Modeling extreme values with a GEV mixture probability distributions

Pascal Alain Dkengne Sielenou

September 28th, 2023

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

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

```r
xfun::in_dir(dir = path, expr = source("./src/generate_gev_sample.R"))
xfun::in_dir(dir = path, expr = source("./src/calculate_gev_inverse_cdf.R"))
xfun::in_dir(dir = path, expr = source("./src/estimate_gev_mixture_model_parameters.R"))
xfun::in_dir(dir = path, expr = source("./src/plot_gev_mixture_model_pdf.R"))
xfun::in_dir(dir = path, expr = source("./src/plot_several_standardized_block_maxima_mean.R"))
xfun::in_dir(dir = path, expr = source("./src/estimate_gev_mixture_model_quantile.R"))
```

```r
n <- 100000
```

```r
loc <- 1
scale <- 0.5
shape <- 0
set.seed(1122)
x <- generate_gev_sample(n = n, loc = loc, scale = scale, shape = shape)
```

```r
nlargest <- 1000
```

```r
gev_mixture_model <- estimate_gev_mixture_model_parameters(x,
                                                           nsloc = NULL,
                                                           std.err = FALSE,
                                                           block_sizes = NULL,
                                                           minimum_nblocks = 50,
                                                           threshold = NULL,
                                                           nlargest = nlargest,
                                                           confidence_level = 0.95,
                                                           log_mv = TRUE,
                                                           log_pw = TRUE,
                                                           trace = FALSE)
