Extremile Regression: A concrete application in geology to extreme seismic moments of earthquakes conditional on their geographical locations

Abdelaati Daouia, Thibault Laurent, Gilles Stupfler

The data example in the paper **Extremile Regression** written by A. Daouia, I. Gijbels and G. Stupfler (2018): A concrete application in geology to extreme seismic moments of earthquakes conditional on their geographical locations

Abdelaati Daouia (Toulouse School of Economics, University of Toulouse 1-Capitole), Thibault Laurent (Toulouse School of Economics, CNRS) and Gilles Stupfler (University of Nottingham, University Park)

November 2018

Contents

This web page presents a detailed illustration of the application in the paper "Extremile Regression" written by Abdelaati Daouia, Irène Gijbels and Gilles Stupfler (2018). We consider the earthquakes with moment magnitudes larger than 2.5, which occurred from 1960 (January 1st) to 2018 (October 3rd). We propose to evaluate the estimates of three rival extreme risk measures over a regular grid that can be specified by the user. These three estimated measures, referred to as regression quantiles (Value-at-Risk) $\hat{q}^*_{1-p_n}(x)$, regression extremiles $\hat{\xi}^*_{1-p_n}(x)$, and regression tail conditional means (Expected Shortfall) $\hat{\nu}^*_{1-p_n}(x)$, allow to assess extreme seismic moments in the presence of covariates, namely the longitude and latitude of earthquakes.

The .pdf of this page is available here.

The model assumption

For both regression extremiles and regression Expected Shortfall to be well defined, Daouia *et al.* (2018) assume that $\mathbb{E}(Y|X = x) < \infty$. They also consider the maximum domain of attraction of heavy-tailed conditional distributions that better describe the tail structure and sparseness of the seismic moment data. The model assumption of heavy tails can be expressed in terms of the conditional survival function $\overline{F}(\cdot|x)$ as

$$\overline{F}(y|x) = y^{-1/\gamma(x)}\ell(y|x)$$

where $0 < \gamma(x) < 1$ and $\ell(\cdot|x)$ is a slowly varying function at infinity, *i.e.*

$$\forall y > 0, \ \lim_{t\to\infty} \frac{\ell(ty|x)}{\ell(t|x)} = 1.$$

The index $\gamma(x) > 0$ tunes the tail heaviness of the conditional survival function $\overline{F}(\cdot|x)$, with higher positive values indicating heavier conditional tails. Note that the assumption $0 < \gamma(x) < 1$ is tailored to the requirement that $\mathbb{E}(Y|X = x) < \infty$. Note also that, when the seismic moment scale is expressed in Dyne-cm units, Goegebeur *et al.* (2014) have found that the estimates of $\gamma(x)$ can be much larger than 1 such as, for instance, in Indonesia and its surroundings. By contrast, when the seismic moment scale is expressed in Newton metres units, as is the case in Daouia *et al.* (2018), the estimates of $\gamma(x)$ are found to be typically smaller than 1, and hence the model assumption is well satisfied.

Before starting

```
install.packages(c("rgdal", "raster", "classInt",
                   "RColorBrewer", "snowfall", "rgeos",
                   "GISTools"))
```

First install and then load the following packages:

```
library("rgdal")
library("raster")
library("classInt")
library("RColorBrewer")
library("snowfall")
library("rgeos")
library("GISTools")
```

Information about the session:

```
sessionInfo()
```

```
## R version 3.5.1 (2018-07-02)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.5 LTS
##
## Matrix products: default
## BLAS: /usr/lib/openblas-base/libblas.so.3
## LAPACK: /usr/lib/libopenblasp-r0.2.18.so
##
## locale:
##  [1] LC_CTYPE=fr_FR.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=fr_FR.UTF-8        LC_COLLATE=fr_FR.UTF-8
##  [5] LC_MONETARY=fr_FR.UTF-8    LC_MESSAGES=fr_FR.UTF-8
##  [7] LC_PAPER=fr_FR.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=fr_FR.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
```

```
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] GISTools_0.7-4     MASS_7.3-51.1      maptools_0.9-4
##  [4] rgeos_0.3-28       snowfall_1.84-6.1  snow_0.4-3
##  [7] RColorBrewer_1.1-2 classInt_0.2-3     spData_0.2.9.4
## [10] raster_2.6-7       rgdal_1.3-4        sp_1.3-1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.18    knitr_1.20      magrittr_1.5    lattice_0.20-38
##  [5] stringr_1.3.1   tools_3.5.1     parallel_3.5.1  grid_3.5.1
##  [9] e1071_1.7-0     htmltools_0.3.6 class_7.3-14    yaml_2.2.0
## [13] rprojroot_1.3-2 digest_0.6.17   evaluate_0.11   rmarkdown_1.10
## [17] stringi_1.2.4   compiler_3.5.1  backports_1.1.2 foreign_0.8-71
```

Depending on the utilized machine, you may need to run the following code:

```
require("gpclib")
gpclibPermit()
```

1 Preparation code

Import first the world country boundaries into R (the source of the data can be found here):

```
world <- readOGR(dsn="Donnees/World WGS84", layer="Pays_WGS84")
```

1.1 Choosing a region

Users may select one country (or several countries) in **world** object. It is possible to select any country, but we strongly recommend to choose an area with a high seismic activity so that the sample size is large enough, and hence the results would be more reliable.

The list of available countries is given in:

```
world@data$NOM
```

For our illustration purposes, we use Indonesia as an example of the geographical zone of interset:

```
country <- "Indonesia"
```

By changing only very few commands presented hereafter, users can make their own analysis for any other country or region in the world.

1.2 Importing data

Import the earthquakes' data recorded in Indonesia during the period from January 1st, 1960 to October 3rd, 2018. The source of the data (on the moment magnitude scale) can be found here. Note that we restrict ourselves to earthquakes with moment magnitudes larger than 2.5.

```
seisme <- read.csv2("data/indonesia.csv")
```

Convert the earthquakes as a **Spatial** object and define the coordinate reference system (CRS) of the geographical data :

```
coordinates(seisme) <- ~ Longitude + Latitude
proj4string(seisme) <- CRS("+proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0")
```

Import the fault lines observed in the world (Source : here).

```
lyr <- ogrListLayers("data/doc.kml")
faults1 <- readOGR("data/doc.kml", lyr[1])
faults2 <- readOGR("data/doc.kml", lyr[2])
faults1 <- spTransform(faults1,  proj4string(seisme))
faults2 <- spTransform(faults2,  proj4string(seisme))
```

1.3 Choosing the appropriate CRS

The geographical coordinates are initially given in Longitude-Latitude in the World Geodetic System (WGS 84). Users have to determine an appropriate CRS (**new.CRS** object) so that the coordinates can be projected on a plane without deforming too much the boundaries or the areas. Moreover, in the new CRS, the axes will be scaled in meters so that the distances between earthquakes and evaluation points can easily be calculated. The package **rgdal** contains an interesting function which gives all known CRS that are used in almost all countries.

```
EPSG <- make_EPSG()
```

By making use of this list, one can check whether some CRS are used for the country of interest.

```
head(EPSG[grep(country, EPSG[,"note"]),])
```

```
##       code                                                        note
## 25    4021 # Unknown datum based upon the Indonesian National Spheroid
## 3557 23830                          # DGN95 / Indonesia TM-3 zone 46.2
## 3558 23831                          # DGN95 / Indonesia TM-3 zone 47.1
## 3559 23832                          # DGN95 / Indonesia TM-3 zone 47.2
## 3560 23833                          # DGN95 / Indonesia TM-3 zone 48.1
## 3561 23834                          # DGN95 / Indonesia TM-3 zone 48.2
##
## 25                                                                    +proj=longlat +a=6378160 +
## 3557  +proj=tmerc +lat_0=0 +lon_0=94.5 +k=0.9999 +x_0=200000 +y_0=1500000 +ellps=WGS84 +towgs84=0,0,
## 3558  +proj=tmerc +lat_0=0 +lon_0=97.5 +k=0.9999 +x_0=200000 +y_0=1500000 +ellps=WGS84 +towgs84=0,0,
## 3559 +proj=tmerc +lat_0=0 +lon_0=100.5 +k=0.9999 +x_0=200000 +y_0=1500000 +ellps=WGS84 +towgs84=0,0,
## 3560 +proj=tmerc +lat_0=0 +lon_0=103.5 +k=0.9999 +x_0=200000 +y_0=1500000 +ellps=WGS84 +towgs84=0,0,
## 3561 +proj=tmerc +lat_0=0 +lon_0=106.5 +k=0.9999 +x_0=200000 +y_0=1500000 +ellps=WGS84 +towgs84=0,0,
```

Note that the list of possible CRS can be long. To choose the appropriate CRS, users will probably have to dig a little bit here. In the case of Indonesia, we will pick out the CRS indexed by the following number:

```
epsg <- 23835
new.CRS <-  CRS(EPSG[which(EPSG$code == epsg), "prj4"])
```

```
epsg <- 6875
new.CRS <-  CRS(EPSG[which(EPSG$code == epsg), "prj4"])
```

1.4 Boundary of the region

We now define the coordinates of the country's borders:

```
boundary.84 <- world[world@data$NOM%in%country, ]
```

We transform these coordinates into the appropriate CRS defined above:

```
boundary <- spTransform(boundary.84, new.CRS)
```

We compute the coordinates of the bounding-box :

```
bb <- bbox(boundary)
xmin <- bb[1, 1]
xmax <- bb[1, 2]
ymin <- bb[2, 1]
ymax <- bb[2, 2]
```

1.5 Evaluation points

Let us now define a regular grid of evaluation cells in the zone of interest. First, we fix the window size:

```
length.xy <- abs(apply(bb, 1, diff))
```

Then, we should give the resolution cells where the risk measures will be evaluated. For Indonesia, we define each cell as a square of size 100 km:

```
length.grid <- 100000
```

Next, we determine the resolution of the grid over which we will evaluate the risk measures, and define the dimension of the exported figures:

```
dim.cell <- round(rev(length.xy)/length.grid)
width <- 12
height <- 7
```

Now, we prepare a raster object:

```
nrows <- dim.cell[1]
ncols <- dim.cell[2]
grille <- raster(nrows = nrows, ncols = ncols,
 xmn = bb[1,1], xmx = bb[1,2], ymn = bb[2,1],
 ymx = bb[2,2], crs = new.CRS)
```

The number of evaluation points (locations) in the regular grid is given by :

```
(n.grille <- nrows*ncols)
```

```
##   y
## 954
```

1.6 Zone of observations

We define such a zone, say W^*, as the bounding box of the country's borders, expanded by a security distance (**security.d** parameter) equal here to 400 km. This implies that only the observations falling into the zone W^* will be used for the computation of the estimated risk measures. The motivation behind this idea is to avoid some vexing troubles due to border effects (commonly known in the spatial point pattern field). The choice of the security distance of 400 km corresponds to the global bandwidth selected in an optimal way by Goegebeur *et al.* (2014, 2017):

```
security.d <- 400000
```

We define the window in which we are interested (i.e. the bounding box plus the security distance):

```
Sr <- Polygon(cbind(c(seq(xmin - security.d, xmax + security.d,
                          length.out = 10), xmax + security.d,
                      seq(xmax + security.d, xmin - security.d,
                          length.out = 10), xmin - security.d),
                    c(rep(ymin - security.d, 10), ymax + security.d,
                      rep(ymax + security.d, 10), ymin - security.d)),
              hole = F)
Srs1 <- Polygons(list(Sr), "s1")
SpP <- SpatialPolygons(list(Srs1), proj4string = CRS(proj4string(boundary)))
```

