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 <- 100000
```

```
loc <- 1
scale <- 0.5
shape <- 0.1
set.seed(1122)
x <- generate_gev_sample(n = n, loc = loc, scale = scale, shape = shape)
```

```
nlargest <- 1000
```

```
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.
```

```
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  4.41093011553657 0.638103481891605  0.11510222627281867
## 9  3.77629681338124 1.073089413254506 -0.04338622244019737
## 10 4.07852787288485 0.817121090692655  0.04837240201853115
## 11 4.02214000220968 0.908269360824931  0.01044469958217165
## 12 4.32778618779844 0.727979412170171  0.07007496716097716
## 13 4.26271439516313 0.734193509833696  0.07185389175570311
## 14 3.89334750030963 0.951704808250416  0.00385713023120079
## 15 3.86779380075390 0.964524224552648 -0.00150125840308322
## 16 4.09359138997512 0.836871571673760  0.03773286282616022
## 17 3.95801695846076 0.959969463357112 -0.00503486830680257
## 18 3.28839965519033 1.403813693362337 -0.10692997347133250
## 19 3.70112855387292 1.032095428268794 -0.02039043724145559
## 20 3.20893837498915 1.403623344840168 -0.10137802358167794
```

```
gev_mixture_model$weighted_normalized_gev_parameters_object
```

```
##                             loc_star        scale_star          shape_star
## identic_weights     3.91458550927121 0.957796830997908 0.00606287664638566
## pessimistic_weights 4.02110439215748 1.012850989405943 0.00993324702508542
## automatic_weights   4.06763183308287 0.769227493893544 0.08007891259251206
```

```
gev_mixture_model$automatic_weights_mw_statistics
```

```
## $function_value
## [1] 0.00118951250924189
##
## $gradient_value
## [1] 2.24603314680216e-06
##
## $function_reduction
## [1] 0.0169652368521281
##
## $number_iterations
```

```
## [1] 1615
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_pw_statistics
```

```
## $function_value
## [1] 0.000759241903749239
##
## $gradient_value
## [1] 2.10149745189514e-05
##
## $function_reduction
## [1] 0.0249243792756868
##
## $number_iterations
## [1] 2566
##
## $convergence
## [1] 0
##
## $message
## [1] "Successful convergence"
```

```
gev_mixture_model$automatic_weights_mw
```

```
##                 8                 9                10                11
## 0.127596316598068 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 0.872403683401932
##                20
## 0.000000000000000
```

```
gev_mixture_model$pessimistic_weights_pw_shape
```

```
##                  8                  9                 10                 11
## 0.0856187372372765 0.0730698405348967 0.0800918681661356 0.0771110526834069
##                 12                 13                 14                 15
## 0.0818490660333114 0.0819947989354235 0.0766047477664741 0.0761953675453930
##                 16                 17                 18                 19
## 0.0792442447529389 0.0759265979830743 0.0685711541190995 0.0747696077297098
##                 20
## 0.0689529165128599
```