```

```
##  Successful convergence.
##  Successful convergence.
```

```r
names(gev_mixture_model)
```

```
##  [1] "data"
##  [2] "data_largest"
##  [3] "block_sizes"
##  [4] "equivalent_block_sizes"
##  [5] "rejected_block_sizes"
##  [6] "block_maxima_indexes_object"
##  [7] "gev_models_object"
##  [8] "extremal_indexes"
##  [9] "normalized_gev_parameters_object"
## [10] "weighted_normalized_gev_parameters_object"
## [11] "identic_weights_mw"
## [12] "pessimistic_weights_mw"
## [13] "pessimistic_weights_pw_shape"
## [14] "pessimistic_weights_pw_scale"
## [15] "pessimistic_weights_pw_loc"
## [16] "automatic_weights_mw"
## [17] "automatic_weights_mw_statistics"
## [18] "automatic_weights_pw_shape"
## [19] "automatic_weights_pw_scale"
## [20] "automatic_weights_pw_loc"
## [21] "automatic_weights_pw_statistics"
```

```
gev_mixture_model$block_sizes
```

```
##  [1]  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```
gev_mixture_model$normalized_gev_parameters_object
```

```
##            loc_star        scale_star          shape_star
## 8  3.58840880254155 0.392768818465400  0.0208210460504149
## 9  3.22225763231416 0.660549913813329 -0.1352208573828976
## 10 3.38569158607671 0.515180964211473 -0.0526471700095747
## 11 3.35801595751743 0.564914264867210 -0.0871466496718430
## 12 3.52917127957394 0.454225467860067 -0.0285597160633812
## 13 3.48408231014543 0.465405303926418 -0.0304892473893983
## 14 3.26289067711469 0.610085554921145 -0.1020049726307863
## 15 3.25188099487815 0.614544516089340 -0.1046625012256916
## 16 3.36032726105798 0.548454663855359 -0.0742929567754220
## 17 3.29768255960161 0.612307232276290 -0.1095602193702250
## 18 2.96217959767506 0.852990106298593 -0.1974375317015732
## 19 3.14399999745866 0.665779727416213 -0.1246177808584009
## 20 2.90270767063140 0.866726146459984 -0.1954785219627826
```

