events. This theory plays a crucial role in applications where rare, high-impact events are of interest, such as finance, insurance, environmental studies, and engineering.

In the insurance industry, the modeling of extreme losses is critical for risk assessment and pricing. The **Danish Reinsurance Dataset**, which contains large losses from the reinsurance market in Denmark, is a widely studied dataset in the field of EVT. Its heavy-tailed nature makes it an ideal case study for tail index estimation, allowing researchers and practitioners to quantify the likelihood of extreme insurance claims.

The tail index of a distribution is a key parameter that characterizes the heaviness of its tail. A heavier tail indicates a higher likelihood of extreme values, which has significant implications for risk management. Several estimators have been proposed for this purpose, each with unique assumptions and strengths. In this project, we focus on two widely used estimators:

1. **Hill Estimator**: A classical and popular method for estimating the tail index, particularly suited for heavy-tailed distributions.
2. **Pickands Estimator**: A robust approach that uses specific order statistics to calculate the tail index.

This project aims to: - Explore the theoretical foundations of the Hill and Pickands estimators. - Implement these methods using R. - Apply the estimators to the Danish Reinsurance Dataset. - Compare the performance and reliability of the estimators in capturing the tail behavior of the dataset.

By analyzing the Danish Reinsurance Dataset, this project seeks to provide practical insights into the effectiveness of the Hill and Pickands estimators in real-world settings. The findings will contribute to a deeper understanding of EVT and its applications in risk management for the insurance industry.

Dataset Overview

- **Univariate (danishuni)**: Contains two columns:
  - `Date`: The date of claim occurrence.
  - `Loss`: The total loss amount in mDKK.

All columns are numeric except the `Date` columns, which are of class Date.

Loading Libraries

At the first we are going to loading needed packages in R.

```
library(ggplot2)
library(dplyr)
library(fitdistrplus)
```

Importing Data

Danish reinsurance data are available in fitdistrplus package. now we are loading the data and we can see top 5 rows of that.

```
# Load data
data(danishuni, package = "fitdistrplus")
head(danishuni)
```

```
##         Date     Loss
## 1 1980-01-03 1.683748
## 2 1980-01-04 2.093704
## 3 1980-01-05 1.732581
## 4 1980-01-07 1.779754
## 5 1980-01-07 4.612006
## 6 1980-01-10 8.725274
```

Exploratory Data Analysis for Extremes

Summary Statistics of Dataset

Now we are going to see summary of the dataset.

```
summary(danishuni)
```

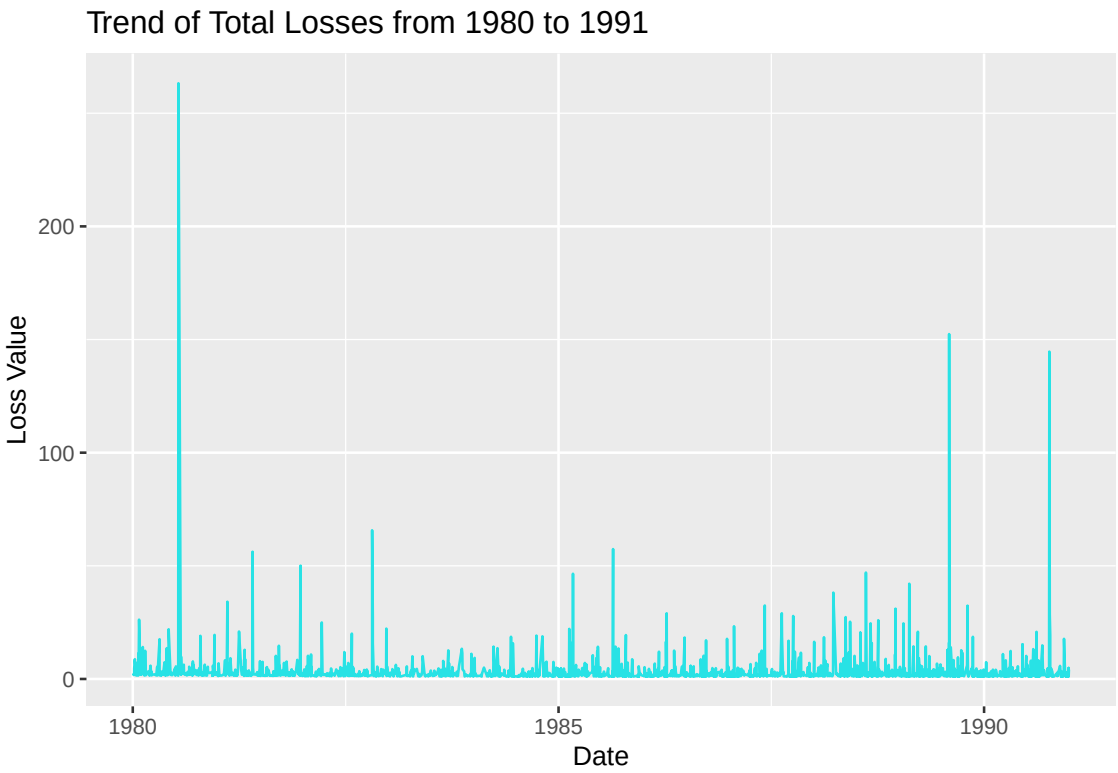
```
##       Date                 Loss        
##  Min.   :1980-01-03   Min.   :  1.000  
##  1st Qu.:1983-03-19   1st Qu.:  1.321  
##  Median :1986-03-03   Median :  1.778  
##  Mean   :1985-11-19   Mean   :  3.385  
##  3rd Qu.:1988-07-08   3rd Qu.:  2.967  
##  Max.   :1990-12-31   Max.   :263.250
```

According to the above table we can see we have two column which the Loss column as the significant different between Quantile 0.75 and Maximum it's heavy tail (i think that).

TimeSeries Plot of Data

Now we want to see the Loss Values since 1980 to 1991.

```
ggplot(danishuni, aes(x = Date , y = Loss)) +
  geom_line(binwidth = 3 , color = 85) +
  labs(title = "Trend of Total Losses from 1980 to 1991", x = "Date", y = "Loss Va
```

