Modeling extreme values with a GEV mixture probability distributions

Pascal Alain Dkengne Sielenou

September 28th, 2023

```
# library(xfun)
```

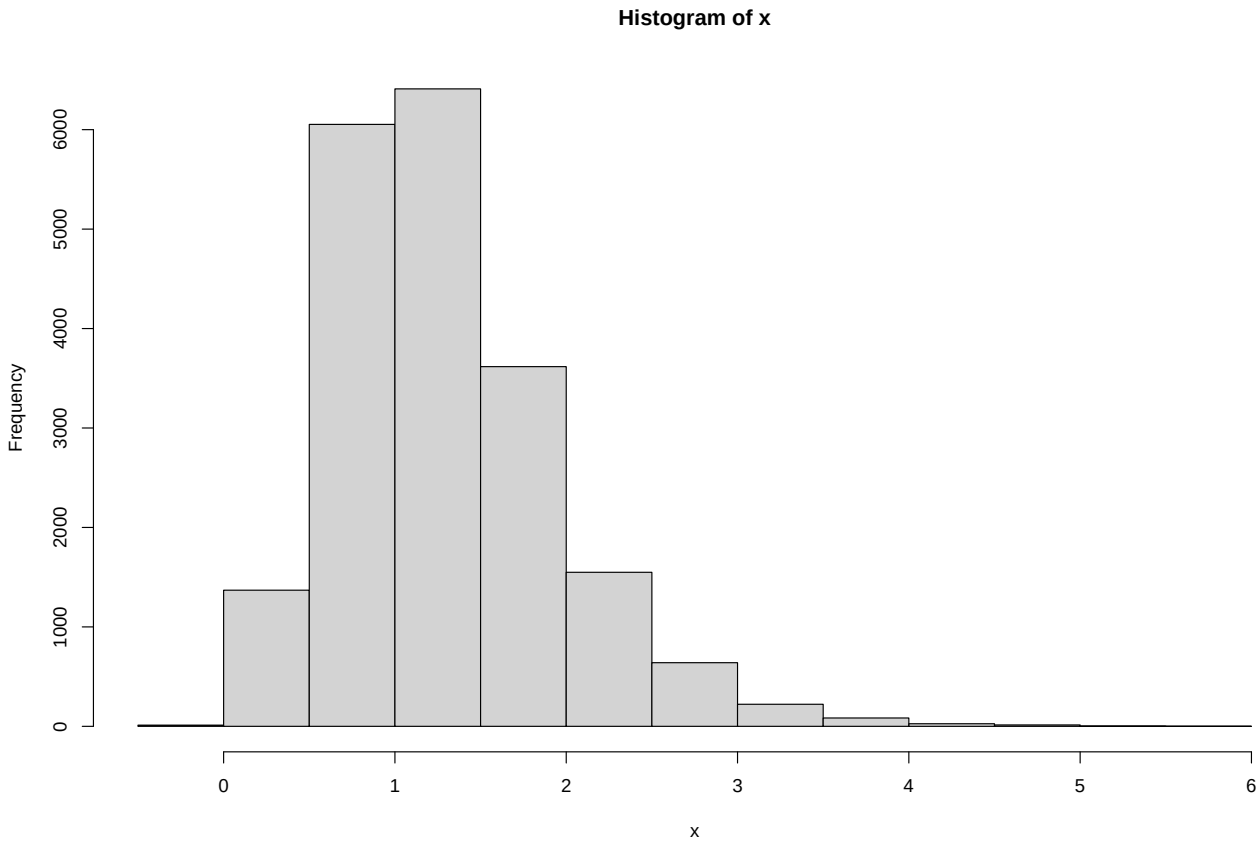
```
path <- ".."
```

```
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"))
```