```
gev_mixture_model$weighted_normalized_gev_parameters_object
```

```
##                              loc_star        scale_star          shape_star
## identic_weights     3.28840740973744 0.601840975420063 -0.0939459291531970
## pessimistic_weights 3.32414012491867 0.620783092316425 -0.0902830901150795
## automatic_weights   3.39041800568140 0.477653869695952 -0.0144226284719552
```

```
gev_mixture_model$automatic_weights_mw_statistics
```

```
## $function_value
## [1] 0.00132245678136426
##
## $gradient_value
## [1] 4.77802622145251e-06
##
## $function_reduction
## [1] 0.0163013844992505
##
## $number_iterations
## [1] 1571
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_pw_statistics
```

```
## $function_value
## [1] 0.00076028447726084
##
## $gradient_value
## [1] 2.83687914739428e-05
##
## $function_reduction
## [1] 0.0244518192756481
##
## $number_iterations
## [1] 3641
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_mw
```

```
##                8                9               10               11
## 0.155395798855892 0.000000000000000 0.000000000000000 0.000000000000000
##               12               13               14               15
## 0.000000000000000 0.000000000000000 0.000000000000000 0.000000000000000
##               16               17               18               19
## 0.000000000000000 0.000000000000000 0.000000000000000 0.844604201144109
##               20
## 0.000000000000000
```

```
gev_mixture_model$pessimistic_weights_pw_shape
```

```