```
gev_mixture_model$pessimistic_weights_pw_scale
```

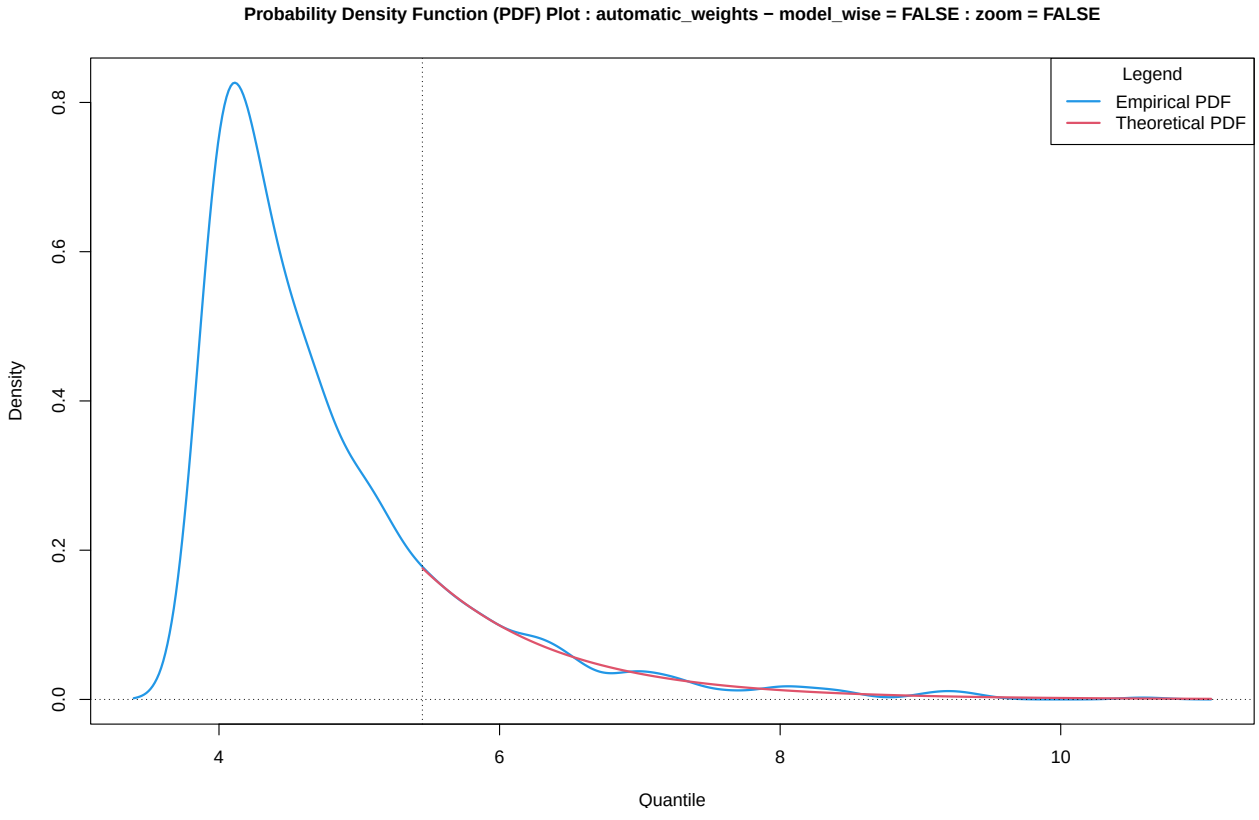
```
##                  8                  9                 10                 11
## 0.0543965328799853 0.0840394515057471 0.0650605273630687 0.0712693454184568
##                 12                 13                 14                 15
## 0.0595119033166772 0.0598828675061495 0.0744331751022149 0.0753935072578855
##                 16                 17                 18                 19
```

