An add-on to the R-package GAMLSS for Generalized Additive Modelling of Extended Generalized Pareto Distributions

Noémie Le Carrer, Carlo Gaetan *

Dipartimento di Scienze Ambientali, Informatica e Statistica
Università Ca' Foscari di Venezia, Venice, Italy

March 24, 2022

Abstract

This article presents an add-on to the R-package `GAMLSS`, to model non-stationary fields whose margins are EGPD distributed, in the sense of Naveau et al. [15]. The `GAMLSS` package provides univariate distributional regression models, where parameters of the assumed distribution for the response can be modeled as additive functions of the explanatory variables. It is flexible enough to allow the encoding of novel families of distribution, as long as their density, cumulative, quantile and random generative function and, optionally, first and second (cross-) derivatives of the likelihood can be computed. We seize this potential to implement the EGPD in a generic way, allowing to introduce new parametric form, up to 4 parameters so far, satisfying the requirement of the EGPD family. We show the potential of this add-on by means of a reproducible toy example that can be associated with space-time modelling of e.g. hourly rainfalls.

1 Introduction

Exceedances of a high threshold are traditionally modelled by means of the generalized Pareto Distribution (GPD) that models [6]. Several R packages have been developed to fit GPDs and contributed to the Comprehensive R Archive Network (CRAN) in the field of extreme value analysis [25]. Among them, `ismev` [12], providing tools for the extreme value analyses presented in Coles [5], `evd` [24], `evir` [17], `extRemes` [10] and `mev` [1], have offered various approaches for fitting univariate and multivariate GPDs. Gilleland et al. [11] review them while the Dutang and Jaunatre [7] CRAN Task View provides an up-to-date list of the R packages in this field. In particular, some of them provide regression-based models for extremes GPD parameters are allowed to vary with covariates. Beyond the linear forms, i.e when the parameters or their transformations are linear combinations of the covariates, provided in the packages `ismev` and `evd`, a few R packages allow to fit GPDs with parameters in the additive forms i.e. when the when the parameters or their transformations can be modelled as additive smooth functions of the explanatory variables. In the sequel we term this form Generalized Additive Model (GAM).

Thus `VGAM` [29] and `gamlss` [19] provides this functionality for GPDs. A detailed overview of the alternatives to fit GPD parameters of GAM form is offered in Youngman [30]. When it comes to

*We thank Nicolas Berthier for helping us to develop some parts of the R code.

GAM form, the amount of smoothing is one of the most critical part of the fitting. `VGAM` requires the user to specify the degrees of freedom for smooths, from which smoothing parameter estimates are derived, following the idea that degrees of freedom are more intuitive to specify than smoothing parameters themselves. In `gamlss`, the function `find.hyper()` automatically minimizes a generalized Akaike information criterion (AIC) to find optimal degrees of freedom. Finally, the R-package `evgam` seizes the flexibility of the different smooth functions (e.g. thin plate regression splines, attractive for multidimensional processes such as spatial ones) available in `mgcv` for fitting GPDs with parameters of GAM form.

However powerful and made convenient through the above-mentioned R packages, the GPD distribution suffers from the fact that it only models excesses beyond a threshold. Although `evgam` provides the asymmetric Laplace distribution to estimate a quantile of GAM form and use it as threshold [31], the question of the threshold choice and then simulation of the whole spectrum of the random variable considered remains problematic. Forming mixtures of Exponential distributions [27, 13] or conditional Gamma distributions [14, 2] remains a competitive alternative e.g. for heavy-tailed precipitation modelling and simulation. Approaches of this type are implemented in the package `GSM`, that uses a Bayesian approach for estimating a mixture of Gamma distributions in which the mixing occurs over the shape parameter and the procedure only requires to specify a prior distribution for a single parameter. Package `evmix` fits an extreme value mixture model with normal behaviour for the bulk distribution up to the threshold and conditional GPD above the threshold. Options for profile likelihood estimation for threshold and fixed threshold approach are provided. Finally, `distributionsrd` provides the double Pareto-lognormal distribution. Although interesting, in practice these packages do not allow to fit nonlinear models as flexible as the GAM options allowed for the GPD alone. Besides, the mixture approaches tend to quickly inflate the number of parameters and the inference of the latter may be problematical operationally [15]. Papastathopoulos and Tawn [16] opened another route in the field of heavy-tail distribution modelling by replacing the uniform draws in the inverse cumulative distribution function (CDF) transform to generate simulations of the GPD, by draws from a richer distribution e.g. type Beta distribution. Using this approach, Naveau et al. [15] presented a concise parametric formulation (later extended to a semi-parametric formulation [15, 26]) for simulating the whole range of a positive random variable (namely precipitations), that makes use of GPD to model not only the high quantiles but the smallest ones as well and bypass the issue of threshold selection. Four parametric families of this Extended Generalized Pareto distribution (EGPD) are implemented in the R package `mev`. However, to our knowledge, no extension has been implemented yet to allow the fitting of nonlinear models. The `GAMLSS` package provides univariate distributional regression models, where parameters of the assumed distribution for the response can be modeled as additive functions of the explanatory variables. It is flexible enough to allow the encoding of novel families of distribution, as long as their density, cumulative, quantile and random generative function and, optionally, first and second (cross-) derivatives of the likelihood can be computed. We seize this potential to implement the parametric EGPD family in a generic way, allowing to introduce new parametric forms, up to 4 parameters so far (due to limitations of GAMLSS), satisfying the requirement of the EGPD family. The remaining of this article will first introduce the EGPD and the GAM form for distribution parameter regression (Section 2), followed by the key commands to use the EGPD within the GAMLSS package and our add-on (Section 3). Section **??** implements them to perform a modelling and analysis of hourly positive rainfalls on the North-west of France. Finally, Section **??** pursue the analysis to discuss specific advantages and limitations of GAMLSS modelling and its implementation in R, while Section 5 concludes the paper.

2 A GAM framework for the EGPD

2.1 Extended-GPD

The Extended Generalized Pareto Distribution (EGPD) Naveau et al. [15] accounts simultaneously and in a parsimonious way for both extreme and non extreme values of a random variable X . Its cumulative distribution function (CDF) reads:

$$F(x) = G\{H_\xi(x/\sigma)\}\ ,\ \forall x > 0 \tag{1}$$

where H_ξ represents the CDF of the GPD of shape and scale parameters ξ and $\sigma > 0$, namely:

$$H_\xi(z) = \begin{cases} 1 - (1 + \xi z)_+^{-1/\xi} & \text{for } \xi \neq 0 \\ 1 - \exp\{-z\} & \text{for } \xi = 0 \end{cases} \tag{2}$$

where $a_+ = \max(a, 0)$.

G is a continuous CDF on the unit interval. It is constrained by several assumptions: 1) we need to ensure that the upper-tail behavior of F follows a GPD; 2) the low values of X, seen as the upper tail of $-X$ when $x \to 0$, follows a GPD as well.