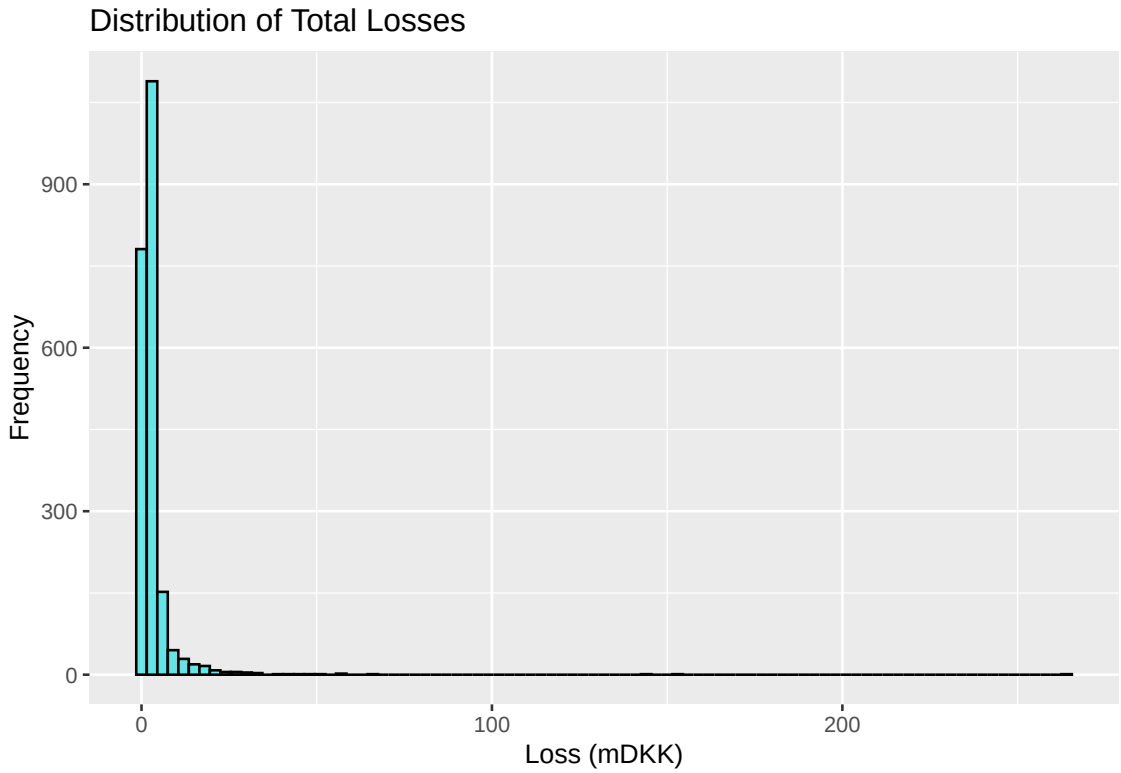


According to the above figure we can see that we had three times high loss values which are more than 100.

Histogram Plot of Data

Now we want to see the histogram plot of Data (Loss Values) to recive an interview of loss density.

```
ggplot(danishuni, aes(x = Loss)) +
  geom_histogram(binwidth = 3, fill = 85, alpha = 0.7 , color = "black") +
  labs(title = "Distribution of Total Losses", x = "Loss (mDKK)", y = "Frequency")
```

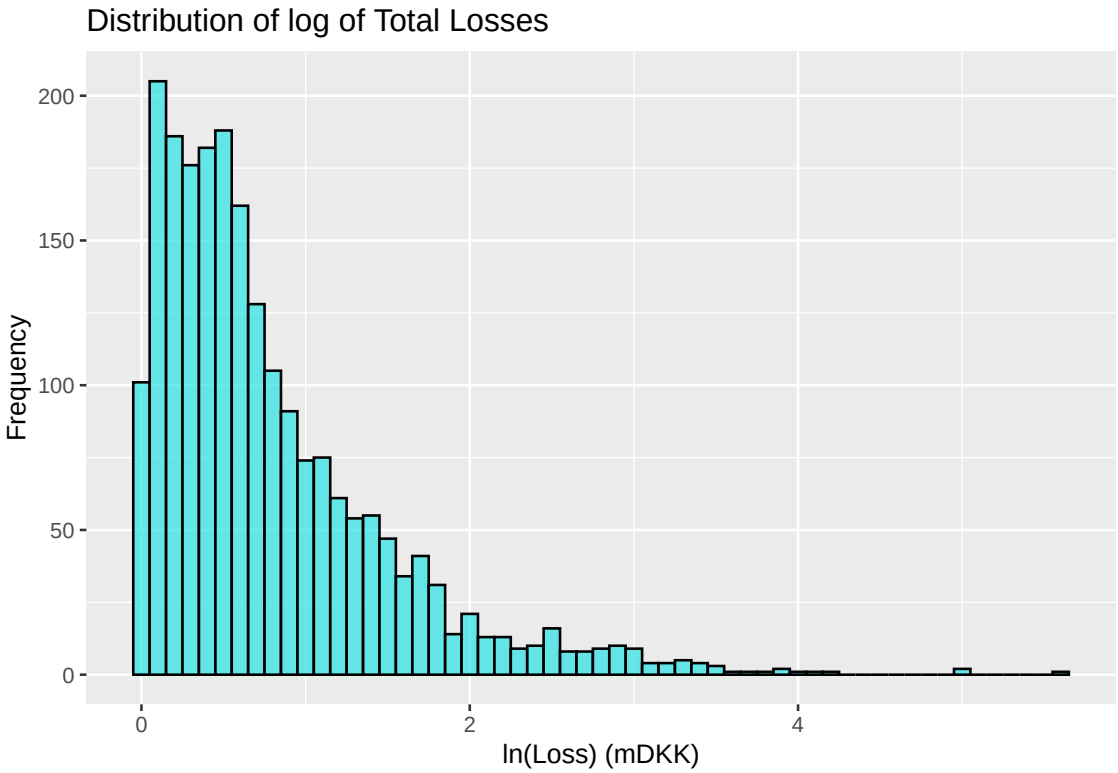


According to the above Figure we can see that Loss values following very heavy tail distribution. it means that we can see very huge loss values in the histogram at right side of that.

Histogram of Transformation data (ln)

for better interview we can use a transformation like ln from the loss values, then we can again draw the histogram plot. in the following figure we can see that.

```
ggplot(danishuni, aes(x = log(Loss))) +
  geom_histogram(binwidth = 0.1, fill = 85, alpha = 0.7 , color = "black") +
  labs(title = "Distribution of log of Total Losses", x = "ln(Loss) (mDKK)", y = "
```



Like as the first histogram we can see again a heavy tail distribution for ln(loss). it means that the loss distributions is probably pareto or exponential.