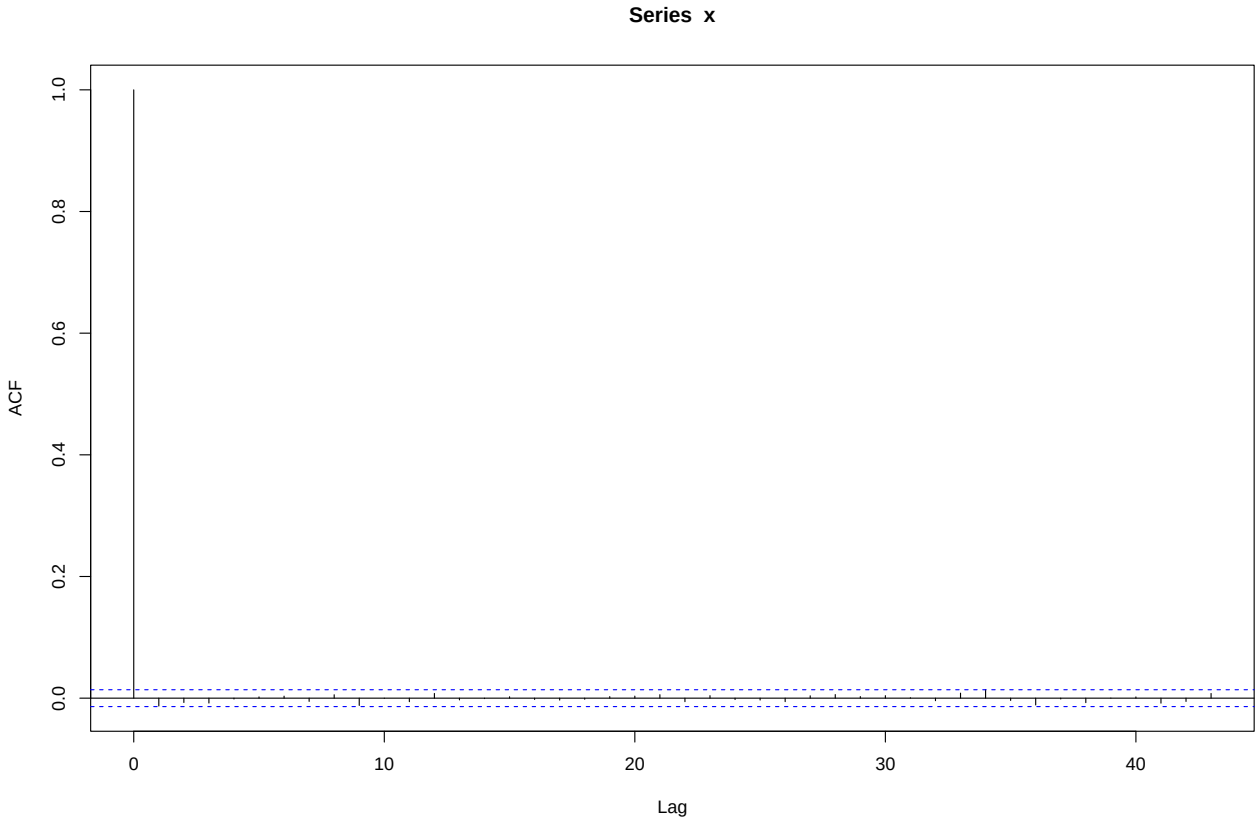
```
n <- 20000
```

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

```
hist(x)
```


Histogram of x
Frequency
0
1000
2000
3000
4000
5000
6000
0
1
2
3
4
5
6
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 2.36957061747757 0.648446505549683 -0.1108339993056009
## 14 2.58675104855494 0.526884116097328 -0.0660476072913345
## 15 2.48108225173543 0.566945800642789 -0.0795787767900970
## 16 2.47147185616499 0.598966396387417 -0.0948820023252217
## 17 2.32049567266110 0.667003423554400 -0.1177164743486123
```

```
## 18 2.66952408078310 0.464234802264545 -0.0294872665841885
## 19 2.10040561789880 0.797105734716942 -0.1536577861213932
## 20 2.69372505825355 0.458377925950788 -0.0282186580104815
```

```
gev_mixture_model$weighted_normalized_gev_parameters_object
```

```
##                            loc_star        scale_star          shape_star
## identic_weights     2.46162827544119 0.590995588145487 -0.0850528213471162
## pessimistic_weights 2.49384739855353 0.602523879660153 -0.0833986764999047
## automatic_weights   2.44731844537383 0.583250816315517 -0.0899371174369438
```