This leads to the following consequences for any candidate function G:

1. the upper-tail behavior of $1 - F(x)$ is equivalent to the original GPD tail used to build $F(x)$;
2. when $x \to 0$, $F(x) \sim \dfrac{c}{\sigma^s} x^s$ where $c = \lim\limits_{x \to 0} \dfrac{G(u)}{u^s}$ is positive and finite for some real s.

Four parametric families that satisfies the above-mentioned constraints are proposed in Naveau et al. [15] for $G(v)$, where $v \in [0, 1]$:

1. $G(v) = v^\nu$, $\nu > 0$
2. $G(v) = pv^{\nu_1} + (1 - p)v^{\nu_2}$, $\nu_2 \geq \nu_1 > 0$ and $p \in [0, 1]$
3. $G(v) = 1 - Q_\delta\{(1 - v)^\delta\}$, $\delta > 0$ where Q_δ is the CDF of a Beta random variable with parameters $1/\delta$ and 2, that is: $Q_\delta(v) = \frac{1+\delta}{\delta} v^{1/\delta}\left(1 - \frac{v}{1+\delta}\right)$
4. $G(v) = [1 - Q_\delta\{(1 - v)^\delta\}]^{\nu/2}$, $\nu, \delta > 0$

In model 1, ξ controls the rate of upper tail decay, ν the shape of the lower tail and σ is a scale parameter. $\nu = 1$ allows us to recover the GPD model, while varying it gives flexibility in the description of the lower tail. Model 2, as a mixture of power laws, allows to increase the flexibility provided by model 1. Therein, ν_1 controls the lower tail behavior while ν_2 modifies the shape of the density in its central part. Model 3 is connected to the work of Falk et al. [9]. ξ keeps controlling the upper extreme tail, δ the central part of the distribution in a threshold tuning way. However, this model imposes a behavior type X^2 for the lower tail (null density at 0) instead of allowing the data to constrain it. Hence model 4, which adds an extra parameter ν to circumvent this limitation. In this last model, ν, δ, ξ control respectively the lower, moderate and upper parts of the distribution. By construction, the lower and upper tails are GPD of shape parameters ν and ξ respectively. To simulate from Equation (1), the user randomly draws a uniform variable U in $[0, 1]$ and then applies the quantile function $X = F^{-1}(U)$, given for $p \in [0, 1]$ by:

$$x_p = F^{-1}(p) = \begin{cases} \frac{\sigma}{\xi}[\{1 - G^{-1}(p)\}^{-\xi} - 1] & \text{for } \xi \neq 0 \\ -\sigma \log\{1 - G^{-1}(p)\} & \text{for } \xi = 0 \end{cases} \tag{3}$$

2.2 Generalized additive model framework

In practice, X may be a random variable indexed by covariate $\boldsymbol{y}$. `GAMLSS` (standing for Generalized Additive Models for Location, Scale and Shape) allows a straightforward fitting of multiple built-in and well-known family distributions with parameters that vary flexibly with $\boldsymbol{y}$. Parameters are predicted from a sum of smooth parametric functions of the covariates. Such a representation is particularly flexible and allows to capture complex dependence patterns between distribution parameters and explanatory variables $\boldsymbol{y}$, along with uncertainty estimation. Conversely, it is less powerful than a purely stochastic modeling of the parameters variations (e.g. considering each EGPD parameter as a random field) for extrapolation beyond the borders of the training space, due to the limitations of spline modelling `per se`.

Let $\boldsymbol{\theta} = (\theta_1, ..., \theta_p)$ be the vector of p parameters, y_{k_j} the j-th covariate for parameter θ_k, η_{θ_k} monotonic link functions and h_j unknown, parametric or not, functions that are different for each parameter θ_k. The GAM form of `GAMLSS` then reads, for each distribution parameter θ_k:

$$\eta_{\theta_k}(\boldsymbol{y}_k) = \beta_0 + h_1(y_{k_1}) + +h_2(y_{k_2}) + h_3(y_{k_3}, y_{k_4}) + ..., \text{ for } k = 1, ..., p \tag{4}$$

which in the case of a space time field $X(\boldsymbol{s}, t)$ boils down to:

$$\eta_{\theta_k}(\boldsymbol{s}, t) = h_1(\boldsymbol{s}) + h_2(t). \tag{5}$$

The unknown h_j are expressed as sums of known basis functions: $h_j(y_k) = \sum_{i=1}^{d_j} \beta_{ji} b_{ji}(y_k)$ where the β_{ji} are the coefficients to be estimated and d_j the dimension of the basis [28]. These smoothers h_j, or splines, can be of different types, thoroughly presented e.g. in Stasinopoulos et al. [23] and summarized in the following section.

2.3 Additive terms

To account for the varied effects of explanatory variables on the parameters of the specified distribution, GAMLSS models allow linear or nonlinear parametric functions, or nonparametric smoothing functions of explanatory variables. Additive terms included in the gamlss package are P-splines (penalized B-splines, a GAMLSS implementation of the Eilers and Marx [8]'s P-splines methodology) `pbc()`, monotone P-splines `pbm()`, cycle P-splines `pbc()`, varying coefficient P-splines `pbc()` in the multivariate case, P-splines with coefficient shrinking to 0 if needed `pbz()` (i.e. the fitted parameter becomes a constant), cubic smoothing splines `cs()`, `css()` and their cyclic version `cy()`, locally weighted scatterplot smoothing [3] `lo()`, neural networks `nn()` and decision trees `tr()` via the `gamlss.add` package giving access to the packages `nnet` and `rpart`, fractional polynomials `fp()` and `bfp()`, random effects `random()` or `re()` at the factor level, ridge regression `ri()` and nonlinear parametric fits `nl()`. The `gamlss.add` package also allows to use the power of the package `mgcv` and in particular its interface function `ga()` giving access to more than one-dimensional smoothers, such as thin plate cubic splines `s()` and tensor products `tp()`, efficiently implemented within `mgcv` but limited to a response variable distribution within the exponential family therein. This family of additive terms is particularly interesting for smooth surface fitting, e.g. in the case of spatial coordinates used as covariates. We refer the reader to the GAMLSS documentation [22] for implementation details on each of these terms.

2.4 Inference

As described in Rigby et al. [22], the GAMLSS model can be written:

$$\begin{aligned} X &\overset{\text{ind}}{\sim} \mathcal{D}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\nu}, \boldsymbol{\tau}) \\ \boldsymbol{\eta}_1 &= g_1(\boldsymbol{\mu}) = \boldsymbol{Y}_1\boldsymbol{\beta}_1 + s_{11}(\boldsymbol{y}_{11}) + ... + s_{1J_1}(\boldsymbol{y}_{1J_1}) \\ \boldsymbol{\eta}_2 &= g_2(\boldsymbol{\sigma}) = \boldsymbol{Y}_2\boldsymbol{\beta}_2 + s_{21}(\boldsymbol{y}_{21}) + ... + s_{2J_2}(\boldsymbol{y}_{2J_2}) \\ \boldsymbol{\eta}_3 &= g_3(\boldsymbol{\nu}) = \boldsymbol{Y}_3\boldsymbol{\beta}_3 + s_{31}(\boldsymbol{y}_{31}) + ... + s_{3J_3}(\boldsymbol{y}_{3J_3}) \\ \boldsymbol{\eta}_1 &= g_4(\boldsymbol{\tau}) = \boldsymbol{Y}_4\boldsymbol{\beta}_4 + s_{41}(\boldsymbol{y}_{41}) + ... + s_{4J_4}(\boldsymbol{y}_{4J_4}) \end{aligned} \tag{6}$$

