Statistical Methods for Extremal Events

Mehrab Atighi

2025-01-11

Contents

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
```

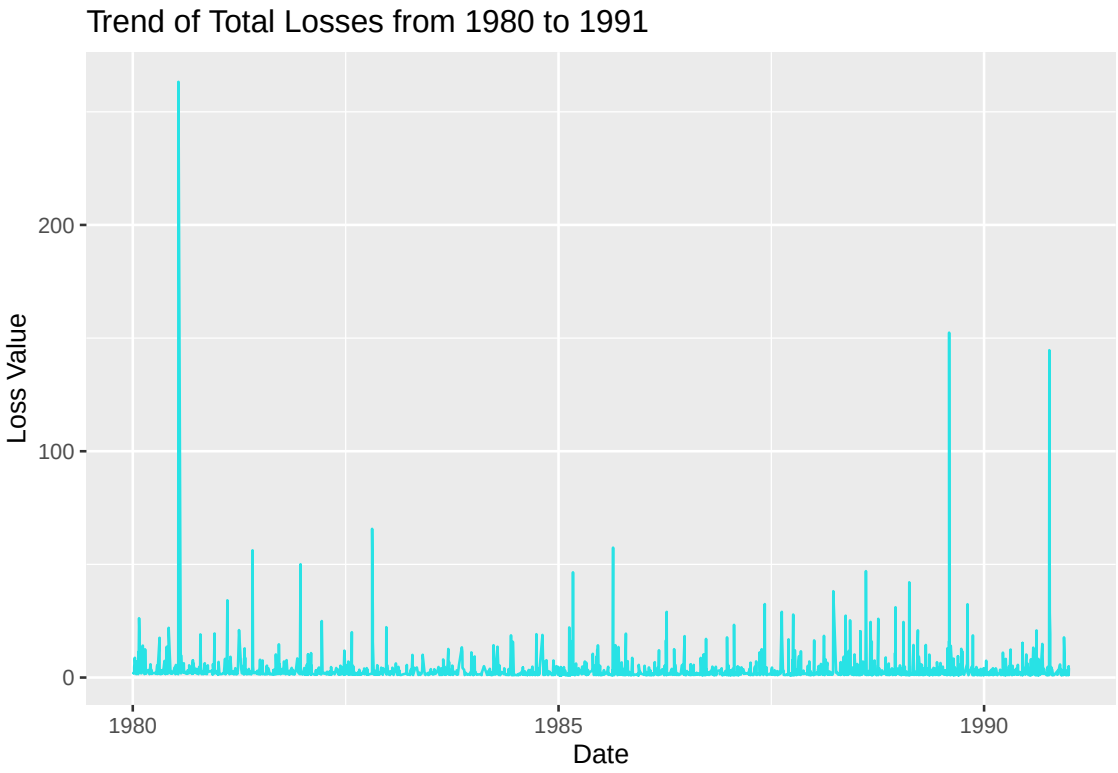
```
##  Median :1986-03-03   Median :  1.778
##  Mean   :1985-11-19   Mean   :  3.385
##  3rd Qu.:1988-07-08   3rd Qu.:  2.967
##  Max.   :1990-12-31   Max.   :263.250
```

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
```

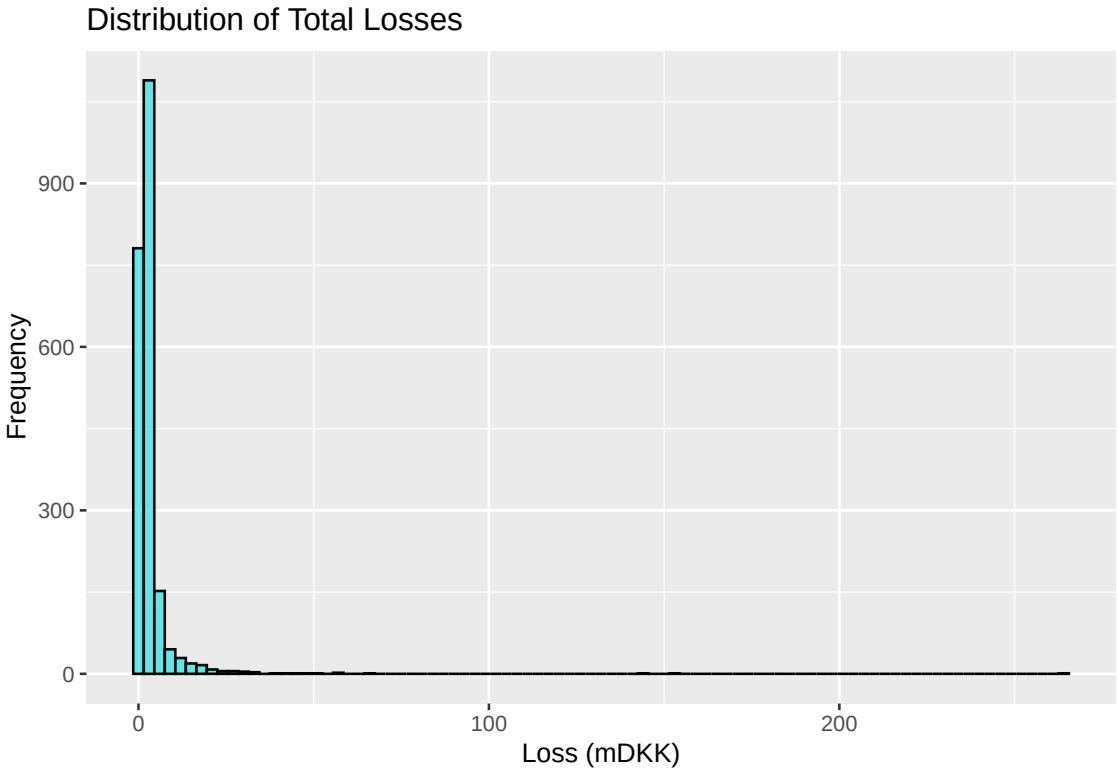


According to the above figure we can see that we had three times high loss values which are more than 100.