We should then convert the CRS of the bounding box in WGS84 (World Geodetic System 1984):

```
SpP.84 <- spTransform(SpP, proj4string(seisme))
```

1.7 Selecting earthquakes and fault lines

To select the earthquakes falling into W^*:

```
ind <- over(seisme, SpP.84)
seisme.bd <- seisme[which(!is.na(ind)),]
```

To transform the coordinates of earthquakes' locations into the appropriate CRS:

```
seisme.bd <- spTransform(seisme.bd, new.CRS)
```

We proceed in the same way for the fault lines:

```
ind.faults1 <- over(faults1, SpP.84)
ind.faults2 <- over(faults2, SpP.84)
faults1 <- faults1[which(!is.na(ind.faults1)), ]
faults2 <- faults2[which(!is.na(ind.faults2)), ]
faults1  <- spTransform(faults1, new.CRS)
faults2  <- spTransform(faults2, new.CRS)
```

The sample size given by the number of earthquakes in W^*:

```
(n <- nrow(seisme.bd))
```

```
## [1] 69458
```

1.8 Converting magnitudes

The response variable Y is the earthquake's **seismic moment** in Newton metres (N-m) units. When the available data Y_i are rather defined in the moment magnitude (M_w) scale, as is the case here in our setup, where:

```
M_w <- seisme.bd@data$Magnitude
```

they can be converted to the seismic moment (M_S) in N-m via the standard formula:

```
ytab <- exp(1.5 * M_w + 9.1)
```

Note that when the seismic moments (M_S) are available, but in dyne-cm (10^{-7} N-m), they can be converted to M_w via the transformation $M_w = (-32/3) + (2/3)\log(M_S)/\log(10)$.

The 2-dimensional covariate X is given by the geographical location in terms of latitude and longitude of earthquakes:

```
xtab <- coordinates(seisme.bd)
Longitude <- xtab[, 1]
Latitude <- xtab[, 2]
```

1.9 Computing distances

We now calculate the matrices of distances between the evaluation points and the observed earthquakes' locations (**dist.pts.grille**). Also, we specialize the discussion to three specific evaluation points, namely the islands of Sumatra, Lombok and Sulawesi, where the great 9.1-magnitude earthquake (2004) and the very recent 6.9 and 7.5-magnitude earthquakes (August and September 2018) have respectively happened.

We define the indices of Sumatra, Lombok, Sulawesi, and prepare the related matrices of distances:

```
  ind_sumatra <- which(M_w == max(seisme$Magnitude))
  ind_lombok <- which(seisme.bd@data$place == "0km SW of Loloan, Indonesia")
  ind_celebes <- which(seisme.bd@data$place == "78km N of Palu, Indonesia")
  ind.big <- c(ind_sumatra, ind_lombok, ind_celebes)
  seisme.bd.eval <- seisme.bd[ind.big, ]
  n.eval.earthquake = nrow(seisme.bd.eval)
  dist.pts.earthquake <- matrix(0, n, n.eval.earthquake)
```

We initialize the matrices of distances between the evaluation points and the earthquakes' locations:

```
  dist.pts.grille <- matrix(0, n, n.grille)
```

We then compute the desired distances:

```
pb <- txtProgressBar(min = 0, max = n, initial = 0, char="=",
                     width=(getOption("width")) , style = 1)
counter <- 0

 for(k in 1:n){
   dist.pts.grille[k,] <- distanceFromPoints(grille, seisme.bd[k,])@data@values
   dist.pts.earthquake[k,] <- pointDistance(seisme.bd.eval, seisme.bd[k,])
   counter = counter + 1
   setTxtProgressBar(pb, counter)
 }
```

2 Exploratory analysis

2.1 Map of recorded earthquakes

We represent the observed earthquakes with different colors depending on the following quantiles:

```
  bk <- quantile(ytab, c(0, 0.25, 0.5, 0.75, 0.95, 0.99, 0.995, 0.999, 1),
                 na.rm = TRUE)
  bk_w <- quantile(M_w, c(0, 0.25, 0.5, 0.75, 0.95, 0.99, 0.995, 0.999, 1),
                 na.rm = TRUE)
```

We add the boundaries of the neighboring countries:

```
SpP.wgs84 <- spTransform(SpP, CRS(proj4string(world)))
boundary.map <-  world[which(gIntersects(SpP.wgs84, world, byid=T)), ]
boundary.map <- spTransform(boundary.map, new.CRS)
```

We define the colors:

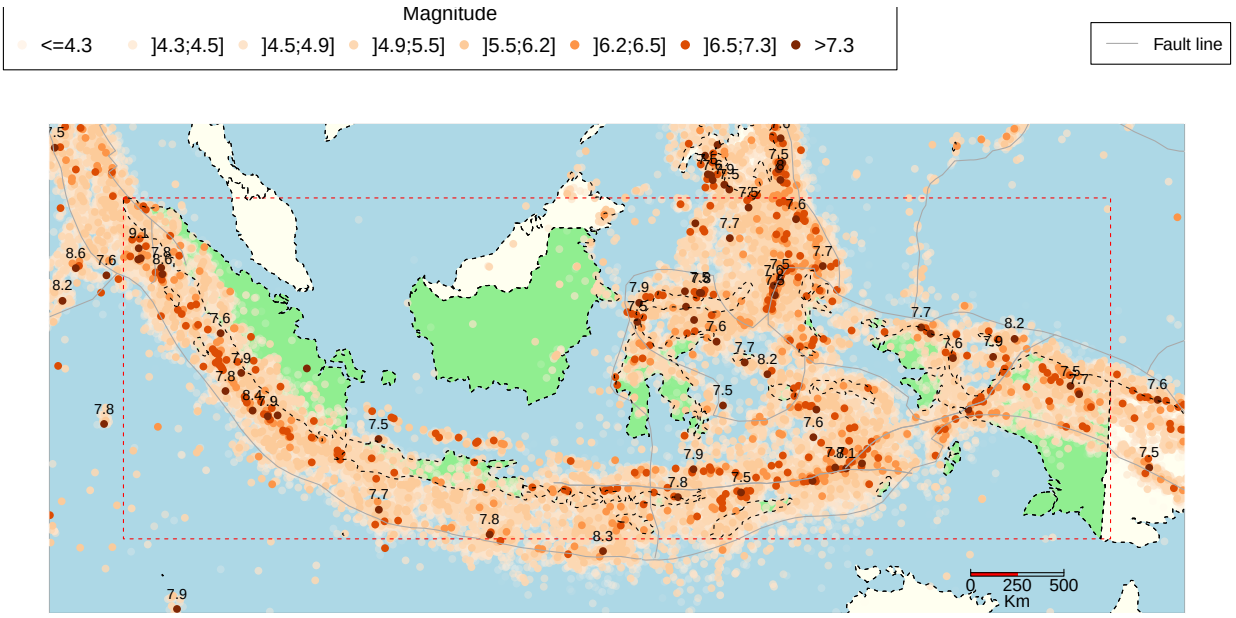
```
nb.col <- length(bk) - 1
plotclr <- colorRampPalette(brewer.pal(9, "Oranges"))(nb.col*2)
plotclr <- plotclr[c(1:(nb.col - 4), seq(nb.col - 3, nb.col * 2,
                                         length.out = 4))]
```

We fix the value of the minimum moment magnitude we want to display on the map:

```
val.min <- 7.5
```

The red dashed line corresponds to the boundary of the grid of evaluation points. The fault lines are graphed in grey.

```
ytab.sort <- order(ytab)
ytab.sort_w <- order(M_w)
ind <- findInterval(M_w, bk_w, all.inside = TRUE)
op <- par(mar = c(1, 0.5, 0.75, 0.5), xpd = TRUE)
plot(SpP,  axes = F, col = "lightblue")
plot(boundary.map, col = "ivory", border = "black", lty = 2, add = TRUE)
plot(boundary, col = "lightgreen", lty = 2, add = T)
plot(seisme.bd[ytab.sort_w, ],
     col = unlist(mapply(adjustcolor, plotclr[ind],
                         ecdf(M_w)(M_w)))[ytab.sort_w],
     add = TRUE, pch = 16)
plot(grille@extent, lty = 2, add = TRUE, col = "red")
plot(faults1, add=T, lwd=1, col="darkgrey")
plot(faults2, add=T, lwd=1, col="darkgrey")
plot(boundary.map, border = "black", lty = 2, add = TRUE)
text(seisme.bd@coords[which(M_w >= val.min), 1],
     seisme.bd@coords[which(M_w >= val.min), 2],
   M_w[which(M_w >= val.min)], pos = 3, cex = 0.8, pch = 16)
masker <- poly.outer(matrix(par()$usr, 2, 2, byrow = TRUE),
                     SpP, extend = 10000000)
add.masking(masker)
map.scale((xmax - 500000) , ymin - security.d/2, 500000, "Km",
          2, 250, sfcol = "red")
bk_w <- round(bk_w, 1)
decoup <- c(paste("<=", bk_w[2], sep = ""),
            paste("]", bk_w[2], ";", bk_w[3], "]", sep = ""),
            paste("]", bk_w[3], ";", bk_w[4], "]", sep = ""),
            paste("]", bk_w[4], ";", bk_w[5], "]", sep = ""),
            paste("]", bk_w[5], ";", bk_w[6], "]", sep = ""),
            paste("]", bk_w[6], ";", bk_w[7], "]", sep = ""),
            paste("]", bk_w[7], ";", bk_w[8], "]", sep = ""),
            paste(">", bk_w[8], sep = ""))
legend("topleft", inset = c(0, -0.04), legend = decoup, pch = 16,
       col = plotclr, title = "Magnitude", horiz = T, cex = 1.2)
legend("topright", legend = c("Fault line"), lty = 1, lwd = 1,
       col = "darkgrey")
```




```
par(op)
```