with $\mathcal{D}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\nu}, \boldsymbol{\tau})$ the distribution of the response variable X, Y_k the design matrices containing the linear additive terms in the model, β_k the linear coefficient parameters and $s_{kj}(\boldsymbol{y}_{kj})$ the smoothing functions for explanatory variables $\boldsymbol{y}_{kj}$, for $k = 1, 2, 3, 4$ and $j = 1, ..., J_k$. The explanatory variables in $\boldsymbol{Y}$ are not necessarily the same as the $\boldsymbol{y}_{kj}$ in the smoothers. The vectors $\boldsymbol{\eta}_1$, $\boldsymbol{\eta}_2$, $\boldsymbol{\eta}_3$ and $\boldsymbol{\eta}_4$ are the predictors of μ, σ, ν, τ respectively and $g_i, i = 1, ..., 4$ are the link functions associated with the corresponding parameter.

In the GAMLSS model, the smooth functions most often take the form $s(\boldsymbol{y}) = Z\gamma$. Z represents the basis matrix, depending on covariates $\boldsymbol{y}$ while γ is a parameter vector to be estimated, subject to a quadratic penalty $\lambda\gamma^\top G\gamma$. $G = D^\top D$ is a known matrix and the hyperparameter λ regulates the amount of smoothing needed for the fit. The type and properties of the smoothing functions are controlled by the form of matrices Z and D.

If we write $\beta = \left(\beta_1^\top, \beta_2^\top, \beta_3^\top, \beta_4^\top,\right)$ the fixed parameters and $\gamma = (\gamma_{11}^\top, ..., \gamma_{1J_1}^\top, \gamma_{21}^\top, ..., \gamma_{4J_4}^\top)$ the random effect parameters, the more general random effects GAMLSS consequently reads:

$$
\begin{aligned}
X|\gamma \overset{\text{ind}}{\sim} \mathcal{D}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\nu}, \boldsymbol{\tau}) & \\
\boldsymbol{\eta}_1 = g_1(\boldsymbol{\mu}) &= \boldsymbol{Y}_1\boldsymbol{\beta}_1 + Z_{11}\gamma_{11} + ... + Z_{1J_1}\gamma_{1J_1} \\
\boldsymbol{\eta}_2 = g_2(\boldsymbol{\sigma}) &= \boldsymbol{Y}_2\boldsymbol{\beta}_2 + Z_{21}\gamma_{21} + ... + Z_{2J_2}\gamma_{2J_2} \\
\boldsymbol{\eta}_3 = g_3(\boldsymbol{\nu}) &= \boldsymbol{Y}_3\boldsymbol{\beta}_3 + Z_{31}\gamma_{31} + ... + Z_{3J_3}\gamma_{3J_3} \\
\boldsymbol{\eta}_1 = g_4(\boldsymbol{\tau}) &= \boldsymbol{Y}_4\boldsymbol{\beta}_4 + Z_{41}\gamma_{41} + ... + Z_{4J_4}\gamma_{4J_4}
\end{aligned}
\tag{7}
$$

The parametric GAMLSS model requires only the estimate of the βs, and is fitted by maximum likelihood estimation w.r.t. parameters β. The random effect model estimates the βs, the γs, and the λs, and uses maximum penalized likelihood estimation, equivalently posterior mode or maximum a posteriori estimation, w.r.t. β and γ for fixed λ. The penalized log-likelihood function for the latter models reads:

$$
l_p = l - \frac{1}{2}\sum_{k=1}^{4}\sum_{j=1}^{J_k}\gamma_{kj}^\top G_{kj}(\lambda_{kj})\gamma_{kj} \tag{8}
$$

where l represents the log-likelihood function for the parametric model, where the observations are assumed independent, is given by:

$$
l = \sum_{i=1}^{n}\log f(x_i|\mu_i, \sigma_i, \nu_i, \tau_i) \tag{9}
$$

with f the parametric PDF for the marginal distributions of x.

The two algorithms provided in GAMLSS [19] for maximizing the penalized likelihood with respect to β and γ for given λ are the RS and CG algorithms. RS generalizes the algorithm used by the authors Rigby and Stasinopoulos [21, 20] to fit mean and dispersion additive models (MADAM). In particular, it does not use the cross-derivatives of the log-likelihood. CG generalizes the Cole and Green [4] and requires information about the first and (expected or approximate) second and cross derivatives of the log-likelihood function with respect to parameters μ, σ, ν and τ and corresponding pairs (cross-derivatives).

Each algorithm works slightly differently but both lead, for a given λ, to the maximum penalized log-likelihood estimates for the βs and γs. To reach the (penalized) maximum likelihood parameter estimates, the RS algorithm maximizes the (penalized) log-likelihood over each of μ, σ, ν and τ in turn, cycling until convergence. Thanks to the information about the cross derivatives, the CG algorithm is able to jointly update (μ, σ). In practice, the RS algorithm is much more stable than the CG algorithm and used as default, however for highly correlated fitted distribution parameters the RS algorithm can be slower, and may converge early before reaching the maximum log-likelihood.

3 Implementing the EGPD family with GAMLSS

We first use the following set of commands to load the `GAMLSS` package [18], the generic implementation of EGPD provided in `GenericEGPDc.R` and create the desired parametric model of EGPD by providing G as a function of z and at maximum two parameters ν and τ (so far, due to limitations of the GAMLSS implementation). In our current implementation we use the symbolic derivation, by means of the package `Deriv`, for remaining flexible and allowing new definitions of parametric distribution G. Therefore the inverse function $G^{-1}(u)$ is consequently computed numerically.

In the following, we use as an illustration the Model 1 from Naveau et al. [15], $G(z) = z^{\nu}, \nu > 0$ (EGPD1).

```
library(gamlss)
source ("GenericEGPDc.R")
EGPD1Family <- MakeEGPD (function (z,nu) z^nu, Gname = "Model1")
EGPDModel1 <- EGPD1Family()
```

The last command is necessary and allows to create the object EGPDModel1 (name concatenating `EGPD` and `Gname`) from the EGPD1Family function.

