Modeling extreme values with a GEV mixture probability distributions

Pascal Alain Dkengne Sielenou

September 28th, 2023

```
# library(xfun)
```

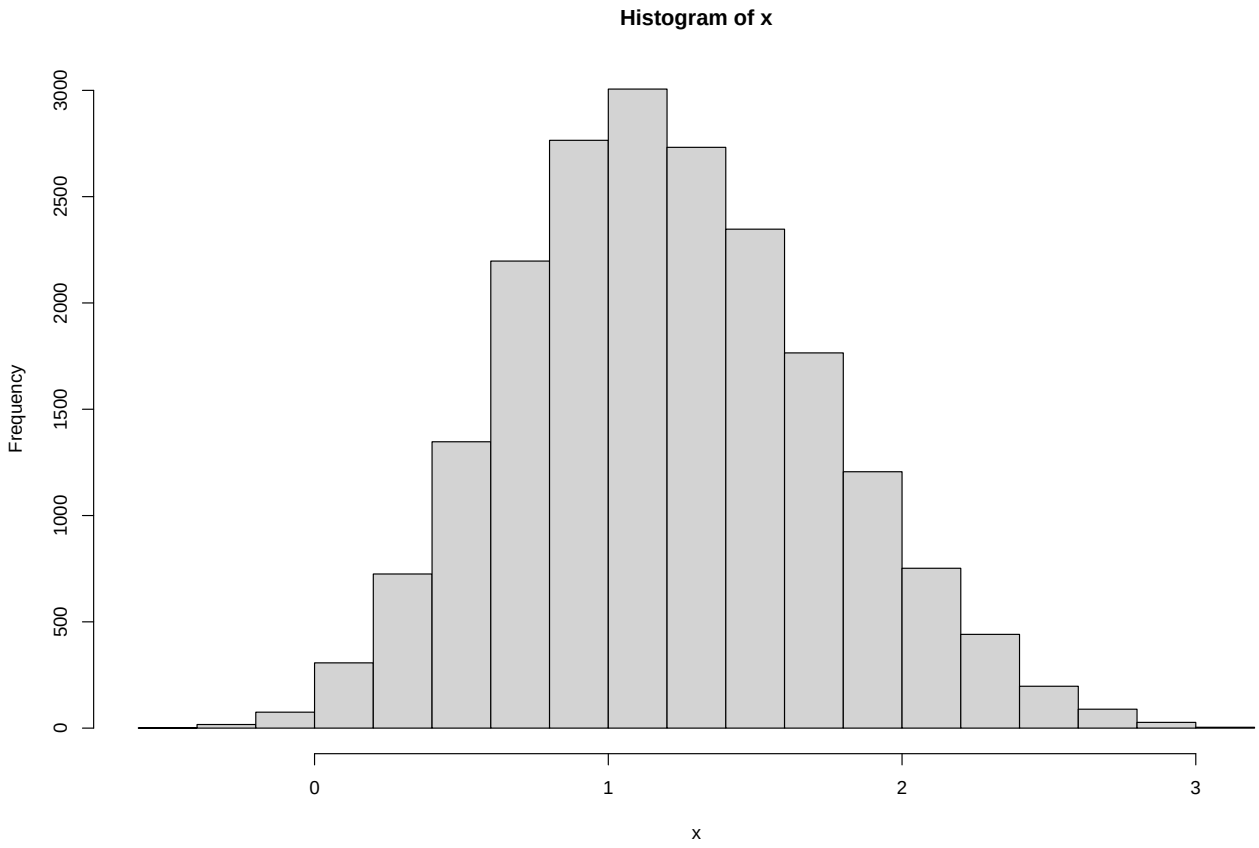
```
path <- ".."
```

```
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"))
```