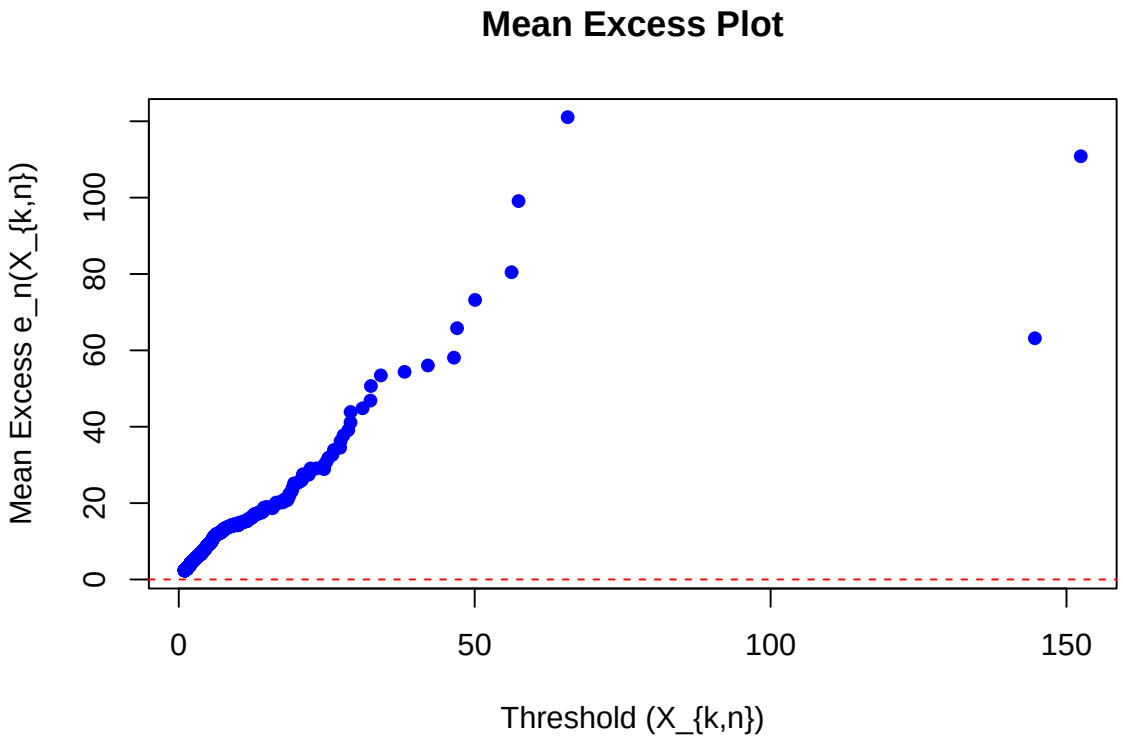
Mean Excess Plot of Data

According to the histogram of the Loss values. we are going to draw Mean Excess Plot which we can see that:(attention that i found two code for this drawing. but i use on of them and the other one is commented in my codes.)

```
# Load the dataset
# Assuming your dataset is a data frame with columns 'date' and 'loss'
# Example of loading a dataset:
library(fitdistrplus)
```

```
data("danishuni")

# Extract the 'loss' column
loss_data <- danishuni$Loss

# Sort the loss data
n <- length(loss_data)
```

```r
sorted_data <- sort(loss_data)

# Function to calculate the mean excess for a given k
mean_excess <- function(data, k) {
  threshold <- data[k]
  excesses <- data[(k+1):n] - threshold
  return(mean(excesses, na.rm = TRUE))
}

# Compute the mean excess values for each k
mean_excess_values <- sapply(1:(n-1), function(k) mean_excess(sorted_data, k))

# Prepare points for the plot
x_points <- sorted_data[1:(n-1)] # X_{k,n}
y_points <- mean_excess_values   # e_n(X_{k,n})

# Create the Mean Excess Plot
plot(x_points, y_points, type = "p", pch = 16, col = "blue",
     xlab = "Threshold (X_{k,n})",
     ylab = "Mean Excess e_n(X_{k,n})",
     main = "Mean Excess Plot"
    #, xlim = c(0 , 70)
     )
abline(h = 0, col = "red", lty = 2)
```




```
# library(fExtremes)
# mePlot(danishuni$Loss)
```

The Mean Excess Plot you provided shows the behavior of the **mean excess function** for varying thresholds. Here's an analysis of the plot:

1. **Axes Explanation**:
   - The **x-axis** represents the threshold ($X_{k,n}$), which is the value above which we calculate the mean excess.
   - The **y-axis** represents the **mean excess values** ($e_n(X_{k,n})$), which are the average values of data points exceeding each threshold.
2. **Shape of the Curve**:
   - The plot starts with a rising trend at lower thresholds. This indicates that as we increase the threshold, the average excess values also increase.
   - Beyond a certain threshold (approximately $X \approx 50$), the plot becomes nearly linear, which suggests that the data beyond this point might follow a **Generalized Pareto Distribution (GPD)**.
   - At higher thresholds (e.g., $X > 100$), there are fewer points, and the variability increases due to fewer observations exceeding these thresholds.
3. **Flatness of the Tail**:
   - If the mean excess plot exhibits a linear behavior beyond a specific threshold, it confirms the suitability of the GPD for modeling the tail.

- The almost straight-line behavior in the range $50 < X < 100$ supports the GPD assumption.

4. **Outliers**:
   - The last few points on the plot (e.g., $X > 120$) deviate significantly, which could be outliers or due to the sparsity of data in this extreme region.

Key Observations:

- **Threshold Selection**:
  - A threshold of around 50 appears reasonable because the mean excess function becomes nearly linear from this point onward.
  - Thresholds lower than this may include non-tail behavior, which could bias the tail analysis.
- **Heaviness of the Tail**:
  - The increasing trend of the mean excess function indicates a **heavy-tailed distribution**. This is typical of datasets in fields like insurance and finance, where extreme values (large losses) are common.
- **Practical Implication**:
  - Selecting the threshold around $X = 50$ would allow you to focus on the extremes while maintaining enough data points for reliable parameter estimation.

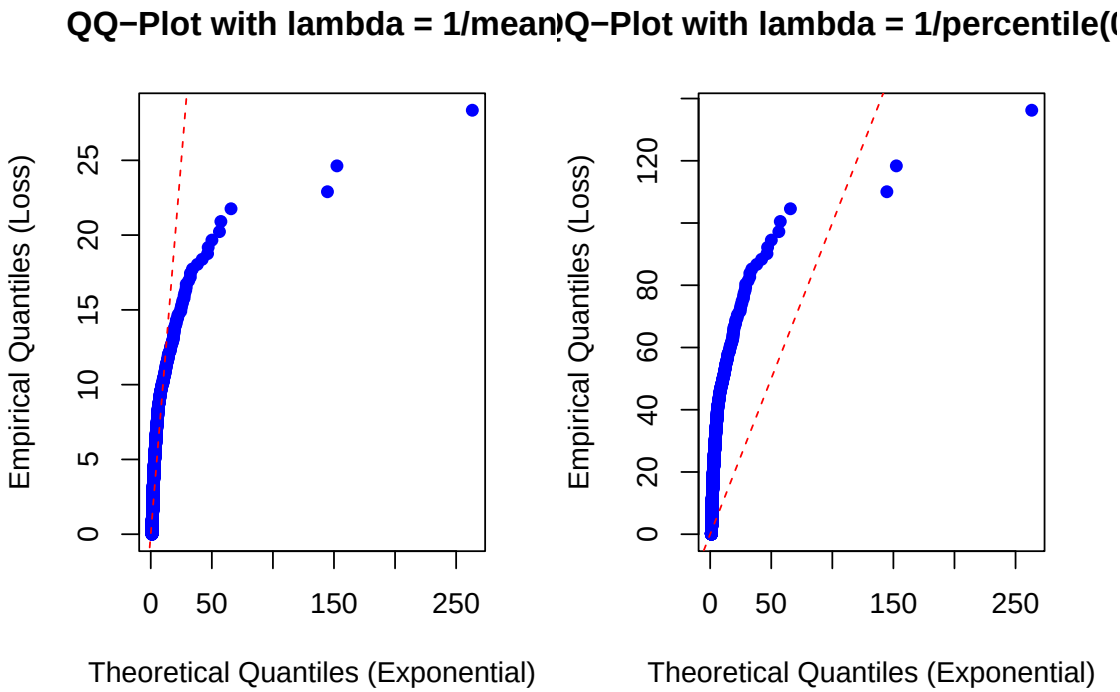
Quantile-Quantile Plot of Data with Exponential Distribution

According to the our goal, we need to draw the QQ plot of Loss values with heavy tail distributions, for example here we do this work with exponential distribution. but you should attention that we set two lambda here, the first one is 1/mean(loss values) and the second is 1/Quantile 0.975 of loss values so we have:

```
# Extract the 'loss' column
library(fitdistrplus)
data("danishuni")
```

```
loss_data <- danishuni$Loss

# Sort the loss data
sorted_loss <- sort(loss_data)

# Generate theoretical quantiles for an exponential distribution
lambda1 <- 1/mean(loss_data)#quantile(loss_data , probs = 0.975) # Estimate rate
```

```r
lambda2 <- 1/quantile(loss_data , probs = 0.975) # Estimate rate parameter (lambda

n <- length(sorted_loss)
exp_quantiles1 <- qexp(ppoints(n), rate = lambda1)  # Theoretical quantiles
exp_quantiles2 <- qexp(ppoints(n), rate = lambda2)  # Theoretical quantiles

# Create the QQ-plot
par(mfrow = c(1,2))
qqplot(y = exp_quantiles1, x = sorted_loss,
       main = "QQ-Plot with lambda = 1/mean",
       xlab = "Theoretical Quantiles (Exponential)",
       ylab = "Empirical Quantiles (Loss)",
       pch = 16, col = "blue")
# Add a 45-degree reference line
abline(0, 1, col = "red", lty = 2)
#add second plot
qqplot(y = exp_quantiles2, x = sorted_loss,
       main = "QQ-Plot with lambda = 1/percentile(0.975)",
       xlab = "Theoretical Quantiles (Exponential)",
       ylab = "Empirical Quantiles (Loss)",
       pch = 16, col = "blue")
# Add a 45-degree reference line
abline(0, 1, col = "red", lty = 2)
```