Given parameters ξ_0, σ_0 and ν_0, we can directly use the density function (PDF), CDF, quantile function (inverse CDF) and EGPD1-distributed random number generation function, respectively:

```
dEGPDModel1(x,mu=mu0,sigma=sigma0,nu=nu0)

pEGPDModel1(x,mu=mu0,sigma=sigma0,nu=nu0)

qEGPDModel1(u,mu=mu0,sigma=sigma0,nu=nu0)

rEGPDModel1(n,mu=mu0,sigma=sigma0,nu=nu0)
```

The general expression for the distribution parameters in GAMLSS is μ ("mu"), σ ("sigma"), ν ("nu"), τ ("tau"). We consequently call the shape parameter ξ, μ ("mu") hereafter. To change the link functions $\eta_i = g_i(\theta_i)$ by means of which each EGPD parameter θ_i (here ξ, σ or ν) is indirectly optimised, or the default starting values for the parameters to fit, or to print these links, we use the commands:

```
EGPD1Family(mu.link="identity",mu.init=1, # log , inverse, own , ...
            sigma.link="log",sigma.init=3,
            nu.link="log",nu.init=0.5)
show.link(EGPD1Family)
```

It is possible to create original link functions "own", by gathering the expressions of the link function η_i, its inverse, the derivative of the latter w.r.t. parameter θ_i and its area of validity. For instance to create a log-link function shifted to the right for μ [1]:

```
# the link function defining the predictor eta  as a function of the current
    distribution parameter (referred to as mu), i.e. eta = g(mu)
own.linkfun <- function (mu) {
    .shift = 0.0001
    log (mu - .shift) }
#  the inverse of the link function as a function of the predictor eta, i.e.mu =  g 1
     (eta)
own.linkinv <- function (eta) {
    shift = 0.0001
    thresh <- - log (.Machine$double.eps)
    eta <- pmin (thresh, pmax(eta, -thresh))
    exp (eta) + shift }
# the first derivative of the inverse link with respect to eta
own.mu.eta <- function (eta) {
    shift = 0.0001
    pmax (exp (eta), .Machine$double.eps) }
# the range in which values of eta are defined.
```

[1] More on link function definition p. 179 of Stasinopoulos et al. [23])

```
own.valideta <- function (eta) TRUE

EGPD1Family(mu.link="own")
```

4 Simulation and fitting of nonstationary EGPD marginal distributions

4.1 Creating PDF, CDF, quantile function and random number generation function

We illustrate our add-on to `GAMLSS` on a synthetic and reproducible experiment. Let us note X the random variable of interest (e.g. hourly precipitations). X has covariates Y_l, Y_L, Y_c, where Y_l and Y_L represents the position in space (e.g. longitude, latitude) and Y_c a cyclic time index, namely the day of the year omitting lap years ($Y_c \in [1, 365]$).

We first simulate a field $X(Y_l, Y_L, Y_c)\}$ over $N_{stat} = 10$ stations identified by their 2-dimensional coordinates ("latitude","longitude") in a square of size 10×10 and over 20 years characterized by their cyclic index "cyc". In a first step, the field is marginally distributed according to the EGPD1 family, with constant parameters $\mu_0 = -0.5, \sigma_0 = 1, \nu_0 = 2$. We use the random generation function rEGPD1Family described above.

```
mu0 <- -0.5
sigma0 <- 1
nu0 <- 2

Ny <- 20
N <- Ny*365
Nstat <- 10
db <- data.frame(matrix(ncol = 4, nrow = N*Nstat))
colnames(db) <- c("longitude","latitude","precip","cyc")
Lat=runif(10,min=0,max=10)
Long=runif(10,min=0,max=10)
time=1:365
for (j in 1:Nstat){
  p <- runif(N)
  pr.Sim <- numeric(N)
  pr.Sim <- qEGPDModel1(p,mu=mu0,sigma=sigma0,nu=nu0)
  db$latitude[((j-1)*N+1):(j*N)] <- Lat[j] # repmat(Lat[j],N,1)
  db$longitude[((j-1)*N+1):(j*N)] <- Long[j] #repmat(Long[j],N,1)
  db$cyc[((j-1)*N+1):(j*N)] <- rep(time,Ny) #repmat(time,1,Ny)
  db$precip[((j-1)*N+1):(j*N)] <-pr.Sim
}
```

Figure 1 shows the CDF and PDF empirically observed and their theoretical counterparts, obtained by using directly pEGPD1Family and dEGPD1Family respectively. As expected for a negative shape parameter μ_0, the function is bounded.

```
Fn <- ecdf(db$precip)
# Compute theoretical CDF and PDF
if(mu0<0){
  q <- 0.1*(1:((-1/mu0)*10)) # support defined for 0 <= z <= -1/mu for mu < 0
}else{
  q <- 0.1*(1:(max(db$precip)*10)) }
simCDF <- numeric(length(q))
simPDF <- numeric(length(q))
simCDF <- pEGPDModel1(q,mu=mu0,nu=nu0,sigma=sigma0)
simPDF <- dEGPDModel1(q,mu=mu0,nu=nu0,sigma=sigma0,log=FALSE)

# Sample directly from random number generation function
indTS <- rEGPDModel1(10000,mu=mu0,sigma=sigma0,nu=nu0)
Fn <- ecdf(indTS)
```

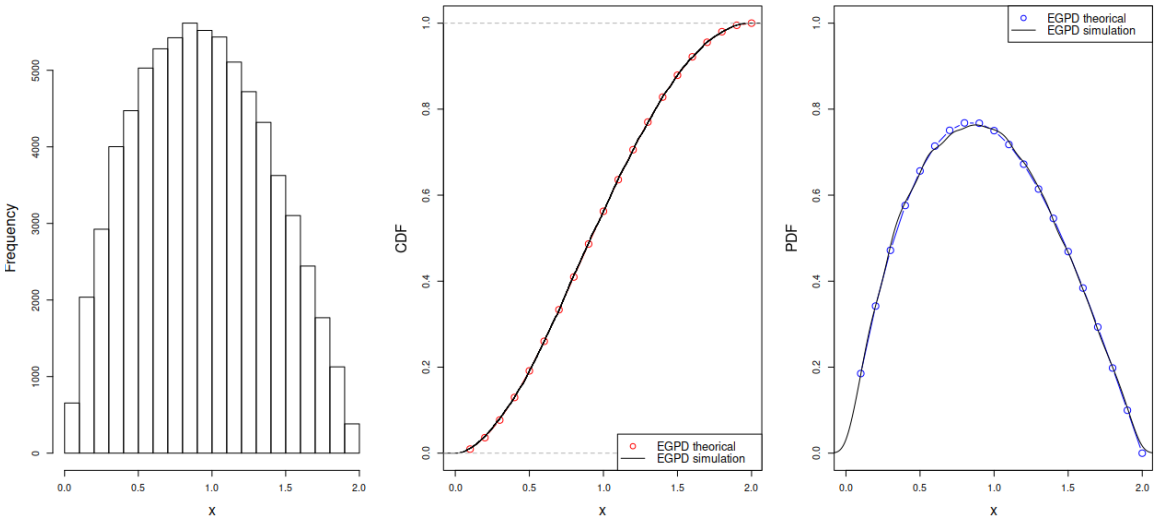

Figure 1: Empirical distributions of the simulations of EGPD1 and associated theoretical curves.

4.2 Model fitting

Fitting the parameters in the constant hypothesis (which in terms of regression reads $x \sim 1$, stationarity in both space and time) is performed by using the `gamlss()` function. `predict()` and `fitted` allow to extract the fitted parameters. The former can also be used to predict the response variable distribution parameters for old or new data values of the covariates as will be illustrated later.