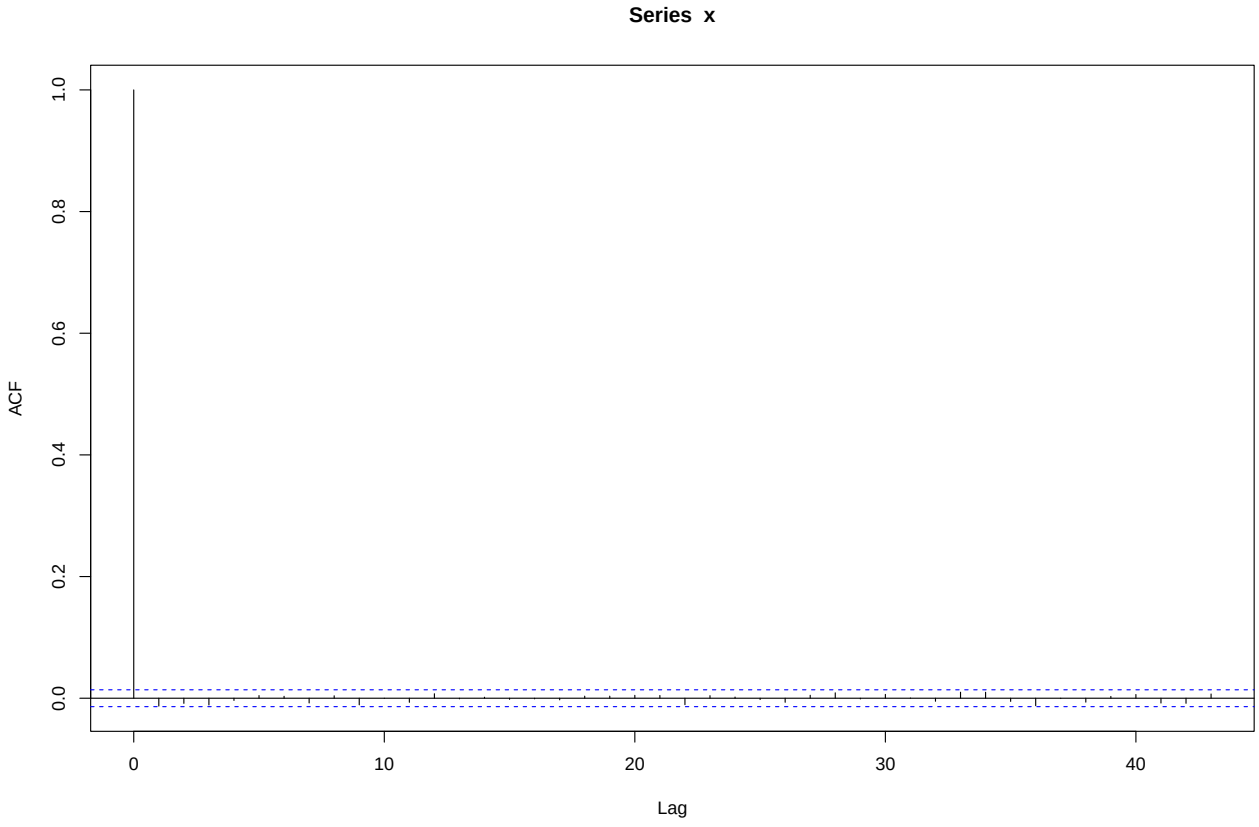
```
n <- 20000
```

```
loc <- 1
scale <- 0.5
shape <- -0.2
set.seed(1122)
x <- generate_gev_sample(n = n, loc = loc, scale = scale, shape = shape)
```

```
hist(x)
```


Histogram of x
Frequency
0
500
1000
1500
2000
2500
3000
0
1
2
3
x

```
acf(x)
```


Series x
ACF
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
Lag

```r
nlargest <- 1000

#
y <- extract_nlargest_sample(x, n = nlargest)
```

```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"
```

```r
gev_mixture_model$block_sizes
```

```
## [1] 13 14 15 16 17 18 19 20
```

```r
gev_mixture_model$normalized_gev_parameters_object
```

```
##            loc_star        scale_star         shape_star
## 13 2.08956208366854 0.327524489778421 -0.280892660028354
## 14 2.18876058945531 0.261019472501933 -0.231905090645186
## 15 2.13980732107708 0.285368631869109 -0.246729305521402
## 16 2.14172215905879 0.295501383126525 -0.260565738627570
## 17 2.07397694731061 0.332693344076672 -0.281953731129788
```

```
## 18 2.21599847189440 0.237485143400370 -0.204814695040313
## 19 1.98340141622742 0.396586953855325 -0.318699195455933
## 20 2.22768244776162 0.232744771719530 -0.202176224559891
```

```
gev_mixture_model$weighted_normalized_gev_parameters_object
```

```
##                            loc_star        scale_star         shape_star
## identic_weights     2.13261392955672 0.296115523790986 -0.253467080126055
## pessimistic_weights 2.13834076865898 0.298804358433140 -0.252049075531200
## automatic_weights   2.10619000395671 0.313700081328084 -0.281857068662443
```

```
gev_mixture_model$automatic_weights_mw_statistics
```

```
## $function_value
## [1] 0.00208766107239488
##
## $gradient_value
## [1] 6.19101561524782e-06
##
## $function_reduction
## [1] 0.00920994683146792
##
## $number_iterations
## [1] 1681
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_pw_statistics
```

```
## $function_value
## [1] 0.00187314434515873
##
## $gradient_value
## [1] 1.82123275336843e-05
##
## $function_reduction
## [1] 0.0102752076462562
##
## $number_iterations
## [1] 3025
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_mw
```

```
##                13                14                15                16
## 0.000000000000000 0.000000000000000 0.758509096115114 0.000000000000000
##                17                18                19                20
## 0.000000000000000 0.000000000000000 0.241490903884887 0.000000000000000
```

```
gev_mixture_model$pessimistic_weights_pw_shape
```

```
##                13                14                15                16
## 0.121532100772252 0.127633898921254 0.125755781778245 0.124027752763616
##                17                18                19                20
## 0.121403214962822 0.131138812129335 0.117023163793625 0.131485274878852
```