##                 8                 9                10                11
## 0.0861198845202670 0.0736775712671250 0.0800196408782187 0.0773060822236086
##                12                13                14                15
## 0.0819705117239118 0.0818124995474774 0.0761659348023052 0.0759637903738795
##                16                17                18                19
## 0.0783061644693524 0.0755926507499697 0.0692332856740073 0.0744629364792395
##                20
## 0.0693690472906378
```

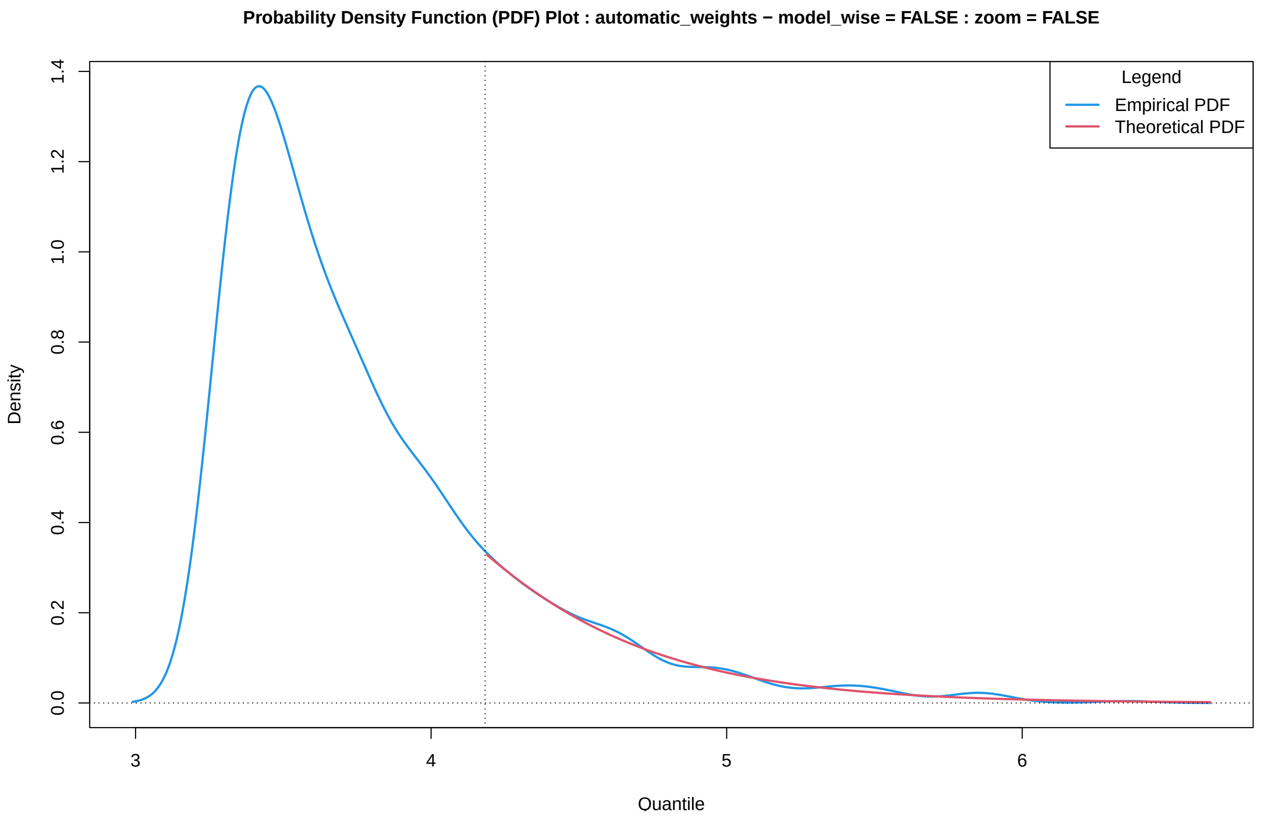
```
gev_mixture_model$pessimistic_weights_pw_scale
```

```
##                 8                 9                10                11
## 0.0618294491557781 0.0808148608370768 0.0698808689631223 0.0734441476123430
##                12                13                14                15
## 0.0657484715180107 0.0664876529155226 0.0768377849709341 0.0771811666642523
##                16                17                18                19
## 0.0722451805739313 0.0770086835082346 0.0979640977787443 0.0812386146063727
##                 20
## 0.0993190208956772
```

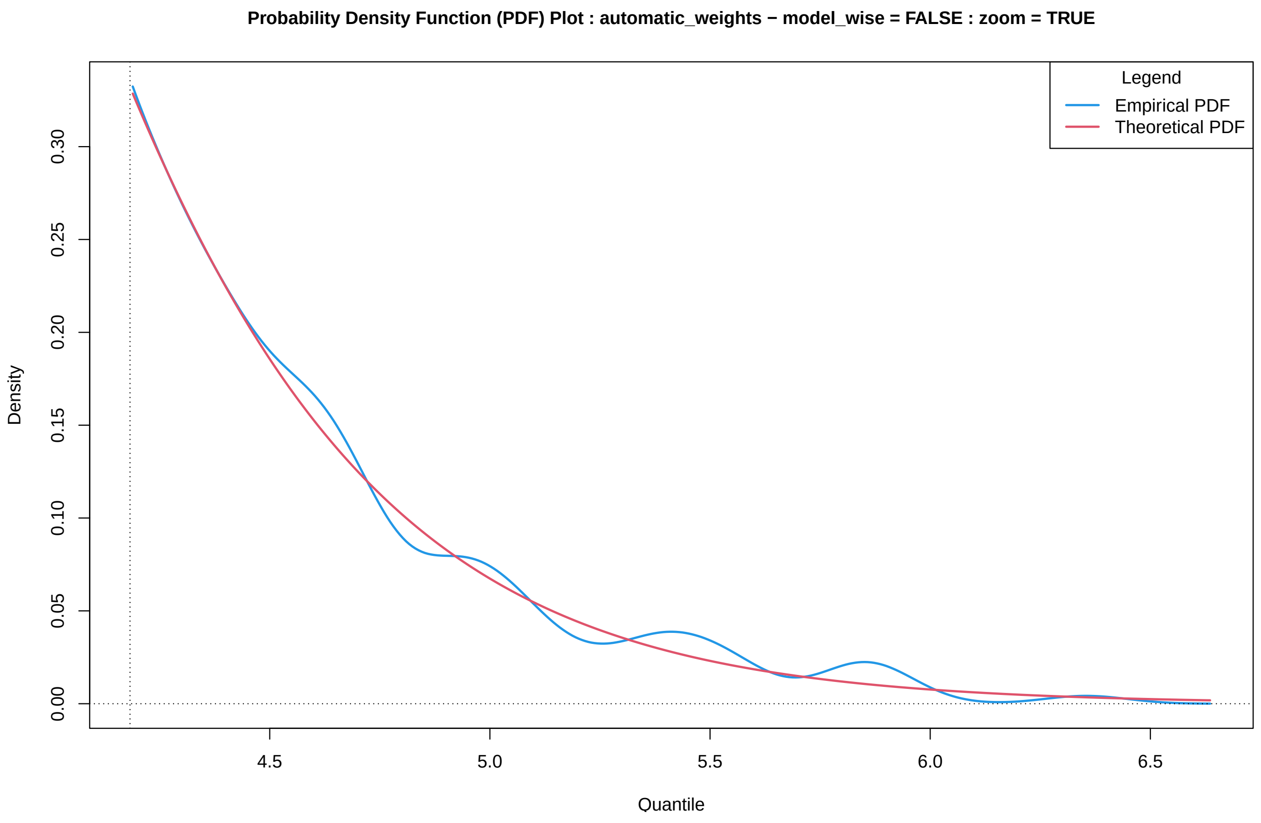
```r
gev_mixture_model$pessimistic_weights_pw_loc
```

```
##                  8                  9                 10                 11
## 0.1019652221906463 0.0707024782975647 0.0832555322215614 0.0809829753085424
##                 12                 13                 14                 15
## 0.0961004763963149 0.0918636400186987 0.0736345003605594 0.0728282543103013
##                 16                 17                 18                 19
## 0.0811703680233970 0.0762414709844394 0.0545110040990736 0.0653804308458378
##                 20
## 0.0513636469430631
```

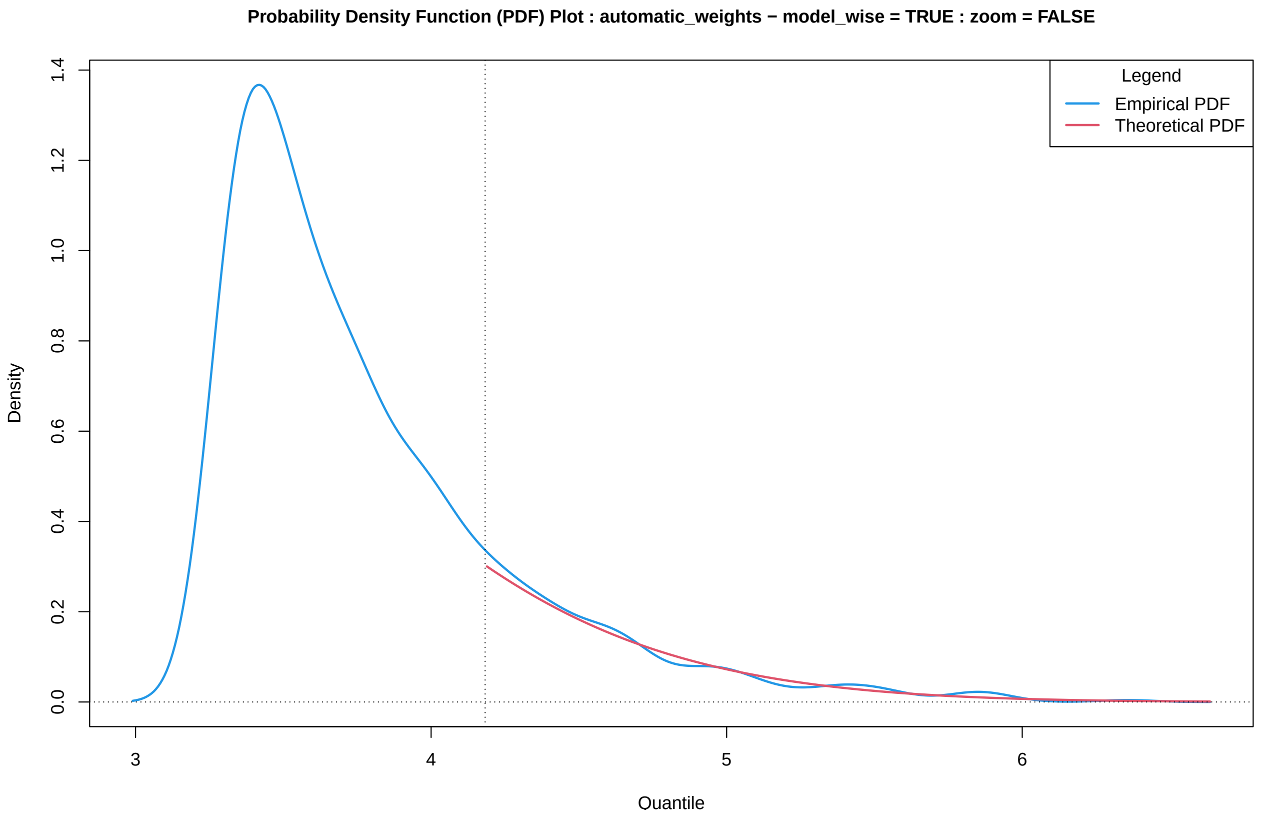
```r
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = FALSE,
                           xlab = "Quantile",