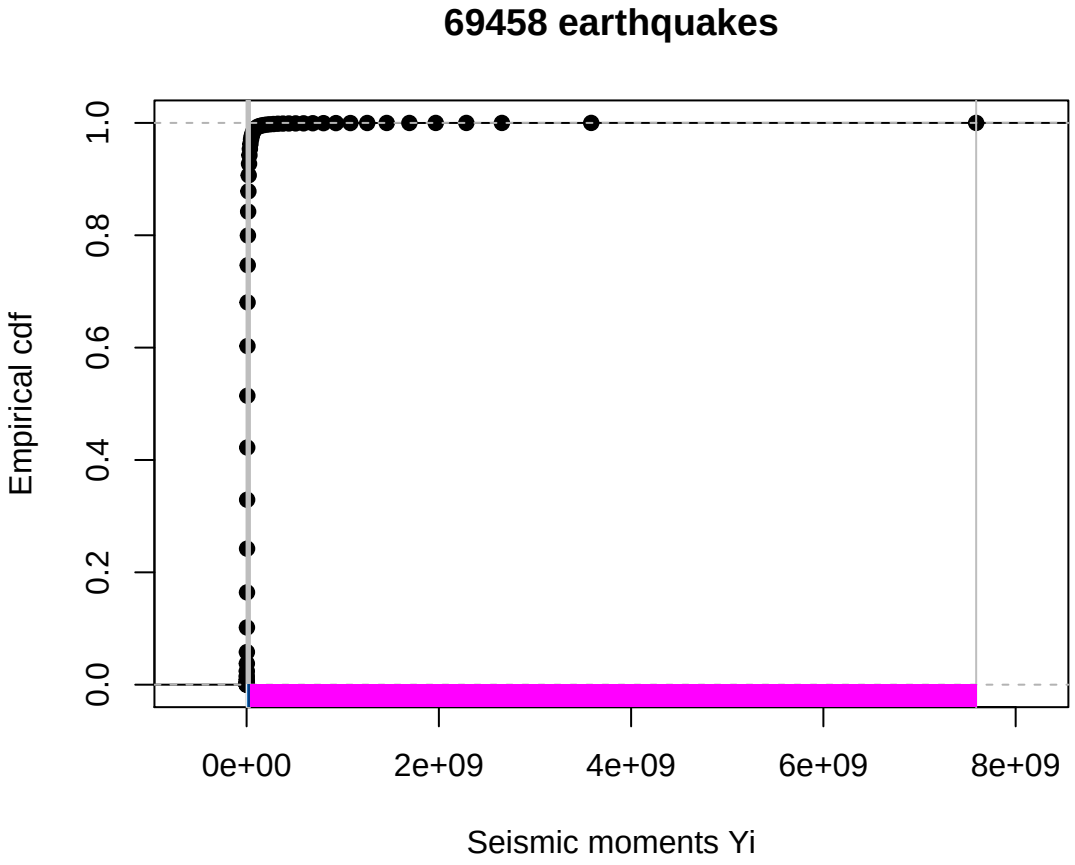
Remark: The zones with a high seismic activity typically cover the locations of fault lines. Also, the most violent earthquakes seem to belong to these risky zones.

The evaluation points excluded from the analysis are those whose neighborhoods (cells) contain less than 1000 observations.

2.2 Marginal distribution of Y

We first represent the empirical cumulative distribution function of the response Y:

```
n.class <- 20
plotclr <- colorRampPalette(brewer.pal(9,"Blues"))(n.class)
plotclr[20] <- "magenta"
plot(classIntervals(ytab, n.class, "quantile", intervalClosure = "right"),
  pal = plotclr, main = paste(n,"earthquakes"),
  xlab = "Seismic moments Yi", ylab = "Empirical cdf")
```



The empirical distribution of seismic moments appears to be heavy-tailed. It is obtained thanks to the square matrix of size n in which cell (i, j) returns 1 if $(Y_i \leq Y_j)$ and 0 otherwise.

Note that this matrix becomes large as n increases. Note also that, due to the discrete nature of the response Y, it is not necessary to determine $1(Y_i \leq Y_j)$ for all $i, j = 1, ..., n$, but only for $i = 1, ..., n$ and $j = 1, ..., n_L$, where n_L is the number of distinct observations Y_i.

The unique values of Y are given by:

```
ytab.unique <- sort(unique(ytab))
table.unique <- table(ytab)
```

Their number n_L is then equal to:

```
n.unique <- length(ytab.unique)
```

We create a vector in order to index the values of Y:

```
  ytab.number <- vector("integer", n)
  for (k in 1:n.unique)
   ytab.number <- replace(ytab.number , which(ytab == ytab.unique[k]), k)
```

We fill in the matrix:

```
  bigmat <- matrix(0, n, n.unique)
  for(k in 1:n.unique)
   bigmat[, k] <- (ytab <= ytab.unique[k])*1
```

By summing the columns of this matrix and dividing by n, we obtain the unconditional empirical distribution:

```
tab.F <- apply(bigmat, 2, sum)/n
```

2.3 Conditional distribution

It is characterized by the Nadaraya-Watson estimator of the conditional distribution function $F_{Y|X}(y|x) = P(Y \leq y|X = x)$:

$$\hat{F}_{NW}(y|x) = \sum_{i=1}^{n} 1(Y_i \leq y) L\left(\frac{x - X_i}{h_n}\right) / \sum_{i=1}^{n} L\left(\frac{x - X_i}{h_n}\right) = \sum_{i=1}^{n} 1(Y_i \leq y) \ell\left(\frac{||x - X_i||}{h_n}\right) / \sum_{i=1}^{n} \ell\left(\frac{||x - X_i||}{h_n}\right),$$

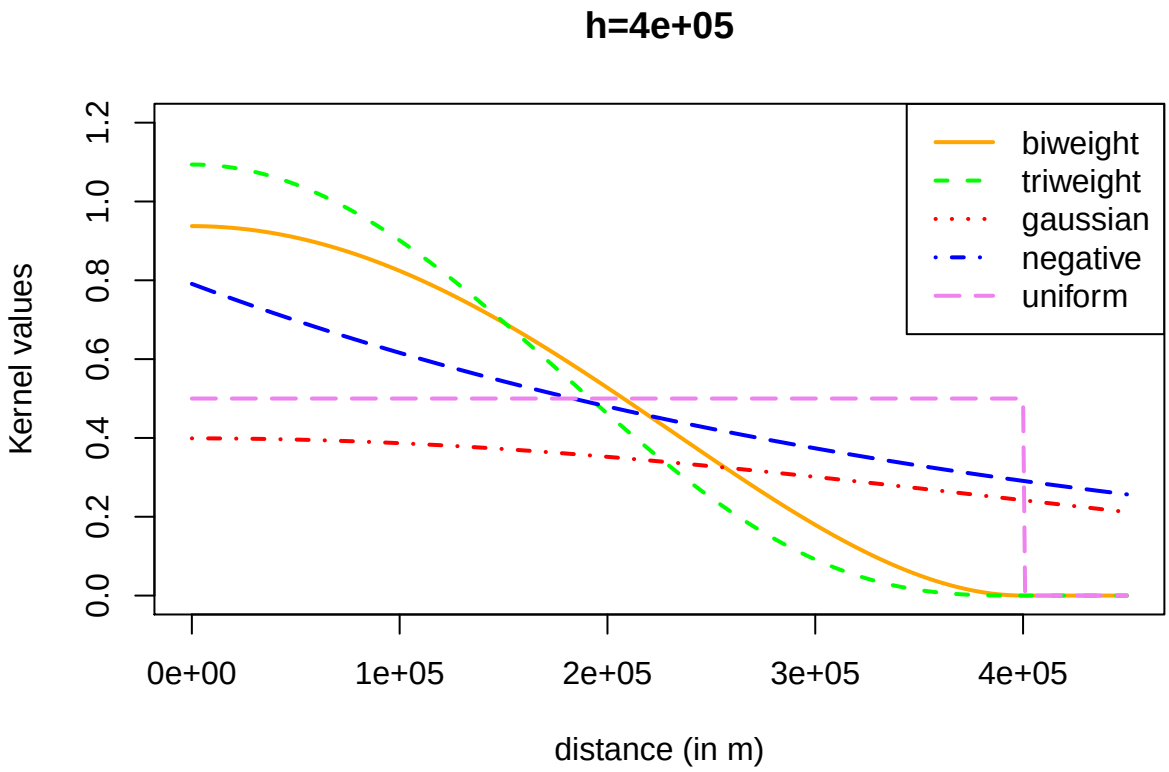
where h_n is a bandwidth to be selected in an optimal way and $L(\cdots) = \ell(||\cdots||)$, with ℓ being a univariate kernel and $||x - X_i||$ is the Euclidean distance between the evaluation point located at x (a cell of the grid) and the earthquake located at X_i. Goegebeur *et al.* (2014, 2017) obtained the global bandwidth $h_n = 400$ km via cross-validation. As regards the kernel ℓ, there exist many kernel functions in the literature. Here we compare the effect of 5 examples of such kernels (see the formulas here):

- biweight
- triweight
- negative exponential
- gaussian
- uniform.

The values of $\ell\left(\frac{||x-X_i||}{h_n}\right)$ are obtained by using the following instructions:

```
FK <- function(x, h = h.opt, type = "bi"){
 switch(type, bi = 15/16*(1-(x/h)^2)^2*ifelse((x/h)^2<=1, 1, 0),
              tri = 35/32*(1-(x/h)^2)^3*ifelse((x/h)^2<=1, 1, 0),
              neg = exp(1)/(2*(exp(1)-1))*exp(-(x/h)),
              gauss = 1/sqrt(2*pi)*exp(-0.5*(x/h)^2),
              logistic = 1/(exp(x/h)+2+exp(-(x/h))),
              unif = 0.5*ifelse((x/h)^2<=1, 1, 0),
              tricube = 70/81*(1-(x/h)^3)^3*ifelse((x/h)^2<=1, 1, 0) )
    }
```

2.3.1 Illustration in 1-D



If we concentrate on the blue curve which corresponds to the negative exponential kernel, the weight given to an earthquake located at the evaluation point (i.e. $||x - X_i|| = 0$) is equal to 0.79. Whereas, for an earthquake located at 100 km (i.e. $||x - X_i|| = 100$ km), the weight is 0.48, and for an earthquake located at 200 km, the weight becomes 0.30.

