Modeling extreme values with a single GEV probability distribution

Pascal Alain Dkengne Sielenou

September 08th, 2023

```r
# library(xfun)
```

```r
path <- ".."
```

```r
xfun::in_dir(dir = path, expr = source("./src/extract_block_maxima_with_indexes.R"))
xfun::in_dir(dir = path, expr = source("./src/generate_gev_sample.R"))
xfun::in_dir(dir = path, expr = source("./src/plot_gev_pdf.R"))
xfun::in_dir(dir = path, expr = source("./src/plot_gev_cdf.R"))
xfun::in_dir(dir = path, expr = source("./src/plot_gev_probability.R"))
xfun::in_dir(dir = path, expr = source("./src/plot_gev_quantile.R"))
xfun::in_dir(dir = path, expr = source("./src/plot_block_maxima.R"))
```

```r
x <- rnorm(n = 1000)
```

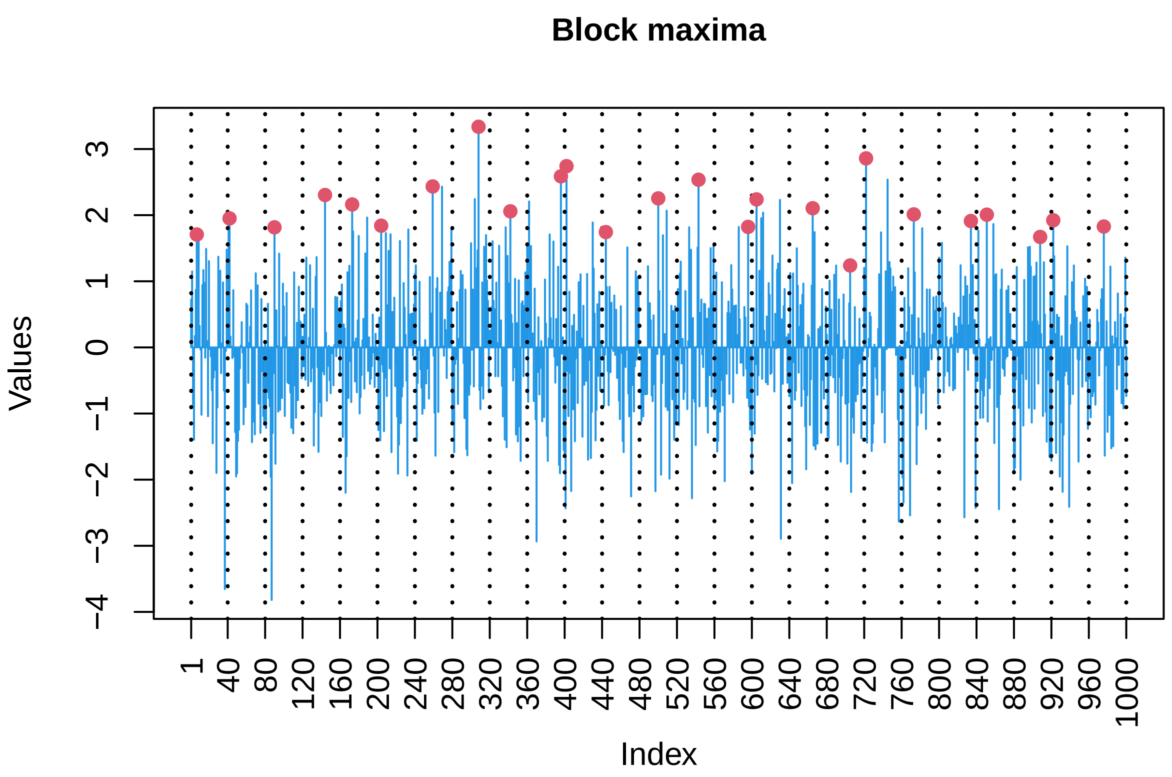
```r
block_size <- 40
```

```r
extremes <- extract_block_maxima_with_indexes(x, block_size)

extremes
```

```
## $block_maxima
##  [1] 1.70764416065627 1.95143467230461 1.81615662718137 2.30491769860035
##  [5] 2.16197430249763 1.84125061797702 2.43443115417944 3.33690215588613
##  [9] 2.05818451032608 2.58763527037433 2.74142898785237 1.74440593058229
## [13] 2.25392519476657 2.53583781250326 1.82282477069182 2.23859262182981
## [17] 2.10512712930803 1.23918428967039 2.85929609723200 2.01235346086732
## [21] 1.91164647234364 2.00771558527949 1.67062705945532 1.92252106179793
## [25] 1.82862129728129
##
## $block_maxima_indexes
##  [1]   7  42  90 144 173 204 259 308 342 396 402 444 500 543 596 605 665 705 722
## [20] 773 834 851 908 922 976
```

