EURIA
Euro-Institut d'Actuariat

Master 1 d'actuariat - EURIA

Bureau d'Etudes M1

Modélisation d'extrêmes de séries temporelles : une étude empirique

Auteurs :
M. Idris Tioguim
M. Dimitri Delcaillau

Encadrants :
M. Nicolas Raillard
M. Marc Prevosto
M. Pierre Ailliot

Version finale du
7 juin 2018

Remerciements

Nous souhaitons tout d'abord remercier les personnes grâce auxquelles nous avons pu réaliser ce projet ainsi que ce rapport.

Premièrement, nous tenons à remercier M. Prevosto et M. Raillard, nos tuteurs travaillant à IFREMER (l'Institut Français de Recherche pour l'Exploitation de la Mer) pour leur grande disponibilité. Les nombreuses réunions nous ont permis de bien appréhender le sujet et de cibler nos recherches.

En second lieu, nous remercions M. Ailliot, professeur à l'UBO et à l'Euria, qui nous a consacré beaucoup de temps et nous a apporté une précieuse aide tout au long de l'année.

En dernier lieu, nous remercions M. Vermet, qui nous aura donné de nombreux conseils, pour mener à bien notre projet.

Table des matières

Table des figures

Liste des sigles et acronymes

TVE	*Théorie des valeurs extrêmes*
GEV	*Generalized Extreme Value (Distribution)*
GPD	*Generalized Pareto Distribution*
AM	*Annual Maxima*
POT	*Peak Over Threshold*
HS	*Significant (Wave) Height (Hauteur significative)*
WSPD	*Wind Speed (Vitesse du vent)*
ARMA	*Auto Regressive Moving Average (Model)*
MLE	*Maximum Likelihood Estimation*
PWM	*Probability Weighted Moments*
IID	*(Variables aléatoires) indépendantes et identiquement distribuées*
(P)ACF	*(Partial) Autocorrelation function (Fonction d'autocorrélation (partielle))*
ZWSPD	*Jeu de données de vitesse du vent en Angola*
ZHS	*Jeu de données de hauteurs signicatives en Angola*
XWSPD	*Jeu de données de vitesse du vent en Bretagne*
XHS	*Jeu de données de hauteurs signicatives en Bretagne*

Introduction

Les évènements extrêmes sont par définition ceux qui causent le plus de dommages aux personnes, aux structures et aux infrastructures, raison pour laquelle la question de leur prise en compte est cruciale. Divers domaines d'application nécessitent la prise en compte d'évènements rares, par exemple l'étude de risques d'inondations, de risques de dégâts liés aux vents ou encore les risques de dévaluations boursières. Par exemple, la fiabilité d'une structure est assurée en vérifiant sa capacité à endurer une force de vent qui est dépassée en moyenne une fois tous les cent ans. Or, dans de nombreuses applications, les données disponibles sont de trop courte durée pour pouvoir estimer empiriquement avec précision des valeurs aussi peu fréquentes. Pour cette raison, il est généralement fait appel à des méthodes statistiques d'extrapolation afin de calculer des quantiles qui sortent du corps de la distribution de la quantité observée. Le sujet de ce BE se place dans le contexte de l'évaluation de quantiles élevés à partir de séries temporelles de données environnementales de vent, vagues ou de niveaux d'eaux, données qui présentent la particularité d'être autocorrélées et d'être marquées par une forte saisonnalité.

Notre travail est séparé en deux parties distinctes. La première porte essentiellement sur les séries temporelles et a pour objectif principal de trouver un modèle reflétant les données que nous avions à notre dispostion. Pour cela, nous avons utilisé la théorie des modèles ARIMA. Nous avons également dû tenir compte de la saisonnalité des séries dont nous disposions, ainsi que le principe de copules afin d'inclure la dépendance temporelle de nos données dans notre modèle final. La deuxième partie portait sur la théorie des valeurs extrêmes. Notre objectif était, à partir du modèle trouvé au préalable, de comparer les différentes méthodes d'estimations de quantiles élevés, à savoir la méthode du dépassement de seuil (POT), et la méthode des maxima par bloc (AM). Nous avons également voulu comparer les estimateurs couramment utilisés pour mettre en place les méthodes évoquées ci-dessus, à savoir l'estimateur du maximum de vraisemblance (MLE), et l'estimateur des moments de probabilité pondérée (PWM). La comparaison reposait essentiellement sur le calcul du biais et RMSE de chaque estimateur, à partir de réalisations de Monte Carlo du modèle initial. Cette étude avait pour but d'indiquer, à toute personne voulant estimer le niveau de retour à 100 ans de données environnementales, quelle méthode (POT ou AM) et quel estimateur (MLE ou PWM) utiliser en fonction de la taille du jeu de données dont elle dispose et de ses caractéristiques.

Chapitre 1

Théorie classique des valeurs extrêmes : résultats probabilistes

Sommaire

1.1 Introduction à la TVE

La théorie des valeurs extrêmes a pour objectif essentiel la maîtrise des risques. Ses domaines d'application sont nombreux : on peut citer notamment la finance, l'actuariat ou encore l'hydrologie.
En hydrologie, par exemple, l'étude des pluies et débits extrêmes est essentielle afin d'anticiper des catastrophes telles que des inondations et ainsi permettre l'aménagement du territoire. La quantité qui est généralement calculée afin de mesurer le risque est le niveau de retour centennal, qui correspond à la valeur qui est dépassée en moyenne une fois tous les 100 ans. Cette quantité est donc un quantile extrême de la distribution.
En actuariat, depuis la réforme solvabilité II, les entreprises ont une exigence de fonds propres, qui est basée sur le calcul du SCR (Solvency Capital Requirement). Ce dernier représente le capital nécéssaire pour absorber le choc provoqué par un risque majeur tel qu'une baisse brutale du cours d'un actif. Les assureurs et réassureurs sont donc contraints de mesurer leurs risques et d'être solvables dans 99.5 % des cas, ce qui s'appelle une VaR (Value At Risk) d'ordre 0.995. Ceci correspond en réalité au calcul d'un quantile de la

queue de la distribution.
La TVE se place exactement dans ce contexte, car l'un de ses principaux objectifs est l'estimation de quantiles de niveau élevé. La problématique de la TVE est ainsi l'estimation de risques pour lesquels les données disponibles sont trop courtes pour pouvoir être calculées empiriquement. La TVE utilise alors des méthodes statistiques d'extrapolation pour le calcul de quantiles sortant du corps de la distribution.

1.2 Loi GEV(Generalized Extreme Value) et méthode de maxima par blocs

La théorie de la valeur extrême traite le comportement stochastique des valeurs extrêmes dans un processus. Pour un seul processus donné, le comportement des maxima peut être décrit par les trois distributions de valeurs extrêmes - Gumbel, Fréchet et Weibull et dans ce chapitre, nous présenterons les différentes lois de la théories des valeurs extrêmes et leurs propriétés.