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")
```

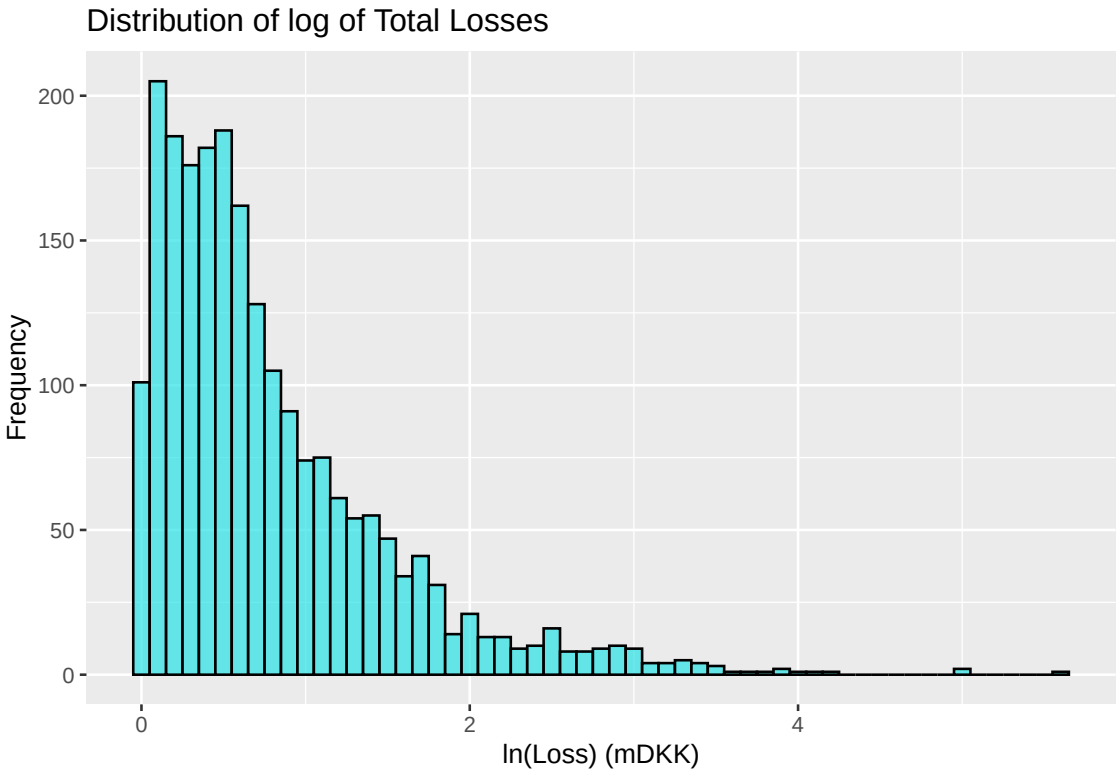


According to the above Figure we can see that Loss values following very heavy tail distribution. it means that we can see very huge loss values in the histogram at right side of that.

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.

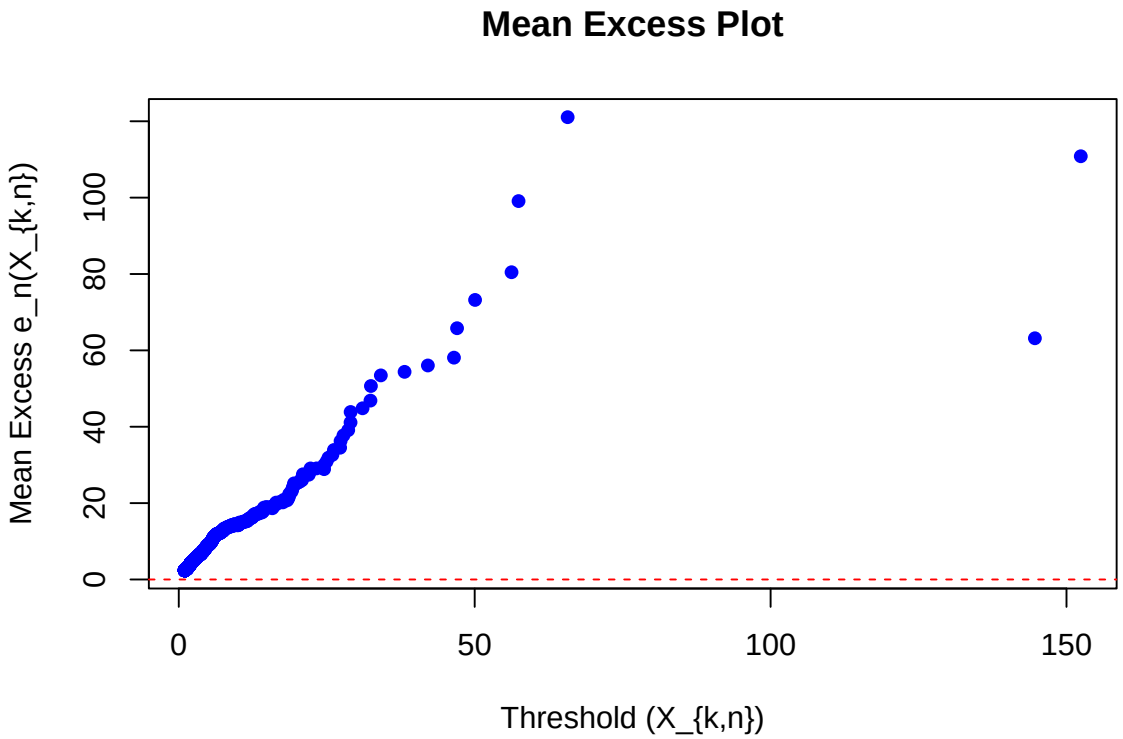
Mean Excess Plot of Data

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.

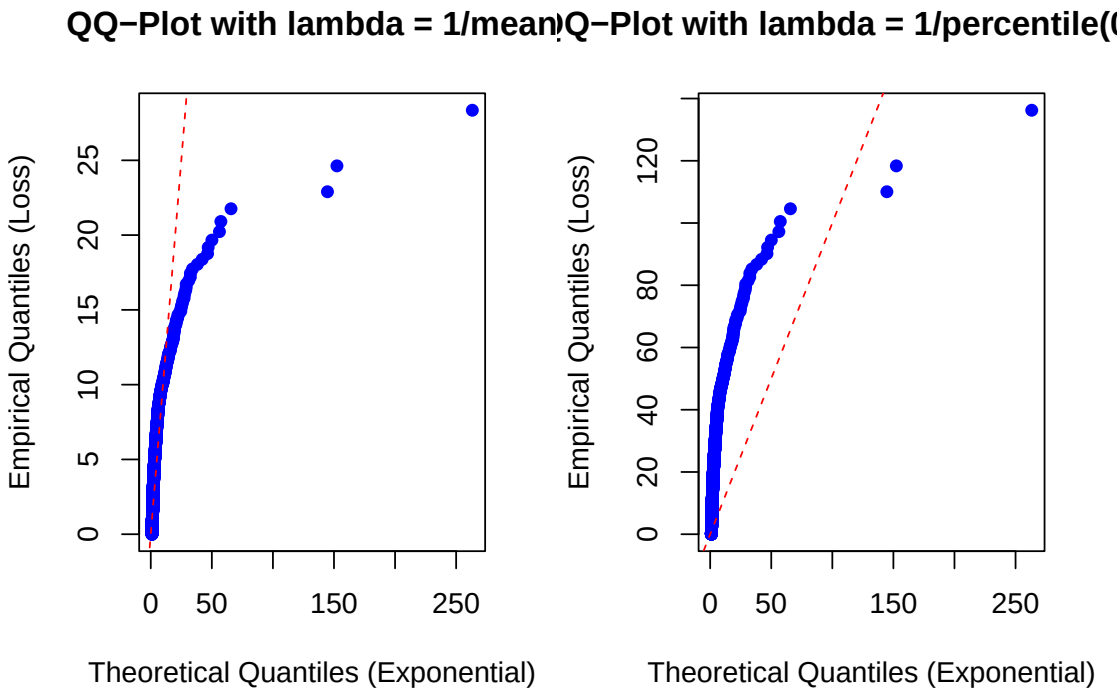
Quantile-Quantile Plot of Data with Exponential Distribution

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.

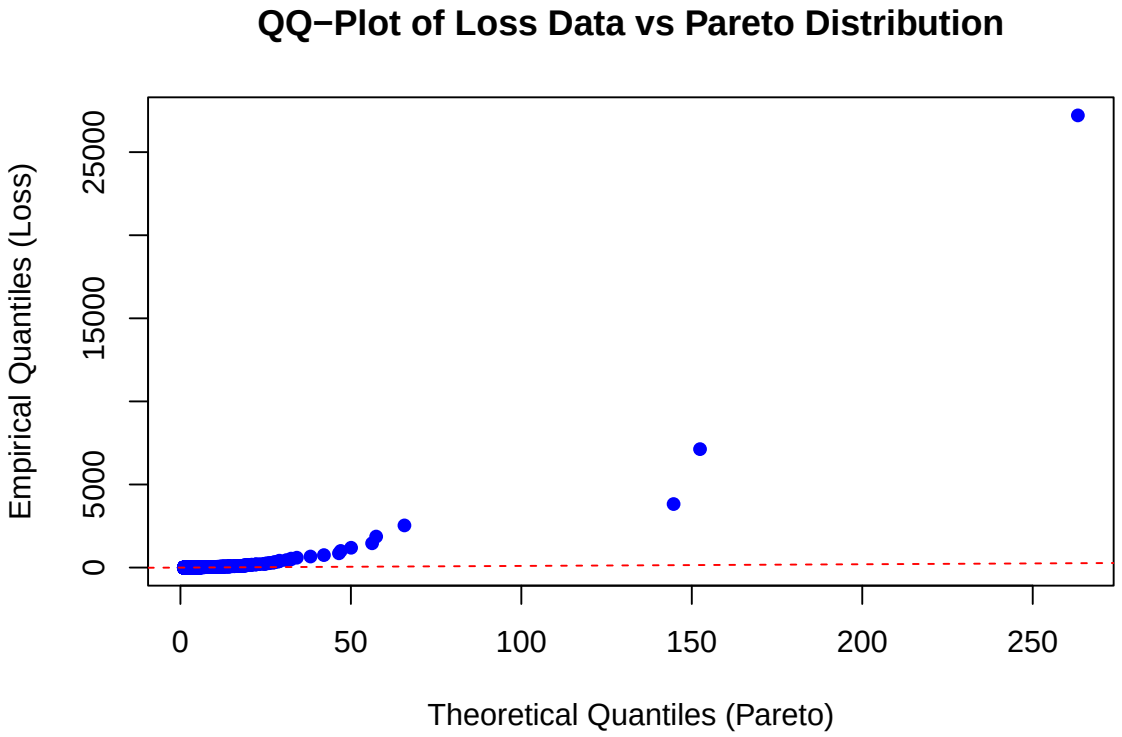
Quantile-Quantile Plot of Data with Pareto Distribution

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:

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