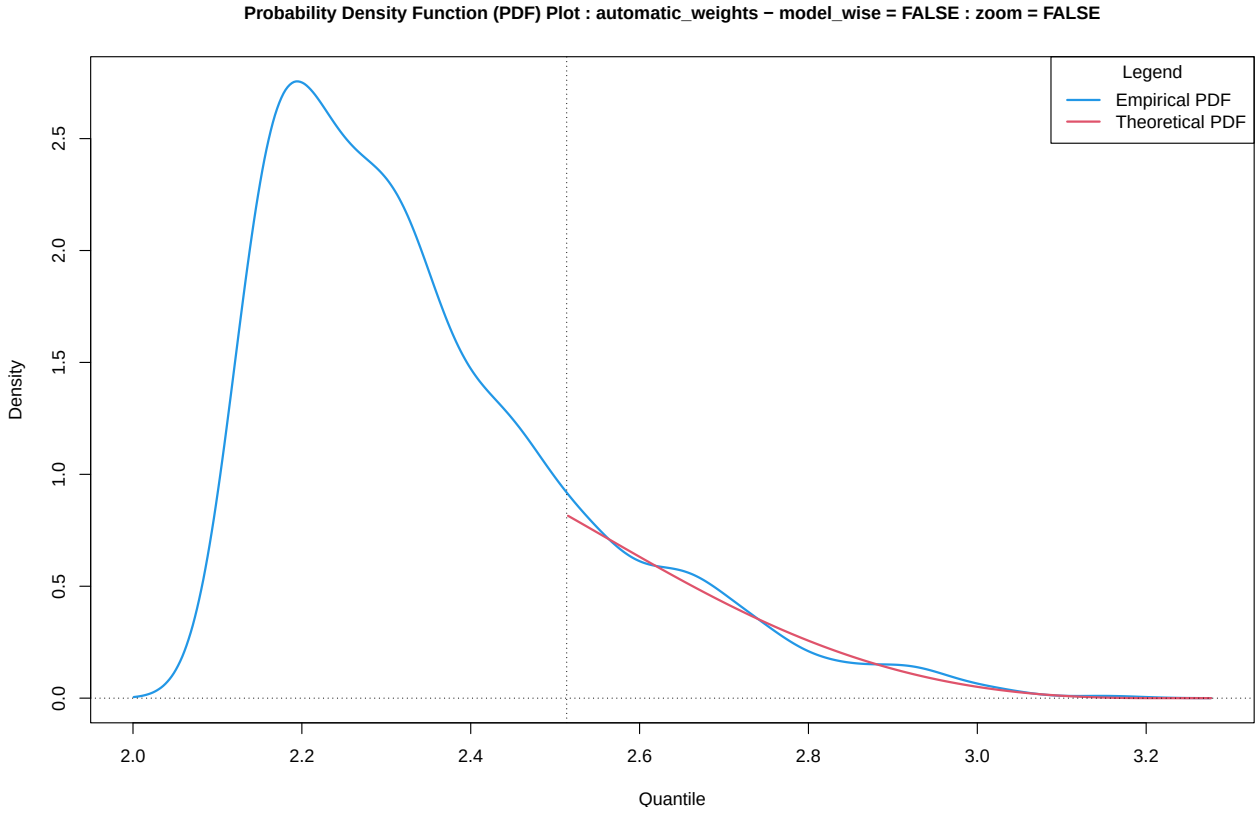
```
gev_mixture_model$pessimistic_weights_pw_scale
```

```
##                13                14                15                16
## 0.128815903128541 0.120527658807812 0.123498427060395 0.124756167325833
##                17                18                19                20
## 0.129483457522553 0.117724238914513 0.138026644363725 0.117167502876628
```