Through the control option "con" we provide starting values (all 1) for the parameters, we limit the number of cycles to 100, we put the default step-size to 0.1 for all parameters and we allow (autostep=TRUE, which is the default in `GAMLSS`) the steps to be halved automatically if the new global deviance is greater than the old one. Maximum (penalized) likelihood estimation is used to fit the models and a Newton-Raphson/Fisher scoring algorithm is used to maximize the (penalized) likelihood. We use the `RS()` algorithm for fitting mean and dispersion in additive models. It is slower than the `CG()` algorithm but we found that when it comes to EGPD the latter did not converge well, even when combined to `RS()` in the option `method=mixed()`. Note that the default link function for μ is the log link, which does not allows negative values for μ. If negative values are a possibility, the link function must be set to "identity" as done below.

```
con <- gamlss.control (n.cyc = 100,
                       mu.step = 0.1, sigma.step = 0.1, nu.step = 0.1,
                       tau.step = 0.1, trace = FALSE, autostep=TRUE)

mod.egpd.0 <- gamlss(x ~ 1, data = db, family = EGPD1Family(mu.link = "identity"),
                        control = con,
                        mu.start=1, sigma.start=1, nu.start=1,
                        method=CG())

muFit <- fitted(mod.0,"mu")[1]
sigmaFit <- predict(mod.egpd.0,what="sigma", type="response")[[1]]
nuFit <- predict(mod.egpd.0,what="nu", type="response")[[1]]

print(muFit)
[1] -0.5032065
print(sigmaFit)
[1] 1.004812
```

```
print(nuFit)
[1] 1.987257
```

4.3 Simulating and fitting nonstationary EGPD marginal distributions

In practice, the field $X(Y_L, Y_l, Y_c)$ often has margins whose parameters vary in space and time (e.g. due to seasonality or relief variations). Modelling such field consists in regressing the parameters of the marginal distributions as additive functions of possible explanatory variables, here space (Y_L, Y_l) and time or a time-index capturing seasonality Y_c. We consequently reproduce the previous experiment yet we add a linear trend to the parameter μ, setting *a priori* $\mu_0 = \mu(y_L = 0)$ and $\mu_{10} = \mu(y_L = 10)$. We also consider that the shape parameter varies with the day of the year, that is: $\mu(y_L, y_l, y_c) = \mu(y_L) * (1 + 0.4 * \sin(2 * y_c * \pi/200))$. We then fit the EGPD-based model to this database (*db* hereafter), this time assuming a linear dependence between μ and Y_L and a nonlinear cyclic dependence between μ and Y_c.

```
mu0 <- 0.1
mu10 <- 0.5
sigma0 <- 1
nu0 <- 2

Ny <- 20
time=1:365
N <- Ny*length(time)
Nstat <- 10
db <- data.frame(matrix(ncol = 4, nrow = N*Nstat))
colnames(db) <- c("longitude","latitude","x","cyc")
Lat=runif(10,min=0,max=10)
Long=runif(10,min=0,max=10)
for (j in 1:Nstat){
  mu_lat <- ((mu10-mu0)/10)*Lat[j]+mu0
  mu_lat_year <- mu_lat * (1+ 0.4* sin(2*time*pi/200))
  mu_lat_allyears <- rep(mu_lat_year,Ny)
  p <- runif(N)
  pr.Sim <- numeric(N)
  for (i in 1:N) {
    pr.Sim[i] <- qEGPDModel1(p[i],mu=mu_lat_allyears[i],sigma=sigma0,nu=nu0)
  }
  db$latitude[((j-1)*N+1):(j*N)] <- Lat[j]
  db$longitude[((j-1)*N+1):(j*N)] <- Long[j]
  db$cyc[((j-1)*N+1):(j*N)] <- rep(time,Ny)
  db$x[((j-1)*N+1):(j*N)] <-pr.Sim
}

# Fit parameters (here only for mu, others are considered constant), without printing
    the trace of inner iterations (con.i):
con.i=glim.control(glm.trace = FALSE)
mod.egpd.1 <- gamlss(x~latitude+pbc(cyc),
                              data=db,
                              family = EGPD1Family,
                              control = con,
                              i.control = con.i,
                              mu.start=1,sigma.start=1,nu.start=1,
                              method=CG())

muFit <- fitted(mod.egpd.1,"mu")
sigmaFit <- predict(mod.egpd.1,what="sigma", type="response")[[1]]
print(sigmaFit)
[1] 0.983516
> nuFit <- predict(mod.egpd.1,what="nu", type="response")[[1]]
print(nuFit)
[1] 2.018579

```

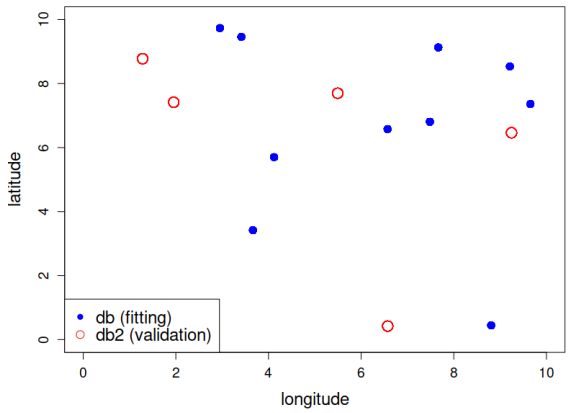

Figure 2: Location of the stations used in the fitting and validation sets respectively.

```
# See fitted parameters as functions of covariates
fittedPlot(mod.egpd.1, x=db$cyc, line.type = TRUE,xlab="Cyclic time index")
```

The function `fittedPlot()` allows to visualize the fitted parameters as a function of the covariate of interest. Information about the model, significance of parameters, among others, are obtained by means of the function `summary()`. However when fitting a smooth nonparametric term in `GAMLSS`, the displayed coefficient of the smoothing term and its standard error (s.e.) only refer to the *linear component* of the term. Approximate significance of the smoothing terms is better obtained using the function `drop1()`, but this may be slow for large datasets and many smoothing terms. `term.plot()` allows to visualise the whole smoothing function and assess the effect of each covariate on a given parameter.

To obtain the effective degrees of freedom for all the distribution parameters, we use `edfAll()`. When it comes to diagnostic plots, useful commands include `plot()` and `wp()` that plot 1) the normalized quantile residuals against fitted values and against index, the density estimate vs quantile residuals and normalized QQ plots and 2) worm plots respectively.