```r
plot_block_maxima(x, block_size, xlab = "Index", ylab = "Values", main = "Block maxima")
```



```
model <- estimate_single_gev_model(x, block_size, nsloc = NULL)
```

```
names(model)
```

```
## [1] "data"                              "block_maxima_indexes"
## [3] "gev_model"                         "block_size"
## [5] "extremal_index"                    "normalized_gev_parameters"
```

```
model$gev_model
```

```
##
## Call: evd::fgev(x = x, nsloc = nsloc, prob = NULL, std.err = std.err,      corr = FALSE, method = "Bl
## Deviance: 27.285524689802
##
## Estimates
##            loc           scale           shape
##  1.942176076899   0.379505102406  -0.103784067890
##
## Optimization Information
##   Convergence: successful
##   Function Evaluations: 22
##   Gradient Evaluations: 8
```

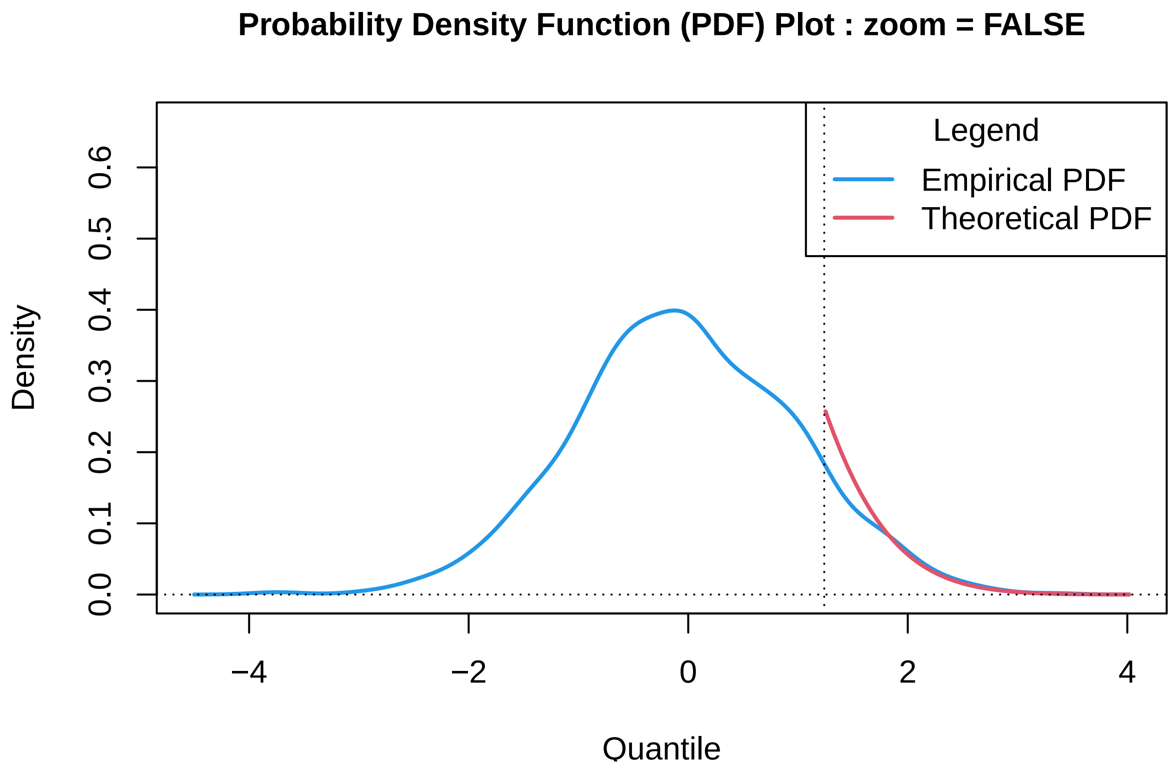
```
names(model$gev_model)
```

```
##  [1] "estimate"    "std.err"     "fixed"       "param"       "deviance"
##  [6] "corr"        "var.cov"     "convergence" "counts"      "message"
## [11] "data"        "tdata"       "nsloc"       "n"           "prob"
## [16] "loc"         "call"
```

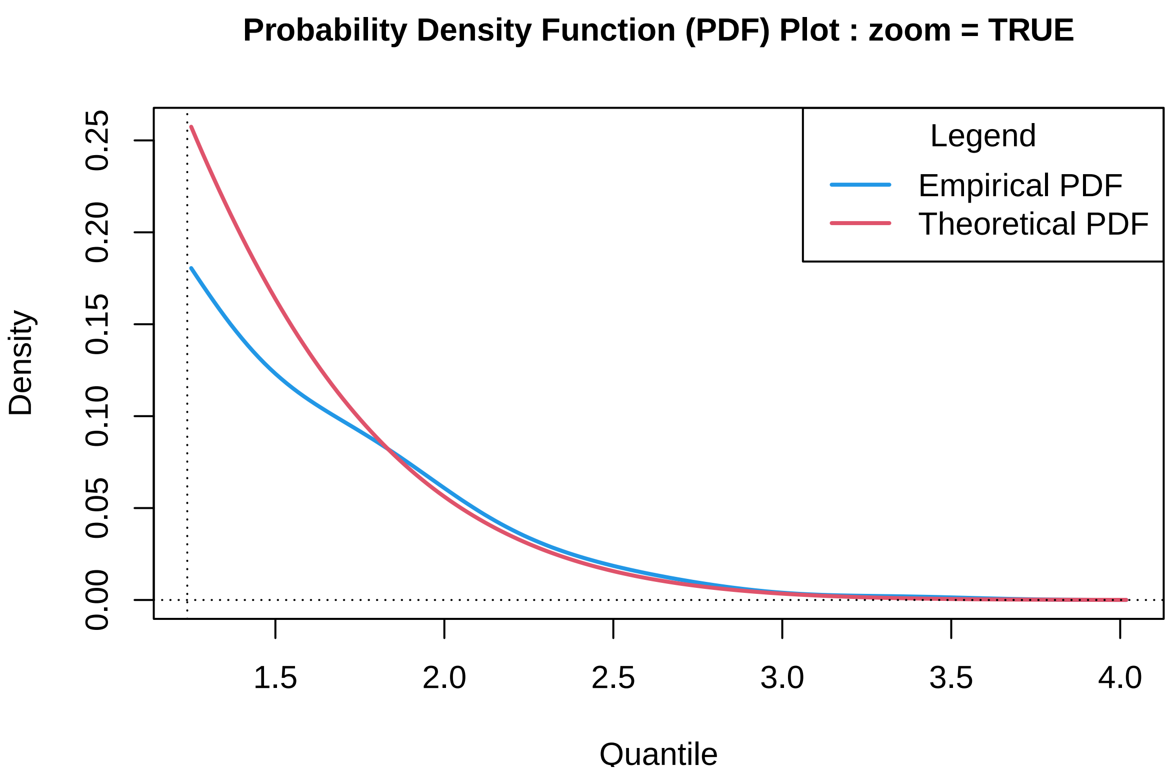
```
model$normalized_gev_parameters
```

```
##          loc_star         scale_star         shape_star
##  0.236504915734205  0.556526593993843 -0.103784067889558
```