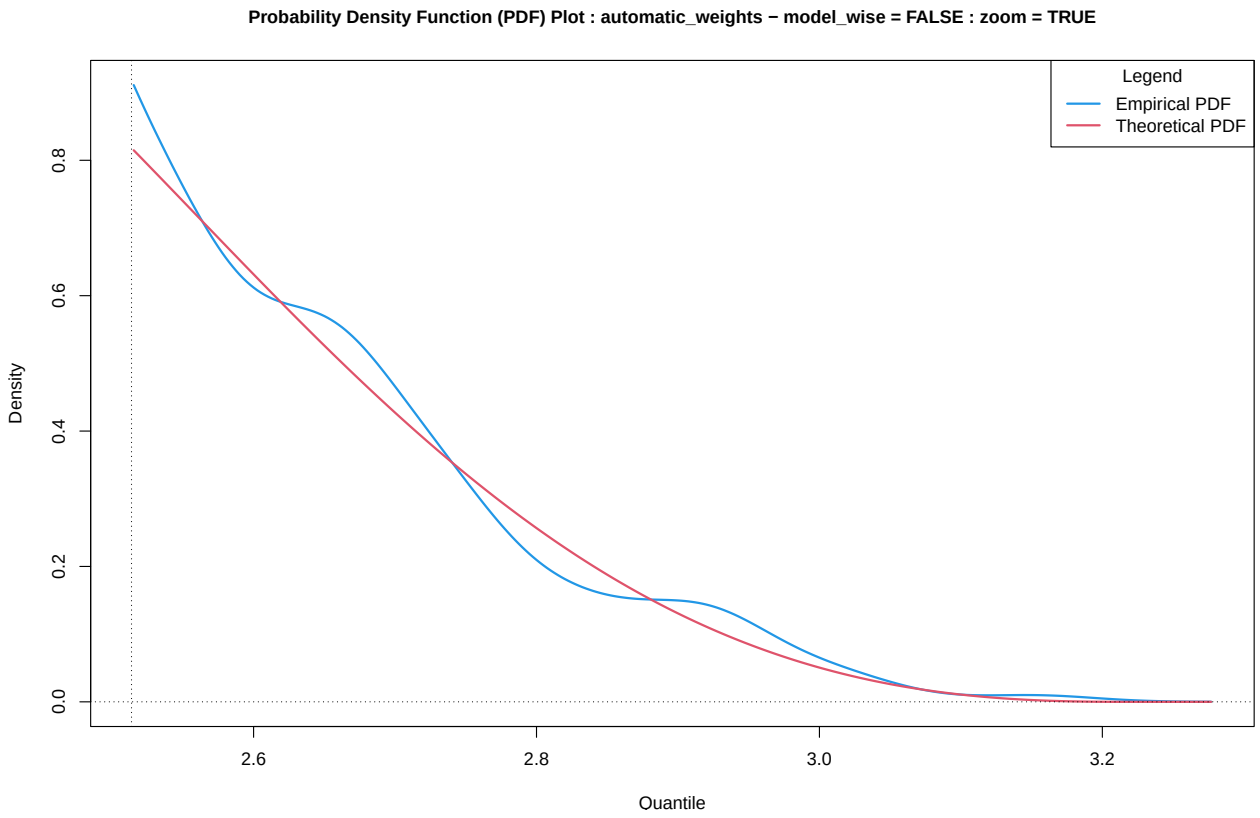
```
gev_mixture_model$pessimistic_weights_pw_loc
```

```
##                13                14                15                16
## 0.119387794221073 0.131838208228066 0.125539720585689 0.125780339110104
##                17                18                19                20
## 0.117541543573210 0.135478554443590 0.107363073665721 0.137070766172546
```