4.4 Prediction

`predictAll()` predicts parameter values for new covariate values. In the example below, we create a second database *db*2 containing a similarly EGPD1-distributed field $X(Y_L, Y_l, Y_c)$ over 20 years and 5 stations, whose locations are represented Figure 2. We then predict the EGPD1 parameters associated with these new samples from *mod.egpd*.1 and the covariates values in *db*2.

```
Nstat <- 5
db2 <- data.frame(matrix(ncol = 4, nrow = N*Nstat))
colnames(db2) <- c("longitude","latitude","x","cyc")
Lat=runif(10,min=0,max=10)
Long=runif(10,min=0,max=10)
for (j in 1:Nstat){
  mu_lat <- ((mu10-mu0)/10)*Lat[j]+mu0
  mu_lat_year <- mu_lat * (1+ 0.4* sin(2*time*pi/200))
  mu_lat_allyears <- rep(mu_lat_year,Ny)
  p <- runif(N)
  pr.Sim <- numeric(N)
  for (i in 1:N) {
    pr.Sim[i] <- qEGPDModel1(p[i],mu=mu_lat_allyears[i],sigma=sigma0,nu=nu0)
  }
```

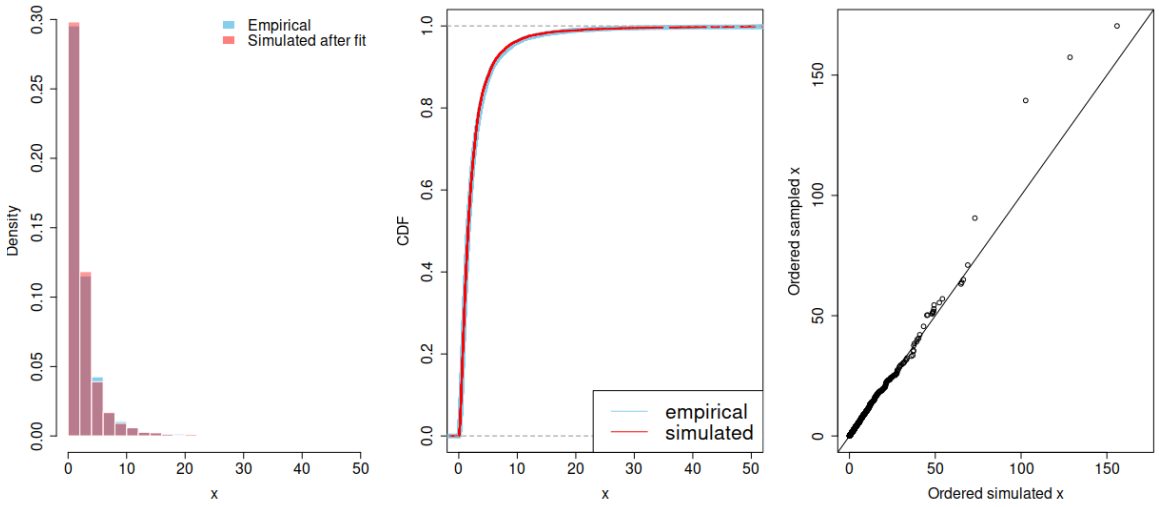

Figure 3: Density, CDF and QQ plot of the empirical observations in station 4 of *db*2 compared to their simulated counterpart after EGPD1 parameter fitting.

```
  db2$latitude[((j-1)*N+1):(j*N)] <- Lat[j]
  db2$longitude[((j-1)*N+1):(j*N)] <- Long[j]
  db2$cyc[((j-1)*N+1):(j*N)] <- rep(time,Ny)
  db2$x[((j-1)*N+1):(j*N)] <-pr.Sim
}
all <- predictAll(mod.egpd.1,newdata=db2,data=db)
muFit2 <- all$mu
sigmaFit2 <- all$sigma
nuFit2 <- all$nu
```

We compare the theoretical PDF and CDF (using the fitted parameters) in a given station (namely station 4 where longitude=1.9, latitude=7.4, that is a station outside the area covered by the set of stations used for fitting) with respect to their empirical counterparts. The result is shown in Figure 4 and . We observe a very good fit, at the exception of extremely large quantiles in the QQ plot. To assess the robustness of the algorithm, we remove a quarter of the number of observations in database *db* and fit a fully nonlinear model (thin splines in latitude, longitude and cyclic splines in the time index, see *mod.egpd.*3 in the next section). We repeat the experiment with this time only 7% of the data from *db*. Figure 4 reports maps over the domain of reference of the predicted μ parameter for days $cyc = 1$ and $cyc = 90$ respectively, while Figure 5 represents the parameters fitted with respect to the covariates latitude and cyclic time index. Even though much of the data is missing, we verify the linear trend for μ w.r.t. latitude and observe a cyclic pseudo-sinusoidal signal w.r.t. the time index, all the more fuzzy than little data is used for the fit.

4.5 Model comparison: information criteria and functions

We now try to select the best EGPD1-based model for fitting our data and compare it to a base Gamma-based model, commonly used for modelling the whole range of e.g. non-null precipitations, that is the traditional and easy-to-implement alternative to the EGPD. We design four models: `mod.egpd.0` has constant parameters and EGPD1-distributed margins, `mod.egpd.1` is linear in both time (month) and space (latitude) for μ and constant for other parameters, `mod.egpd.2` remains linear in space (latitude) and uses cyclic cubic splines to model the dependency in time of the shape parameter while considering others constant. Finally, `mod.egpd.3` models use smooth surface fitting

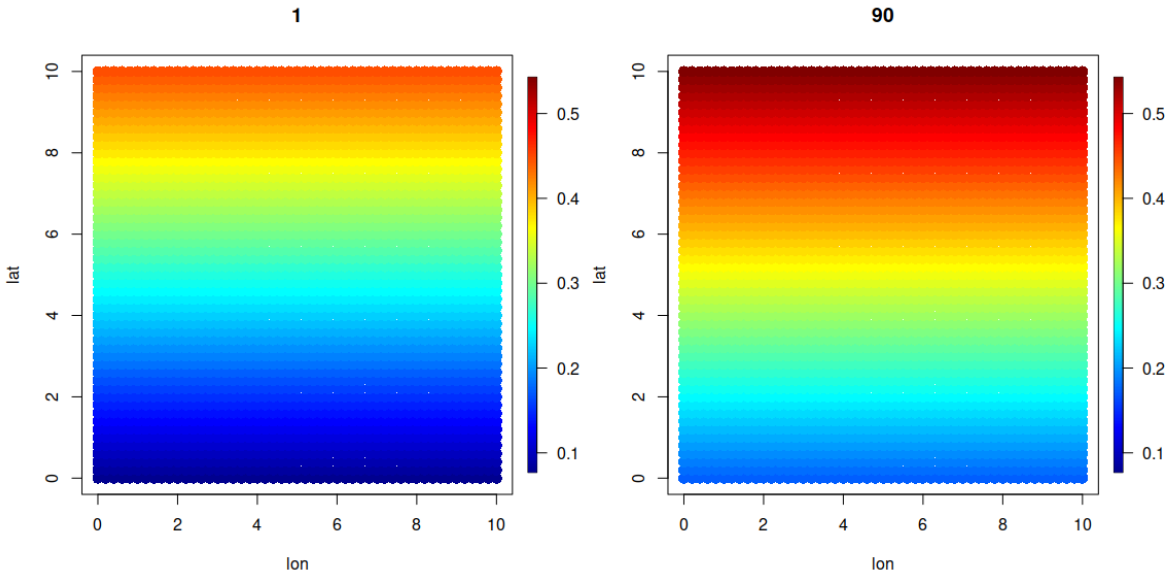

Figure 4: Maps of the predicted parameters for day of the year $cyc = 1$ and $cyc = 90$ (left to right respectively), when only three quarters of db are used for fitting.