```
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

Gumbel's Method of Exceedances is a statistical method used to analyze extreme values by estimating how many future observations exceed a specific threshold derived from past records. The method assumes an infinite sequence of independent and identically distributed (iid) random variables.

Definitions and Formulae

Let $X_1, X_2, \ldots, X_n$ represent a sample of iid random variables arranged in increasing order as:

$$X_{1,n} \leq X_{2,n} \leq \cdots \leq X_{n,n}.$$

The **k-th order statistic**, $X_{k,n}$, is chosen as the threshold, and we analyze how many of the next r observations exceed this threshold.

Number of Exceedances The number of exceedances, denoted $S_r^n(k)$, is defined as:

$$S_r^n(k) = \sum_{i=1}^{r} I\{X_{n+i} > X_{k,n}\},$$

where $I\{\cdot\}$ is an indicator function that equals 1 if the condition inside it is true, and 0 otherwise.

Hypergeometric Distribution The number of exceedances, $S_r^n(k)$, follows a **hypergeometric distribution**:

$$P(S = j) = \frac{\binom{r+n-k-j}{n-k}\binom{j+k-1}{k-1}}{\binom{r+n}{n}}, \quad j = 0, 1, \ldots, r.$$

Where: - n: Sample size. - k: Order statistic defining the threshold $X_{k,n}$. - r: Number of future observations.

R Implementation

Below is an example of how to implement Gumbel's Method of Exceedances in R to compute the probabilities and simulate exceedances.

```
library(dplyr)
n = nrow(danishuni)
r = 25
j = 0:9
```

```r
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]
```

```r
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
```

Interpretation of Results

Probability of No Exceedances ($j = 0$)

- The probabilities for $j = 0$ are very high across all k, with values decreasing as k increases:
  - For $k = 1$, $P(S_r^n(k) = 0) = 0.9886$, meaning that almost 99% of the time, no observations exceed the first largest threshold within the next $r = 25$ observations.
  - For $k = 5$, $P(S_r^n(k) = 0) = 0.9442$, still high but slightly lower, indicating the threshold is less extreme for higher k.

Probability of One Exceedance ($j = 1$)

- The probabilities increase as k increases, suggesting that more exceedances are expected as the threshold $X_{k,n}$ is less extreme for higher k:

- For $k = 1$, $P(S_r^n(k) = 1) = 0.0113$, indicating that a single exceedance is rare.
- For $k = 5$, $P(S_r^n(k) = 1) = 0.0540$, showing a higher likelihood of one exceedance as the threshold becomes less stringent.

Higher Numbers of Exceedances ($j \geq 2$)

- Probabilities for $j \geq 2$ are negligible across all k, indicating that exceedances of these thresholds are exceedingly rare.

Comparison with Example 6.2.16

Your results align well with the behavior seen in the example for smaller k. However, due to your choice of $r = 25$ (higher than $r = 12$ in the example), the probabilities for $j = 0$ are slightly higher in your case, as larger r increases the chance of observing no exceedances when thresholds are set high.

Practical Implication

If these results are used to design thresholds for extreme losses:

1. For $k = 3$ (third largest observation), there is approximately a 96.62% chance that this threshold will **not** be exceeded in the next 25 observations.
2. If a stricter threshold is desired, $k = 1$ (largest observation) can be chosen, with a 98.86% chance of no exceedances.

The Return Period

In extreme value analysis, the return period is a commonly used concept to assess the frequency of extreme events. The return period T is defined as the average interval of time between events that exceed a certain threshold u.

Return Period Formula

The return period T for a threshold u is given by:

$$T = \frac{1}{P(X > u)}$$

where $P(X > u)$ is the probability that the random variable X exceeds the threshold u.

Probability of Exceedance

The probability of exceedance before the return period can be calculated using the following formula:

$$P(L(u) \leq EL(u)) = P(L(u) \leq \left\lfloor \frac{1}{p} \right\rfloor) = 1 - (1-p)^{\lfloor \frac{1}{p} \rfloor}$$

where $L(u)$ is the exceedance level, $EL(u)$ is the expected exceedance level, and $\lfloor x \rfloor$ denotes the integer part of x.

Approximation for High Thresholds

For high thresholds u (i.e., $u \uparrow \infty$ and consequently $p \downarrow 0$), the probability of exceedance is approximated by:

$$\lim_{u \uparrow \infty} P(L(u) \leq EL(u)) = \lim_{p \downarrow 0} \left(1 - (1-p)^{\lfloor \frac{1}{p} \rfloor}\right) = 1 - e^{-1} = 0.63212$$

This indicates that for high thresholds, the mean of $L(u)$ (the return period) is larger than its median.

\subsection*{Interpretation}

The return period provides a useful measure for understanding the frequency of extreme events. It is particularly important in risk assessment, actuarial science, and financial planning, where accurate modeling of extreme events is crucial.

```
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
```

This value represents the probability or rate of exceedance, which is the proportion of the dataset that exceeds the threshold u. In this case, $r_k = 0.046$ suggests that approximately 4.6% of the data points are greater than the threshold u = 10.58.