                           ylab = "Density",
                           main = "Probability Density Function (PDF) Plot")
```



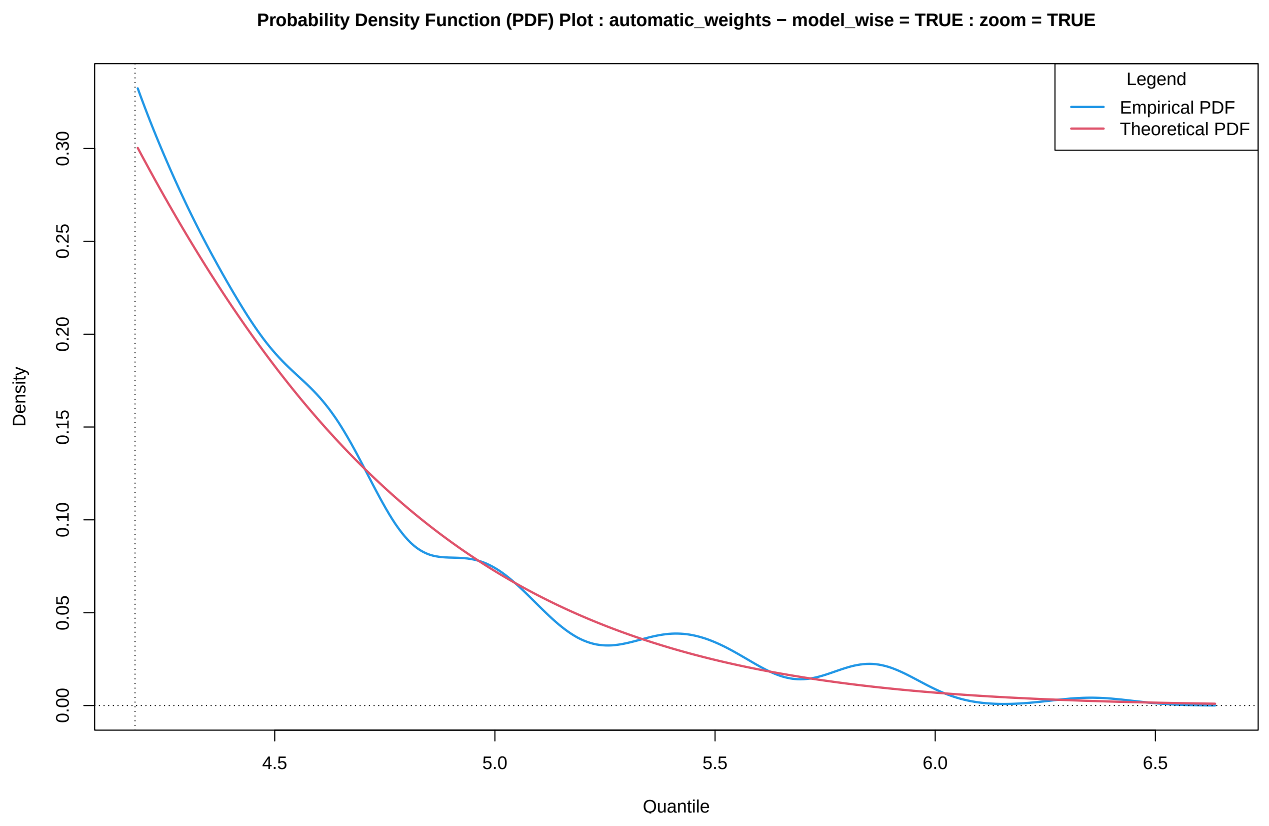
```r
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = TRUE,
                           xlab = "Quantile",
                           ylab = "Density",
                           main = "Probability Density Function (PDF) Plot")
```



```
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = TRUE,
                           zoom = FALSE,
                           xlab = "Quantile",
                           ylab = "Density",
                           main = "Probability Density Function (PDF) Plot")
```



```r
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = TRUE,
                           zoom = TRUE,
                           xlab = "Quantile",
                           ylab = "Density",
                           main = "Probability Density Function (PDF) Plot")
```



```r
estimator_types <- c("automatic_weights_mw",
                     "pessimistic_weights_mw",
                     "identic_weights_mw",
                     "automatic_weights_pw",
                     "pessimistic_weights_pw",
                     "identic_weights_pw",
                     "empirical",
                     "confidence_interval_mw",
                     "confidence_interval_pw")
```

```r
alpha <- 10^(-14)
```

```r
results_mw <- estimate_gev_mixture_model_quantile(gev_mixture_model,
                                                  alpha = alpha,
                                                  confidence_level = 0.95,
                                                  do.ci = TRUE,
                                                  estimator_type = estimator_types[1])