```
## 0.0663582774804612 0.0750508887018099 0.1169808948375882 0.0806639988136976
##                 20
## 0.1169586298162581
```

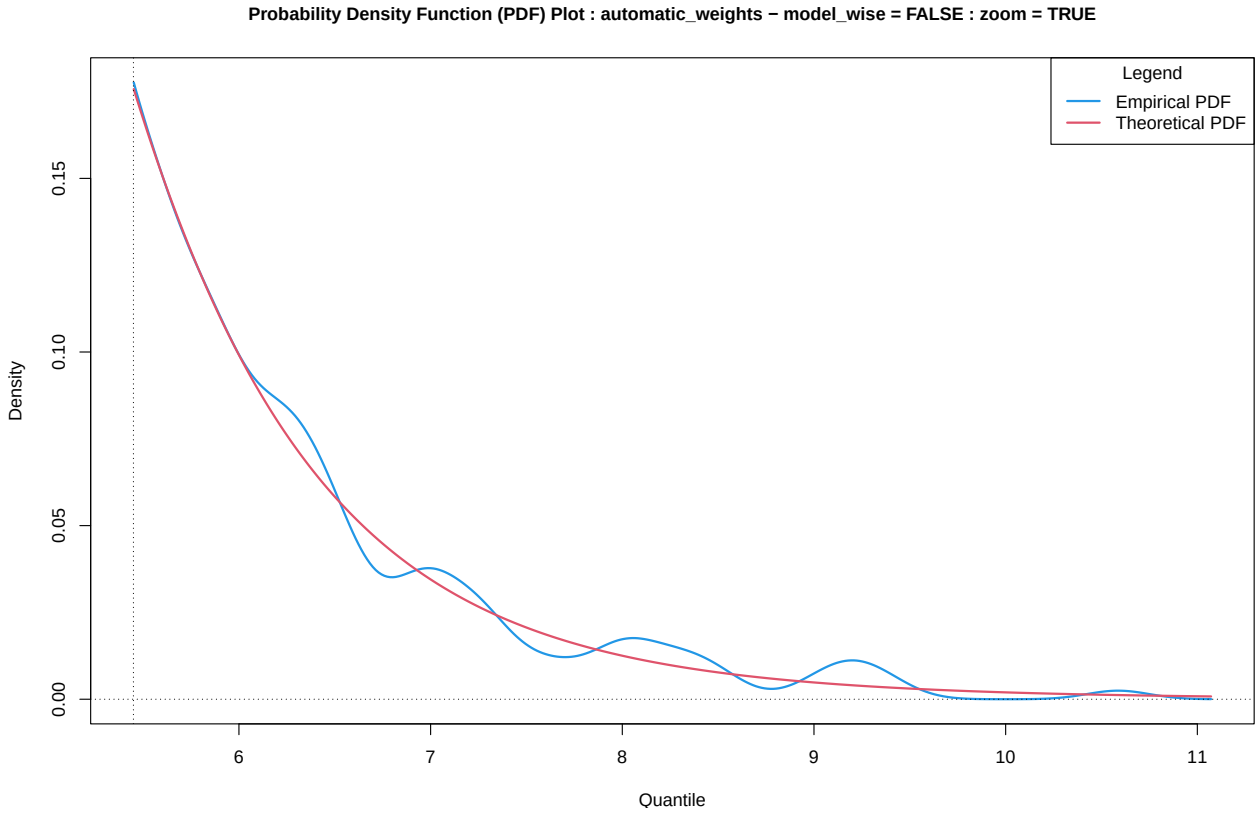
```
gev_mixture_model$pessimistic_weights_pw_loc
```