1.2.1 Loi GEV(Generalized Extremes Values)

Supposons $(X_1, ..., X_n)$ des variables aléatoires indépendantes et identiquement distribuées (i.i.d) avec fonction de répartition commune $F(x) = P[X_i <= x]$. Soit $M_n = \max(X_1, ..., X_n)$ le maximun des n prémières variables aléatoires de notre échantillon. La fonction de répartion de la variable aléatoire M_n est donnée par la relation suivante :

$$F_{M_n}(x) = P[M_n \leq x] = F(x)^n$$

et donc $F_{M_n}(x)$ converge vers 0 ou 1 selon la valeur de $F(x) \in [0, 1]$. La loi limite de M_n est donc dégénérée et on autorise une renormalisation linéaire de M_n pour contourner le problème. Plus précisément, on s'intéresse au comportement limite de

$$M_n^* = \frac{M_n - b_n}{a_n}$$

avec a_n *et* b_n des suites (déterministes) bien choisies pour éviter la dégénérescence de la loi limite. Il existe une analogie avec le théorème central-limite qui s'intéresse au comportement de $\frac{S_n - b_n}{a_n}$ avec S_n définie par $S_n = X_1 + ... + X_n$, $b_n = n\mathbb{E}[\mathbb{X}_{\not\vdash}]$ *et* $a_n = nVar(X_1)$; sous certaines conditions, la loi limite de $\frac{S_n - b_n}{a_n}$ est une loi $\mathcal{N}(0, 1)$

Le théorème de Fisher-Tippet montre que la seule loi limite de M_n^* est la loi GEV définie ci-dessous.

Définition(Distribution GEV) : $X \sim GEV(\mu, \sigma, k)$ avec $\sigma > 0$ si sa focntion de repartition est donnée par

- Si $k \neq 0$: $F(x; \mu, \sigma, k) = exp\left(-\left[1 + k\frac{x-\mu}{\sigma}\right]^{-1/k}\right)$ définie pour x vérifiant $1 + k\frac{x-\mu}{\sigma} > 0$
- Si $k = 0$: $F(x; \mu, \sigma, k) = exp\left(-exp\left(-\left[\frac{x-\mu}{\sigma}\right]\right)\right)$ définie pour tout $x \in \mathcal{R}$

Remarques :

- μ est un paramètre de position et σ paramètre d'échelle : si $X \sim GEV(\mu, \sigma, k)$ alors $\frac{X-\mu}{\sigma} \sim GEV(0, 1, k$
- k est un paramètre de forme.
  * Le cas $k = 0$ correspond à la loi de Gumbel et on peut montrer que c'est bien la limite des autres cas lorsque $k \to 0$.
  * Le cas $k < 0$ correspond à la loi de Weibull. La formule ci-dessus définit la fonction de répartition pour $x < \mu - \sigma/k = x_+$ et on pose $F(x; \mu, \sigma, k) = 1$ si $x \geq x_+$: la variable X est donc à valeurs dans $]-\infty, x_+]$ (loi à support majoré).
  * Le cas $k > 0$ correspond à la loi de Fréchet. La formule ci-dessus définit la fonction de répartition pour $x > \mu - \sigma/k = x_-$ et on pose $F(x; \mu, \sigma, k) = 0$ si $x \geq x_-$: la variable X est donc à valeurs dans $]x_-, +\infty[$
- la densité $f(.; \mu, \sigma, k)$ de la loi $GEV(\mu, \sigma, k)$ s'obtient en dérivant la fonction de répartition par rapport à x . Si $k \neq 0$, on obtient

$$f(.; \mu, \sigma, k) = \begin{cases} \frac{1}{\sigma}(1 + k\frac{x-\mu}{\sigma})^{-1/k-1} exp(-(1 - k\frac{1-\mu}{\sigma})^{-1/k}) & \text{si } 1 + k(\frac{x-\mu}{\sigma}) > 0 \\ 0 & \text{sinon} \end{cases}$$

- on en déduit que X a un moment d'ordre $k > 0$ ssi $k < 1/k$. En particulier X a une espérance finie ssi $k < 1$ et une variance finie ssi $k < 1/2$.

Théorème(Fisher-Tippet(1928)). S'il existe des séquences $(a_n)_{n\in\mathbb{N}}$ et $(b_{nn\in\mathbb{N}}$ telles que

$$P[\frac{M_n - b_n}{a_n} \leq z] \to G(z)$$

pour tout $z \in \mathbf{R}$ avec G non dégénérée , alors G est la fonction de répartition d'une loi GEV.

1.2.2 Méthode des maxima par blocs

Le théorème de Fisher-Tippet donne la loi approchée du maximum d'un grand nombre d'observations i.i.d.. En pratique, si on dispose de n observations $(x_1, ..., x_n)$, on commence par regrouper les données en k blocs de longueur l et on calcule le maximum sur chaque bloc :

$$m_i = \max(x_{(i-1)l+1}, ..., x_{il}) \;\; pour \; i \in 1, ..., k$$

On approche ensuite la loi de la variable aléatoire M_i par une loi GEV puis on estime les paramètres de cette loi en utilisant l'échantillon $(m_1, ..., m_k)$. Il faut alors trouver un bon compromis entre la taille des blocs l , qui doit être assez grande pour que l'approximation par la loi GEV soit réaliste, et le nombre de blocs k qui doit être assez grand pour avoir assez d'informations pour estimer les trois paramètres de la GEV. Pour les données météorologiques, on considère souvent des blocs de taille 1 an ce qui a pour avantage de gommer les effets saisonniers.

1.3 Loi GPD (Generalized Pareto Distribution) et méthode de dépassement de seuil

1.3.1 Loi de Pareto généralisée : GPD

Ici, on cherche un modèle paramétrique pour décrire la forme de la fonction de répartition F d'une variable aléatoire X au-dessus d'un niveau u élevé. D'après le théorème de Fisher et Tippet, il est naturel de supposer que pour n "grand" :

$$P[\max(X_1, ..., X_n) \leq x] = F^n(x) \approx exp\left(-\left[1 + k\frac{x-\mu}{\sigma}\right]^{-1/k}\right)$$

Et donc :

$$nlog(F(x)) \approx -\left[1 + k\frac{x-\mu}{\sigma}\right]^{-1/k}$$

comme on s'intéresse à la queue de la distribution ($x \geq u$ avec u grand implique que $F(x) \approx 1$) on a

$$log(F(x)) \approx -(1 - F(x))$$

et donc finalement

$$P[X \geq x] = 1 - F(x) \approx \frac{1}{n}\left[1 + k\frac{x-\mu}{\sigma}\right]^{-1/k}$$

En pratique, tronquer au niveau u revient à s'intéresser à la loi conditionnelle

$$P[X \geq u + y \backslash X \geq u]$$

avec $y \geq 0$.
En utilisant l'approximation précédente, on obtient alors :

$$P[X \geq u + y \backslash X \geq u] \approx \left[1 + \frac{ky}{\tilde{\sigma}}\right]^{-1/k}$$

avec $\tilde{\sigma} = \sigma + k(u - \mu)$. Finalement on obtient pour une loi conditionnelle de $X - u$ (les dépassements du niveau u) sachant que $X > u$ (le niveau u est dépassé)

$$P[X - u \leq y \backslash X \geq u] \approx 1 - \left[1 + \frac{ky}{\tilde{\sigma}}\right]^{-1/k}$$

Le terme de droite correspond à la fonction de répartition de la loi de Pareto généralisée (Generalized Pareto Distribution, GPD). Ceci justifie (de manière informelle) de modéliser la loi des dépassements d'un niveau élevé par une loi GPD. Les paramètres $(\tilde{\sigma}, k)$ de la loi GPD s'expriment en fonction du seuil u et des paramètres (μ, σ, k) de la loi GEV pour la loi des maxima. En particulier, le paramètre de forme k, qui caractérise la lourdeur de la queue de la distribution (c.f ci-dessous), est identique pour les deux lois et ne dépend pas du seuil u choisi.