Records as an Exploratory Tool

Suppose that the random variables X_i are i.i.d. with distribution function (df) F. Recall the definitions of **records** and **record times** from the theory of extreme values.

Definition of a Record

A record X_n occurs if:

$$X_n > M_{n-1} = \max\{X_1, X_2, \ldots, X_{n-1}\}.$$

By convention, the first observation X_1 is always considered a record, as there are no prior observations to compare it to.

Record Times

The **record times** L_n are the random times at which the process M_n jumps, i.e., when a new record is set.

Record Counting Process

Define the **record counting process** as:

$$N_n = \sum_{k=1}^{n} \mathbb{I}\{X_k > M_{k-1}\},$$

where $\mathbb{I}\{\cdot\}$ is the indicator function. The random variable N_n counts the total number of records observed in the sequence $\{X_1, X_2, \ldots, X_n\}$ up to time n.

Moments of N_n

The following result on the mean and variance of N_n may seem surprising:

Moments of N_n

Suppose X_i are i.i.d. random variables with a continuous distribution function F, and let N_n be defined as above. Then:

$$\mathbb{E}[N_n] = \sum_{k=1}^{n} \frac{1}{k},$$
$$\mathrm{Var}(N_n) = \sum_{k=1}^{n} \frac{1}{k} - \sum_{k=1}^{n} \frac{1}{k^2}.$$

Explanation of Results

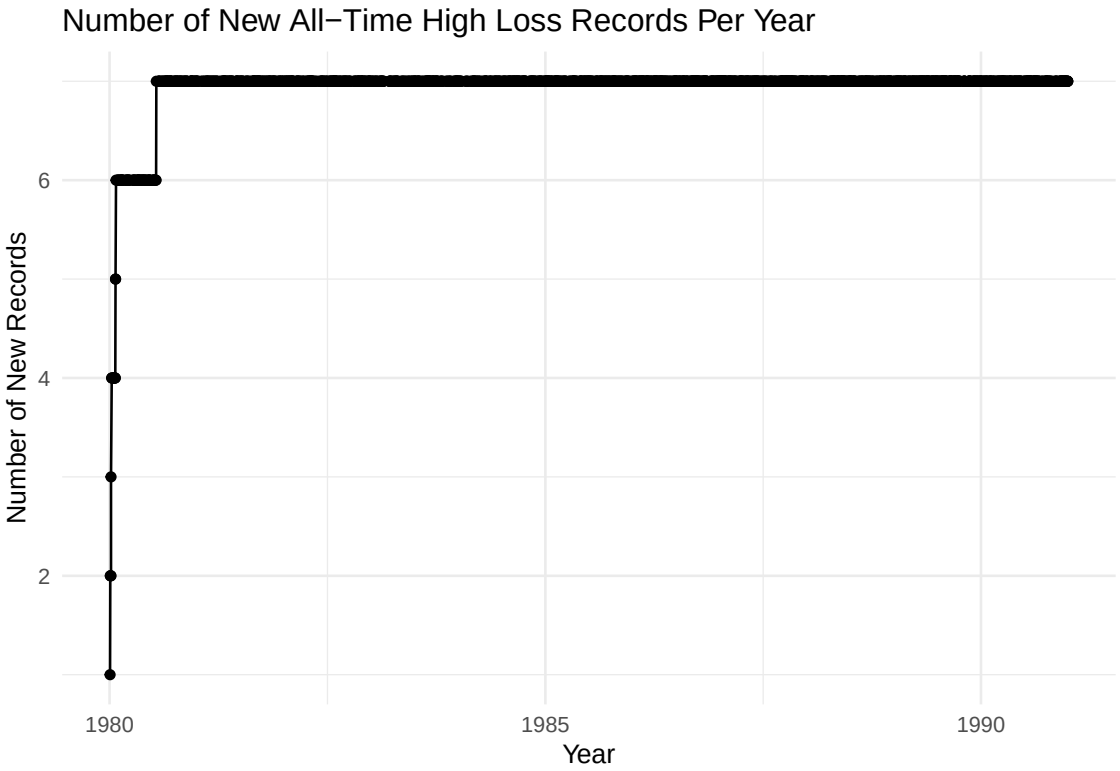
- The expected number of records $\mathbb{E}[N_n]$ grows logarithmically with n, approximately as $\ln(n) + \gamma$ for large n, where γ is the Euler-Mascheroni constant.
- The variance $\mathrm{Var}(N_n)$ is smaller than the mean, highlighting the relative stability of record counts in large samples.

implications

These results are foundational in extreme value theory. They show that while records are relatively rare events, their occurrence is predictable and follows well-defined statistical properties.