(thin planes for latitude and longitude) and cyclic cubic splines to model the combined dependency in space and time for both μ and ν parameters. For each of these models, a Gamma-based equivalent, `mod.ga.i`, $i \in \{0, 1, 2, 3\}$ is fitted for comparison. Database db (fitting set, see Figure **??**) is used to fit each of these models. We then compare them based on their fitting performance on the fitting set and validate them with the performances on the validation set ($db2$).

The `GAIC(model,k=a)` function allows to compute both the global deviance GD (minus twice the fitted log-likelihood, $a = 0$), GD hereafter, the Akaike Information Criterion ($a = 2$) and the Bayesian Information Criterion (BIC) ($a = \log(\text{number of observations})$) for each model and to compare them. From results reported in Tables 1, all models based on EGPD have significantly lower GD and BIC than the Gamma-based models, showing that EGPD is more appropriate to model the marginal distributions at hand.

```
GAIC(mod.egpd.0,mod.egpd.1,mod.egpd.2,mod.egpd.3,mod.ga.0,mod.ga.1,mod.ga.2,mod.ga.3,k
    =0)

GAIC(mod.egpd.0,mod.egpd.1,mod.egpd.2,mod.egpd.3,mod.ga.0,mod.ga.1,mod.ga.2,mod.ga.3)

GAIC(mod.egpd.0,mod.egpd.1,mod.egpd.2,mod.egpd.3,mod.ga.0,mod.ga.1,mod.ga.2,mod.ga.3,k
    =log(length(db$x)))
```

Performances can be as well assessed on the validation set, by means of the `getTGD()` and `TGD()` functions, providing the validation global deviance, that is the global deviance (minus twice the fitted log-likelihood) evaluated using predictive values for the parameters at the validation sample:

```
gg.egpd.0 <- getTGD(mod.egpd.0,newdata=db2)
gg.ga.0 <- getTGD(mod.ga.0,newdata=db2)
...
TGD(gg.egpd.0,gg.ga.0,gg.egpd.1,gg.ga.1,gg.egpd.2,gg.ga.2,gg.egpd.3,gg.ga.3)
```

In terms of GD on the fitting set, results show that the two best-fit among EGPD-based models, almost equal when compared to the remaining models, are those that allow the nonlinear cyclic modelling for μ. $mod.egpd.3$ is slightly ahead of $mod.egpd.2$, which is surprising as the correct generative

Table 1: Effective degree of freedom (df) and information criteria for the models fitted on database *db*: global deviance (GD), Akaike information criterion (AIC), Bayesian information criterion (BIC) and global deviance on validation set (GD-v). The smallest values in each column are indicated in bold font.

Model	df	GD	AIC	BIC	GD-v
mod.egpd.0	3	274413.2	274419.2	274446.7	135617.1
mod.egpd.1	5	272807.8	272817.8	272863.8	134732.1
mod.egpd.2	21.0	**271892.2**	**271934.2**	**272127.3**	**134302.7**
mod.egpd.3	46.3	**271864.4**	271957.1	272383.2	**134302.8**
mod.ga.0	2	293062.7	293066.7	293085.1	144905.3
mod.ga.1	4	289822.1	289830.1	289866.9	143223.9
mod.ga.2	13.9	287444.7	287472.6	287600.7	142050.0
mod.ga.3	17.6	287371.4	287406.7	287568.8	142057.0

model is *mod.egpd.*2 yet additional degrees of freedom may provide overfitting that reduce the global deviance. Results are slightly similar on the validation set, however this time the global deviance of *mod.egpd.*3 is slightly larger than *mod.egpd.*2's.

However, as soon as models are penalized with respect to the number of parameters, *mod.egpd.*2 is correctly ranked first before *mod.egpd.*3, *mod.egpd.*1, *mod.egpd.*0 and the four Gamma-based models by order of decreasing complexity. We observe similar results when we fit models on subsets of *db* up to 7% of *db*'s initial size.

```
edf(mod.egpd.2, "mu")

drop1(mod.egpd.2)

summary(mod.egpd.2)
```

5 Conclusions

This article introduced an add-on to the R-package `GAMLSS`, to model non-stationary fields whose margins are EGPD distributed. A number of R packages target the modelling of heavy-tails distributions. Yet, although some allow distributional regression models, for which distribution parameters can be modeled as additive functions of covariates, none of these packages address the extended-GPD distribution nor another form of unique and closed-form formulation for heavy-tailed distributions that models small, intermediate and large values in a synthetic manner. This add-on implements the EGPD family in a generic way in the `GAMLSS` package, allowing to test any new parametric form, up to 4 parameters so far, satisfying the requirement of the EGPD family. Applications include space-time modelling of rainfall marginal distributions.

References

[1] L Belzile, JL Wadsworth, PJ Northrop, SD Grimshaw, and R Huser. mev: Multivariate extreme value distributions, 2017. *URL https://CRAN. R-project. org/package= mev. R package version*, 1.

[2] Jie Chen, François P Brissette, and Xunchang J Zhang. A multi-site stochastic weather generator for daily precipitation and temperature. *Transactions of the ASABE*, 57(5):1375–1391, 2014.

[3] William S Cleveland and Susan J Devlin. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American statistical association*, 83(403):596–610, 1988.

[4] Timothy J Cole and Pamela J Green. Smoothing reference centile curves: the lms method and penalized likelihood. *Statistics in medicine*, 11(10):1305–1319, 1992.

[5] Stuart Coles. *An Introduction to Statistical Modeling of Extreme Values*. Springer, New York, 2001.

[6] Anthony C Davison and NI Ramesh. Local likelihood smoothing of sample extremes. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(1):191–208, 2000.

[7] Christophe Dutang and Kevin Jaunatre. Cran task view: extreme value analysis. Technical report, Version 2022-02-08, URL https://CRAN. R-project. org/view= ExtremeValue, 2022.

[8] Paul HC Eilers and Brian D Marx. Flexible smoothing with b-splines and penalties. *Statistical science*, 11(2): 89–121, 1996.

[9] Michael Falk, Jürg Hüsler, and Rolf-Dieter Reiss. *Laws of Small Numbers: Extremes and Rare Events*. Springer, Basel, 2010.

[10] Eric Gilleland and Richard W Katz. extremes 2.0: an extreme value analysis package in r. *Journal of Statistical Software*, 72:1–39, 2016.

[11] Eric Gilleland, Mathieu Ribatet, and Alec G Stephenson. A software review for extreme value analysis. *Extremes*, 16(1):103–119, 2013.

[12] Janet E Heffernan, Alec G Stephenson, and Eric Gilleland. Ismev: An introduction to statistical modeling of extreme values. *R package version*, 1:41, 2016.

[13] Denise E Keller, Andreas Marc Fischer, Christoph Frei, Mark A Liniger, Christof Appenzeller, and Reto Knutti. Implementation and validation of a wilks-type multi-site daily precipitation generator over a typical alpine river catchment. *Hydrology and Earth System Sciences*, 19(5):2163–2177, 2015.