```
##                  8                  9                 10                 11
## 0.1195285626955343 0.0633656590307636 0.0857257389683890 0.0810256075949297
##                 12                 13                 14                 15
## 0.1099924177032042 0.1030629164239071 0.0712341790418545 0.0694369430441415
##                 16                 17                 18                 19
## 0.0870268451170782 0.0759930739660641 0.0389011923115678 0.0587771867700854
##                 20
## 0.0359296773324808
```

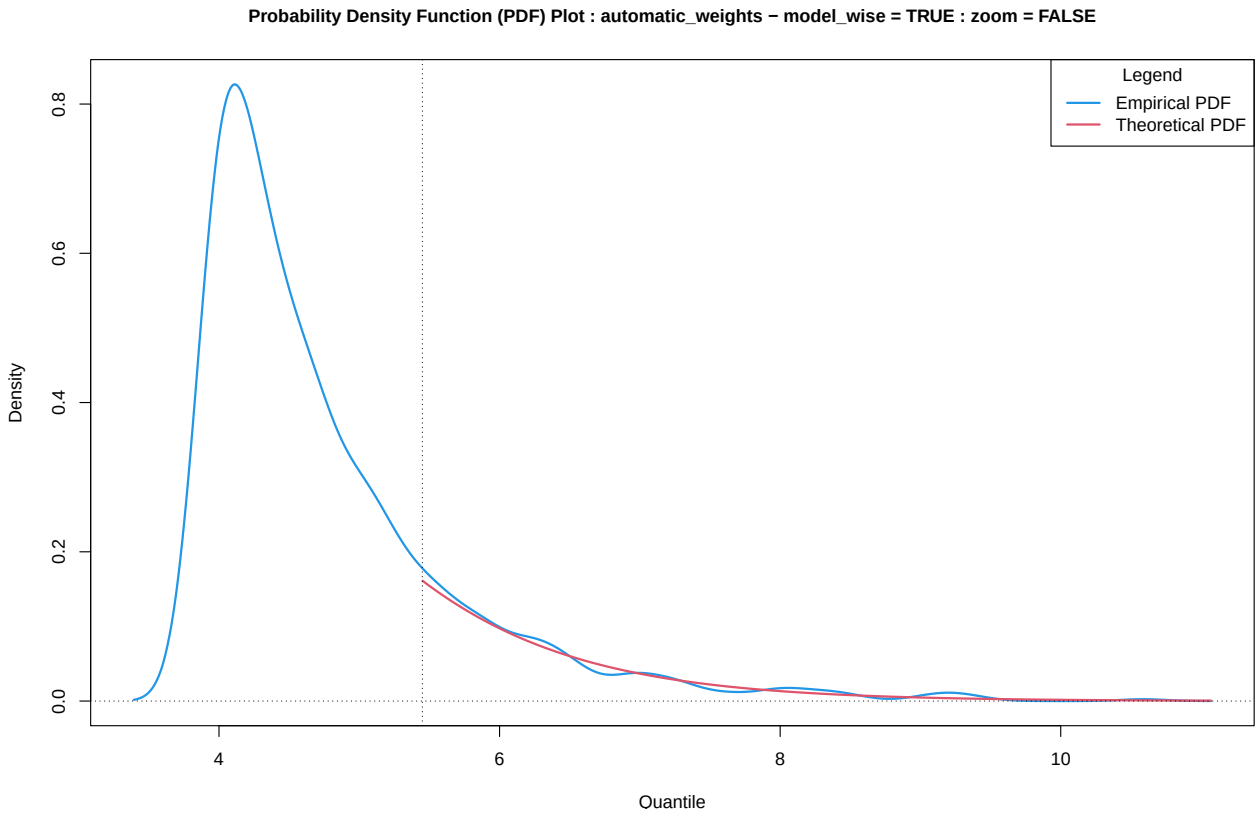
```
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = FALSE,
                           xlab = "Quantile",
                           ylab = "Density",
                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