QQ-Plot with Lambda = 1/Mean The left QQ-Plot shows the empirical quantiles of the loss data against the theoretical quantiles of an exponential distribution with a lambda equal to the inverse of the mean of the data. In this plot:

The data points should ideally follow the 1-1 line if the data fits the exponential distribution well.

The regression line provides a visual indication of the overall trend.

The 95% confidence bar helps visualize the variability around the theoretical quantiles.

In this case, if the data points deviate significantly from the 1-1 line, especially at higher quantiles, it suggests that the loss data has a heavier tail than what would be expected under an exponential distribution with this lambda value.

QQ-Plot with Lambda = 1/Percentile(0.975) The right QQ-Plot uses a lambda value set to the inverse of the 99th percentile of the data. This can sometimes provide a better fit for the extreme values. In this plot:

Similarly, data points should follow the 1-1 line for a good fit.

The regression line and 95% confidence bar provide additional context.

Again, if the data points, particularly at the higher end, deviate from the 1-1 line, it indicates that the loss data has a heavy tail, which means it has higher probabilities for extreme values compared to the exponential distribution with the given lambda.

So we can say that:\ From both plots, if you observe a consistent pattern where the empirical quantiles are above the theoretical quantiles at higher values, it indeed suggests that your data has a heavy tail. This means that extreme losses are more frequent in your dataset than what would be expected from a simple exponential distribution.

Heavy tails are common in financial and insurance datasets, and they often require distributions that can model extreme events more accurately, such as the Generalized Pareto Distribution (GPD) or the Generalized Extreme Value (GEV) distribution.

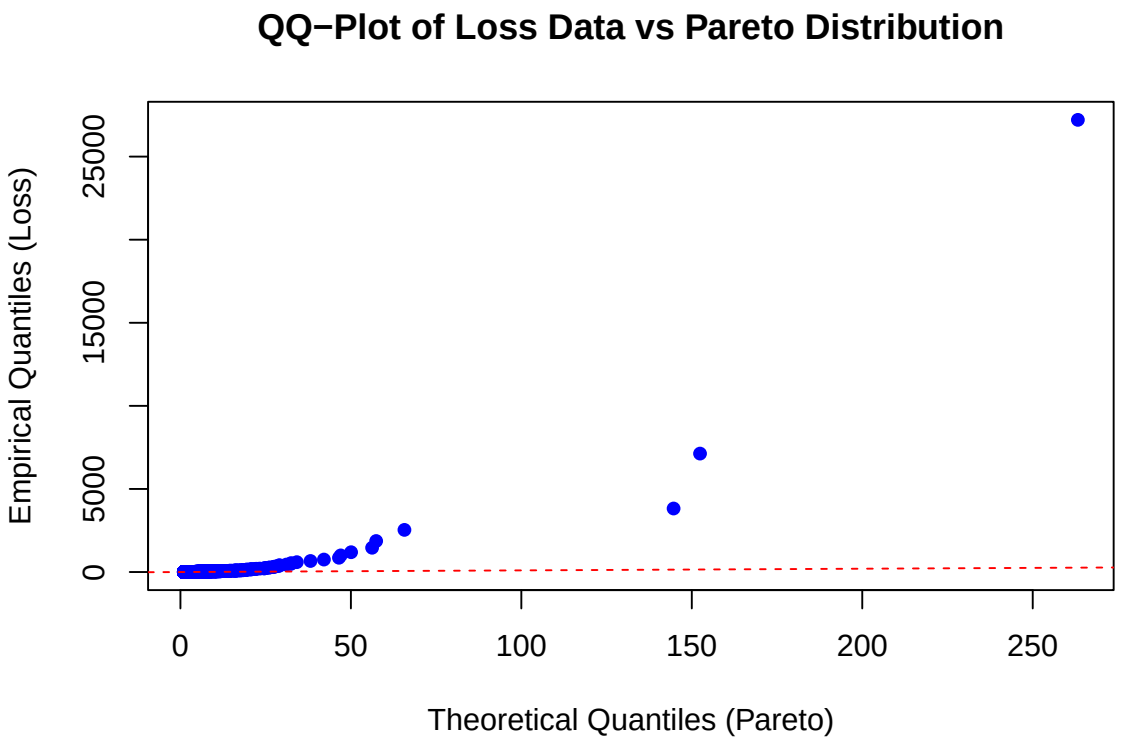
Quantile-Quantile Plot of Data with Pareto Distribution

According to the our goal, we need to draw the QQ plot of Loss values with more heavy tail distributions, for example here we do this work with pareto distribution which the sigma and alpha parameter is set as bottem. so we have:

```r
# Load the dataset
library(fitdistrplus)
data("danishuni")
```

```r
loss_data <- danishuni$Loss

# Sort the loss data
sorted_loss <- sort(loss_data)
n <- length(sorted_loss)

# Estimate Pareto parameters
# We'll estimate the scale (sigma) and shape (alpha) using the method of moments
sigma <- min(loss_data)  # Scale parameter (minimum value of the data)
alpha <- 1 / (log(mean(loss_data / sigma)))  # Shape parameter

# Generate theoretical quantiles for the Pareto distribution
pp <- ppoints(n)  # Proportions for quantiles
pareto_quantiles <- sigma * (1 - pp)^(-1 / alpha)  # Theoretical quantiles

# Create the QQ-plot
par(mfrow = c(1,1))
qqplot(y = pareto_quantiles, x = sorted_loss,
       main = "QQ-Plot of Loss Data vs Pareto Distribution",
       xlab = "Theoretical Quantiles (Pareto)",
       ylab = "Empirical Quantiles (Loss)",
```

```
       pch = 16, col = "blue")
# Add a 45-degree reference line
abline(0, 1, col = "red", lty = 2)
```



From examining the QQ-Plot, there are a few key points we can discuss about the data and its fit to the Pareto distribution:\ The data points closely follow the 1-1 line at the lower quantiles. This indicates a **good fit to the Pareto distribution in the lower quantile range, meaning that the majority of your loss data aligns well with this distribution**. At the higher quantiles, however, the data points begin to deviate significantly from the 1-1 line. This suggests that the Pareto distribution may not adequately capture the extreme values in your dataset. The deviations at higher quantiles confirm that your dataset has a heavy tail, where extreme losses occur more frequently than would be expected under the Pareto distribution. This is an important characteristic in risk management and financial modeling. Given the heavy tail in your data, you might consider fitting your data to distributions specifically designed to handle extreme values, such as the Generalized Pareto Distribution (GPD) or the Generalized Extreme Value (GEV) distribution.