Définition $X \sim GPD(\sigma, k)$ avec $\sigma \geq 0$ si sa fonction de répartition vérifie :
— $P[X \leq x] = G(x; \sigma, k) = 1 - \left[1 + \frac{kx}{\sigma}\right]^{-1/k}$ *si* $k \neq 0$, $x \geq 0$ *et* $1 + \frac{kx}{\sigma} \geq 0$

— $P[X \leq x] = G(x; \sigma, 0) = 1 - exp(-\frac{x}{\sigma})$ *si* $k = 0$ *et* $x \geq 0$

Remarques

— σ est le paramètre d'échelle et k le paramètre de forme.

— la densité $g(.; \sigma, k)$ de la loi $GPD(\sigma, k)$ s'obtient en dérivant la fonction de répartition. Pour $k \neq 0$:

$$g(.; \sigma, k) = \begin{cases} \frac{1}{\sigma}(1 + k\frac{x}{\sigma})^{-1/k-1} & \text{si } x > 0 \text{ et } 1 + k(\frac{x}{\sigma}) > 0 \\ 0 & \text{sinon} \end{cases}$$

— si $k < 0$ alors la fonction de répartition est définie pour $x < -\sigma/k = x_+$ et on pose $F(x; \sigma, k) = 1$ *si* $x \geq x_+$. La variable aléatoire X est donc à valeurs dans $]0, x_+]$ (bornée p.s.).

— le cas $k < 0$ correspond aux queues lourdes : X est alors à valeurs dans $\mathbb{R}^+$.

— Le cas $k = 0$ correspond à la limite des autres cas lorsque $k \to 0$ et on retrouve la loi exponentielle.

— $Y \sim GPD(\sigma, k)$ avec $k < 1$ alors

$$\mathbb{E}[Y] = \frac{\sigma}{1 - k}$$

— $X \sim GPD(\sigma, k)$ alors

$$P[X - u \backslash X \geq u] \sim GPD(\sigma + ku, k)$$

La loi GPD est stable par censure : la loi des excès d'une GPD au-dessus d'un seuil u est donc une loi GPD avec le même paramètre de forme k et un paramètre d'échelle qui évolue linéairement avec u. En particulier, si $k = 0$ on retrouve la propriété d'absence de mémoire de la loi exponentielle.

1.3.2 Méthode de dépassement de seuil (Peak Over Threshold : POT)

Dans la méthode des maxima par blocs, des blocs de taille identique sont constitués puis seulement le maximum de chacun d'eux est utilisé pour ajuster une loi GEV. Ce choix des blocs est généralement arbitraire, et on perd généralement de l'information sur

les événements extrêmes qui par nature sont déjà peu observés. Par exemple plusieurs événements extrêmes intéressants peuvent être mis dans un même bloc alors qu'un autre bloc ne contient pas d'événement "extrême" ! En pratique, cela signifie qu'il faut beauccoup de données pour pouvoir mettre en place la méthode des maxima par blocs (typiquement quelques décennies si on fait des blocs annuels), ce qui n'est généralement pas le cas. Une alternative à la méthode des maxima par blocs consiste à conserver toutes les observations qui dépassent un niveau élevé puis à ajuster une loi appropriée à ces dépassements qui représente les événements "extrêmes". Cette méthode est généralement appelée méthode des dépassements de seuil (ou "Peak Over Threshold", POT).

Chapitre 2

Méthodes statistiques dans la théorie des valeurs extrêmes

Sommaire

Dans cette partie nous étudierons différentes méthodes permettant d'estimer les paramètres d'une distribution de probabilité à partir d'échantillons de cette loi. Ceci nous permettra par la suite d'estimer les différents paramètres des lois GEV et GPD.

2.1 MLE : méthode du maximum de vraisemblance

2.1.1 Principe général

L'estimateur du maximum de vraisemblance, proposée par Ronald Aylmer Fisher en 1912, est l'un des estimateurs les plus utilisés en statistiques. Son principe est expliqué dans ce qui suit. Nous supposons qu'un seul paramètre est à estimer, pour simplifier. Ceci se généralise facilement pour plusieurs paramètres. Considérons une variable aléatoire X réelle, discrète ou continue. Soit $X_1, X_2, ..., X_n$ un échantillon (supposé i.i.d) de taille n $\in \mathbb{N}$ de même loi que X. On note $x_1, x_2, ..., x_n$ une observation de cet échantillon. Notre objectif est d'estimer un paramètre θ inconnu, lié à cette loi. Soit f la fonction définie par :

$$\forall x \in \mathbb{R},\ f(x|\theta) = \begin{cases} f_\theta(x),\ si\ X\ est\ une\ variable\ aléatoire\ continue \\ \mathbb{P}_\theta[X = x],\ si\ X\ est\ une\ variable\ aléatoire\ discrète \end{cases} \quad (EQ1)$$

Dans la formule précédente, f_θ et $\mathbb{P}_\theta$ représentent respectivement la densité de X si X est continue et une probabilité ponctuelle si X est discrète (lorsque θ est le paramètre théorique). Nous définissons alors la vraisemblance associée à l'échantillon $(X_i)_{1\leq i\leq n}$: $L(x_1,...,x_n|\theta) = f(x_1|\theta)\times ...\times f(x_n|\theta)$, soit par indépendance des observations, on obtient : $L(x_1,...,x_n|\theta) = \prod_{i=1}^{n} f(x_i|\theta)$. La log-vraisemblance est très souvent utilisée, du fait du passage d'un produit à une somme, grâce au passage au logarithme. On la note : $l(x_1,...,x_n|\theta) = \sum_{i=1}^{n} ln\, f(x_i|\theta)$. Nous voulons maximiser cette fonction, et l'estimateur du maximum de vraisemblance sera alors le paramètre $\widehat{\theta}$ qui la maximise. Pour simplifier nous supposons L deux fois dérivable. Une condition nécessaire et suffisante pour avoir un maximum en $\widehat{\theta}$ est alors : la dérivée première en ce point s'annule et la dérivée seconde est négative.