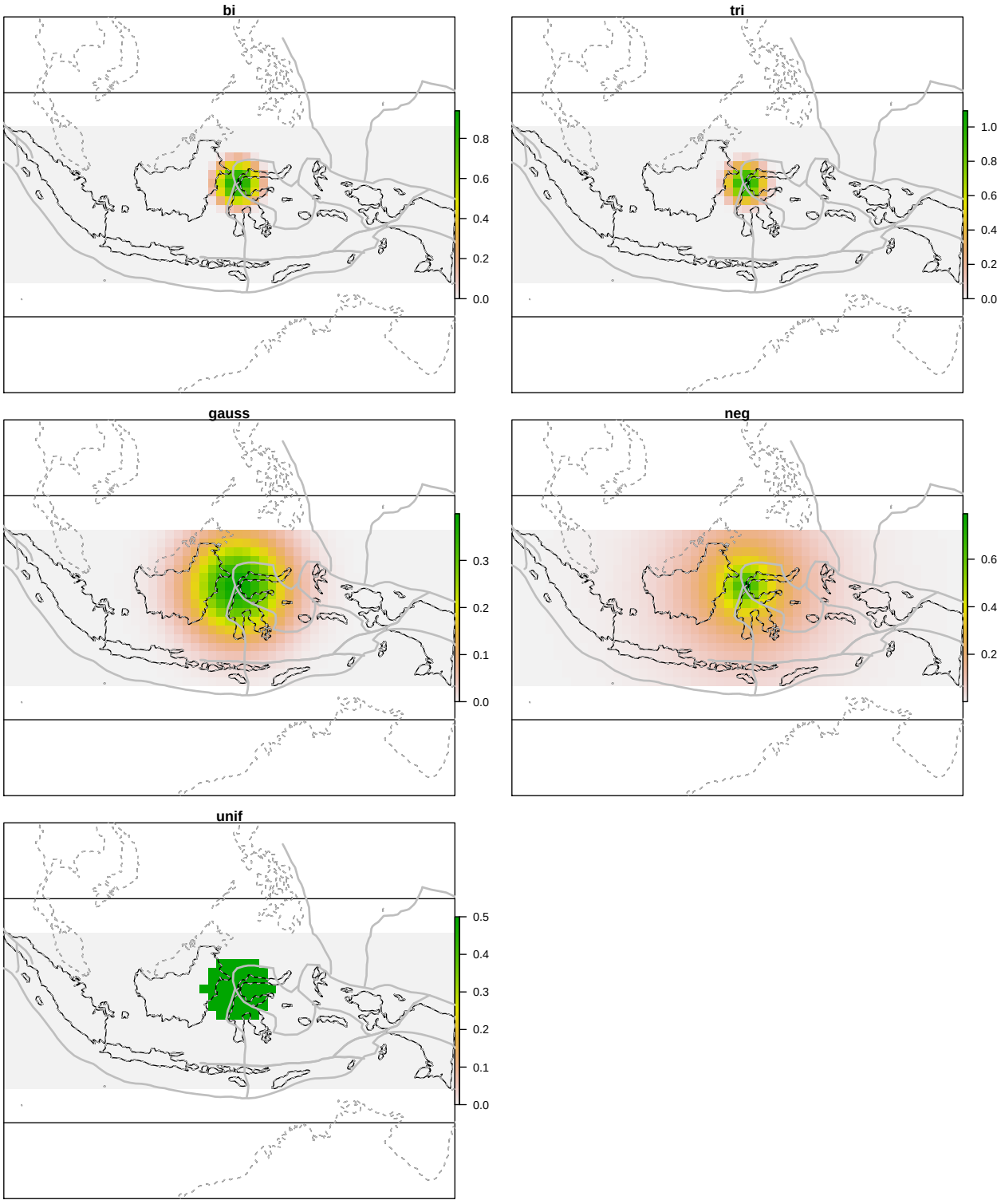
2.3.2 Illustration in 2-D

From now on, we choose the evaluation point x to be the island of Sulawesi and we compute the kernel values $\ell(\frac{||x-X_i||}{h_n})$. The colors in the next figure represent the weights given to earthquakes depending on their distance to the evaluation point.

```
kernel.raster <- grille
```

```
pt.central <- cellFromRowCol(grille, 16, 24)
pt.central_sumatra <- cellFromRowCol(grille, 3, 1)
pt.celebes <- cellFromRowCol(grille, 7, 28)
dist.to.c <- distanceFromPoints(grille, xyFromCell(grille,
                            pt.celebes))@data@values
```

We obtain the following maps for the different kernels:



Remarks

- In the case of biweight and triweight kernels, it can be seen that, beyond the threshold distance (bandwidth) h_n, the weights are equal to 0, and hence the isolated earthquakes from the fixed evaluation point (Sulawesi) are not taken into account. Moreover these kernel functions decay quickly: For example if we look at the previous figure, an earthquake 75 km away from the evaluation point x has a weight 10 times less important than an earthquake located at the same place as x.

- The uniform kernel gives the same weight to all earthquakes falling into the circle of radius h_n and 0 to those lying outside the circle.
- The negative exponential and gaussian kernels are not compactly supported. This implies that whatever the location of an earthquake, its weight cannot be 0. Moreover, these functions decrease quite slowly: For example, an earthquake 75 km away from the evaluation point x has a weight only two times less important than an earthquake very close to the location x.

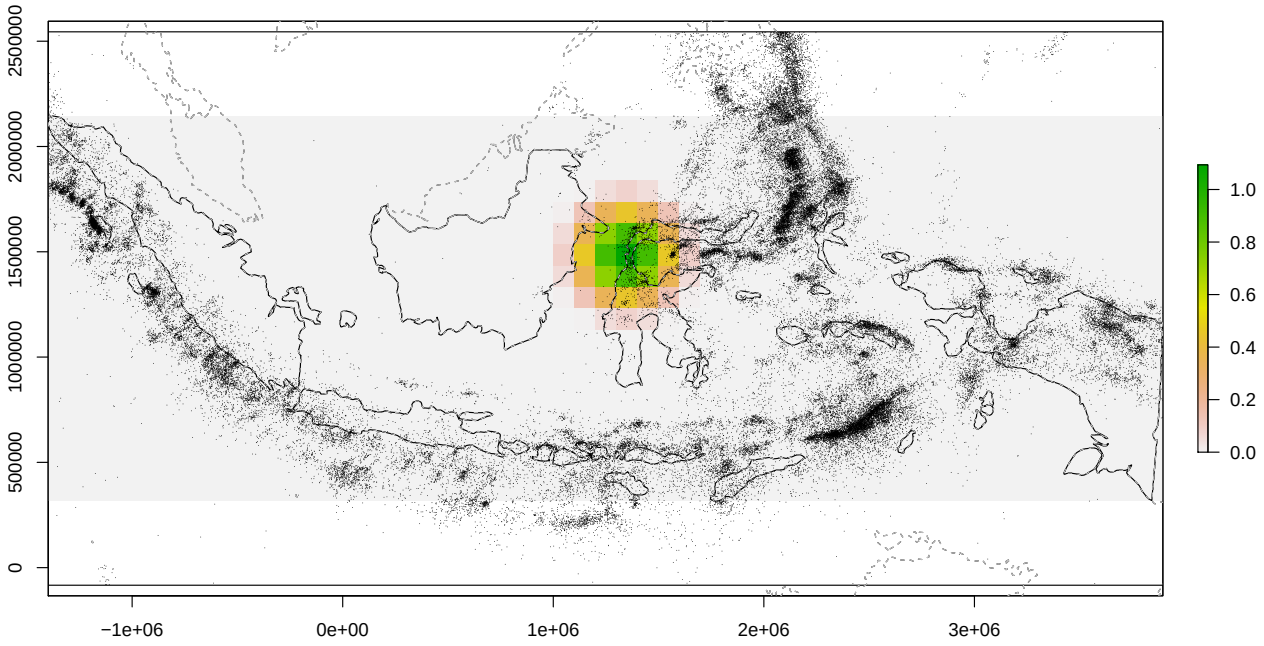
2.3.3 Kernel and bandwidth choice

To ensure a fair comparison between the estimated regression risk measures, Daouia *et al.* (2018) used in their construction the same Triweight kernel ℓ and the same global bandwidth, $h_n = 400$ km, obtained via cross-validation in Goegebeur *et al.* (2014, 2017):

```
h.opt <- 400000
K.mat.dist <- FK(dist.eval.pts, h = h.opt, type = "tri")
```

2.3.4 Illustration

We represent in the following figure the zone in which we are interested, with the evaluation point x being the island of Sulawesi as before. The colors correspond to the weights given to the observed earthquakes according to their distance from the evaluation point.



2.3.5 Computing $\hat{F}_{NW}(y|x)$

First, we associate a weight (thanks to the kernel function) to each earthquake depending on its distance to the evaluation point:

```
dist.to.earthquake <- dist.pts.grille[, which(ind.grille.eval == pt.celebes)]
kern <- FK(dist.to.earthquake, h, "tri")
```

Then, we employ the formula above to calculate the Nadaraya-Watson estimates:

```
num <-  colSums(bigmat*kern)
den <- sum(kern)
hatF_k <- num/den
```

Note that the number $n_L(x)$ of possible distinct values of $\hat{F}_{NW}(y|x)$ is less than or equal the number n_L of distinct observations Y_i. The number $n_L(x)$ depends on the location of earthquakes around the evaluation point x.

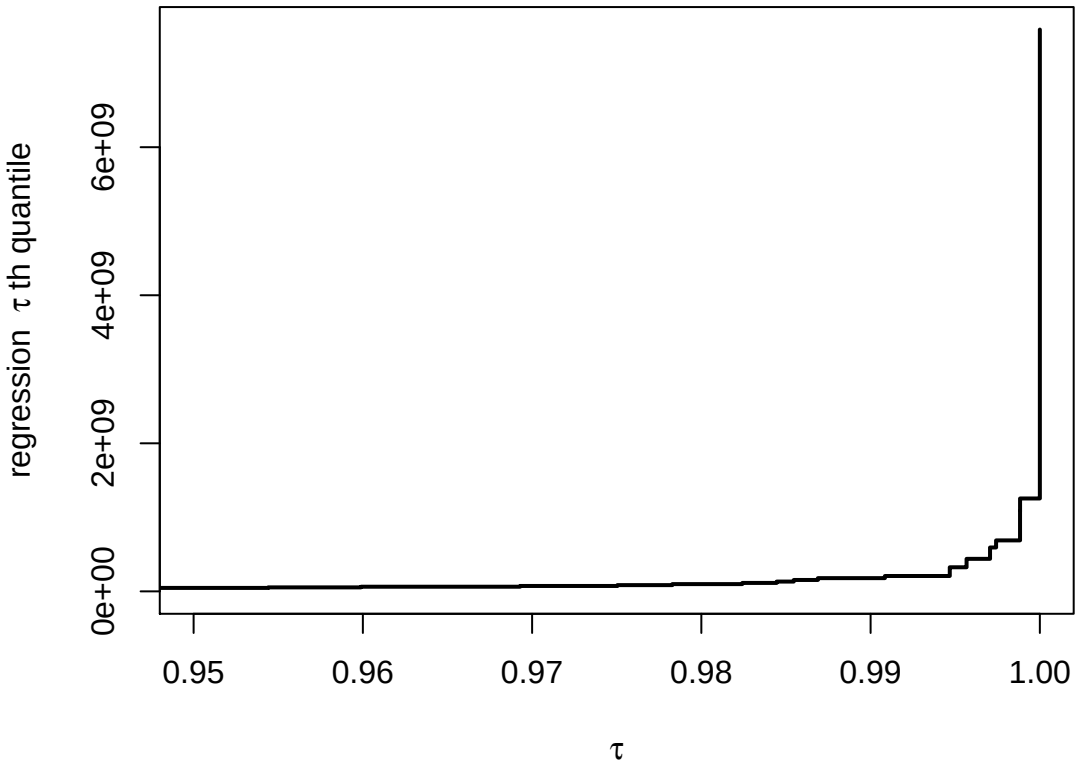
3 The regression risk measures

3.1 Regression quantiles

For a given order $\tau \in (0,1)$, we consider the standard nonparametric estimator of the regression τth quantile $q_\tau(x) = F^{-1}_{Y|X}(\tau|x) = \inf\{y : F_{Y|X}(y|x) \geq \tau\}$, given by:

$$\hat{q}_\tau(x) = \hat{F}^{-1}_{NW}(\tau|x) = \inf\{y : \hat{F}_{NW}(y|x) \geq \tau\}.$$

This is the generalized inverse of the Nadaraya-Watson estimator of the conditional distribution function. The evolution of this risk measure (on the seismic moment scale) in terms of the security level $\tau \geq 0.95$, at the same evaluation point as before (Sulawesi), is shown in the following figure. The graph exhibits the jumps corresponding to the $n_L(x)$ distinct values of $\hat{F}_{NW}(\cdot|x)$.