```
gev_mixture_model$automatic_weights_mw_statistics
```

```
## $function_value
## [1] 0.0022036799712727
##
## $gradient_value
## [1] 9.95587442687018e-06
##
## $function_reduction
## [1] 0.00997080672649736
##
## $number_iterations
## [1] 1747
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_pw_statistics
```

```
## $function_value
## [1] 0.0018488150779445
##
## $gradient_value
## [1] 6.54722768513583e-05
##
## $function_reduction
## [1] 0.010914006087512 4
##
## $number_iterations
## [1] 1254
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_mw
```

```
##                13                14                15                16
## 0.000000000000000 0.000000000000000 0.757605400046192 0.000000000000000
##                17                18                19                20
## 0.000000000000000 0.000000000000000 0.242394599953808 0.000000000000000
```

```
gev_mixture_model$pessimistic_weights_pw_shape
```

```
##                13                14                15                16
## 0.121717797502392 0.127293013525720 0.125582191001631 0.123675008650271
##                17                18                19                20
## 0.120882953989050 0.132033009434655 0.116615411964073 0.132200613932209
```

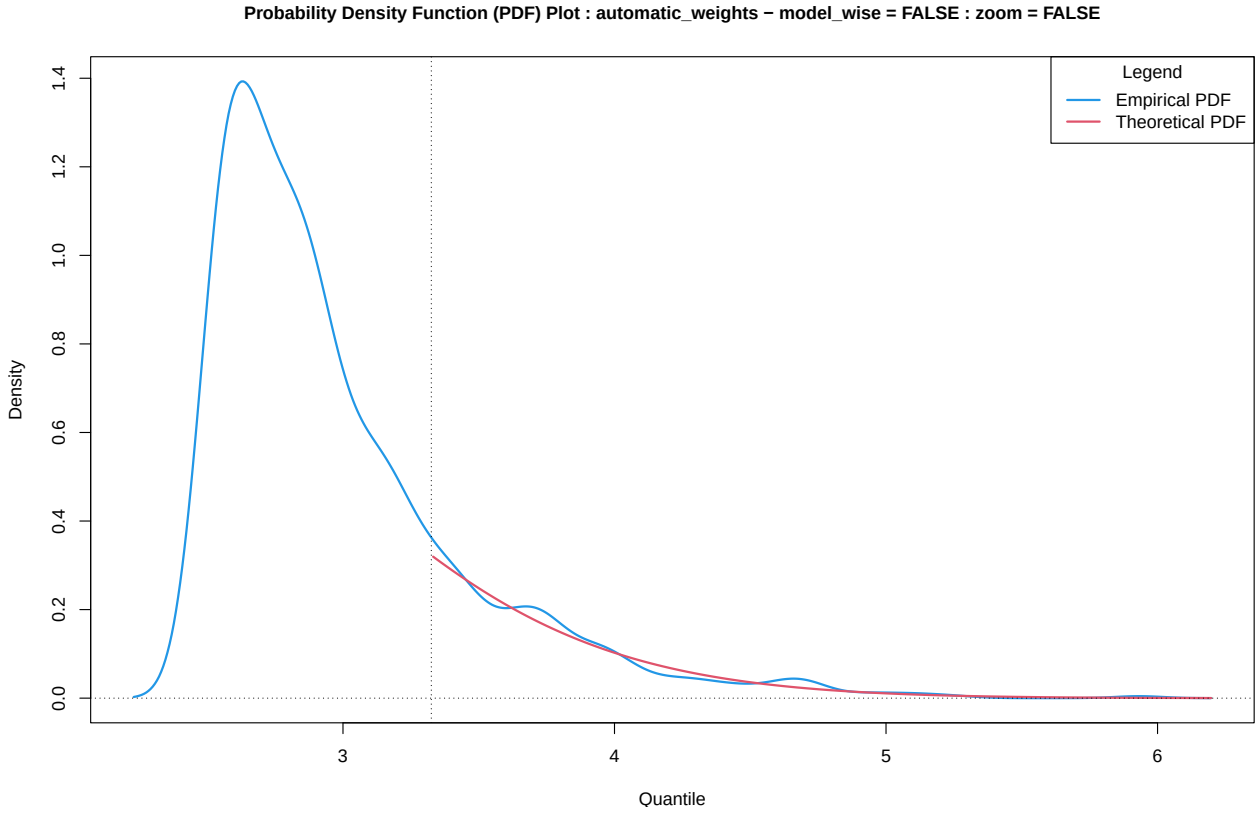
```
gev_mixture_model$pessimistic_weights_pw_scale
```