Soit, en $\theta = \widehat{\theta}$:

$$\begin{cases} \frac{\partial L(x_1,...x_n|\theta)}{\partial\theta} = 0 \\ \frac{\partial^2 L(x_1,...x_n|\theta)}{\partial\theta^2} < 0 \end{cases}$$

Comme signalé précédemment, ceci se généralise aisément pour plusieurs paramètres à estimer. Pour cela, la condition nécéssaire devient : la dérivée partielle par rapport à chaque paramètre est égale à 0 et comme condition suffisante que la matrice hessienne associée soit définie négative (ce qui assure un maximum global).

Calcul pour la loi GEV

Rappelons l'expression de la fonction caractéristique d'une loi GEV :

$$\forall x\in\mathbb{R},\ F(x;\mu,\sigma,\xi) = \exp\left\{-\left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]_+^{-1/\xi}\right\}$$

Nous avons donc en dérivant, la densité associée :

$\forall x\geq 0\,, tel\, que :\ 1+\kappa\frac{x}{\sigma}\geq 0$:

$$f(x;\mu,\sigma,0) = \begin{cases} \frac{1}{\sigma}\left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{(-1/\kappa)-1}\exp\left\{-\left[1+\kappa\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\kappa}\right\}\ si\,\kappa\neq 0 \\ \frac{1}{\sigma}\exp\left(-\frac{x-\mu}{\sigma}\right)\exp\left[-\exp\left(-\frac{x-\mu}{\sigma}\right)\right]\ si\,\kappa = 0 \end{cases}$$

La fonction de vraisemblance de la loi GEV est :

$$ln(L(x|\theta)) = l(x|\theta) = -n\,ln(\sigma) + \sum_{i=1}^{n}[(\frac{1}{\kappa}-1)ln(y_i) - (y_i)^{\frac{1}{\kappa}}]$$

$avec$: $\theta = (\mu,\sigma,\kappa)\, et\, y_i = [1-\frac{\kappa}{\sigma}(x_i-\mu)]$.

En dérivant successivement par rapport à μ, σ et κ, le maximum de vraisemblance en $\theta = \widehat{\theta} = (\widehat{\mu}, \widehat{\sigma}, \widehat{\kappa})$, on obtient le système de 3 équations suivant :

$$\begin{cases} \frac{1}{\sigma}\sum_{i=1}^{n}\left\{\frac{1-\kappa-y_i^{\frac{1}{\kappa}}}{y_i}\right\} = 0 \\ -\frac{n}{\sigma} + \frac{1}{\sigma}\sum_{i=1}^{n}\left\{\frac{1-\kappa-(y_i)^{\frac{1}{\kappa}}}{y_i}(\frac{x_i-\mu}{\sigma})\right\} = 0 \\ -\frac{1}{\kappa^2}\sum_{i=1}^{n}\left\{ln(y_i)(1-\kappa-(y_i)^{\frac{1}{\kappa}}) + \frac{1-\kappa-(y_i)^{\frac{1}{\kappa}}}{y_i}\kappa(\frac{x_i-\mu}{\sigma})\right\} = 0 \end{cases}$$

La solution de cette équation peut être calculée à l'aide de la méthode de Newton, en vérifiant qu'il s'agit bien d'un maximum global, à l'aide de la condition suffisante vue précédemment.

Notons bien que ce système n'admet pas toujours de solution. C'est pourquoi un paramètre de taux d'erreur sera introduit par la suite, lorsque nous étudierons les performances des estimateurs.

Calcul pour la loi GPD

Nous allons appliquer la méthode du maximum de vraisemblance à la loi $GPD(\sigma, \kappa)$ dont l'objectif est d'estimer ces paramètres à partir d'un échantillon de taille $n \in \mathbb{N}$ supposé i.i.d.

Rappelons d'abord l'expression de la fonction caractéristique d'une loi GPD :

$$\forall x \geq 0, \, tel\, que \, : 1 + \kappa\frac{x}{\sigma} \geq 0, G(x;\sigma,\kappa) = \mathbb{P}[X \leq x] = \begin{cases} 1 - (1+\kappa\frac{x}{\sigma})^{\frac{1}{\kappa}} \, si\, \kappa \neq 0 \\ 1 - exp(-\frac{x}{\sigma}) \, si\, \kappa = 0 \end{cases}$$

Nous avons en dérivant, la densité associée :

$$g(x) = \frac{1}{\sigma}\left(1 + \frac{\kappa x}{\sigma}\right)^{\left(\frac{1}{\kappa}-1\right)} \, pour\, x \geq 0\, si\, \kappa > 0, \, et\, \mu \leq x \leq -\frac{\sigma}{\kappa}\,, si\, \kappa < 0$$

Et pour $\kappa = 0$, $g(x) = \frac{1}{\sigma}exp(-\frac{x}{\sigma})$

On en déduit alors la fonction de log-vraisemblance de la loi GPD :

$$l(x|\theta) = n\left(-log\sigma + (\frac{1}{\kappa-1})\frac{1}{n}\sum log(1 - \frac{\kappa x_i}{\sigma})\right)$$

Pour $\kappa = 0, l(x|\sigma) = nlog(\sigma) - \frac{1}{\sigma}\sum_{i=1}^{n} x_i$

On obtient pour $\kappa = 0$, en dérivant la fonction de vraisemblance, l'estimateur suivant :

$$\widehat{\sigma} = \frac{n}{\sum_{i=1}^{n} x_i}$$

Pour $\kappa \neq 0$, on obtient le système suivant :

$$\begin{cases} -\frac{n}{\sigma} + (\frac{1}{\kappa} - 1) \sum_{i=1}^{n} \frac{\kappa x_i}{\sigma^2 - \kappa x_i \sigma} = 0 \\ -\sum_{i=1}^{n} log(1 - \frac{\kappa x_i}{\sigma}) - \kappa(\kappa - 1) \sum_{i=1}^{n} \frac{x_i}{\sigma - \kappa x_i}) = 0 \end{cases}$$

On peut alors, par des algorithmes numériques, trouver une solution approchée de ce système. Notons qu'une telle solution n'existe pas toujours, comme pour la loi GEV.