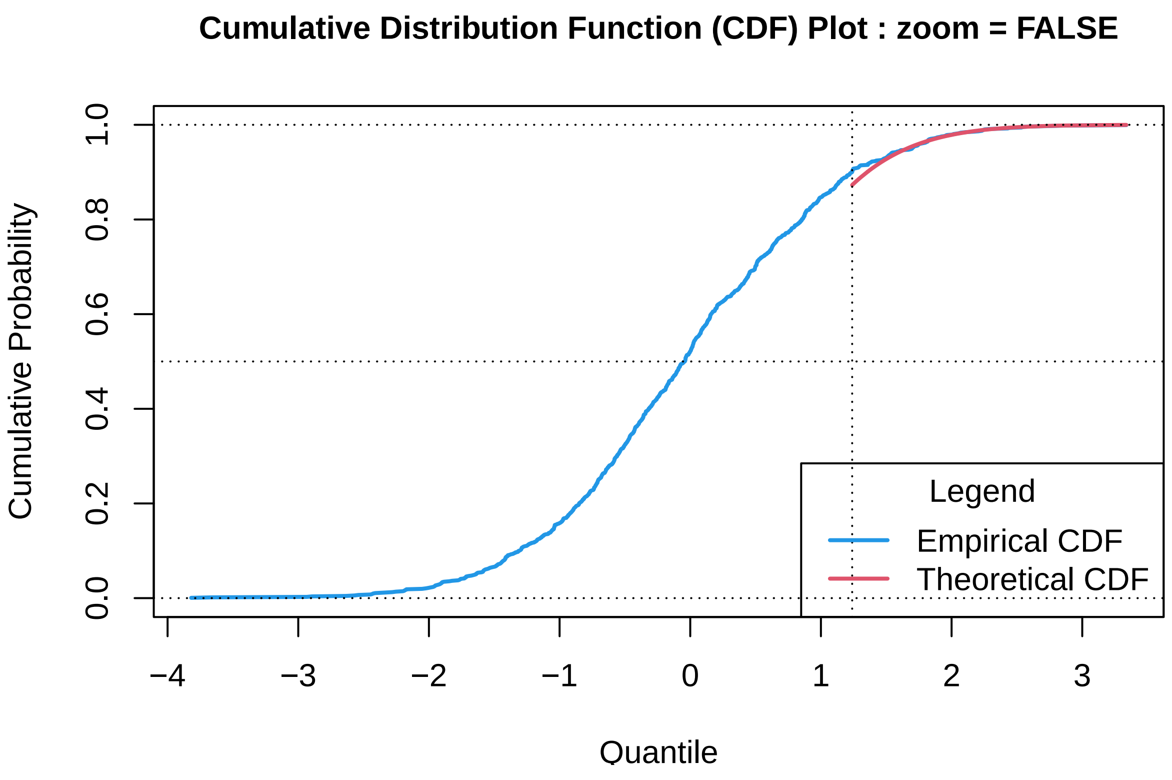
```
plot_gev_pdf(model,
             zoom = FALSE,
             xlab = "Quantile",
             ylab = "Density",
             main = "Probability Density Function (PDF) Plot")
```



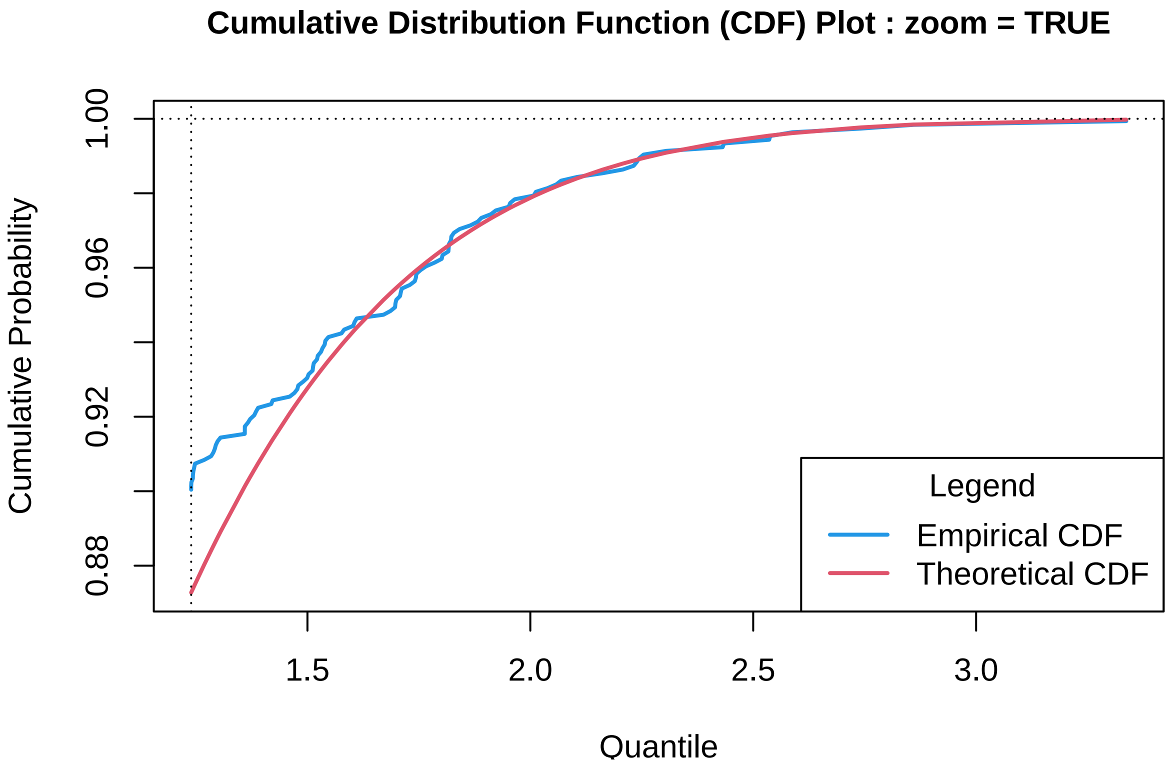
```
plot_gev_pdf(model,
             zoom = TRUE,
             xlab = "Quantile",
             ylab = "Density",
             main = "Probability Density Function (PDF) Plot")
```



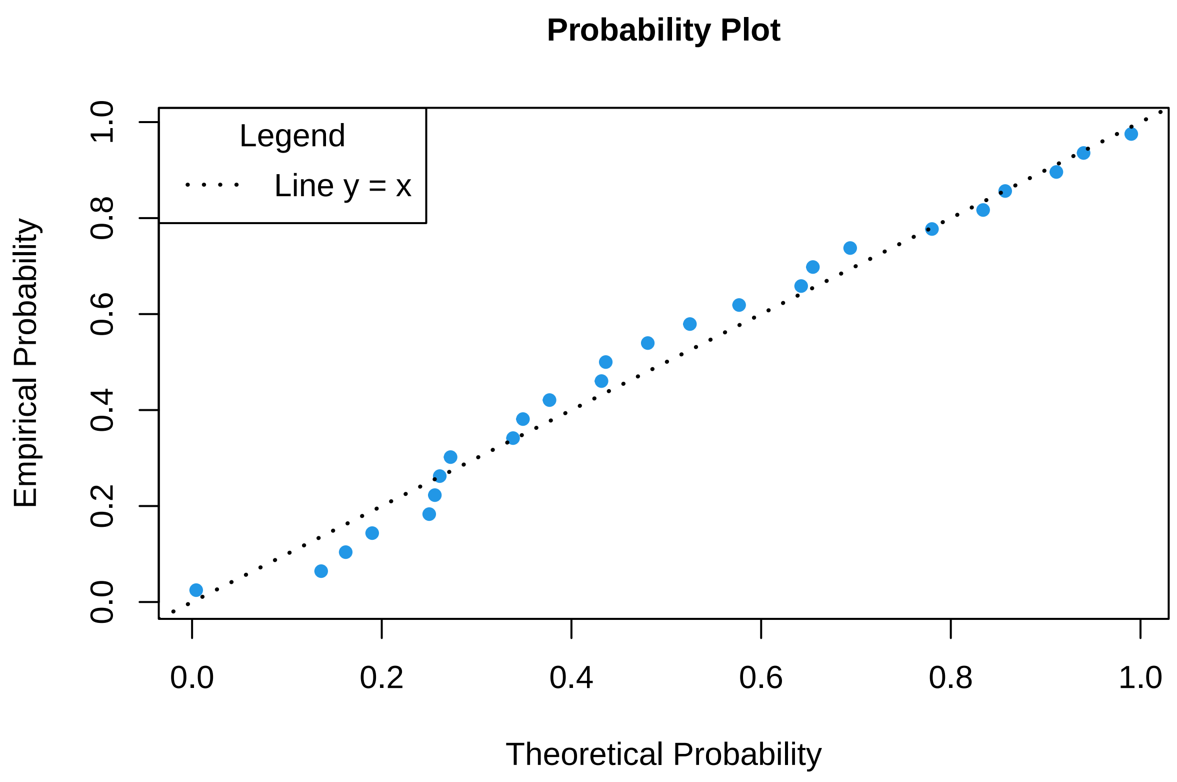
```r
plot_gev_cdf(model,
             zoom = FALSE,
             xlab = "Quantile",
             ylab = "Cumulative Probability",
             main = "Cumulative Distribution Function (CDF) Plot")