Gumbels Method of Exceedances

```
library(dplyr)
n = nrow(danishuni)
r = 25
j = 0:9
df = data.frame(probability = 0:9)
k_max = length(which(danishuni$Loss >= quantile(danishuni$Loss , 0.95)))

for(k in 1:k_max){
  df1 = data.frame(K = round( choose((r + n - k - j) , (n-k) ) * choose((j + k - 1
  df <- bind_cols(df , df1)

}
df = df[,-1]
```

```
rownames(df) = c(paste0("j = " , 0:9))
colnames(df) = c(paste0("k = " , 1:k_max))
df[1:9,1:5]
```

```
##         k = 1  k = 2  k = 3  k = 4  k = 5
## j = 0 0.9886 0.9773 0.9662 0.9551 0.9442
## j = 1 0.0113 0.0223 0.0331 0.0437 0.0540
## j = 2 0.0001 0.0004 0.0007 0.0012 0.0018
## j = 3 0.0000 0.0000 0.0000 0.0000 0.0000
## j = 4 0.0000 0.0000 0.0000 0.0000 0.0000
## j = 5 0.0000 0.0000 0.0000 0.0000 0.0000
## j = 6 0.0000 0.0000 0.0000 0.0000 0.0000
## j = 7 0.0000 0.0000 0.0000 0.0000 0.0000
## j = 8 0.0000 0.0000 0.0000 0.0000 0.0000
```

The Return Period

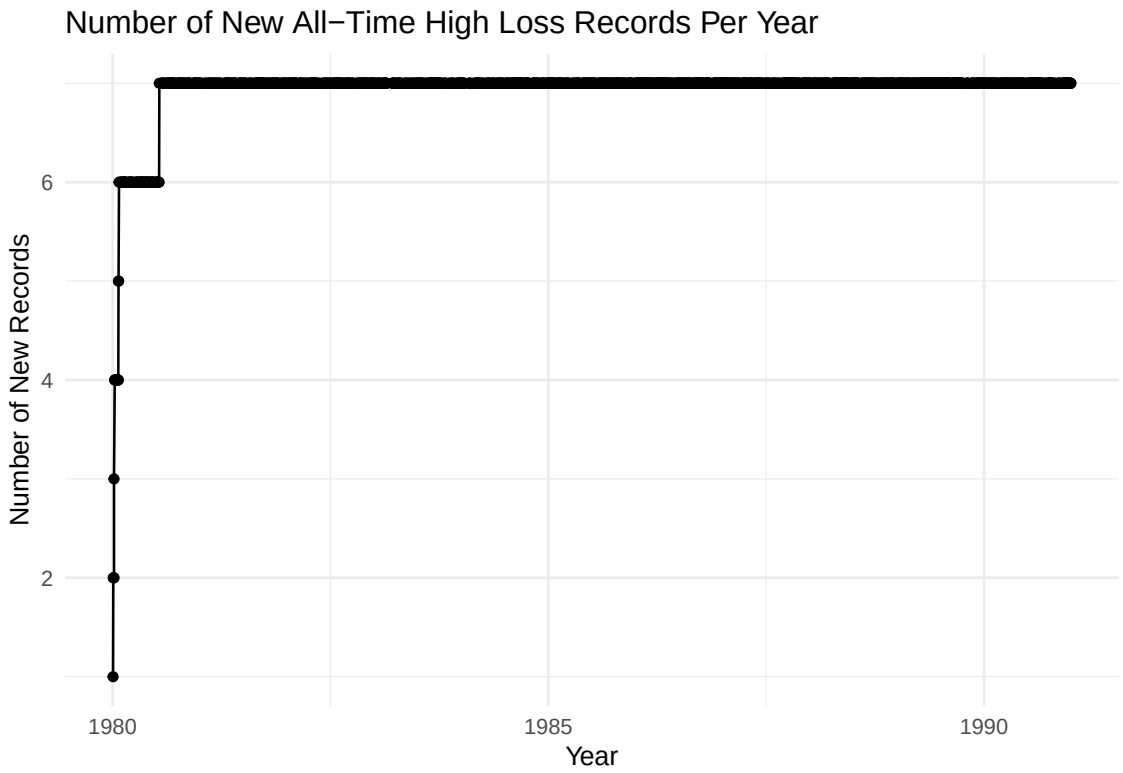
```
k = 100
u = 10.58 # or you can use this: sort(danishuni$Loss , decreasing = T)[k]
p = as.numeric(substr(names(which.min(abs(u - quantile(danishuni$Loss , probs = se
r_k = sum((1 - p)^(1:(k-1))) * p
r_k
```

```
## [1] 0.046
```

Records as an Exploratory Tool

```r
# Load necessary libraries
library(ggplot2)
```

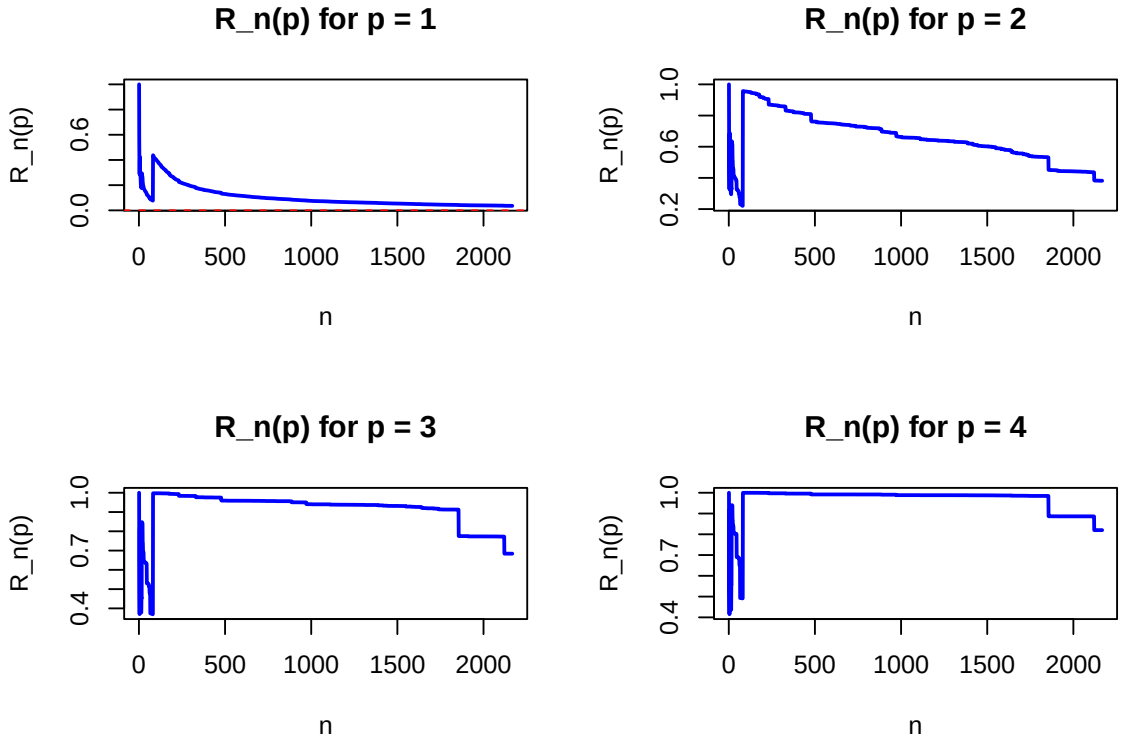
```r
# Ensure your date column is in Date format (replace 'Date' with the actual colum

# Find new all-time high loss values
data <- danishuni %>%
  arrange(Date) %>%
  mutate(is_new_record = cumsum(Loss == cummax(Loss)))

# Count the number of new records per year
new_records_per_year <- data %>%
  filter(is_new_record == 1) %>%
  mutate(year = format(Date, "%Y")) %>%
  group_by(year) %>%
  summarise(count = n())

# Plot the number of new records per year
ggplot(data, aes(x = Date, y = is_new_record)) +
  geom_line() +
  geom_point() +
  labs(title = "Number of New All-Time High Loss Records Per Year",
       x = "Year",
       y = "Number of New Records") +
  theme_minimal()
```