2.2 PWM : méthode des moments de probabilité pondérée

2.2.1 Principe général

Greenwood a démontré qu'une fonction de répartition d'une variable aléatoire X est entièrement caractérisée par les moments de probabilité pondérée (c.f référence **??**), définie par :

$$\forall i, j, k \in \mathbb{R},\ M_{i,j,k} = \int_0^1 x^i F(x)^j (1 - F(x))^k) dF(x)$$

La formule générale est donnée par :

$$\beta_r = n^{-1} \sum_{j=1}^{n} \binom{j-1}{r} x_{(j)}$$

Un estimateur sans biais de M_k, appelé Maciunas Landwher, est donné par :

$$\forall k \in \mathbb{N},\ M_k = \frac{1}{n} \sum_{i=1}^{n} \frac{\binom{i-1}{k}}{\binom{n-1}{k}} x_{(i)}$$

avec : $x_{(1)} < ... < x_{(n)}$, correspond aux observations rangées dans l'ordre croissant. On peut montrer que :

$$\forall k \in \mathbb{N},\ \beta_k = n^{-1} \sum_{i=1}^{n} (\frac{i - 0.35}{n})^k \times x_{(j)}$$

2.2.2 Calcul pour la loi GEV

Pour une distribution GEV, on peut relier les moments de probabilité pondérée aux paramètres caractérisant cette loi. Raynal-Villasenor a en effet montré la relation suivante (c.f référence 17) :

$$M_k = \frac{\mu - \sigma/\kappa}{1+k} + \frac{\sigma\Gamma(1+\kappa)}{\kappa(1+k)^{\kappa}}$$

On en déduit les estimateurs $(\widehat{\mu}, \widehat{\sigma}, \widehat{\kappa})$ des paramètres de la loi, en résolvant le système :

$$\begin{cases} \widehat{\sigma} = \frac{(M_0-2M_1)^2\widehat{\kappa}}{\Gamma(1+\widehat{\kappa})(M_0+4M_3-M_1)} \\ \widehat{\mu} = M_0 - \left\{\frac{(M_0-2M_1)^2}{M_0+4M_3-M_1}\right\}\left\{1 - \frac{1}{\Gamma(1+\widehat{\kappa})})\right\} \\ \widehat{\kappa} = ln(\frac{(M_0-2M_1)^2}{M_0+4M_3-M_1})/ln(2) \end{cases}$$

2.2.3 Calcul pour la loi GPD

En reprenant l'expression de β_k vue précédemment, on obtient dans le cas de la loi GPD les valeurs suivantes :

$$\begin{cases} \widehat{\kappa} = \frac{\beta_0}{\beta_0-2\beta_1} - 2 \\ \widehat{\sigma} = \frac{2\beta_0\beta_1}{\beta_0-2\beta_1} \end{cases}$$

2.3 Comment assurer l'indépendance d'échantillons POT et AM ?

Dans cette partie, nous expliquerons les méthodes utilisées afin d'assurer l'indépendance de nos échantillons POT que l'on prélève. Les échantillons utilisés dans les méthodes des maxima annuels et du dépassement de seuil doivent être extraits de la série temporelle initiale de sorte que l'hypothèse d'indépendance soit réaliste. En effet, tous les calculs et les résultats obtenus dans la partie théorique précédente sur les valeurs extrêmes supposent l'indépendance des échantillons sélectionnés.

2.3.1 Méthode AM

Dans le cas de la méthode AM, le fait de prendre le maximum de chaque année assure de manière quasiment automatique l'indépendance. Ceci peut être vérifié par un simple graphique. Il faut néanmoins veiller à ce que le maximum d'une année donnée n'appartienne

pas à une tempête qui chevauche cette année et la suivante ; dans le cas contraire : le maximum de l'année suivante ne doit pas être selectionné dans cette tempête.

2.3.2 Méthode POT : déclustering

Dans le cas de la méthode POT, l'hypothèse d'indépendance est assurée à l'aide de clusters, dans lesquels seules les observations maximales d'une tempête sont gardées. Pour cela, il faut définir la notion de tempête. Comme dans le choix du seuil dans la méthode POT classique, il faut choisir une valeur, notée u à partir de laquelle, on considère qu'une tempête a réellement lieu. Par exemple, pour le cas de données de vitesse de vent, on peut considérer qu'une tempête commence à partir du moment où une vitesse de 80 km/h est dépassée. Une fois cette valeur définie, il faut également considérer une autre valeur, notée u_{min}, pour laquelle lorsqu'au cours d'une tempête on passe en dessous de celle-ci, nous considérons alors que la tempête est terminée. Une dernière valeur est requise, notée d_{clust}, il s'agit d'une distance minimum nécéssaire entre deux tempêtes successives afin d'assurer l'indépendance de ces dernières. D'après des études sur la durée des tempêtes, nous avons considéré que les maxima appartenaient au même cluster (i.e. même tempête) si une distance inférieure à 2-3 jours les séparait.

Afin d'expliquer clairement ce processus de clustering, nous avons réalisé une simulation avec les valeurs suivantes : $u = 4.2$, et $u_{min} = 3.8$, sans considérer de distance minimale d_{clust} pour simplifier. Pour ce faire, nous avons utilisé le jeu de données 'portpirie', disponible sous R, contenant les maxima annuels de niveaux de mer (en m) à Port Pirie, en Afrique du Sud entre 1923 et 1987. Le processus est détaillé sur les figures 2.1 et 2.2.

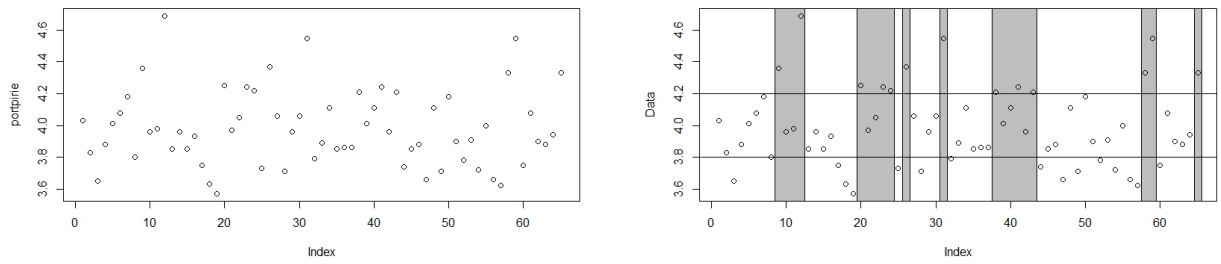

FIGURE 2.1 – Données initiales et mise en place des clusters

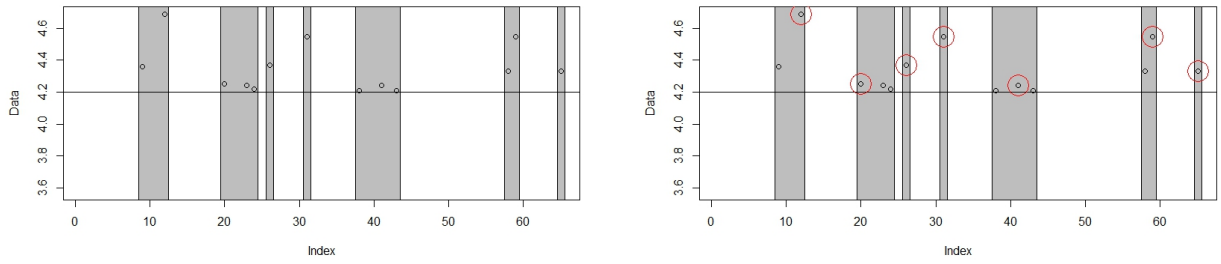

FIGURE 2.2 – Clusters et échantillons finalement retenus

2.4 Application de la théorie des valeurs extrêmes en finance et actuariat

Comme introduit au début de notre projet, l'une des applications classiques de la théorie des valeurs extrêmes est la détermination de la Value At Risk (VaR), qui représente en finance la perte potentielle maximale sur la valeur d'un actif ou d'un portefeuille d'actifs compte tenu d'un horizon de détention et d'un intervalle de confiance. Cette notion n'étant plus limitée en finance, elle est utilisée en assurance notamment pour le calcul du quantile de niveau "petit" :VaR à 75% pour le niveau des provisions en solvabilité 2 et aussi pour le calcul du niveau élevé : contrôler la probabilité de ruine à un an pour qu'elle ne dépasse pas 0.05%. Dans cette partie notre but est juste de présenter une méthode de calcul de la VaR sans toutefois entrer dans les détails en utilisant les notions des valeurs extrêmes vues précédemment.