```
##                13                14                15                16
## 0.131636767516472 0.116569070696996 0.121333828910281 0.125281882582651
##                17                18                19                20
## 0.134102346235420 0.109490158117044 0.152735183878190 0.108850762062946
```

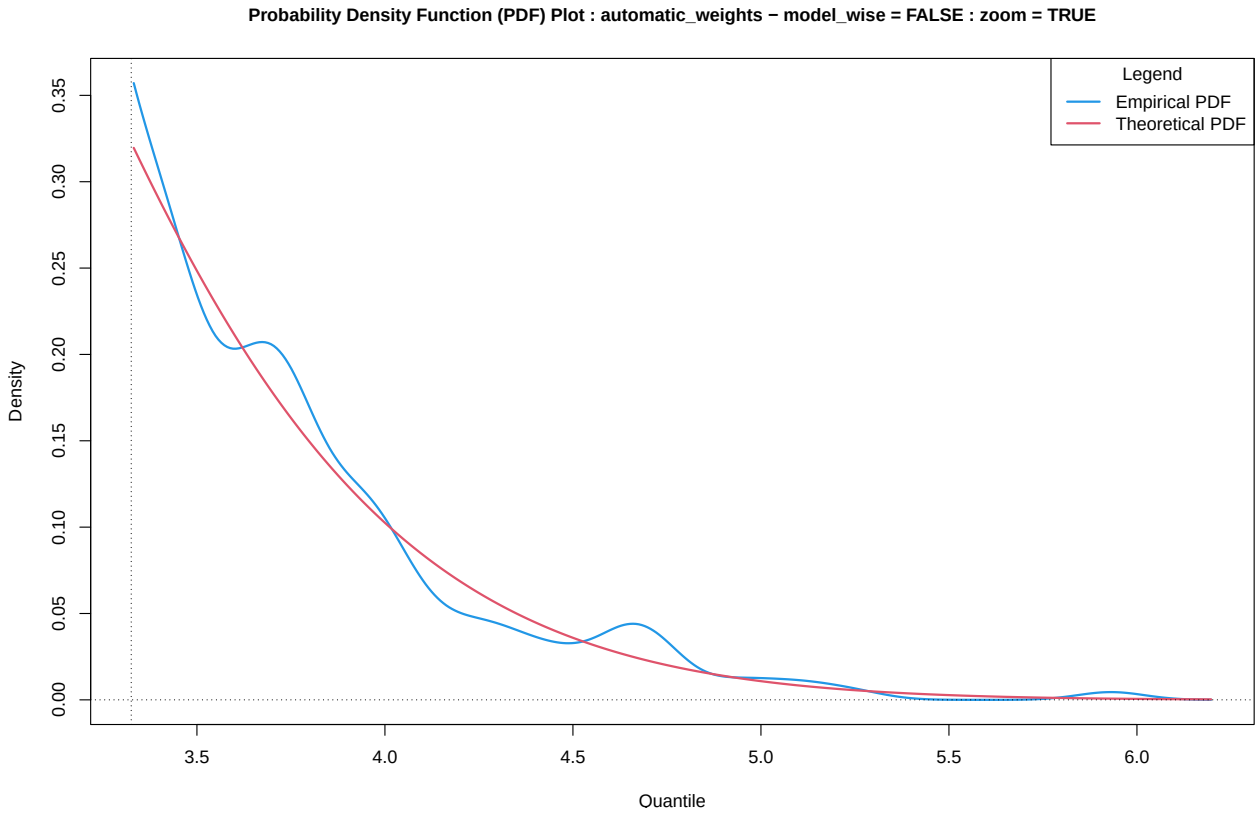
```
gev_mixture_model$pessimistic_weights_pw_loc
```