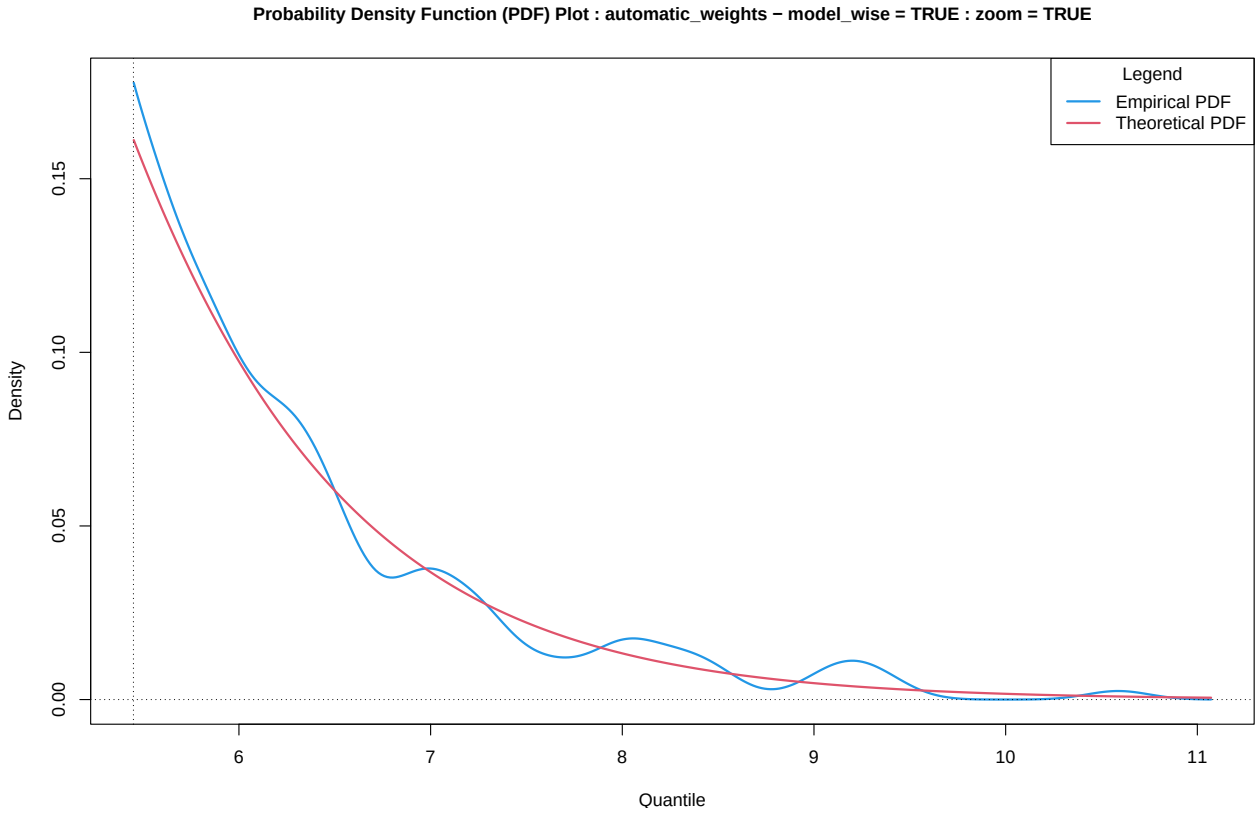

```
plot_gev_mixture_model_pdf(gev_mixture_model,
                           type = "automatic_weights",
                           model_wise = FALSE,
                           zoom = TRUE,
                           xlab = "Quantile",
                           ylab = "Density",
                           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")
```




```
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)
```

```
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 104.090592410215    NA
```

```
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 82.2597086394559    NA
```

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

```
##             100%
## 10.5868368546653
```

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

```
## [1] 121.604364466665
```

```
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   -238.81336923665 132.248595965334 503.310561167319
## 9   2.45701060722118 21.0511442025568 39.6452777978925
## 10 -70.6684597153869 51.4778714569968  173.62420262938
## 11 -30.7088167844667 33.1134414247796 96.9356996340258
## 12 -132.769479639366 65.9592145127243 264.687908664815
## 13  -154.71278319739 68.4501374426045 291.613058082599
## 14 -42.0304546291799 31.6419910741868 105.314436777553
## 15 -36.9043501247963 29.9723761953831 96.8491025155624
## 16 -108.306534809138 44.8262378880184 197.959010585175
## 17 -42.9399866174919 28.7201550520043 100.380296721501
## 18  1.28431358521505 15.7325170283517 30.1807204714884
## 19 -33.6766645444586 25.5030950777275 84.6828546999137
## 20 -1.12982486572713 16.2133207167218 33.5564662991708
```

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

```
## [1] -238.813369236650  503.310561167319
```

```
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   -238.81336923665 132.248595965334 503.310561167319
## 19 -33.6766645444586 25.5030950777275 84.6828546999137
```

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

```
## [1] -238.813369236650  503.310561167319
```

```r
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)
```