The Ratio of Maximum and Sum

```r
loss_data <- danishuni$Loss          # Extract the loss column
n <- length(loss_data)               # Number of observations

# Values of p to consider
p_values <- seq(1,4 , 1)

# Preallocate storage for R_n(p)
R_n_matrix <- matrix(NA, nrow = n, ncol = length(p_values))
colnames(R_n_matrix) <- paste0("p=", p_values)

# Calculate R_n(p) for each value of p
for (j in seq_along(p_values)) {
  p <- p_values[j]
  abs_loss_p <- abs(loss_data)^p

  for (i in 1:n) {
    S_n <- sum(abs_loss_p[1:i])       # Sum up to the i-th element
```

```r
    M_n <- max(abs_loss_p[1:i])        # Max up to the i-th element
    R_n_matrix[i, j] <- M_n / S_n      # Ratio
  }
}

# Plot R_n(p) against n for each p
par(mfrow = c((round(length(p_values)/2 , 0)), 2))  # One plot for each p
for (j in seq_along(p_values)) {
  plot(1:n, R_n_matrix[, j], type = "l",
       main = paste("R_n(p) for p =", p_values[j]),
       xlab = "n", ylab = "R_n(p)", col = "blue", lwd = 2)
  abline(h = 0, col = "red", lty = 2) # Add reference line at 0
}
```



Parameter Estimation for the Generalised Extreme Value Distribution (GEV)

introduction

```
library(fitdistrplus)
library(ismev)       # For GEV functions
library(extRemes)    # Additional extreme value analysis tools
library(EnvStats)
```

Maximum Likelihood Estimation

for all data

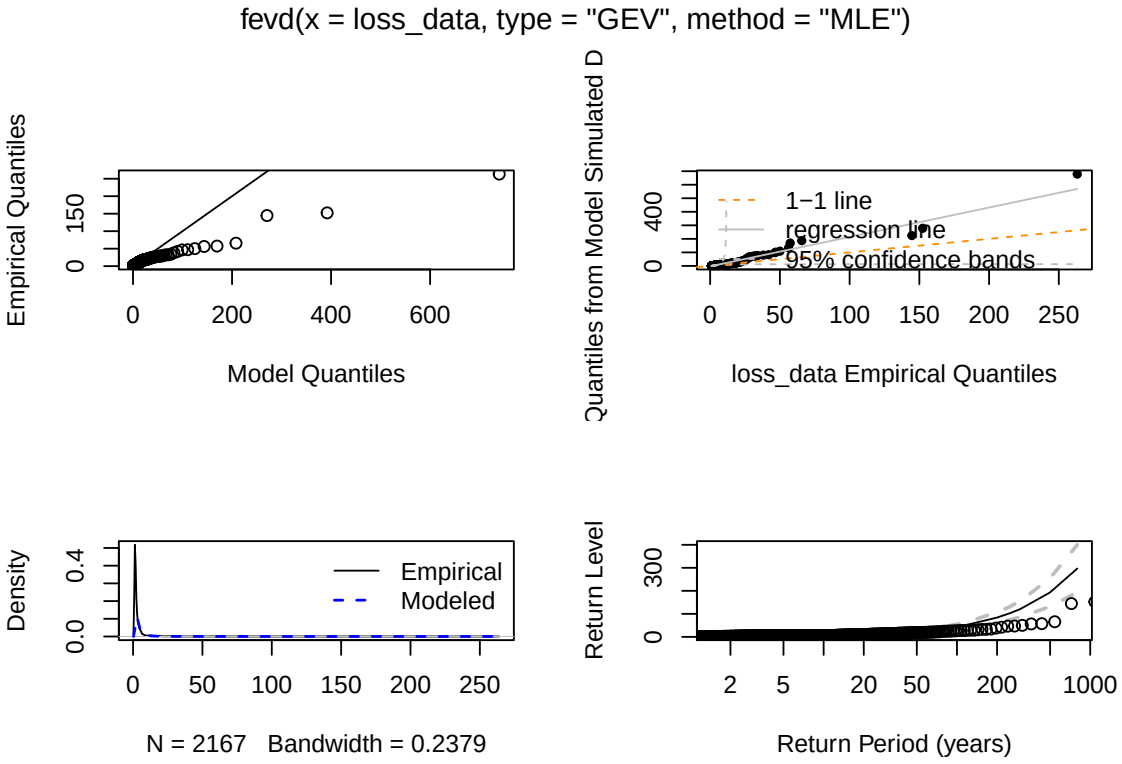
```
library(extRemes)    # Additional extreme value analysis tools
```

```
#GEV ON ALL DATA WITH MLE method
loss_data <- danishuni$Loss
gev_fit_all_data_mle <- fevd(loss_data, method = "MLE", type = "GEV")
summary(gev_fit_all_data_mle)
```

```
##
## fevd(x = loss_data, type = "GEV", method = "MLE")
##
## [1] "Estimation Method used: MLE"
##
##
##  Negative Log-Likelihood Value:  3392.418
##
##
##  Estimated parameters:
##  location     scale     shape
## 1.4833406 0.5928882 0.9165839
##
##  Standard Error Estimates:
##   location      scale      shape
## 0.01507648 0.01865780 0.03034083
##
##  Estimated parameter covariance matrix.
##               location        scale         shape
```

```
## location  0.0002273004 2.389179e-04 -1.135461e-04
## scale     0.0002389179 3.481137e-04  9.290683e-05
## shape    -0.0001135461 9.290683e-05  9.205658e-04
##
##  AIC = 6790.835
##
##  BIC = 6807.878
```

```
plot(gev_fit_all_data_mle)
```




```
return_level <- return.level(gev_fit_all_data_mle, return.period = 100)
print(return_level)
```

```
## fevd(x = loss_data, type = "GEV", method = "MLE")
## get(paste("return.level.fevd.", newcl, sep = ""))(x = x, return.period = return
##
##  GEV model fitted to  loss_data
## Data are assumed to be  stationary
## [1] "Return Levels for period units in years"
## 100-year level
##       44.68651
```