```
##                 13                 14                 15                 16
## 0.1121504599303331 0.1393546040796036 0.1253804852714684 0.1241813007596357
##                 17                 18                 19                 20
## 0.1067795487591039 0.1513802412107043 0.0856848784383149 0.1550884815508362
```

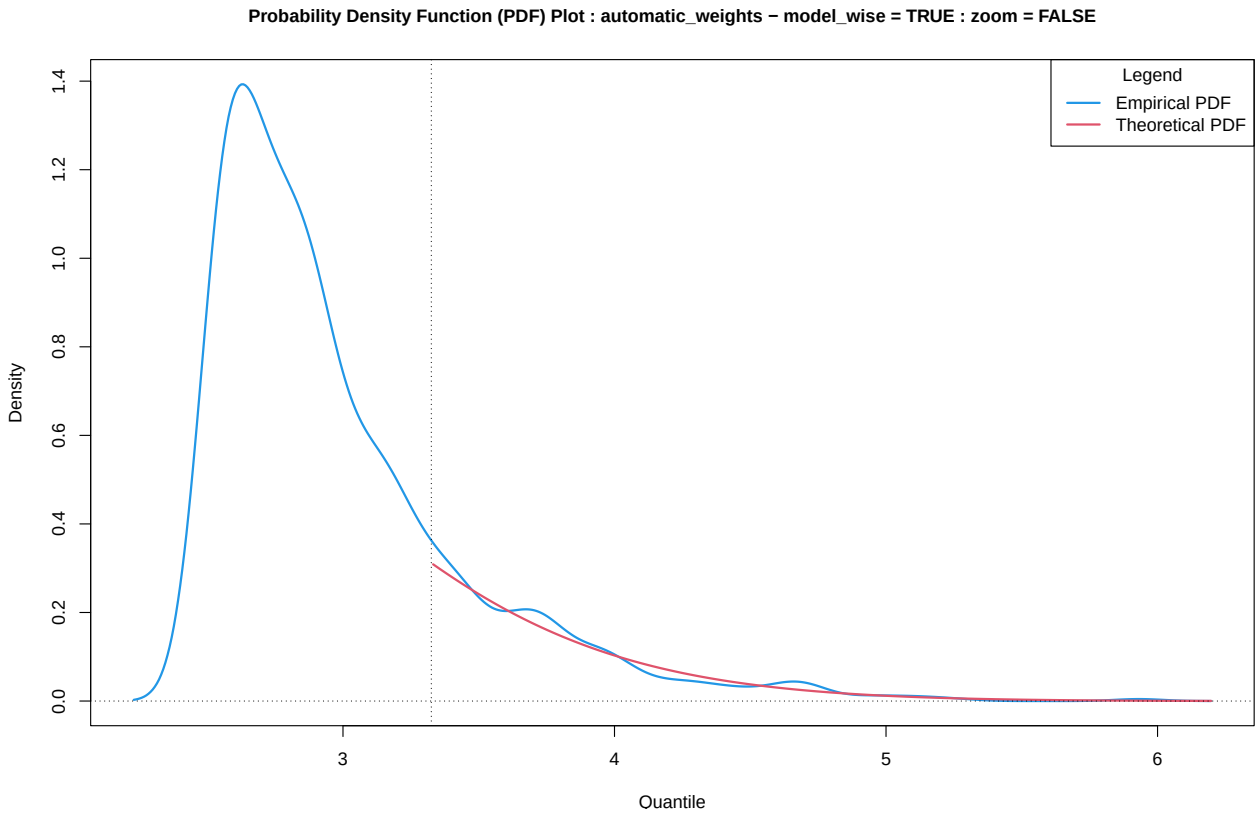
```
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = FALSE,
                           xlab = "Quantile",
                           ylab = "Density",
                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