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

```
# 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

In this section, we consider a simple but effective tool for detecting heavy tails in a distribution and for providing a rough estimate of the order of its finite moments.

Definitions

Suppose that the random variables $X_1, X_2, \ldots, X_n$ are i.i.d., and define for any positive p the following quantities:

$$S_n^{(p)} = |X_1|^p + |X_2|^p + \cdots + |X_n|^p, \quad M_n^{(p)} = \max\{|X_1|^p, |X_2|^p, \ldots, |X_n|^p\}, \quad n \geq 1.$$

For simplicity, we write $M_n = M_n^{(1)}$ and $S_n = S_n^{(1)}$, slightly abusing the standard notation.

Functionals of $S_n^{(p)}$ and $M_n^{(p)}$

To study the underlying distribution of X_i, we investigate the asymptotic behavior of the ratio:

$$R_n^{(p)} = \frac{M_n^{(p)}}{S_n^{(p)}}, \quad n \geq 1.$$

This ratio captures how the maximum compares to the total sum, giving insights into the heaviness of the distribution's tail.

Limit Behavior

We summarize the known limit behavior of the ratio $\frac{M_n}{S_n}$ as follows:

- **Almost sure convergence:**

$$\frac{M_n}{S_n} \xrightarrow{\text{a.s.}} \frac{\mathbb{E}[|X|]}{n}, \quad \text{as } n \to \infty.$$

- **Convergence in probability:**

$$\frac{M_n}{S_n} \xrightarrow{\mathbb{P}} \frac{\mathbb{E}[|X|\mathbb{I}\{|X| > x\}]}{\mathbb{E}[|X|]} \quad \text{for } x > 0.$$

- **Convergence in distribution:**

$$\frac{M_n}{S_n} \xrightarrow{d} Y^{(1)}, \quad \text{where } Y^{(1)} = \frac{|X|}{\sum_{i=1}^{n} |X_i|} \text{ and } X \sim F.$$

Behavior for Higher Moments

For general p, let:

$$R_n^{(p)} = \frac{M_n^{(p)}}{S_n^{(p)}}.$$

The following results hold:

- If $\mathbb{E}[|X|^p] < \infty$, then $R_n^{(p)} \to 0$ as $n \to \infty$.
- If $R_n^{(p)}$ deviates significantly from zero for large n, this indicates that $\mathbb{E}[|X|^p]$ is infinite.

Practical Implications

1. By plotting $R_n^{(p)}$ against n for various values of p, one can infer whether $\mathbb{E}[|X|^p]$ is finite:
    - If $R_n^{(p)}$ converges to zero, then $\mathbb{E}[|X|^p]$ is finite.
    - Significant deviations of $R_n^{(p)}$ from zero for large n suggest $\mathbb{E}[|X|^p]$ is infinite.

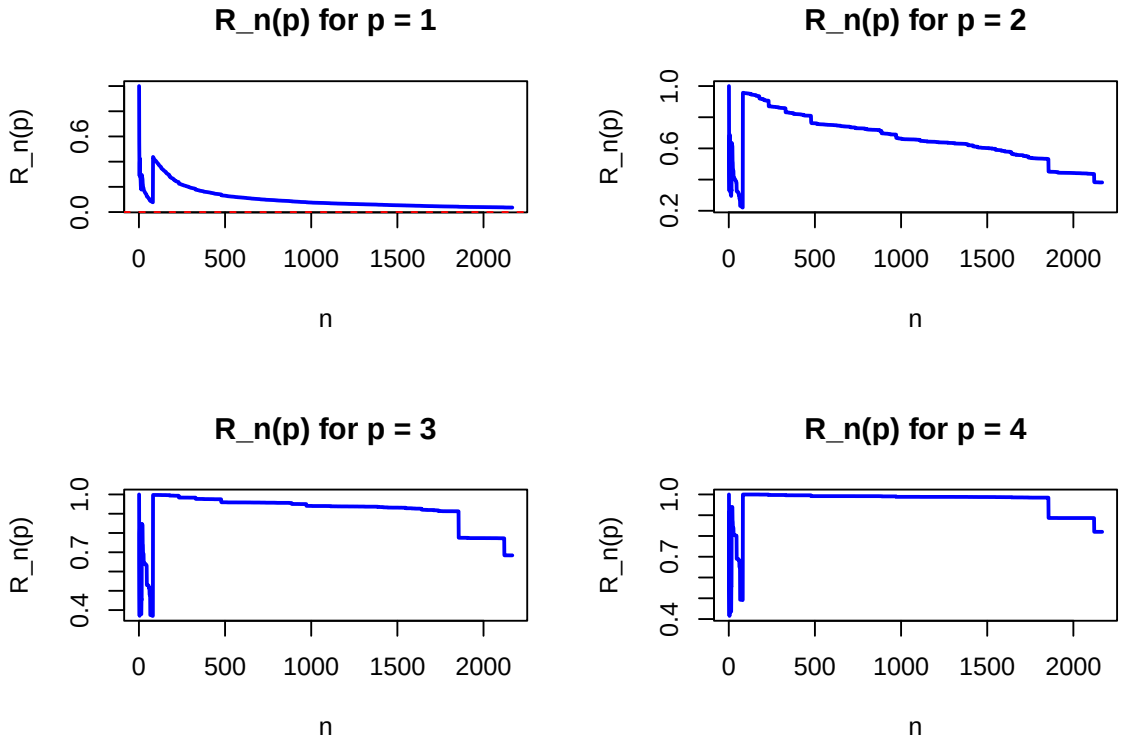
2. The method can also be adapted for the positive or negative parts of X_i to explore the right or left tail behavior of the distribution. Simply replace $|X_i|^p$ with $(X_i^+)^p$ or $(X_i^-)^p$, where $X_i^+ = \max\{X_i, 0\}$ and $X_i^- = \max\{-X_i, 0\}$.
3. More sophisticated functionals, such as upper order statistics, can also be used to refine the analysis of the tail behavior. For example, the empirical large claim index discussed in related sections allows for more subtle discrimination of distributions.

Conclusion for using this method.

The ratio of maximum to sum provides a simple yet powerful exploratory tool for analyzing the tail behavior of distributions. It can identify whether finite moments exist and highlight differences between heavy-tailed and light-tailed distributions. This method can be extended and refined to analyze specific characteristics of the data.

```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
    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)) {
```

```r
  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
}
```



Analysis of Ratio of Maximum and Sum Plots

The provided plots depict the ratio $R_n(p)$, where $R_n(p)$ represents the ratio of the maximum value to the sum of the values for different values of p as n increases.

Observations:

1. **Top-left plot** ($p = 1$):
   - The ratio $R_n(p)$ starts high and rapidly decreases, approaching zero as n increases.
   - This indicates that for $p = 1$, the sum of the values grows significantly faster than the maximum value, resulting in a diminishing ratio.
2. **Top-right plot** ($p = 2$):
   - The ratio $R_n(p)$ also starts high and decreases, but at a slower rate compared to $p = 1$.

- This suggests that for $p = 2$, the sum and the maximum value grow at a more comparable rate, but the sum still grows faster over time.

3. **Bottom-left plot** ($p = 3$):
   - The ratio $R_n(p)$ starts high and decreases very slowly, maintaining values closer to 1 for a longer range of n.
   - This implies that for $p = 3$, the maximum value and the sum of the values grow at nearly the same rate.
4. **Bottom-right plot** ($p = 4$):
   - The ratio $R_n(p)$ starts high and remains very close to 1 throughout the range of n, with only slight decreases.
   - This indicates that for $p = 4$, the maximum value and the sum of the values grow almost at the same rate, resulting in a nearly constant ratio.

Interpretation:

- **Convergence to Infinity**:
- For lower values of p (e.g., $p = 1$), the sum of the values grows much faster than the maximum value, leading to the ratio approaching zero. This can be interpreted as indicating that the sum has higher moments compared to the maximum.
  - For higher values of p (e.g., $p = 4$), the maximum value and the sum of the values grow at similar rates, suggesting convergence to a stable ratio. This indicates that the higher moments of the distribution lead to the maximum and the sum growing at comparable rates.

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

The Generalised Extreme Value (GEV) distribution is a fundamental tool in extreme value theory. It combines three types of extreme value distributions

(Gumbel, Fréchet, and Weibull) into a single framework, allowing for the modelling of extreme events. The GEV distribution is expressed as:

$$H_{\xi,\mu,\psi}(x) = \exp\left\{-\left(1+\xi\frac{x-\mu}{\psi}\right)^{-1/\xi}\right\}, \quad 1+\xi\frac{x-\mu}{\psi} > 0.$$

Here: - ξ is the shape parameter. - μ is the location parameter. - ψ is the scale parameter. For the special case where $\xi = 0$, the distribution reduces to the Gumbel distribution:

$$H_{0;\mu,\psi}(x) = \exp\left\{-e^{-(x-\mu)/\psi}\right\}, \quad x \in \mathbb{R}.$$

Block Maxima Approach

The block maxima method is a classical approach for applying the GEV distribution. In this method, data is divided into blocks (e.g., annual maxima, monthly maxima), and the maximum value from each block is extracted. These block maxima are then used to fit the GEV distribution.

1. **Divide Data into Blocks**: - Suppose you have daily loss data. You can divide it into monthly blocks and extract the maximum loss value for each month. 2. **Fit the GEV Distribution**: - Using the maximum values from each block, fit the GEV distribution to estimate the parameters ξ, μ, and ψ.

Parameter Estimation

To estimate the parameters of the GEV distribution, we use the Maximum Likelihood Estimation (MLE) method. This involves finding the parameter values that maximize the likelihood function given the observed data.

Return Level Estimation

Return levels are a crucial aspect of extreme value analysis. They represent the value expected to be exceeded once every T months, where T is the return period. The return level z_T for a given return period T is given by:

$$z_T = \mu + \frac{\psi}{\xi}\left(\left(-\log\left(1-\frac{1}{T}\right)\right)^{-\xi} - 1\right), \quad \xi \neq 0.$$

Interpretation of Results

The analysis of the GEV distribution fitted to the block maxima data provides insights into the behavior of extreme losses. For instance:

- **Empirical Quantiles vs. Model Quantiles**: The closeness of points to the 1-1 line indicates the fit quality.
- **Density Plot**: Compares empirical and model densities.
- **Return Level Plot**: Visualizes expected extreme values for different return periods. The GEV model helps in understanding the frequency and severity of extreme events, which is critical for risk assessment in finance, insurance, and environmental sciences.

application for all data

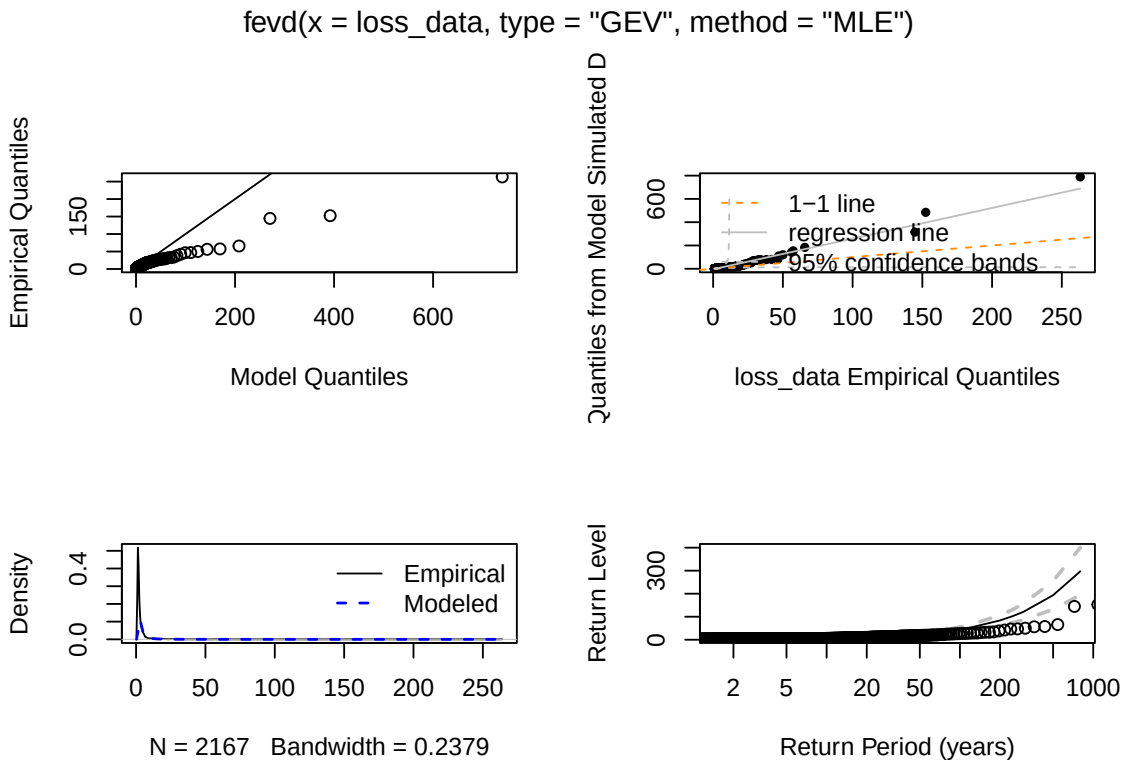
```
library(extRemes)    # Additional extreme value analysis tools
```