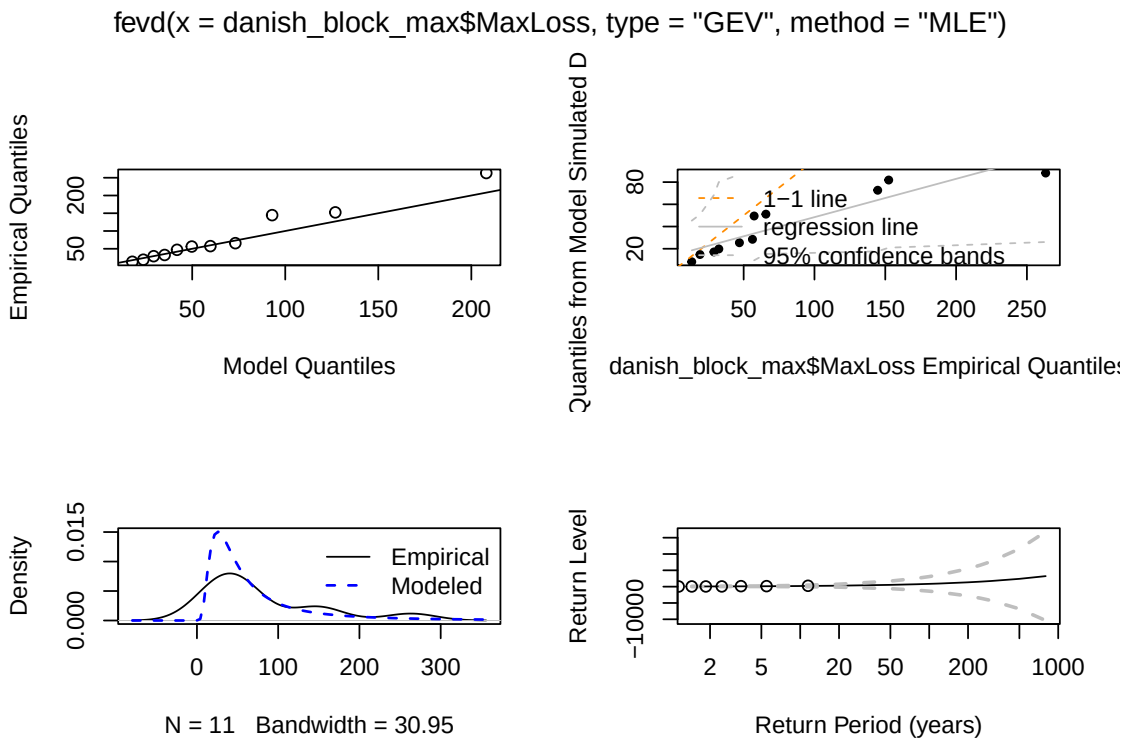
for Monthly maxima data(BlockMaximum)

```
#GEV ON BLOCK MAX DATA with MLE method
library(dplyr)
```

```
danish_block_max <- danishuni %>%
  group_by(year = lubridate::year(Date)) %>%
  summarise(MaxLoss = max(Loss))
gev_fit_blockMax_data_mle <- fevd(danish_block_max$MaxLoss, method = "MLE"   , type
summary(gev_fit_blockMax_data_mle)
```

```
##
## fevd(x = danish_block_max$MaxLoss, type = "GEV", method = "MLE")
##
## [1] "Estimation Method used: MLE"
##
##
##  Negative Log-Likelihood Value:  58.23331
##
##
##  Estimated parameters:
##   location      scale      shape
## 37.8259460 28.9673297  0.6380642
##
##  Standard Error Estimates:
##   location      scale      shape
## 10.7251910 11.0547413  0.4140545
##
##  Estimated parameter covariance matrix.
##            location       scale      shape
## location 115.029722  95.6378074 -1.6604664
## scale     95.637807 122.2073057 -0.1589988
## shape     -1.660466  -0.1589988  0.1714411
##
##  AIC = 122.4666
##
##  BIC = 123.6603
```

```
plot(gev_fit_blockMax_data_mle)
```




```
return_level <- return.level(gev_fit_blockMax_data_mle, return.period = 100)
print(return_level)
```

```
## fevd(x = danish_block_max$MaxLoss, type = "GEV", method = "MLE")
## get(paste("return.level.fevd.", newcl, sep = ""))(x = x, return.period = return
##
##  GEV model fitted to  danish_block_max$MaxLoss
## Data are assumed to be  stationary
## [1] "Return Levels for period units in years"
## 100-year level
##       847.063
```

Method of Probability Weighted Moments

introduction

some information

for all data

```
loss_data <- danishuni$Loss
gev_fit_all_data_pwm <-egevd(loss_data, method = "pwme")
```

```
gev_fit_all_data_pwm
```

```
##
## Results of Distribution Parameter Estimation
## ---------------------------------------------
##
## Assumed Distribution:            Generalized Extreme Value
##
## Estimated Parameter(s):          location =  1.5551821
##                                  scale    =  0.7156472
##                                  shape    = -0.6709899
##
## Estimation Method:               Unbiased pwme
##
## Data:                            loss_data
##
## Sample Size:                     2167
```

for Monthly maxima data(BlockMaximum)

```
#GEV ON BLOCK MAX DATA WITH PWM method
gev_fit_BlockMax_pwm <-egevd(danish_block_max$MaxLoss, method = "pwme")
gev_fit_BlockMax_pwm
```

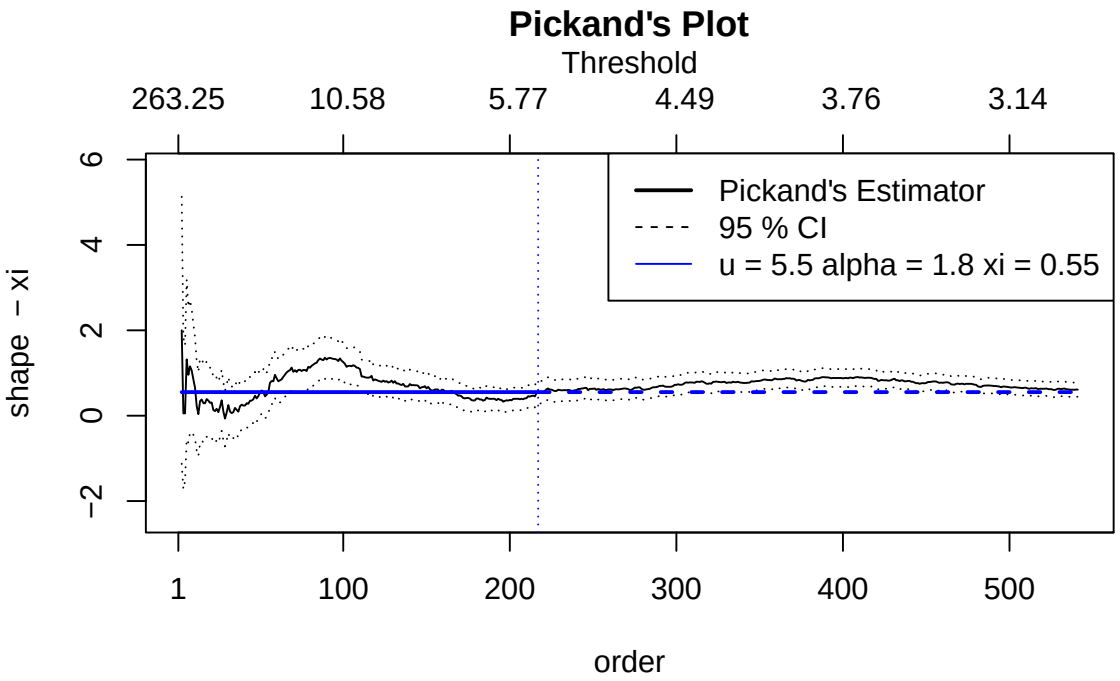
```
##
## Results of Distribution Parameter Estimation
## ---------------------------------------------
##
## Assumed Distribution:            Generalized Extreme Value
##
## Estimated Parameter(s):          location = 39.0416710
##                                  scale    = 34.8869268
##                                  shape    = -0.3818574
##
## Estimation Method:               Unbiased pwme
##
## Data:                            danish_block_max$MaxLoss
##
## Sample Size:                     11
```