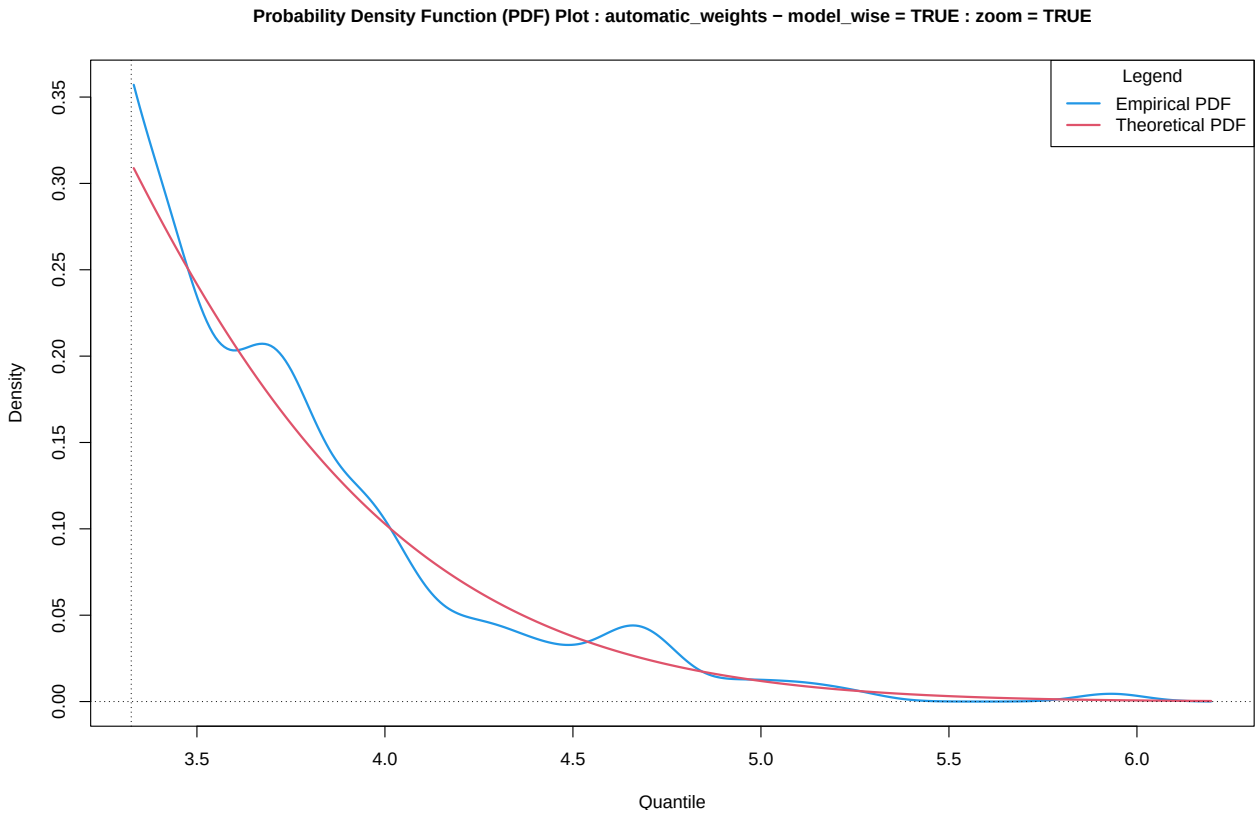
```r
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = TRUE,
                           xlab = "Quantile",
                           ylab = "Density",
                           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 8.8945875287946    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 8.4648964340765    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 5.93091768565999    NA
```

```
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
## 13  3.88362170381808 7.98769228590381 12.0917628679895
## 14  1.58509254217846 9.40784910998744 17.2306056777964
## 15  2.29426686575851  8.9106364265772 15.5270059873959
## 16  2.89360285957268 8.39075803436588 13.8879132091591
## 17  3.97954761873479 7.80515398155196 11.6307603443691
## 18 -6.01462738344398 11.7656683886257 29.5459641606954
## 19   4.5359530581328 7.22968576463637 9.92341847113993
## 20 -6.99553957425548  11.819797245357 30.6351340649695
```

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

```
## [1] -6.99553957425548 30.63513406496946
```

```
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 2.29426686575851  8.9106364265772 15.5270059873959
## 19  4.5359530581328 7.22968576463637 9.92341847113993
```

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

```
## [1]  2.29426686575851 15.52700598739588
```

```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)
```