Méthode de calcul de la VaR en utilisant la TVE

Notre objectif ici est de présenter comment utiliser la théorie de la valeur extrême pour calculer la VaR. Comme déja vu dans le premier chapitre, plutôt que de mettre l'accent sur l'ensemble de la distribution, la théorie de la valeur extrême s'intéresse uniquement à la zone de queue de la distribution et, ici, nous utilisons la GPD (distribution généralisée de Pareto) pour décrire la zone de queue. En se concentrant sur la valeur extrême, cette méthode permet de décrire plus précisément la zone de queue de la distribution.

Soit $X \sim GPD(\sigma, k)$ et notons $G_{\sigma,k}$ sa fonction de répartition. Nous définissons la fonction d'excès moyen au-dessus d'un seuil u par la relation suivante :

$$F_u(y) = P\{X - u \leq y | x > u\} \quad pour \;\; 0 \leq y \leq x_0 - u$$

.

En utilisant le résultat de l'estimation de la distribution d'excès moyen, nous pouvons estimer les queues de la distribution

$$F(x) = (1 - F(u))G_{\sigma,k}(x - u) + F(u)$$

tout en remplaçant $F(u)$ par l'estimateur empirique $\frac{n-N_u}{n}$.
n représente la taille de l'échantillon et N_u le nombre d'observations au-dessus du seuil u choisi.

Ainsi, la fonction de distribution cumulative pour la zone de queue de la distribution $(x > u)$ est :

$$\hat{F}(x) = 1 - \frac{1}{N_u}\left(1 + \hat{k}\frac{x-u}{\hat{\sigma}}\right)^{-1/\hat{k}}$$

où les valeurs $\hat{k}$ et $\hat{\sigma}$ sont calculées en utilisant les formules obtenues dans les paragraphes précédents.

Nous pouvons à présent pour une probabilité $q > F(u)$ donnée estimer la VaR en utilisant la formule suivante :

$$\hat{VaR}^q_{extremeGPD} = u + \frac{\hat{\sigma}}{\hat{k}}\left[\left(\frac{n}{N_u}(1-q)\right)^{-k} - 1\right]$$

.

Lorsque $X \sim GEV(\mu, \sigma, k)$ nous pouvons estimer la Var dans le cas d'une approche paramétrique en utilisant la formule suivante :

$$VaR^p_{extremeGEV} = \hat{\mu} + \frac{\hat{\sigma}}{\hat{k}}\left[\left(-ln(p)^{-k}\right) - 1\right]$$

où les valeurs $\hat{\mu}, \hat{\sigma} \;\; et \;\; \hat{k}$ sont des estimateurs obtenus soit par MLE ou PWM.

Chapitre 3

Comparaison des méthodes POT et AM et des différents estimateurs dans le cas iid

Sommaire

L'objectif à présent est de comparer nos différentes méthodes d'estimation de quantiles extrêmes (AM et POT), ainsi que les différents estimateurs mis à notre disposition (MLE et PWM). Toute la théorie des valeurs des extrêmes présentée auparavant s'appuie sur des données i.i.d. Ce cas idéal n'est jamais parfaitement vérifié dans la réalité et d'autant moins pour des données environnementales, car elles présentent une forte saisonnalité et sont très corrélées. Notre première étude néanmoins s'appuiera sur une hypothèse d'indépendance des données. Cela nous permettra de mieux appréhender le cas plus complexe de données présentant une dépendance temporelle. Ensuite, dans une seconde partie, nous tenterons de modéliser nos données réelles, et de trouver des moyens de se rendre à l'étude de données indépendantes.

3.1 Simulations et résultats des différentes études réalisées auparavant

Les propriétés des estimateurs PWM et MLE des lois GEV et GPD dans le cas de données indépendantes ont déjà été étudiées par de nombreux chercheurs. On peut citer notamment les études de Hoosking en 1985 et les simulations de Hoosking et Wallis en 1987 (c.f référence 12 et 13). Ces derniers ont considéré différentes valeurs du paramètre de forme kappa entre -0.4 et 0.4. Les résultats obtenus sont les suivants :

- pour la loi GEV, l'estimateur PWM est globalement plus performant que l'estimateur du maximum de vraisemblance lorsque la taille de l'échantillon est inférieure à 50. Dans le cas où l'échantillon dépasse une taille de 50, alors les deux estimateurs ont des performances comparables.
- pour la loi GPD, dans le cas de taille d'échantillon inférieure à 100, les performances du PWM sont meilleures, en terme de RMSE, que celles du MLE pour un paramètre de forme κ positif ou nul ; tandis que pour toute valeur de κ, le biais du MLE est moins important, c'est à dire que le MLE est plus performant en terme de biais. Dans le cas de taille d'échantillon supérieure à 100, le comportement des deux méthodes est similaire.

Notons également que pour les deux lois (GEV et GPD), lorsque la taille d'échantillon est trop faible (inférieure à 50), l'estimateur du maximum de vraisemblance n'existe pas toujours, surtout lorsque le paramètre de forme est négatif.

D'autres études ont été réalisées : Cunnane en 1973, Yevjvick et Taesombut en 1978 et Tavares da Silva en 1983 (c.f références 14 et 15). Ils ont étudié les performances de l'estimateur du MLE des distributions de Gumbel et exponentielle (lorsque $\kappa = 0$ pour une GPD ou une GEV), lorsque le nombre de clusters par année, λ, est supérieur à 1. Ils sont arrivés à la conclusion que pour λ supérieur à une certaine valeur fixée (notée α comprise entre 1.5 et 2), la variance dans l'estimation de la loi exponentielle est inférieure à celle de la loi de Gumbel. Les études montrent également, lorsque $\lambda = 1$, c'est-à-dire lorsque il n'y a qu'une tempête en moyenne par année et donc que la taille de l'échantillon GPD est la même que celui de la GEV, l'inverse se produit (la variance est inférieure pour la loi de Gumbel)

3.2 Présentation de la méthode de comparaison

Comme évoqué ci-dessus, nous nous intéressons ici à des données parfaitement i.i.d. Notre méthode d'analyse des performances est la suivante : calcul du biais et RMSE à l'aide de simulations de Monte Carlo. Pour cela nous devons définir clairement les paramètres que l'on cherche à estimer et comment nous réalisons nos simulations de Monte Carlo.