[14] Piet K Kenabatho, NR McIntyre, RE Chandler, and HS Wheater. Stochastic simulation of rainfall in the semi-arid limpopo basin, botswana. *International Journal of Climatology*, 32(7):1113–1127, 2012.

[15] Philippe Naveau, Raphaël Huser, Pierre Ribereau, and Alexis Hannart. Modeling jointly low, moderate, and heavy rainfall intensities without a threshold selection. *Water Resources Research*, 52(4):2753–2769, 2016.

[16] Ioannis Papastathopoulos and Jonathan A Tawn. Extended generalised Pareto models for tail estimation. *Journal of Statistical Planning and Inference*, 143(1):131–143, 2013.

[17] Bernhard Pfaff, Alexander McNeil, and Maintainer Bernhard Pfaff. Package 'evir'. *R News*, 2018.

[18] R. A. Rigby and D. M. Stasinopoulos. Generalized additive models for location, scale and shape,(with discussion). *Applied Statistics*, 54:507–554, 2005.

[19] Robert A Rigby and D Mikis Stasinopoulos. Generalized additive models for location, scale and shape. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 54(3):507–554, 2005.

[20] Robert A Rigby and DM Stasinopoulos. A semi-parametric additive model for variance heterogeneity. *Statistics and Computing*, 6(1):57–65, 1996.

[21] Robert A Rigby and Mikis D Stasinopoulos. Mean and dispersion additive models. In *Statistical theory and computational aspects of smoothing*, pages 215–230. Springer, 1996.

[22] Robert A Rigby, Mikis D Stasinopoulos, Gillian Z Heller, and Fernanda De Bastiani. *Distributions for modeling location, scale, and shape: Using GAMLSS in R*. CRC press, 2019.

[23] Mikis D Stasinopoulos, Robert A Rigby, Gillian Z Heller, Vlasios Voudouris, and Fernanda De Bastiani. *Flexible regression and smoothing: using GAMLSS in R*. CRC Press, 2017.

[24] Alec G Stephenson. evd: Extreme value distributions. *R news*, 2(2):31–32, 2002.

[25] R Core Team et al. R: A language and environment for statistical computing. 2013.

[26] Patricia Tencaliec, A-C Favre, Philippe Naveau, Clémentine Prieur, and Gilles Nicolet. Flexible semiparametric generalized pareto modeling of the entire range of rainfall amount. *Environmetrics*, 31(2):e2582, 2020.

[27] DS Wilks. Multisite generalization of a daily stochastic precipitation generation model. *journal of Hydrology*, 210 (1-4):178–191, 1998.

[28] Simon N Wood. *Generalized additive models: an introduction with R*. CRC press, 2017.

[29] Thomas W Yee and Alec G Stephenson. Vector generalized linear and additive extreme value models. *Extremes*, 10(1):1–19, 2007.

[30] Benjamin D Youngman. Evgam: An r package for generalized additive extreme value models. *arXiv preprint arXiv:2003.04067*, 2020.

[31] Keming Yu and Rana A Moyeed. Bayesian quantile regression. *Statistics & Probability Letters*, 54(4):437–447, 2001.

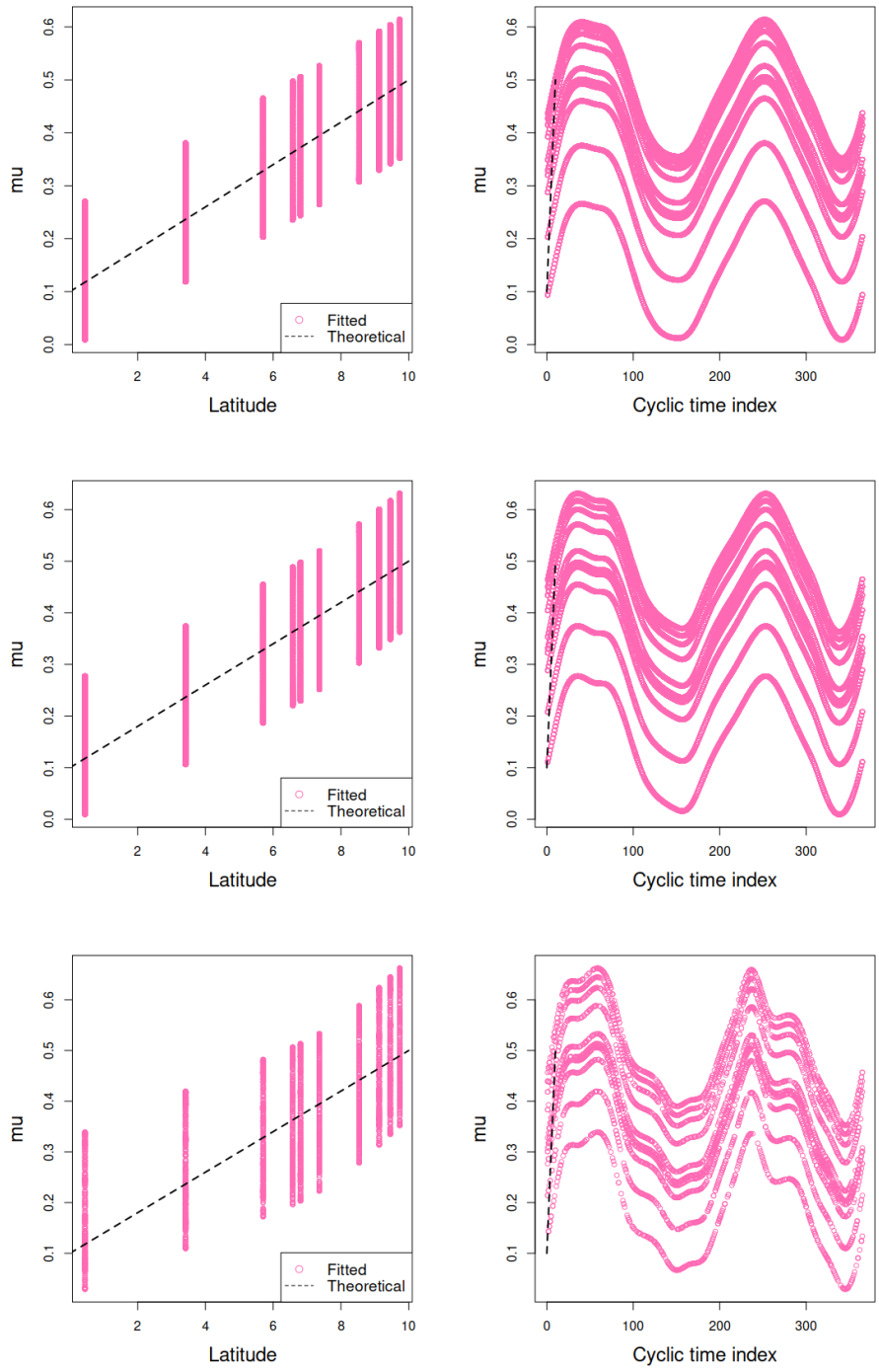

Figure 5: Fitted parameters represented versus the covariates latitude and day of the year respectively, when take the full observation time series db (top), when we remove a quarter of it (middle) and when we only keep 7% of it (bottom).