3.2 Regression tail index estimates

First, we consider the following localized version of the Hill estimator introduced by Daouia *et al.* (2011):

$$LH_k(x) := \hat{\gamma}_k(x) = \frac{1}{k}\sum_{i=1}^{k}(\log(Y^*_{M_x-i+1,M_x}) - \log(Y^*_{M_x-k,M_x}))$$

where $M_x = \sum_{i=1}^{n} 1(||X_j - x|| \leq h_n)$ is the number of earthquakes observed at a distance less than or equal to h_n, and $Y^*_{1,M_x} \leq Y^*_{2,M_x} \leq \cdots \leq Y^*_{M_x,M_x}$ denote the order statistics corresponding to the M_x response variables Y_j such that $||X_j - x|| \leq h_n$. Built on ideas from Gardes and Stupfler (2014), a localized and averaged version of $\hat{\gamma}_k(x)$ is given by:

$$LAH_{k,a}(x) = \frac{1}{\lceil a\,k \rceil + 1}\sum_{j=\lfloor (1-a)k \rfloor}^{k} \hat{\gamma}_j(x),$$

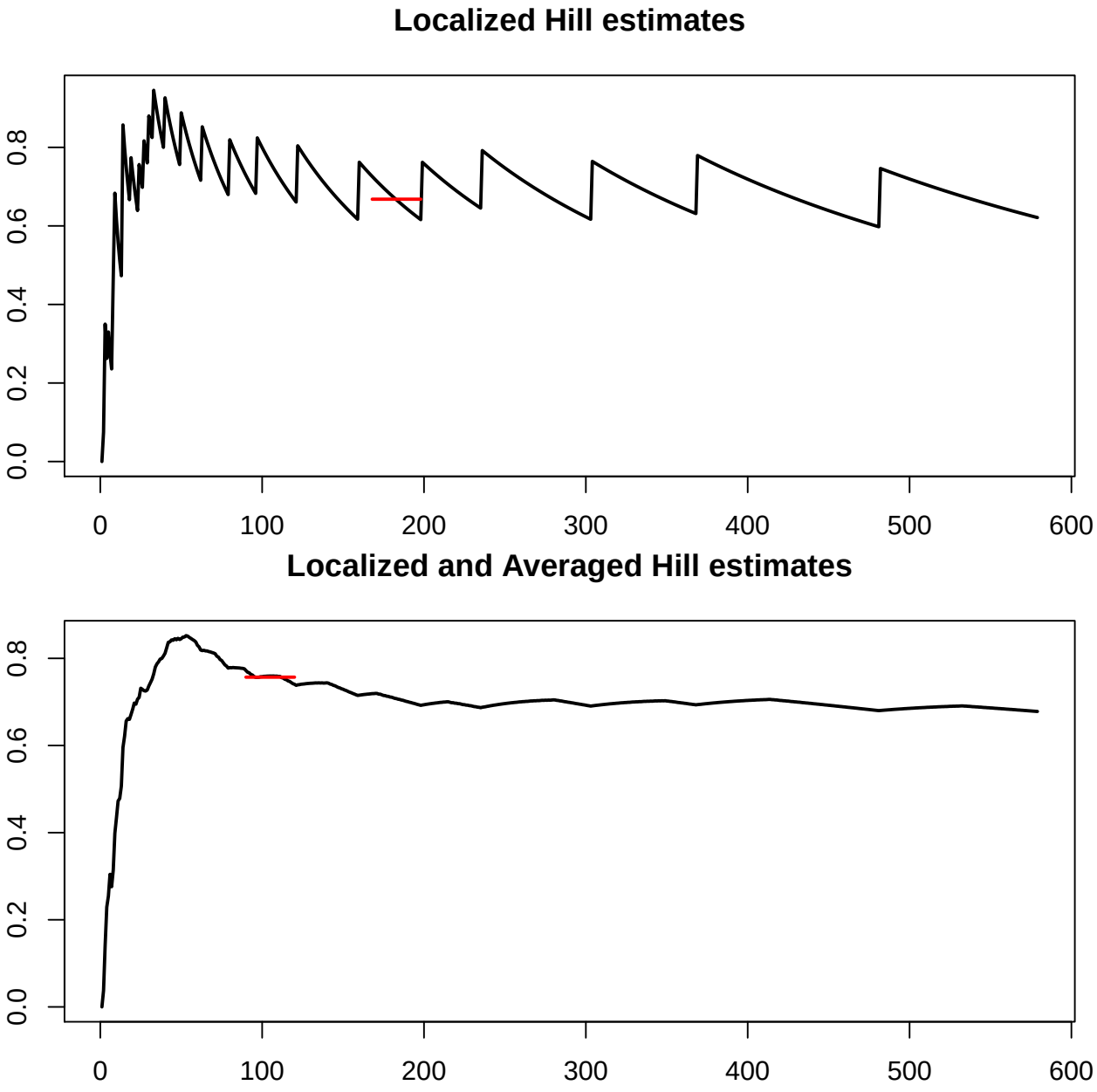
where $a \in [0, 1[$ is a constant to be selected by the user. Following Gardes and Stupfler (2014), the choice $a = 3/7$ generally affords quite decent results, although this choice is flexible to a good extent.

Next, we calculate both estimators $LH_k(x)$ and $LAH_{k,a}(x)$ at the same evaluation point as before (Sulawesi), for the grid values of $k = 1, \ldots, M_x/4$:

```
l <- which(ind.grille.eval == pt.celebes)
kern <- K.mat.dist[, l]
num.mat.int <-  colSums(bigmat*kern)
num <- num.mat.int[ytab.number]
a_stupfler <- 3/7

# number of observations lying inside the local strip around x
nstar <- length(which(kern > 0))
size_candidate <- floor(nstar/4) # nstar

den <- sum(kern)
hatF_k <- num/den

# we simplify the computation for kernel values = 0
hatF_k <- hatF_k[kern > 0]
ytab.n <- ytab[kern > 0]

F_emp <- cumsum(rev(table(ytab.n)))[-1]
hat_gama_candidate_hill <- numeric(size_candidate)
hat_gama_candidate_stupfler <- numeric(size_candidate)

for (k in 1:size_candidate) {
  hat.gama_hill <- 0
  ytab.n_sort <- sort(ytab.n)
  inv.F <- ytab.n_sort[nstar - k + 1]

  for (j in 0:(k - 1)) {
     inv.F.j <- ytab.n_sort[nstar - j]
     hat.gama_hill <- hat.gama_hill + (log(inv.F.j) - log(inv.F))
  }

  hat_gama_candidate_hill[k] <- hat.gama_hill/k

  lower_bound <- floor((1 - a_stupfler) * k)
  upper_bound <- k
```

```
  hat_gama_candidate_stupfler[k] <-
    sum(hat_gama_candidate_hill[lower_bound:upper_bound])/
    (ceiling(a_stupfler * k) + 1)
  }
```

In the following figures we graph the plots of both Localized Hill estimator $k \mapsto LH_k(x)$ and Localized and Averaged Hill estimator $k \mapsto LAH_{k,a}(x)$.



It may be seen that the plot of $k \mapsto LAH_{k,a}(x)$ provides a smoothed variant of the plot of $k \mapsto LH_k(x)$.

3.3 Optimal pointwise estimates

A very simple technique to select an appropriate pointwise value, for each estimator, consists in computing its standard deviations over a *moving window* of (here, $\lfloor 0.05 M_x/4 \rfloor$) successive values of k (this corresponds

to a window covering around 5% of the possible values of k in the selected range $[1, \frac{M_x}{4}]$). Then, we identify the window of k-values where the standard deviation (and hence the variation) of the estimates is minimal. To gain stability in the estimates, we take the average of the estimates corresponding to those k-values and regard that as the final desired estimate. In the figures above, the red segment corresponds to the stable region of each plot and gives the average of the estimates over this region.

To determine the first stable region of the plots, as well as the final averaged estimates, we use the following function:

```
choice_k <- function(Vect, h = 0.05) {
  # 2 inputs:
  #       Vect = vector of n estimates
  #         h  = a factor to select so that the interval's length is approximately 2hn
  #
  # 4 outputs:
  #        opt_val = optimal estimate
  #        ind_opt = index of the optimal estimate
  #        ind_min, ind_max = minimal and maximal indices of the block over which
  #                           the standard deviation of the estimates is minimal

  n_vect <- length(Vect)
  widt <- 2*max(1, ceiling(h*n_vect)) + 1
  kmin <- 1
  na.vect <- which(is.na(Vect))
  if (length(na.vect) != 0)
    kmin <- max(3, na.vect[length(na.vect)])
  kmax <- n_vect - widt
  NbSigma <- kmax - kmin + 1

  VectSd <- numeric(NbSigma)
  AV <- numeric(NbSigma - 2)
  AP <- numeric(NbSigma - 2)

  for (i in 1:NbSigma) {
    VectSd[i] <- sd(Vect[(i + kmin):(i + kmin + widt - 1)], na.rm = T)
  }

  VectSdC <- VectSd[2:(length(VectSd) - 1)]

  Sdmean <- mean(VectSd, na.rm = T)
  M <- as.numeric(VectSdC < Sdmean)

  for (i in 1:(NbSigma-2)) {
    AV[i] <- as.numeric(VectSdC[i] < VectSd[i])
    AP[i] <- as.numeric(VectSdC[i] < VectSd[i + 2])
  }

  ToT <- M + AV + AP

  vecIndicMin <- which(ToT == max(ToT, na.rm = T))
  IndicMin <- vecIndicMin[1] + 1
  IndicMin

  IndicMed <- which(Vect == median(Vect[(IndicMin + kmin):(IndicMin + kmin + widt - 1)],
                                   na.rm=T))
```

```
  ind_opt <- IndicMed + kmin

  opt_val <- Vect[IndicMed + kmin]
  ind_min <- IndicMin + kmin
  ind_max <- IndicMin + kmin + widt - 1

  # Final result:
  result <- list(opt_val = opt_val, ind_opt = ind_opt,
                 ind_min = ind_min, ind_max = ind_max,
                 esti = mean(Vect[ind_min:ind_max]))
  return(result)
}
```

We apply the function *choice_k()* to the estimators $LH_k(x)$ and $LAH_{k,a}(x)$ as follows:

```
k_opti_hill <- choice_k(hat_gama_candidate_hill, h = 0.025)
k_opti_stupfler <- choice_k(hat_gama_candidate_stupfler, h = 0.025)
```

This type of automatic device is commonly used in the literature of conditional extremes. It was first implemented by Daouia *et al.* (2010) and applied in Stupfler (2013), Daouia *et al.* (2013), Gardes and Stupfler (2014), Goegebeur *et al.* (2014, 2017), among others.

3.4 Extrapolated risk measures

The three regression risk measures of interest are:

- the Weissman quantile estimator

$$\hat{q}^*_{1-p_n} = \hat{q}_{1-\alpha_n}(x) \left(\frac{\alpha_n}{p_n}\right)^{\hat{\gamma}(x)},$$

where $p_n \in (0,1)$ is an extreme tail exceedance probability to be fixed by the user, $\alpha_n \in (p_n, 1)$ is an intermediate level to be selected in an optimal way, $\hat{q}_{1-\alpha_n}(x) = \hat{F}^{-1}_{NW}(1-\alpha_n|x)$ and $\hat{\gamma}(x)$ stands for $LH_k(x)$ or $LAH_{k,a}(x)$.

Here, we propose to estimate the severe seismic moment that will be exceeded (on average) only once every $2n$ cases (corresponding to twice the whole available period of observation of all historical data). This translates into choosing the extreme level $p_n = 1/(2n)$. As regards the tuning parameter α_n, we use the re-parametrization $\alpha_n = k/M_x$, where $k \in [1, M_x - 1]$ is an integer to be selected in an optimal way over the range of intermediate values, say, from 1 to $M_x/4$ (this restriction allows to reject too large values of α_n), similarly to the tail index estimators $LH_k(x)$ and $LAH_{k,a}(x)$.