Tout d'abord, nous partons d'une loi GEV et d'une loi GPD avec des paramètres choisis au prélable : $GEV(\mu, \sigma, \kappa)$ et $GPD(\widetilde{\sigma}, \kappa)$. Nous avons, dans un premier temps, décidé de choisir $\mu = 0$, $\sigma = 1, \widetilde{\sigma} = 1$ et différentes valeurs de Kappa, à savoir $\kappa \in \{-0.3; -0.2; -0.1; 0; 0.1\}$
Pour étudier les différentes performances suivant le nombre d'années d'observations dont on dispose, nous avons réalisé des simulations en faisant varier le nombre de données n_y simulées. Nous avons choisi les valeurs suivantes : $n_y \in \{10; 20; 50; 100\}$
Plus précisément, pour chaque choix de n_y, nous avons un échantillon de taille n_y pour la loi GEV et de taille $\lambda.n_y$ pour la loi GPD. Le paramètre λ précédent correspond au nombre de tempêtes observées en moyenne par an et donc au nombre de valeurs au-dessus du seuil choisi avec la méthode POT : nous avons fixé : $\lambda = 5$. Ceci est cohérent avec le fait qu'avec la méthode POT/GPD, nous possédons en général plus de données qu'avec la méthode AM/GEV. En effet, la méthode du dépassement de seuil consiste, comme son nom l'indique, au choix d'un seuil et donc à accepter plus de valeurs par an que la méthode des maxima annuels qui elle ne retient qu'une donnée par an.

Comme nous appliquons la méthode de Monte Carlo, nous allons réaliser ces différentes simulations un nombre $nbMC$ de fois. L'idéal est de choisir un nombre le plus grand possible. Etant donné les contraintes liées aux ordinateurs que nous utilisons, nous avons retenu $nbMC = 10000$ simulations de Monte Carlo. A présent, pour ces 10000 simulations de Monte Carlo, nous calculons, les paramètres suivants, estimés via les estimateurs MLE et PWM : κ, σ, $\widetilde{\sigma}$, et le niveau de retour à 100 ans, noté q_{100}. Comme nous avons fait les choix de ces paramètres au préalable, nous connaissons leurs valeurs théoriques. Ainsi, pour chaque simulation nous calculons le biais (relatif) et le RMSE (relatif), puis nous en faisons une moyenne sur ces 10000 simulations (Monte Carlo). A la fin nous obtenons donc un tableau contenant pour chaque choix de n_y un biais, un RMSE relatif à la loi GEV, GPD et aux estimateurs MLE et PWM. Notre objectif ensuite est d'interpréter ce tableau et de déduire ainsi quelle méthode et quel estimateur conviennent le mieux suivant le nombre d'années d'observations dont nous disposons.

3.3 Interprétation des résultats dans le cas i.i.d

Notre objectif dans cette partie est d'analyser les résultats que nous avons obtenu à l'aide de la méthode décrite précédemment. Il est difficile d'interpréter directement les valeurs inscrites dans les trois tableaux ci-dessous, c'est pourquoi nous avons complété ce tableau avec une approche « graphique ». Dans les tableaux 3.3, 3.2 et 3.1, nous avons inscrit les valeurs des biais relatifs et RMSE relatifs (en pourcentage) obtenus par la méthode de Monte Carlo pour l'estimation des paramètres sigma et kappa ainsi que du niveau de retour à 100 ans. Sur les figures 3.1 et 3.2, nous avons mis le tableau des biais et RMSE des niveaux de retours à 100 ans, sous forme de courbes afin d'en faciliter la compréhension.

Tout d'abord, signalons que l'estimateur MLE n'est pas toujours obtenu, car la fonction

de vraisemblance n'admet pas toujours de maximum. Plus précisément, elle a une probabilité différente de 1 d'en posséder un. Néanmoins, plus la taille de l'échantillon est grand, plus cette probabilité est proche de 1. C'est pour cela que plus n_y est grand, plus le taux d'erreur est faible, quelque soit la méthode choisie. Concernant le PWM, il est toujours obtenu. En effet, son calcul repose sur une formule exacte et ne produit donc pas d'erreur contrairement au MLE.

Analyse du taux d'erreur (du MLE) :
- Nous pouvons noter tout d'abord que le taux d'erreur est plus faible pour la loi GPD que la loi GEV, et ceci quelque soit la taille de l'échantillon. En particulier, pour la GPD, le taux d'erreur est nul dès que n_y est plus grand que 20. Ceci est cohérent avec le fait que nous avons une taille d'échantillons plus grande avec la méthode de dépassement de seuil.
- Nous observons également que plus kappa est petit, plus le taux d'erreur est grand. Ainsi en terme de taux d'erreur, la GPD est plus performante que la GEV pour l'estimateur MLE.

Analyse du niveau de retour :

En terme de *RMSE* :
- la méthode POT est toujours plus performante que la méthode AM, quelque soit le paramètre de forme κ et le nombre d'années n_y.
- concernant les estimateurs MLE et PWM, les deux semblent donner des résultats similaires, bien que le PWM soit légèrement plus performant.

Concernant le *biais* :
- Pour un nombre d'années inférieur à 50 ans (ici 10 ou 20 ans), la méthode POT donne de meilleurs résultats que la méthode AM (les courbes bleues et vertes sont bien en dessous des courbes rouges et noires pour $n_y < 50$) . Ce résultat est d'autant plus frappant pour des valeurs de kappa supérieure à -0.2 pour lesquelles les performances du AM/MLE explosent. De plus, l'estimateur PWM semble donner également des résultats meilleurs que le MLE.

- A partir de 50 ans de données, l'écart de performances entre les quatre méthodes est bien plus faible. Pour $n_y = 50$, la méthode POT/PWM est toujours la plus performante sauf pour $\kappa = 0$ où elle est la moins efficace. Pour $n_y = 100$, les méthodes sont toutes à peu près équivalentes, bien que l'estimateur MLE semble devenir légèrement meilleur.

Conclusion de l'étude :
Pour un jeu de données de taille faible (inférieure à 50 ans), la méthode POT est la plus adaptée afin de calculer le niveau de retour à 100 ans. Ceci est en adéquation avec le fait que les échantillons POT sont plus grands que les échantillons AM, qui ne gardent qu'une valeur par an. L'estimateur statistique associé à la méthode POT qui donne la meilleure performance semble PWM, bien que l'écart avec le MLE ne soit pas très important.

Dès lors que le nombre d'années devient assez conséquent, les performances des estimateurs sont assez similaires et le choix de la méthode dépendra alors du paramètre

kappa.