```
#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
```

Analysis of Generalised Extreme Value (GEV) Distribution Method on Data The provided image shows the results of fitting a Generalised Extreme Value (GEV) distribution to your loss_data using the Maximum Likelihood Estimation (MLE) method. Here's an analysis based on the figures:

Empirical Quantiles vs. Model Quantiles (Top Left Plot):

This plot compares the empirical quantiles of the data to the quantiles predicted by the GEV model. Ideally, the points should lie close to the 1-1 line if the GEV model fits the data well.

A good fit is observed if the points follow the 1-1 line closely, indicating that the model's quantiles match the empirical data.

Quantiles from Model Simulated Data vs. Empirical Quantiles (Top Right Plot):

This plot shows the quantiles from the model-simulated data against the empirical quantiles of the loss_data. The plot includes a 1-1 line, a regression line, and 95% confidence bands.

If the empirical quantiles align well within the confidence bands and the regression line follows the 1-1 line closely, it suggests that the GEV model accurately simulates the data's quantiles.

Density Plot (Bottom Left Plot):

This plot compares the empirical density of the data (black line) to the density predicted by the GEV model (blue dashed line).

A good fit is observed if the modeled density closely follows the empirical density, indicating that the GEV model captures the distribution of the data well.

Return Level Plot (Bottom Right Plot):

This plot shows the return levels for different return periods (in months). The return level is the value expected to be exceeded once in a given return period, with confidence intervals included.

If the return levels and their confidence intervals align well with the empirical data points, it suggests the GEV model's accuracy in predicting extreme values.

Finaly, i think that our model with above shape, scale and location parameters which estimated with MLE method are not very good and dont fit exactly on our data. so we are going to make fitting on monthly maxima data or **blockMaximum** data.

application for Monthly maxima data(BlockMaximum)

In extreme value analysis, data may become available when the X_i can be interpreted as maxima over disjoint time periods of length s say. In hydrology and other fields, this period can consist of one month to compensate for intra-month seasonalities.

Therefore the original data may look like

$$\mathbf{X}^{(1)} = \left(X_1^{(1)}, \ldots, X_s^{(1)}\right)$$

$$\mathbf{X}^{(2)} = \left(X_1^{(2)}, \ldots, X_s^{(2)}\right)$$

$$\vdots$$

$$\mathbf{X}^{(n)} = \left(X_1^{(n)}, \ldots, X_s^{(n)}\right)$$

where the vectors $\mathbf{X}^{(i)}$ are assumed to be iid, but within each vector $\mathbf{X}^{(i)}$ the various components may (and mostly will) be dependent.

The time length s is chosen so that the above conditions are likely to be satisfied. The basic iid sample from H_0 on which statistical inference is to be performed then consists of

$$X_i = \max\left(X_1^{(i)}, \ldots, X_s^{(i)}\right), \quad i = 1, \ldots, n.$$

For historical reasons and since s often corresponds to a 1-month period, statistical inference for H_0 based on data of the form above is referred to as *fitting of monthly maxima*. Below we discuss some of the main techniques for estimating θ in the exact model.

So we are going to use bottem codes on the Danishuni Dataset.

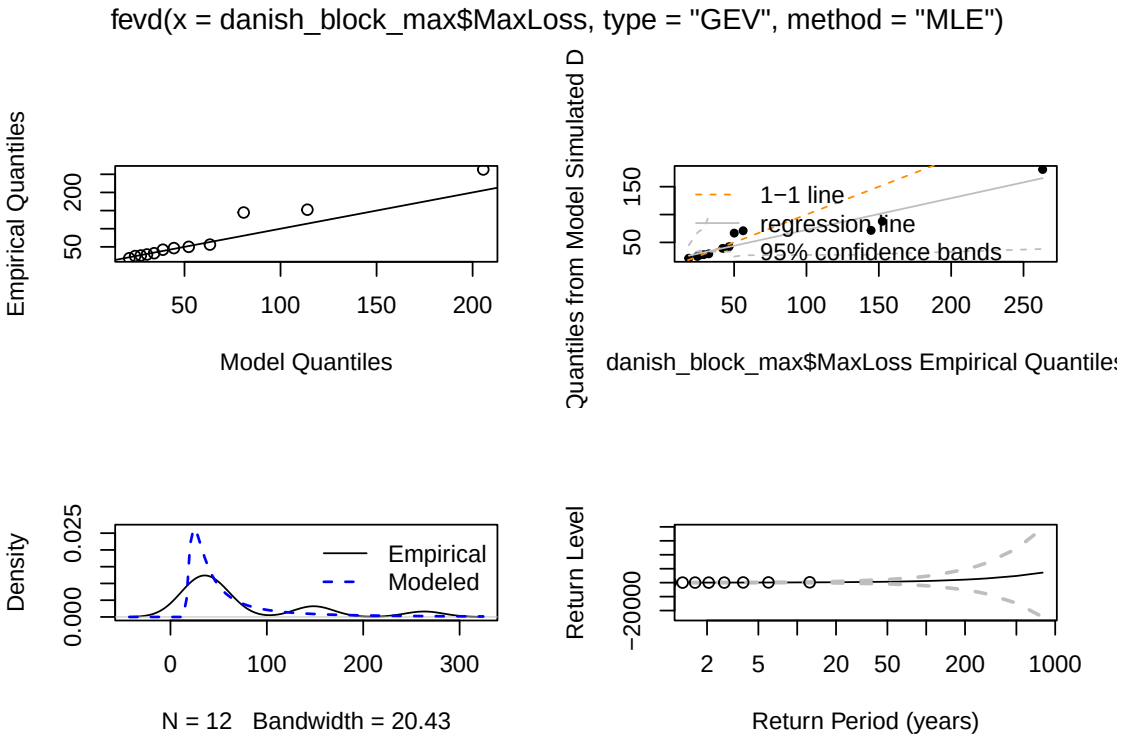
```
#GEV ON BLOCK MAX DATA with MLE method
library(dplyr)
```