- the regression extremile estimator

$$\hat{\xi}^*_{1-p_n}(x) = \hat{q}^*_{1-p_n}(x) G(\hat{\gamma}(x))$$

where

$$G(s) = \Gamma(1-s)(\log 2)^s \quad \text{for} \quad s < 1,$$

with Γ being the gamma function. Putting $\tau := 1 - p_n$, $\widehat{\xi^\star_\tau}(x)$ defines an extrapolated estimator of the conditional order-τ extremile of Y given $X = x$:

$$\xi_\tau(x) = \mathbb{E}\left[Y\, J_\tau\left(F(Y|X)\right)|X = x\right] = \int_0^1 J_\tau(t)\, q_t(x)\, dt = \int_0^1 q_t(x)\, dK_\tau(t),$$

where $J_\tau(\cdot) = K'_\tau(\cdot)$ on $(0,1)$, with

$$K_\tau(t) = \begin{cases} 1-(1-t)^{s(\tau)} & \text{if} \quad 0 < \tau \leq 1/2 \\ t^{r(\tau)} & \text{if} \quad 1/2 \leq \tau < 1 \end{cases}$$

being a distribution function with support $[0,1]$, and $r(\tau) = s(1-\tau) = \log(1/2)/\log(\tau)$.

- the regression expected shortfall estimator

$$\hat{\nu}^*_{1-p_n}(x) = \hat{\nu}_{1-\alpha_n}(x) \left(\frac{\alpha_n}{p_n}\right)^{\hat{\gamma}(x)}$$

where

$$\hat{\nu}_{1-\alpha_n}(x) = \sum_{i=1}^n L\left(\frac{x-X_i}{h_n}\right).Y_i.1(Y_i \geq \hat{q}_{1-\alpha_n}(x))/\sum_{i=1}^n 1(Y_i \geq \hat{q}_{1-\alpha_n}(x))L\left(\frac{x-X_i}{h_n}\right).$$

Putting $\tau := 1-p_n$, this defines an extrapolated estimator of the conditional expected shortfall $\nu_\tau(x) := \mathbb{E}[Y|Y \geq q_\tau(x), X = x]$, *i.e.*, the average seismic moment over the Value at Risk $q_\tau(x)$.

3.5 Computation of the risk estimates

```
# initialization
vec.weiss_hill <- vec.weiss_stupfler <-
  vec.xi_hill <- vec.xi_stupfler <-
  vec.nu_hill <- vec.nu_stupfler <-
  vec.gama_hill <- vec.gama_stupfler <-
  vec.nstar <- numeric(n.eval)

p.n <-  1/(2*n)
a_stupfler <- 3/7

for(l in 1:n.eval) {
# The Nadaraya-Watson estimator
 kern <- K.mat.dist[, l]
 num.mat.int <-  colSums(bigmat*kern)
 num <- num.mat.int[ytab.number]
# number of observations observed into the window
 nstar <- length(which(kern > 0))
 size_candidate <- floor(nstar/4)

 den <- sum(kern)
 hatF_k <- num/den

 # we simplify the computation for kernel values = 0
 hatF_k <- hatF_k[kern > 0]
 ytab.n <- ytab[kern > 0]
```

```
F_emp <- cumsum(rev(table(ytab.n)))[-1]
taux_n <- 1/100

# discrete case
hat_gama_candidate <- numeric(size_candidate)
hat_gama_candidate_stupfler <- numeric(size_candidate)

weiss_candidate_hill <- numeric(size_candidate)
weiss_candidate_stupfler <- numeric(size_candidate)

xi_candidate_hill <- numeric(size_candidate)
xi_candidate_stupfler <- numeric(size_candidate)

nu_candidate_hill <- numeric(size_candidate)
nu_candidate_stupfler <- numeric(size_candidate)

for (k in 1:size_candidate) {
   # the Hill type estimator
   hat.gama <- hat.gama_ksh <- 0
   ytab.n_sort <- sort(ytab.n)
   inv.F <- ytab.n_sort[nstar - k + 1]

   for (j in 0:(k - 1)) {
     inv.F.j <- ytab.n_sort[nstar - j]
     hat.gama <- hat.gama + (log(inv.F.j) - log(inv.F))
   }


   hat_gama_candidate[k] <- hat.gama/k

   lower_bound <- floor((1 - a_stupfler) * k)
  upper_bound <- k
  hat_gama_candidate_stupfler[k] <-
    sum(hat_gama_candidate[lower_bound:upper_bound])/
     (ceiling(a_stupfler * k) + 1)

  hat.gama_hill <- hat_gama_candidate[k]
  hat.gama_stupfler <- hat_gama_candidate_stupfler[k]

  # Weissman type estimator of the extreme regression quantile
  weiss_hill <- inv.F*(k/(n * p.n))^hat.gama_hill
  weiss_candidate_hill[k] <- weiss_hill

  weiss_stupfler <- inv.F*(k/(n * p.n))^hat.gama_stupfler
  weiss_candidate_stupfler[k] <- weiss_stupfler

  # conditional extremile
  s_hill <- hat.gama_hill
  if (s_hill < 1) {
    xi_hill <- weiss_hill*gamma(1 - s_hill)*(log(2))^s_hill
  } else {
    xi_hill <- NA
  }
```

```
  xi_candidate_hill[k] <- xi_hill

  s_stupfler <- hat.gama_stupfler
   if (s_stupfler < 1) {
    xi_stupfler <- weiss_stupfler*gamma(1 - s_stupfler)*(log(2))^s_stupfler
  } else {
    xi_stupfler <- NA
  }
  xi_candidate_stupfler[k] <- xi_stupfler

  hat.nu <- sum(kern * ytab * (ytab >= inv.F))/sum(kern * (ytab >= inv.F))

  nu_hill <-  hat.nu * ((k/n)/p.n)^hat.gama_hill
  nu_candidate_hill[k] <- nu_hill

  nu_stupfler <-  hat.nu * ((k/n)/p.n)^hat.gama_stupfler
  nu_candidate_stupfler[k] <- nu_stupfler

  }

   vec.gama_hill[l] = choice_k(hat_gama_candidate, 0.025)$esti
   vec.gama_stupfler[l] = choice_k(hat_gama_candidate_stupfler, 0.025)$esti

   vec.weiss_hill[l] = choice_k((log(weiss_candidate_hill) - 9.1)/1.5, 0.025)$esti
   vec.weiss_stupfler[l] = choice_k((log(weiss_candidate_stupfler) - 9.1)/1.5, 0.025)$esti

   vec.xi_hill[l] = choice_k((log(xi_candidate_hill) - 9.1)/1.5, 0.025)$esti
   vec.xi_stupfler[l] = choice_k((log(xi_candidate_stupfler) - 9.1)/1.5, 0.025)$esti

   vec.nu_hill[l] = choice_k((log(nu_candidate_hill) - 9.1)/1.5, 0.025)$esti
   vec.nu_stupfler[l] = choice_k((log(nu_candidate_stupfler) - 9.1)/1.5, 0.025)$esti
   vec.nstar[l] = nstar
 }
```

Note that the estimates $LH_k(x)$ and $LAH_{k,a}(x)$ of $\gamma(x)$ are found to be typically smaller than 1 (as required by the model assumption) except for some very few values of k that we drop from the grid of k-values in the calculation of the final estimates:

```
ind_gama_uper_1 <- union(which(vec.gama_hill > 1), which(vec.gama_stupfler > 1))
vec.gama_hill[ind_gama_uper_1] <- NA
vec.weiss_hill[ind_gama_uper_1] <- NA
vec.xi_hill[ind_gama_uper_1] <- NA
vec.nu_hill[ind_gama_uper_1] <- NA
vec.gama_stupfler[ind_gama_uper_1] <- NA
vec.weiss_stupfler[ind_gama_uper_1] <- NA
vec.xi_stupfler[ind_gama_uper_1] <- NA
vec.nu_stupfler[ind_gama_uper_1] <- NA
```

We fill the grid corresponding to the estimated cells :

```
  weiss.raster_hill <- xi.raster_hill <- nu.raster_hill <- gama.raster_hill <-
  weiss.raster_stupfler <- xi.raster_stupfler <- nu.raster_stupfler <-
  gama.raster_stupfler <-  grille

  values(weiss.raster_hill)[ind.grille.eval] <- vec.weiss_hill[1:n_grille_eval]
```

```
  values(xi.raster_hill)[ind.grille.eval] <- vec.xi_hill[1:n_grille_eval]
  values(nu.raster_hill)[ind.grille.eval] <- vec.nu_hill[1:n_grille_eval]
  values(gama.raster_hill)[ind.grille.eval] <- vec.gama_hill[1:n_grille_eval]

  values(weiss.raster_stupfler)[ind.grille.eval] <- vec.weiss_stupfler[1:n_grille_eval]
  values(xi.raster_stupfler)[ind.grille.eval] <- vec.xi_stupfler[1:n_grille_eval]
  values(nu.raster_stupfler)[ind.grille.eval] <- vec.nu_stupfler[1:n_grille_eval]
  values(gama.raster_stupfler)[ind.grille.eval] <- vec.gama_stupfler[1:n_grille_eval]
```

The maximum differences between $\hat{\nu}^*_{1-p_n}(x)$ and $\hat{q}^*_{1-p_n}(x)$ are equal to :

```
max(values(nu.raster_hill) - values(weiss.raster_hill), na.rm = T)
```

```
## [1] 1.336332
```

```
max(values(nu.raster_stupfler) - values(weiss.raster_stupfler), na.rm = T)
```

```
## [1] 1.182161
```

The maximum differences between $\hat{\xi}^*_{1-p_n}(x)$ and $\hat{q}^*_{1-p_n}(x)$ are equal to :

```
max(values(xi.raster_hill) - values(weiss.raster_hill), na.rm = T)
```

```
## [1] 0.8036018
```

```
max(values(xi.raster_stupfler) - values(weiss.raster_stupfler), na.rm = T)
```

```
## [1] 1.072192
```

The estimated values at Sumatra, Lombok and Sulawesi:

```
seisme.bd.eval@data[, c("place", "Magnitude",
                        "gama_hill", "weiss_hill", "xi_hill", "nu_hill",
                        "gama_stupfler", "weiss_stupfler", "xi_stupfler", "nu_stupfler")]
```