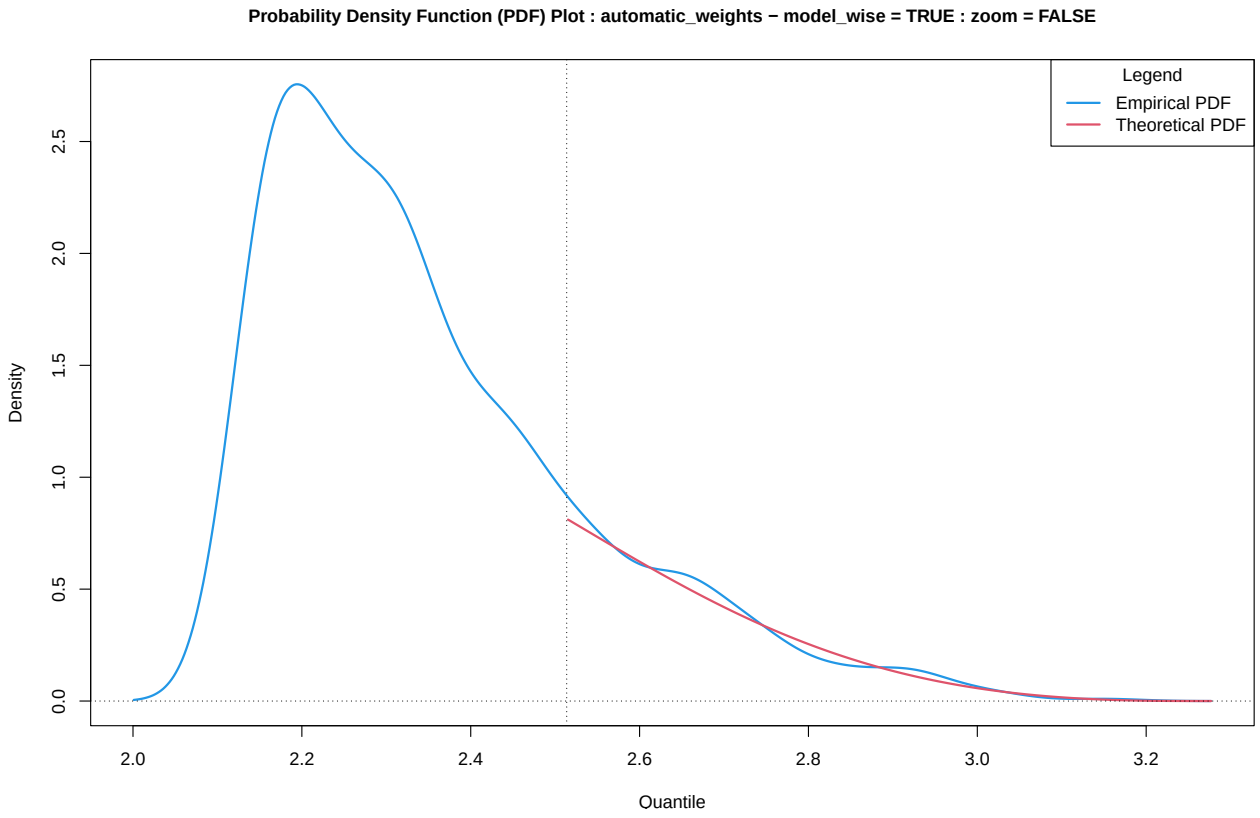
```
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = FALSE,
                           xlab = "Quantile",
                           ylab = "Density",
                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



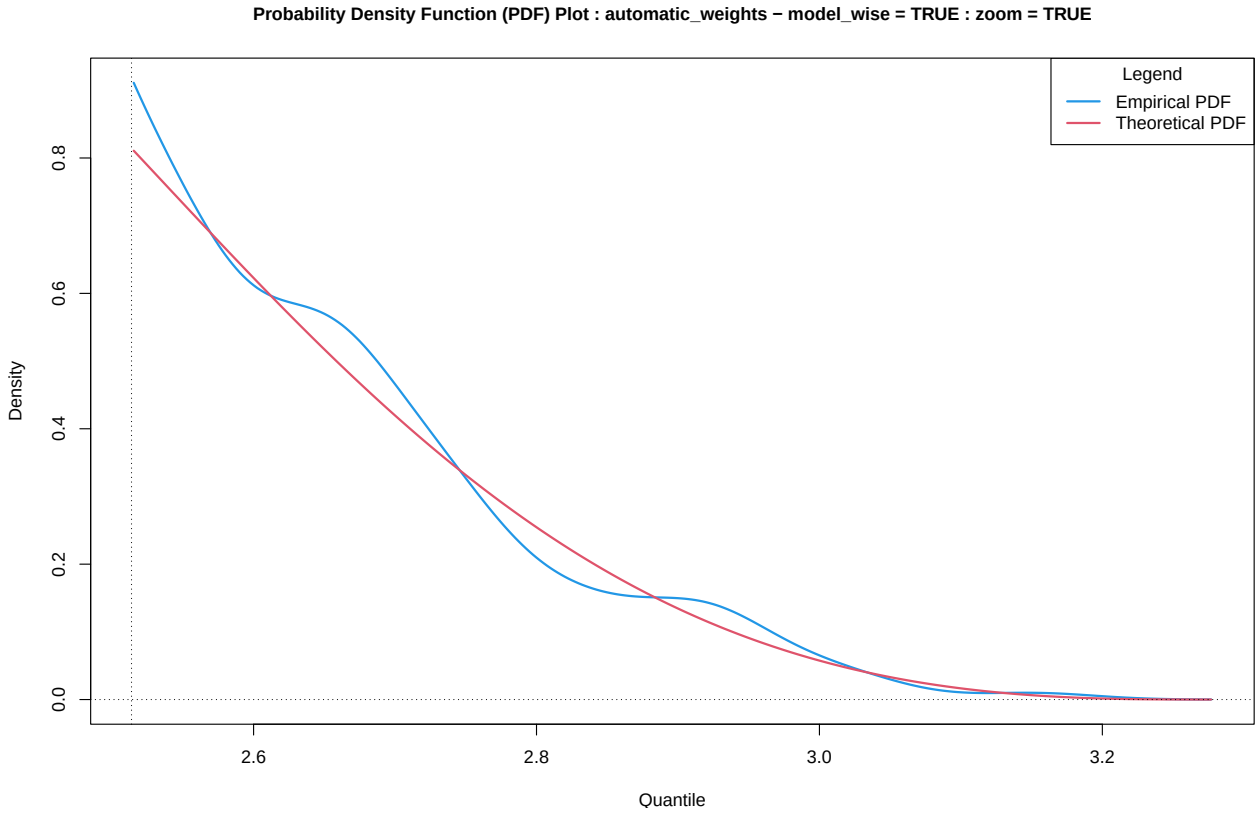
```r
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = TRUE,
                           xlab = "Quantile",
                           ylab = "Density",
                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




```r
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")
```




```
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")
```




```
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")
```