```
danish_block_max <- danishuni %>%
  group_by(month = lubridate::month(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:  59.91762
##
##
##  Estimated parameters:
```

```
##  location     scale     shape
## 33.346556 18.751289  0.869428
##
##  Standard Error Estimates:
##  location     scale     shape
## 6.3386002 7.6777747 0.3865978
##
##  Estimated parameter covariance matrix.
##            location      scale      shape
## location 40.1778530 40.8775823 -0.5592048
## scale    40.8775823 58.9482247  0.5531512
## shape    -0.5592048  0.5531512  0.1494578
##
##  AIC = 125.8352
##
##  BIC = 127.29
```

```
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
##       1188.728
```

According to the above results we can see a significant difference between AIC and BIC of two above models. and in the top left plot, we can see a good 1-1 fitting. so we can say the monthly maximum blocks data are really better thank all of data.

Method of Probability Weighted Moments

introduction

The *Method of Probability Weighted Moments (PWM)* is a technique used in statistics, particularly in the field of hydrology and environmental sciences, to estimate the parameters of probability distributions. It is especially useful when dealing with extreme values and skewed distributions.

Key Concepts of PWM

1. **Probability Weighted Moments (PWMs)**:PWMs are defined as the expected values of the product of a random variable X and a function of its cumulative distribution function (CDF), $F(X)$. Mathematically, PWMs are expressed as:
$$M(p, r, s) = E[X^p F(X)^r (1 - F(X))^s]$$
Special cases include:
$$\alpha_r = M(1, 0, r) = E[X(1 - F(X))^r]$$
$$\beta_r = M(1, r, 0) = E[XF(X)^r]$$
2. **Estimation of Parameters**: PWMs are used to estimate the parameters of a probability distribution by matching the theoretical PWMs with the sample PWMs. The estimators are often considered superior to standard moment-based estimates because they are less sensitive to outliers and have better sampling properties.
3. **Application in Extreme Value Analysis**: PWMs are particularly useful in fitting distributions to block maxima or threshold exceedances, which

are common in extreme value analysis. They provide a robust method for parameter estimation when dealing with extreme events.

4. **Comparison with L-Moments**: PWMs are closely related to L-moments, which are linear combinations of PWMs. L-moments are often used in conjunction with PWMs for parameter estimation and have similar advantages in terms of robustness and sampling properties.

Example Calculation

Suppose we have a sample of annual maximum daily rainfall amounts. We can calculate the PWMs for this sample and use them to estimate the parameters of the GEV distribution.

1. **Calculate Sample PWMs**:Compute the sample PWMs using the formula:

$$\hat{w}_r(\theta) = \frac{1}{n}\sum_{j=1}^{n} X_{j,n}U_{j,n}^r, \quad r = 0, 1, 2$$

   Here, $X_{j,n}$ are the ordered sample values, and $U_{j,n}$ are the corresponding order statistics of a uniform distribution on (0, 1).

2. **Fit the Distribution**: Use the sample PWMs to fit the GEV distribution and estimate its parameters.

Advantages of PWM

- **Robustness**: PWMs are less affected by outliers compared to traditional moments.
- **Better Sampling Properties**: PWMs provide more stable estimates, especially for skewed distributions.
- **Ease of Use**: PWMs can be easily computed and used in parameter estimation.

Key Formulae

Define:

$$w_r(\theta) = E\left(XH_0^r(X)\right), \quad r \in \mathbb{N}_0$$

where H_0 is the GEV and X has the distribution function H_0 with parameter $\theta = (\xi, \mu, \psi)$. For $\xi \geq 1$, H_0 is regularly varying with index $1/\xi$. Hence w_0

is infinite. Therefore, we restrict ourselves to the case $\xi < 1$. Define the empirical analogue to the above formula:

$$\hat{w}_r(\theta) = \int_{-\infty}^{+\infty} x H_\theta^r(x)\, dF_n(x), \quad r \in \mathbb{N}_0$$

where F_n is the empirical distribution function corresponding to the data $X_1, \ldots, X_n$. To estimate θ, we solve the equations:

$$w_r(\theta) = \hat{w}_r(\theta), \quad r = 0, 1, 2.$$

application 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
```