Estimating Under Maximum Domain of Attraction

Pickands's Estimator

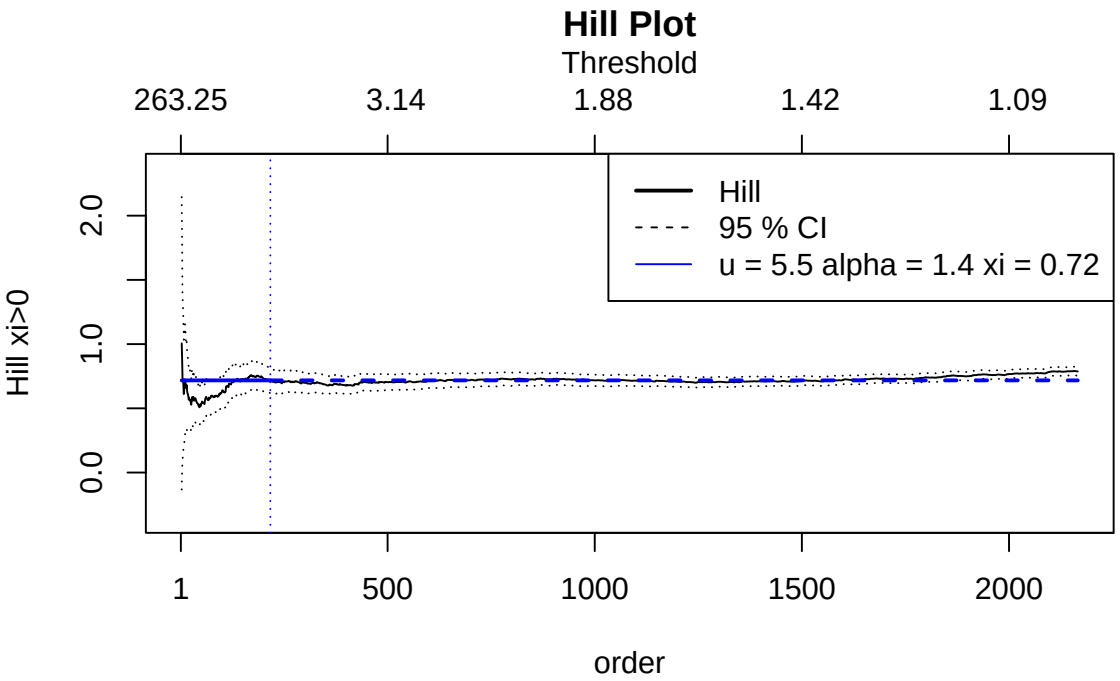
```
library(fExtremes)
library(RobExtremes)
library(evmix)
```

```
pickandsplot(danishuni$Loss)
```



Hill's Estimator

```
hillplot(danishuni$Loss)
```



Fitting Excesses Over a Threshold

Fitting the GPD

Introduction

Maximum Likelihood Estimation

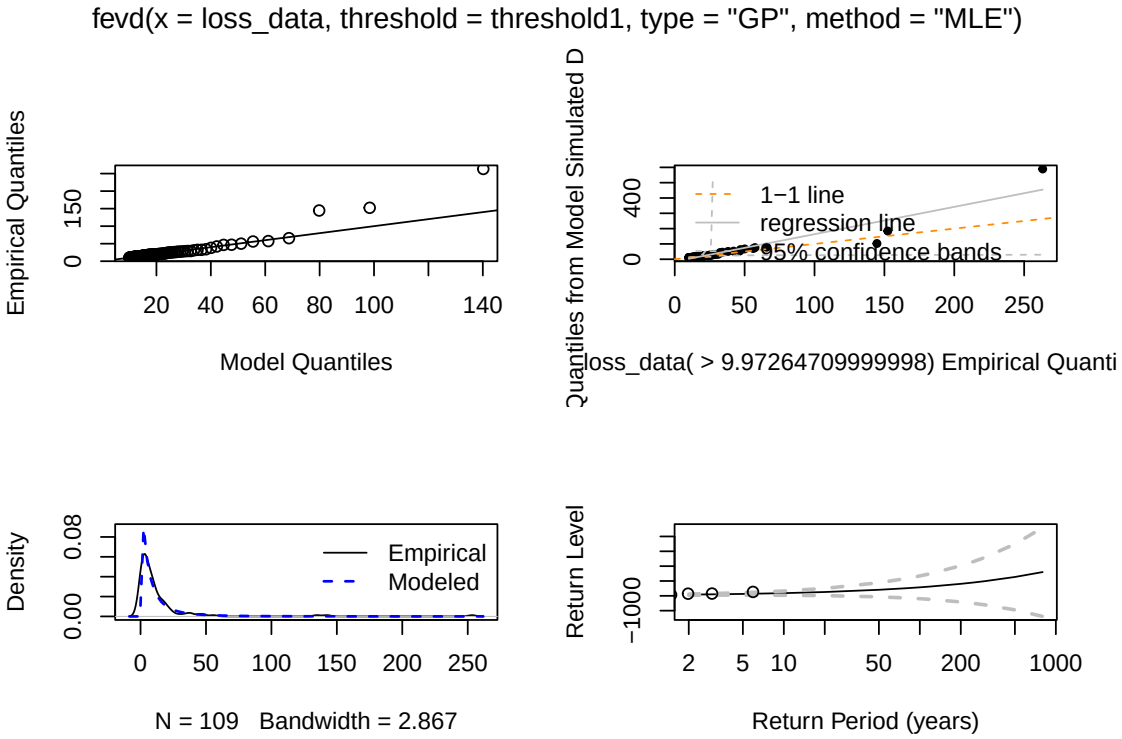
```
library(extRemes)    # Additional extreme value analysis tools
```

```
threshold1 = quantile(danishuni$Loss , probs = 0.95)
gpd_fit_all_data <- fevd(loss_data, threshold = threshold1, type = "GP" , method =
summary(gpd_fit_all_data)
```

```
##
## fevd(x = loss_data, threshold = threshold1, type = "GP", method = "MLE")
##
## [1] "Estimation Method used: MLE"
##
##
##  Negative Log-Likelihood Value:  375.3185
##
##
##  Estimated parameters:
```

```
##    scale    shape
## 7.037527 0.492032
##
##  Standard Error Estimates:
##     scale     shape
## 1.1177516 0.1351766
##
##  Estimated parameter covariance matrix.
##              scale       shape
## scale  1.24936856 -0.08119689
## shape -0.08119689  0.01827271
##
##  AIC = 754.637
##
##  BIC = 760.0197
```

```
plot(gpd_fit_all_data)
```




```
return_level <- return.level(gpd_fit_all_data, return.period = 100)
print(return_level)
```

```
## fevd(x = loss_data, threshold = threshold1, type = "GP", method = "MLE")
```

```
## get(paste("return.level.fevd.", newcl, sep = ""))(x = x, return.period = return
##
##  GP model fitted to  loss_data
## Data are assumed to be  stationary
## [1] "Return Levels for period units in years"
## 100-year level
##       573.0964
```

```
simulated_data5 <- rgev(1000,
                        scale = gpd_fit_all_data$results$par["scale"],
                        shape = gpd_fit_all_data$results$par["shape"])
head(simulated_data5)
```

```
## [1] -4.763579  1.797650 -2.247205  4.119384  3.402476 44.764043
```

Conclusion

The Danish reinsurance dataset provides valuable insights into fire losses and their components, making it an excellent resource for studying loss severity distributions and extreme value theory.