results_mw
```

```
##   lower         estimate upper
## 1    NA 16.9834382448298    NA
```

```r
results_pw <- estimate_gev_mixture_model_quantile(gev_mixture_model,
                                                  alpha = alpha,
                                                  confidence_level = 0.95,
                                                  do.ci = TRUE,
                                                  estimator_type = estimator_types[4])
```

```
results_pw
```

```
##   lower         estimate upper
## 1    NA 14.2758269333393    NA
```

```
quantile(x = x, probs = 1 - alpha)
```

```
##              100%
## 6.35340812153668
```

```
true_rl <- calculate_gev_inverse_cdf(p = 1 - alpha, loc = loc, scale = scale, shape = shape)
true_rl
```

```
## [1] 17.1184954496734
```

```
est_rl_pw <- estimate_gev_mixture_model_quantile(gev_mixture_model,
                                                 alpha = alpha,
                                                 confidence_level = 0.95,
                                                 do.ci = TRUE,
                                                 estimator_type = estimator_types[9])

est_rl_pw
```

```
##                 lower         estimate            upper
## 8  -11.2650226276556 18.2583972083514 47.7818170443583
## 9   5.48061813042822 7.98880416865956 10.4969902068909
## 10 0.345579682074224 10.8863787153817 21.4271777486892
## 11  2.79145403686066 9.25639162155555 15.7213292062504
## 12 -4.25121369107479 12.2090039259783 28.6692215430313
## 13 -5.23121823492217 12.1748809241234 29.5809800831691
## 14   2.1704900754929 8.88772846587113 15.6049668562494
## 15  2.31947565841163 8.79844053233887 15.2774054062661
## 16 -1.45165075168454 