application 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
```

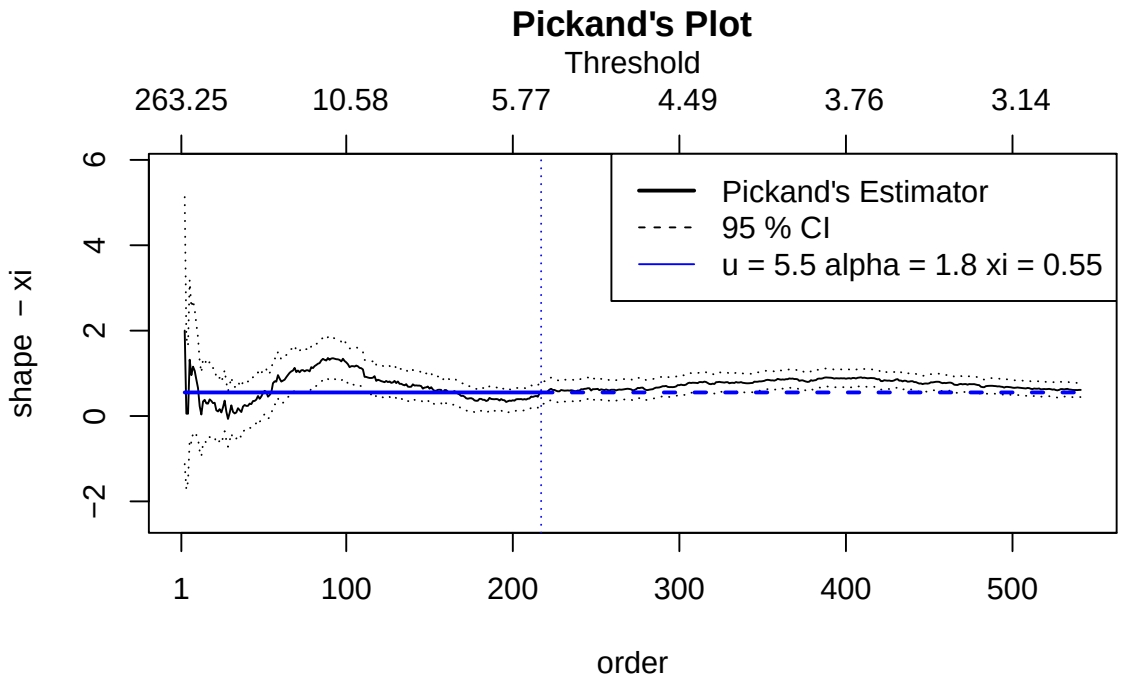
```
## ------------------------------------------
##
## Assumed Distribution:            Generalized Extreme Value
##
## Estimated Parameter(s):          location = 35.0253260
##                                  scale    = 24.6454966
##                                  shape    = -0.5076665
##
## Estimation Method:               Unbiased pwme
##
## Data:                            danish_block_max$MaxLoss
##
## Sample Size:                     12
```

Estimating Under Maximum Domain of Attraction

Pickands's Estimator

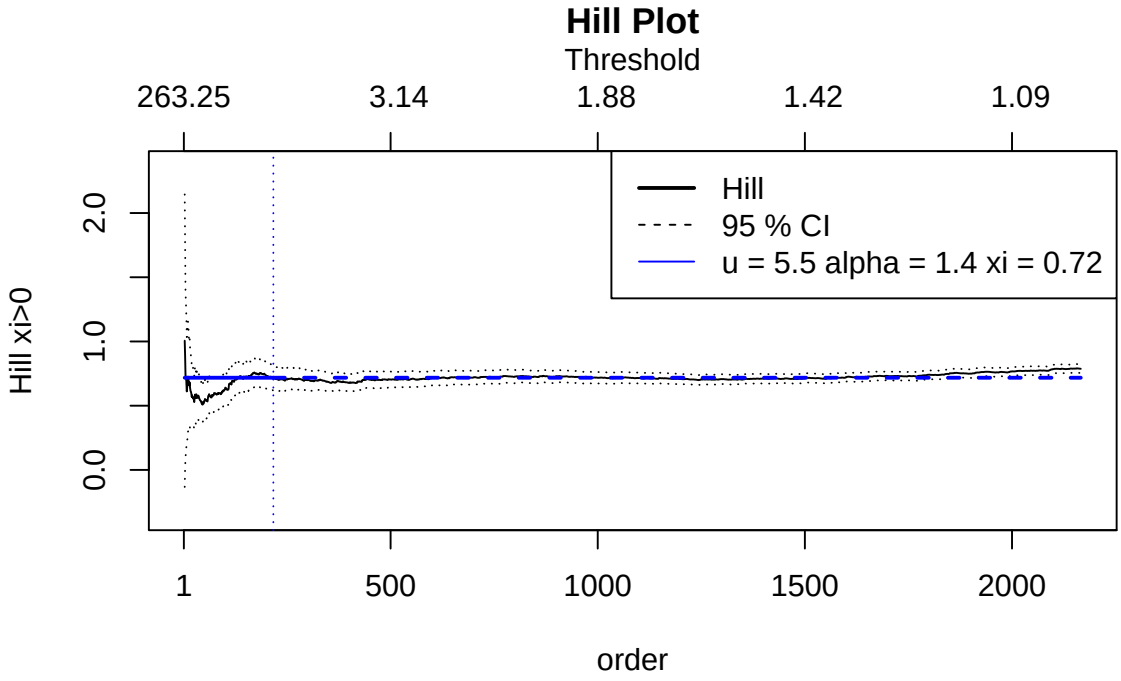
```
library(fExtremes)
library(RobExtremes)
library(evmix)
```

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



Hill's Estimator

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



Fitting Excesses Over a Threshold

Fitting the GPD

Introduction

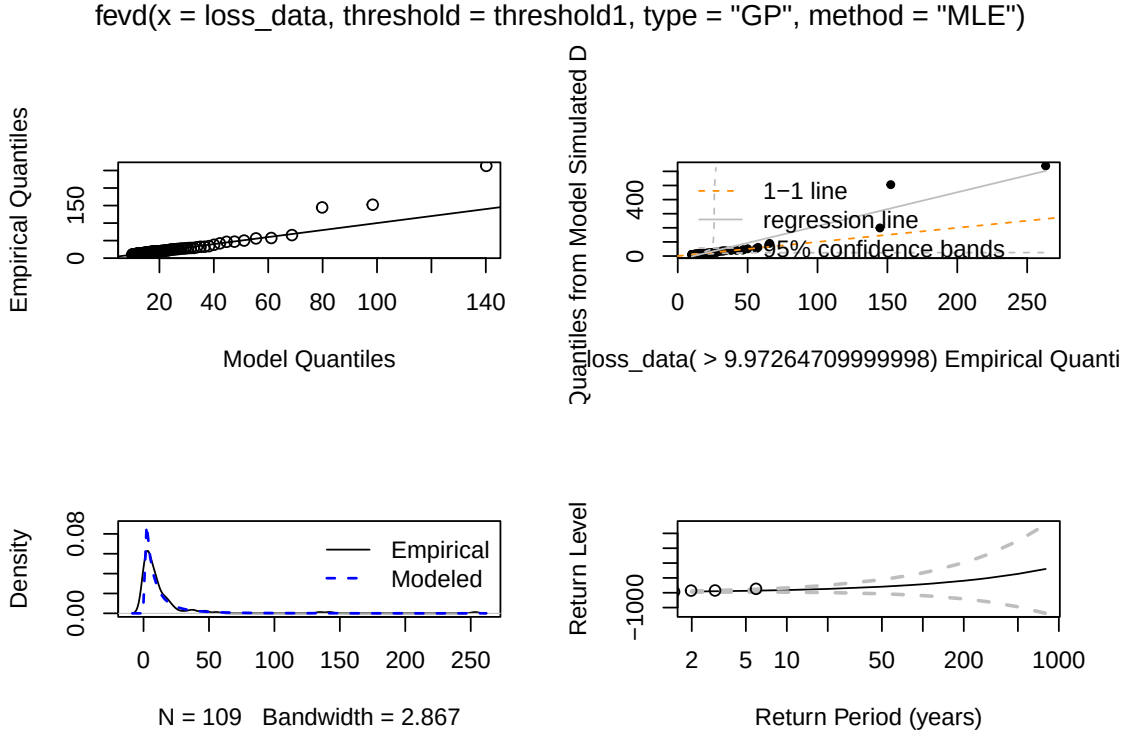
Maximum Likelihood Estimation

```
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
```

```r
plot(gpd_fit_all_data)
```




```r
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
```

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

```
## [1]  9.082932  3.047537 80.038128  1.962233  1.085029 -4.819420
```

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.