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

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

rl_mw
```

```
##   lower         estimate upper
## 1    NA 3.29550226022498    NA
```

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

```
rl_pw
```

```
##   lower         estimate upper
## 1    NA 3.21887255196577    NA
```

```
rl_empirical <- estimate_gev_mixture_model_quantile(gev_mixture_model,
                                                    alpha = alpha,
                                                    confidence_level = 0.95,
                                                    do.ci = TRUE,
                                                    estimator_type = estimator_types[7])

rl_empirical
```

```
##   lower         estimate upper
## 1    NA 3.15218214533986    NA
```

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

```
## [1] 3.49603840060645
```

```
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
## 13 3.06860081151579 3.25525745751543 3.44191410351507
## 14 3.01696472069186 3.31309800576784 3.60923129084382
## 15 3.02894123560677 3.29563087902332 3.56232052243987
## 16 3.04013857020801 3.27531138854189 3.51048420687577
## 17 3.06447115203976 3.25367510619623  3.4428790603527
## 18 2.89342404750546 3.37265259007345 3.85188113264143
## 19 3.07890056434617 3.22762988663039 3.37635920891461
## 20 2.86911170523342 3.37588286738543 3.88265402953744
```

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

```
## [1] 2.86911170523342 3.88265402953744
```

```
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
## 15 3.02894123560677 3.29563087902332 3.56232052243987
## 19 3.07890056434617 3.22762988663039 3.37635920891461
```

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

```
## [1] 3.02894123560677 3.56232052243987
```

```r
matplot(x = rownames(est_rl_pw),
        y = est_rl_pw,
        xlab = "block size",
        ylab = "quantile",
        main = "Estimates of a quantile",
        ylim = range(c(est_rl_pw_range, true_rl)),
        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 = rl_mw[2], col = 7, lwd = 2)
abline(h = rl_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)
```