9.79527074152766 21.0421922347399
## 17  1.90222995161699 8.61516893298617 15.3281079143554
## 18  5.09805629251989 7.26400667557431 9.42995705862873
## 19   2.4704477598636   8.316063725134 14.1616796904044
## 20  4.85547224539689 7.31655952775769 9.77764681011849
```

```
est_rl_pw_range <- range(as.matrix(est_rl_pw))
est_rl_pw_range
```

```
## [1] -11.2650226276556  47.7818170443583
```

```
est_rl_mw <- estimate_gev_mixture_model_quantile(gev_mixture_model,
                                                 alpha = alpha,
                                                 confidence_level = 0.95,
                                                 do.ci = TRUE,
                                                 estimator_type = estimator_types[8])

est_rl_mw
```

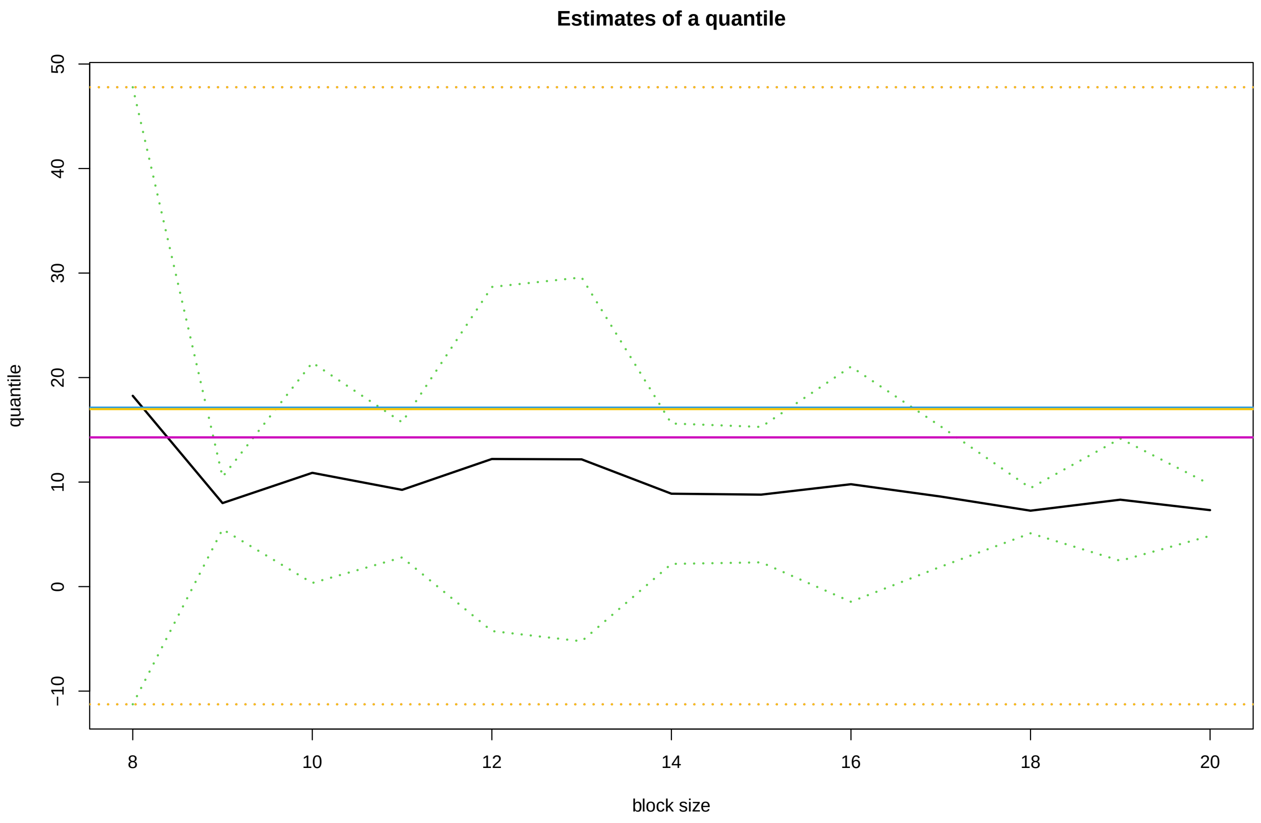
```
##                 lower         estimate            upper
## 8  -11.2650226276556 18.2583972083514 47.7818170443583
## 19   2.4704477598636   8.316063725134 14.1616796904044
```

```
est_rl_mw_range <- range(as.matrix(est_rl_mw))
est_rl_mw_range
```

```
## [1] -11.2650226276556  47.7818170443583
```

```
matplot(x = rownames(est_rl_pw),
        y = est_rl_pw,
        xlab = "block size",
        ylab = "quantile",
        main = "Estimates of a quantile",
        cex = 1,
        cex.lab = 1,
        cex.axis = 1,
        type = "l",
        lty = c("dotted", "solid", "dotted"),
        lwd = c(2,2,2),
        col = c(3, 1, 3))

abline(h = true_rl, col = 4, lwd = 2)
abline(h = results_mw[2], col = 7, lwd = 2)
abline(h = results_pw[2], col = 6, lwd = 2)
abline(h = est_rl_pw_range, col = 6, lty = "dotted", lwd = 2)
abline(h = est_rl_mw_range, col = 7, lty = "dotted", lwd = 2)
```


Estimates of a quantile
quantile
block size