```



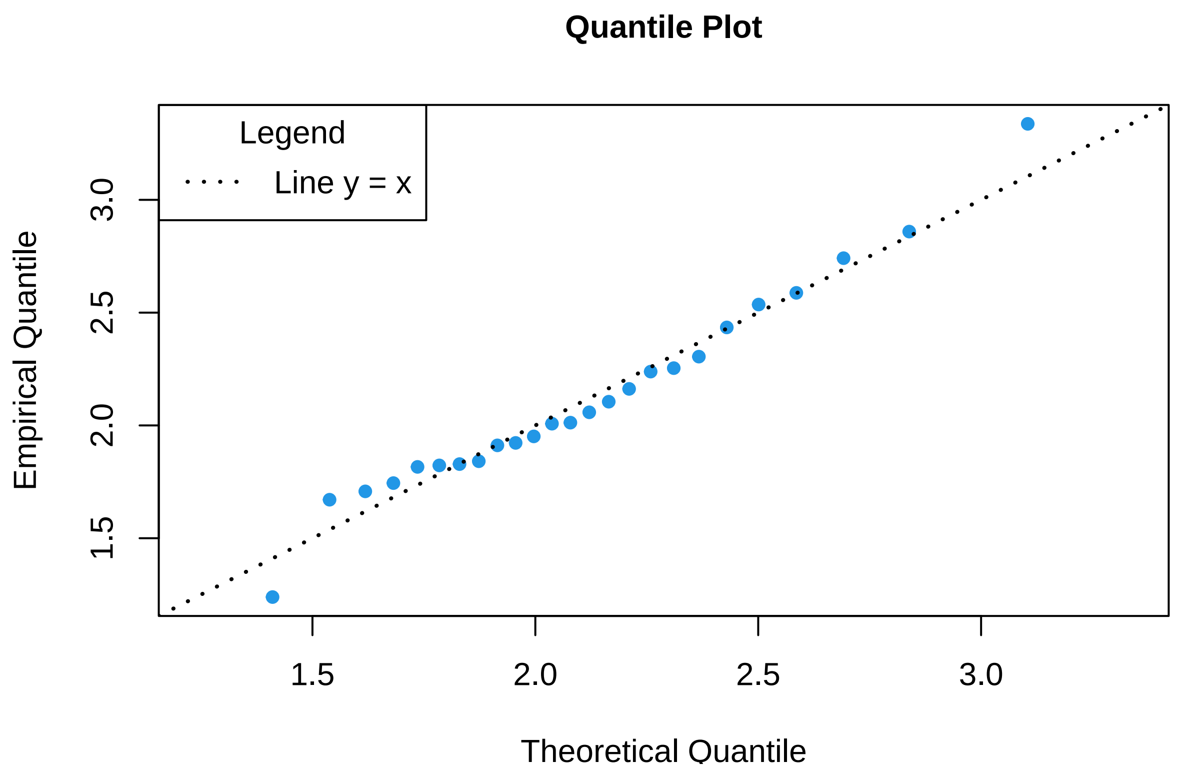
```r
plot_gev_cdf(model,
             zoom = TRUE,
             xlab = "Quantile",
             ylab = "Cumulative Probability",
             main = "Cumulative Distribution Function (CDF) Plot")
```



```r
plot_gev_probability(model,
                     xlab = "Theoretical Probability",
                     ylab = "Empirical Probability",
                     main = "Probability Plot")
```



```r
plot_gev_quantile(model,
                  xlab = "Theoretical Quantile",
                  ylab = "Empirical Quantile",
                  main = "Quantile Plot")
```


Quantile Plot
Legend
Line y = x
Empirical Quantile
Theoretical Quantile