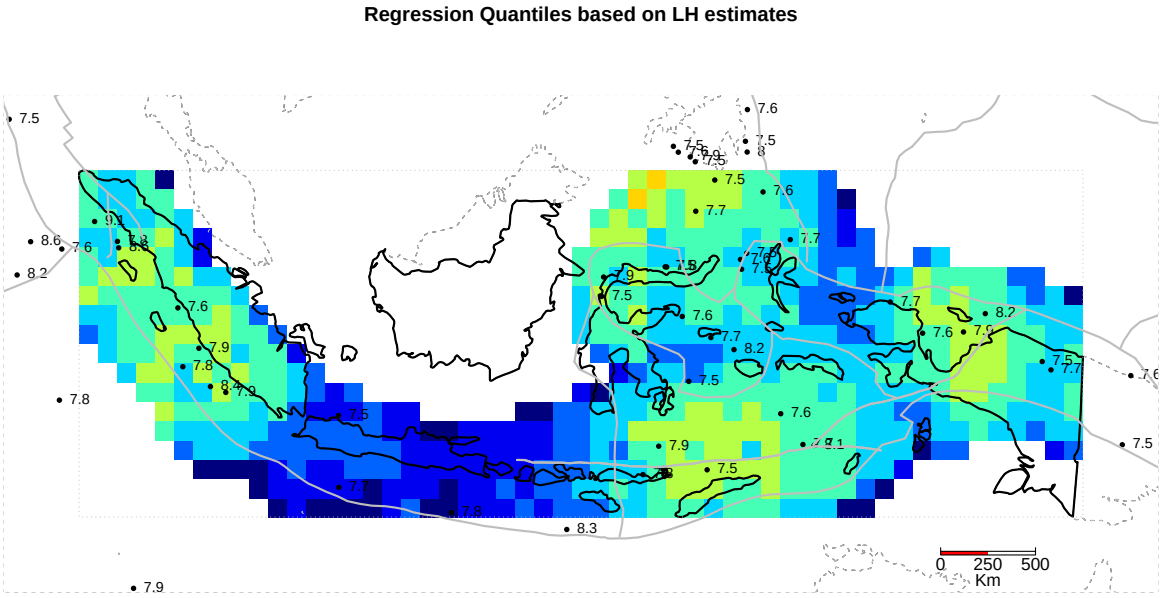
```
##                                       place Magnitude gama_hill
## 48267 off the west coast of northern Sumatra       9.1 0.6220215
## 542              0km SW of Loloan, Indonesia       6.9 0.5823722
## 58                 78km N of Palu, Indonesia       7.5 0.6600383
##       weiss_hill  xi_hill  nu_hill gama_stupfler weiss_stupfler
## 48267   8.085825 8.505945 9.085812     0.6605091       8.275498
## 542     7.650650 7.964717 8.052039     0.5585803       7.687427
## 58      8.588371 8.869303 9.299337     0.7225883       8.332340
##       xi_stupfler nu_stupfler
## 48267    8.777831    9.315622
## 542      8.008320    8.080089
## 58       8.939770    9.059104
```

Choice of the colours :

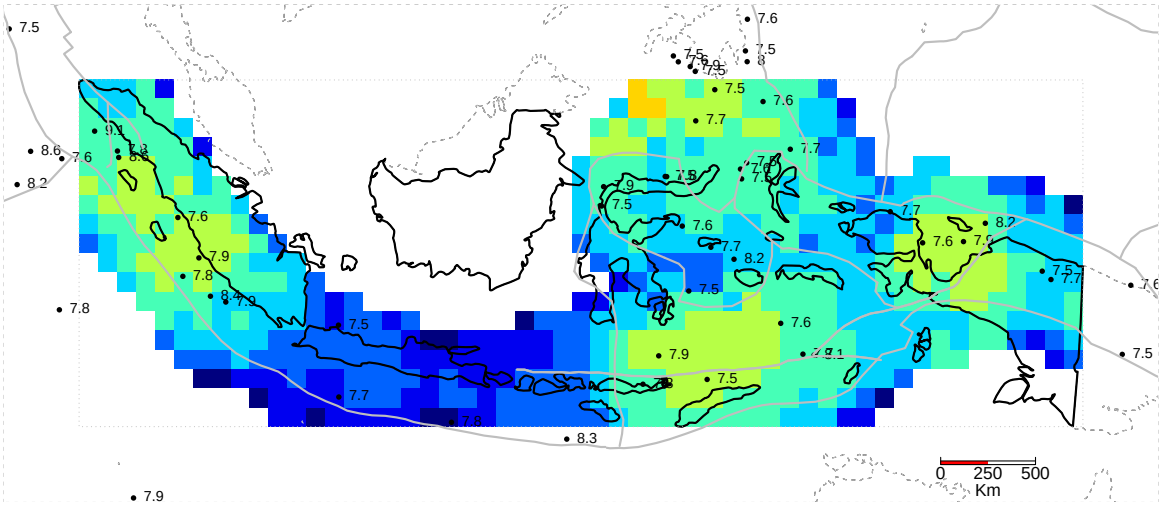
```
vec_values <- c(values(weiss.raster_hill), values(xi.raster_hill), values(nu.raster_hill),
                values(weiss.raster_stupfler), values(xi.raster_stupfler),
                values(nu.raster_stupfler))

bb.neighbour <- seq(min(vec_values,  na.rm = T),
                    max(vec_values, na.rm = T),
                    length.out = 11)
nb.col <- length(bb.neighbour) - 1
plotclr <- colorRampPalette(c("#00007F", "blue", "#007FFF",
```

```
                        "cyan", "#7FFF7F", "yellow",
                        "#FF7F00", "red", "#7F0000"))(nb.col)
```

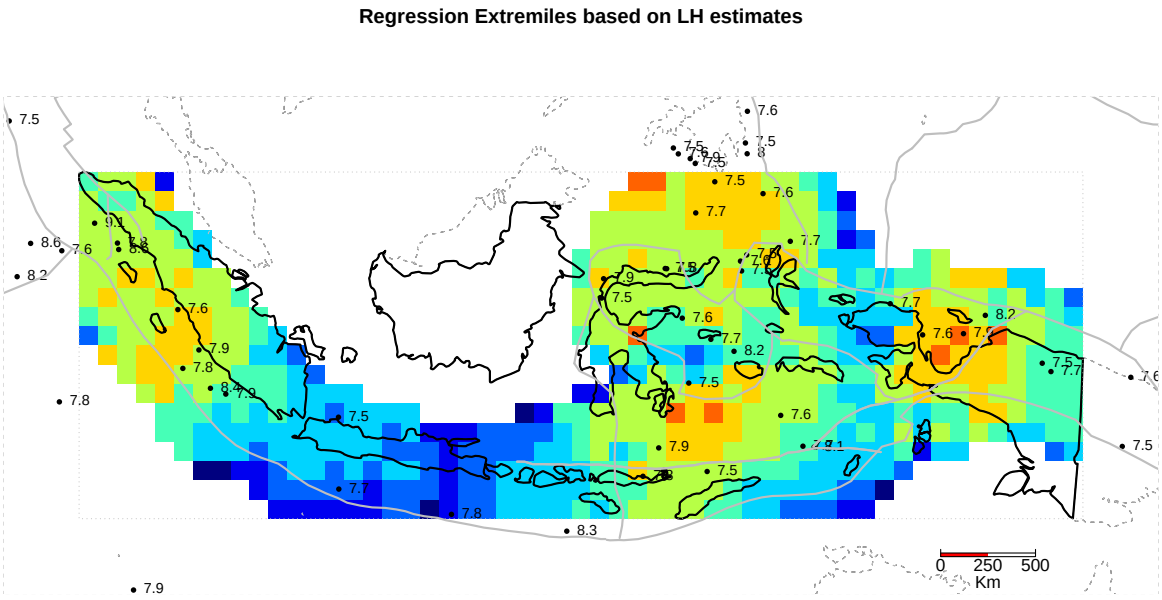
- the regression quantiles $\hat{q}^*_{1-p_n}(x)$ based on the estimates $LH_k(x)$:



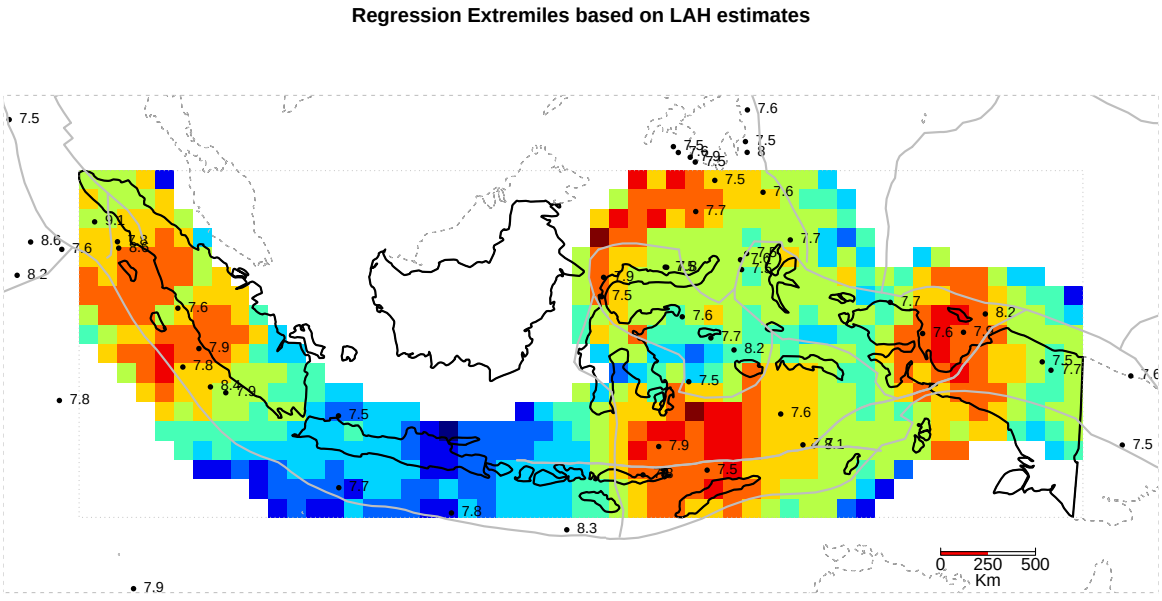
- the regression quantiles $\hat{q}^*_{1-p_n}(x)$ based on the estimates $LAH_{k,a}(x)$:



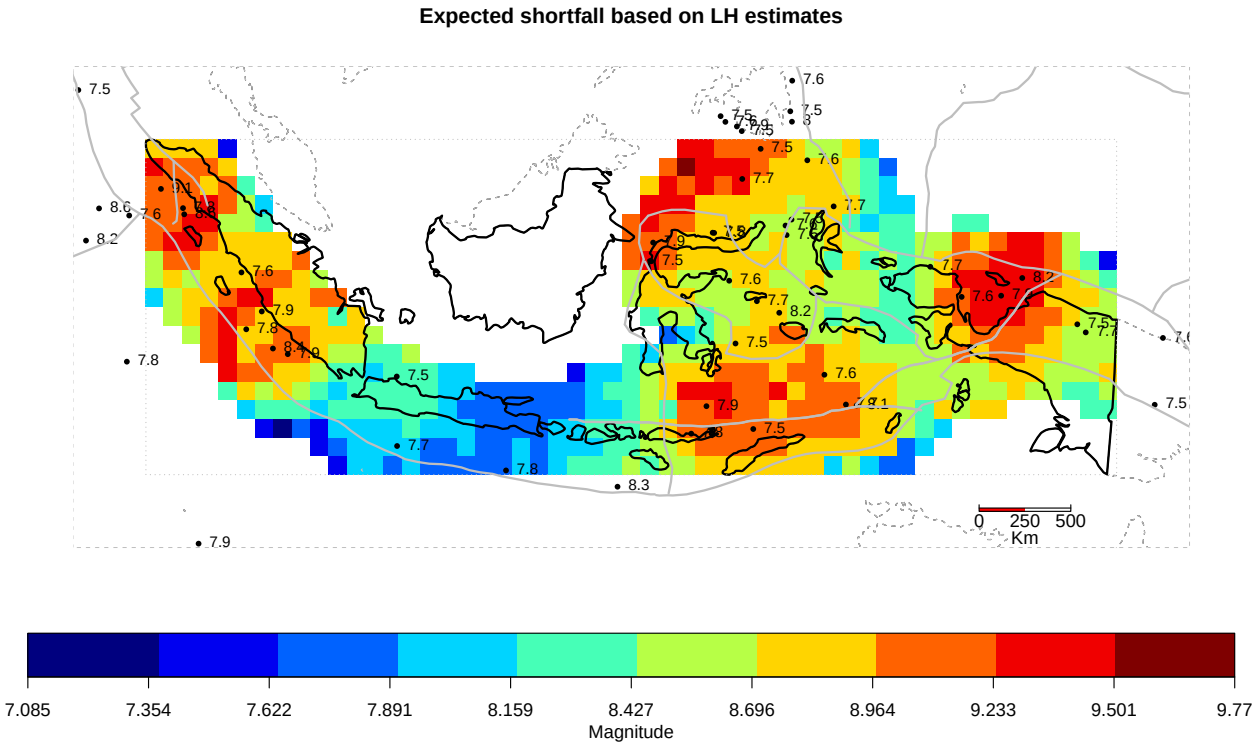
- the regression extremiles $\hat{\xi}^*_{1-p_n}(x)$ based on the estimates $LH_k(x)$:



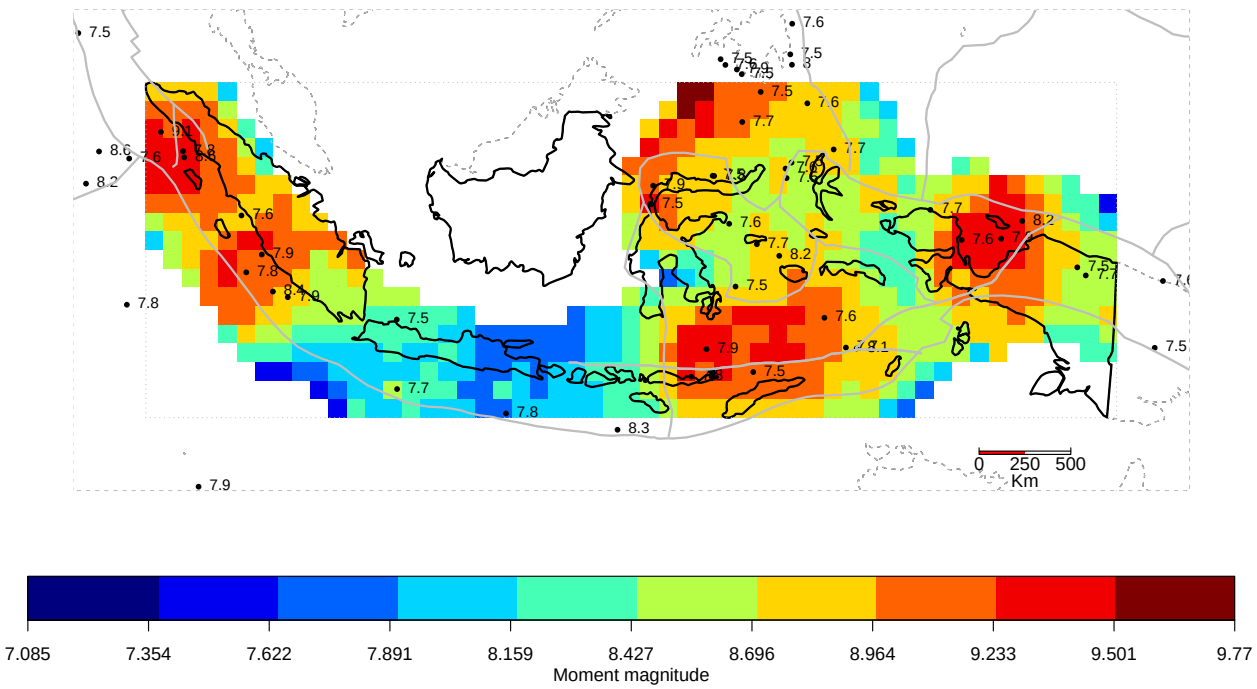
- the regression extremiles $\hat{\xi}^*_{1-p_n}(x)$ based on the estimates $LAH_{k,a}(x)$:



- the regression expected shortfall $\hat{\nu}^*_{1-p_n}(x)$ based on the estimates $LH_k(x)$:



- the regression expected shortfall $\hat{\nu}^*_{1-p_n}(x)$ based on the estimates $LAH_{k,a}(x)$:

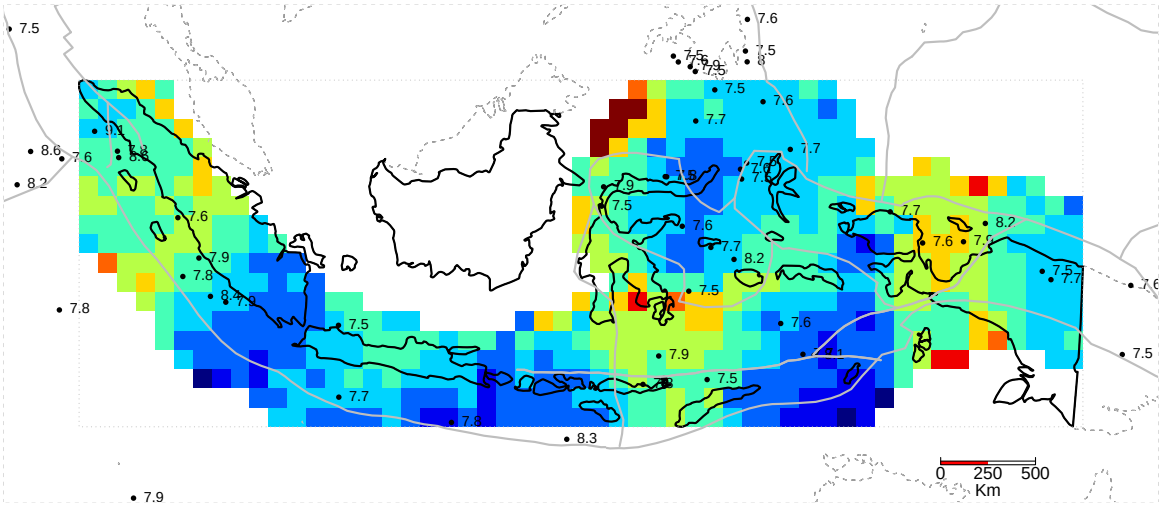


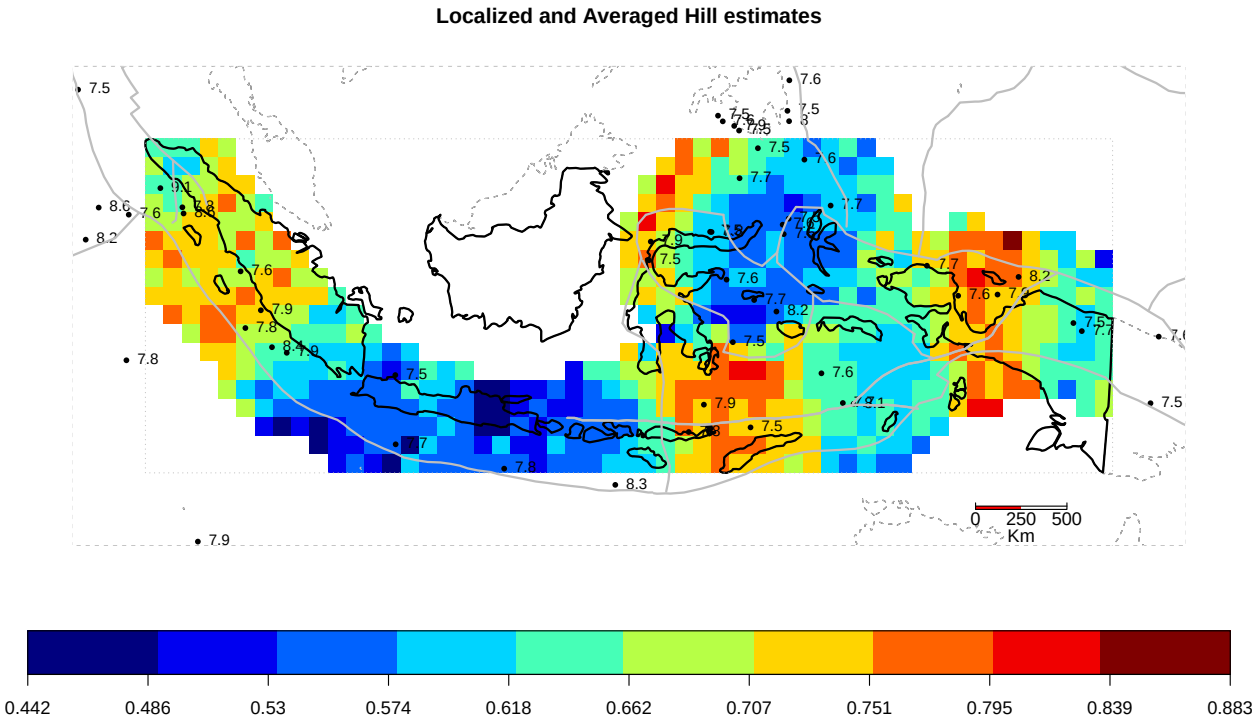
- Finally, we represent the estimates $LH_k(x)$ and $LAH_{k,a}(x)$ themselves :

Choice of the colours :

```
vec_values <- c(values(gama.raster_hill), values(gama.raster_stupfler))
bb.neighbour <- seq(min(vec_values,  na.rm = T),
                    max(vec_values, na.rm = T),
                    length.out = 11)
nb.col <- length(bb.neighbour) - 1
plotclr <- colorRampPalette(c("#00007F", "blue", "#007FFF",
                              "cyan", "#7FFF7F", "yellow",
                              "#FF7F00", "red", "#7F0000"))(nb.col)
```

Localized Hill estimates





4 References

- Daouia, A., Florens, J-P. and Simar, L. (2010). Frontier estimation and extreme value theory. *Bernoulli*, **16**, 1039–1063.
- Daouia, A., Gardes, L. and Girard, S. (2013). On kernel smoothing for extremal quantile regression. *Bernoulli*, **19**, 2557–2589.
- Daouia, A., Gijbels, I. and Stupfler, G. (2017). Extremiles: A new perspective on asymmetric least squares. *J. Amer. Statist. Assoc.*, in press. Available at https://www.tandfonline.com/doi/full/10.1080/01621459.2018.1498348.
- Daouia, A., Gijbels, I. and Stupfler, G. (2018). Extremile Regression. Submitted for publication.
- Gardes, L. and Stupfler, G. (2014). Estimation of the conditional tail index using a smoothed local Hill estimator. *Extremes*, **17**, 45–75.
- Goegebeur, Y., Guillou, A. and Osmann, M. (2014). A local moment type estimator for the extreme value index in regression with random covariates. *Canad. J. Statist.*, **42**, 487–507.
- Goegebeur, Y., Guillou, A. and Osmann, M. (2017). A local moment type estimator for an extreme quantile in regression with random covariates. *Communications in Statistics - Theory and Methods*, **46**, 319–343.
- Stupfler, G. (2013). A moment estimator for the conditional extreme-value index. *Electronic Journal of Statistics*, **7**, 2298–2343.