Notons que la taille des données que l'on a à notre disposition dépasse rarement 50 ans, ce qui nous conduit à conseiller d'utiliser en priorité la méthode POT, ainsi que l'estimateur PWM.

méthode AM/GEV							méthode POT/GPD						
quantile		2.49	3.01	3.69	4.6	5.84	quantile		2.82	3.56	4.63	6.21	8.62
kappa		-0.3	-0.2	-0.1	0	0.1	kappa		-0.3	-0.2	-0.1	0	0.1
n_y	est	Biais					n_y	est	Biais				
10	mle	*	*	*	*	*	10	mle	-2.1	-3.8	-2.83	-2.31	-2.62
20	mle	6.27	5.04	4.94	140.44	5.57	20	mle	-2	-1.8	-1.48	-2.97	-3.17
50	mle	-0.29	0.44	-0.81	-0.25	-0.24	50	mle	-1.76	-1.2	-1.66	-0.31	-1.18
100	mle	-0.47	-0.03	-0.7	0.05	-0.66	100	mle	-0.36	-0.75	-0.33	-0.01	-0.83
10	pwm	4.2	7.96	5.68	5.37	5.35	10	pwm	2.95	1.59	2.34	3	2.13
20	pwm	1.45	2.23	2.2	1.12	3.44	20	pwm	0.68	1.19	2.18	0.65	0.4
50	pwm	0.77	1.15	-0.32	0.5	0.57	50	pwm	-0.12	0.39	-0.35	1.37	1.22
100	pwm	0.05	0.85	0.15	0.87	0.32	100	pwm	0.57	-0.22	0.67	0.87	-0.25
n_y	est	RMSE					n_y	est	RMSE				
10	mle	*	*	*	*	*	10	mle	1.27	1.21	1.28	1.26	1.24
20	mle	2.03	1.79	1.91	135.73	1.78	20	mle	0.8	0.82	0.82	0.77	0.79
50	mle	0.79	0.88	0.83	0.79	0.8	50	mle	0.49	0.48	0.46	0.5	0.48
100	mle	0.54	0.54	0.54	0.53	0.51	100	mle	0.35	0.33	0.33	0.34	0.33
10	pwm	1.8	1.77	1.82	1.8	1.78	10	pwm	1.23	1.16	1.23	1.23	1.19
20	pwm	1.16	1.18	1.16	1.16	1.26	20	pwm	0.86	0.88	0.86	0.82	0.83
50	pwm	0.72	0.79	0.72	0.7	0.73	50	pwm	0.53	0.54	0.51	0.55	0.53
100	pwm	0.51	0.52	0.53	0.53	0.51	100	pwm	0.39	0.37	0.37	0.37	0.37

TABLE 3.1 – Matrice des biais et RMSE relatifs des estimateurs MLE et PWM pour le niveau de retour à 100 ans pour 100 000 simulations, dans le cas i.i.d. *les valeurs sont supérieures à 1000

méthode AM/GEV							méthode POT/GPD						
kappa		-0.3	-0.2	-0.1	0	0.1	kappa		-0.3	-0.2	-0.1	0	0.1
n_y	est	Biais					n_y	est	Biais				
10	mle	0.35	5.4	0.74	4.34	2.55	10	mle	-5.96	-6.19	-5.99	-5.9	-6.02
20	mle	-1.55	-1.95	-2.14	-2.21	-1.54	20	mle	-3.1	-2.72	-3.08	-3.32	-3.53
50	mle	-0.86	-1.03	-1.26	-1.13	-0.99	50	mle	-1.55	-1.49	-1.35	-0.91	-1.02
100	mle	-0.42	-0.26	-0.46	-0.36	-0.56	100	mle	-0.58	-0.61	-0.47	-0.36	-0.65
10	pwm	-5.99	-3.79	-4.75	-4.46	-4.13	10	pwm	-2.94	-2.96	-2.72	-2.8	-3.24
20	pwm	-2.27	-1.97	-2.07	-2.4	-1.61	20	pwm	-2.03	-1.51	-1.44	-1.8	-1.9
50	pwm	-0.73	-1.14	-1.43	-1.17	-1.06	50	pwm	-0.88	-0.91	-0.87	-0.25	-0.6
100	pwm	-0.5	-0.12	-0.4	-0.34	-0.45	100	pwm	-0.23	-0.45	-0.08	-0.02	-0.46
n_y	est	RMSE					n_y	est	RMSE				
10	mle	1.68	2.13	1.69	2.14	1.5	10	mle	0.58	0.56	0.59	0.57	0.59
20	mle	0.76	0.75	0.73	0.8	0.71	20	mle	0.36	0.36	0.37	0.37	0.37
50	mle	0.38	0.39	0.38	0.37	0.38	50	mle	0.23	0.21	0.21	0.22	0.21
100	mle	0.25	0.24	0.24	0.24	0.23	100	mle	0.15	0.14	0.14	0.15	0.15
10	pwm	0.93	0.92	0.94	0.91	0.95	10	pwm	0.54	0.52	0.53	0.53	0.55
20	pwm	0.59	0.59	0.58	0.59	0.58	20	pwm	0.38	0.38	0.38	0.39	0.38
50	pwm	0.35	0.37	0.36	0.34	0.35	50	pwm	0.24	0.24	0.23	0.24	0.24
100	pwm	0.24	0.24	0.24	0.24	0.24	100	pwm	0.17	0.16	0.16	0.16	0.17

TABLE 3.2 – Matrice des biais et RMSE relatifs des estimateurs MLE et PWM pour le paramètre de forme κ pour 100 000 simulations, dans le cas i.i.d.

méthode AM/GEV							méthode POT/GPD						
kappa		-0.3	-0.2	-0.1	0	0.1	kappa		-0.3	-0.2	-0.1	0	0.1
n_y	est	Biais					n_y	est	Biais				
10	mle	-10.11	-10.76	-10.51	-11.33	-12.49	10	mle	6.67	6.71	5.6	6.63	6.73
20	mle	-5.86	-5.6	-4.73	-5.48	-5.49	20	mle	3.57	2.4	3.83	3.4	3.61
50	mle	-2.31	-1.92	-2.26	-1.71	-2.16	50	mle	1.49	1.96	1.17	0.97	0.53
100	mle	-1.27	-1.2	-1.46	-0.69	-1.04	100	mle	0.67	0.47	0.46	0.4	0.51
10	pwm	-0.49	-1.03	-1.55	-2.1	-2.89	10	pwm	3.21	3.07	1.93	3.15	3.56
20	pwm	-1.31	-1.48	-0.58	-1.27	-1.38	20	pwm	2.39	1.05	2.1	1.79	1.88
50	pwm	-0.63	-0.04	-0.4	0.12	-0.26	50	pwm	0.81	1.36	0.68	0.28	0.1
100	pwm	-0.27	-0.43	-0.55	0.22	-0.18	100	pwm	0.31	0.31	0.06	0.06	0.31
n_y	est	RMSE					n_y	est	RMSE				
10	mle	1.01	1.05	1.03	1.02	1.05	10	mle	0.77	0.78	0.75	0.74	0.79
20	mle	0.65	0.61	0.66	0.62	0.63	20	mle	0.49	0.48	0.49	0.51	0.5
50	mle	0.38	0.38	0.38	0.38	0.39	50	mle	0.31	0.29	0.29	0.3	0.3
100	mle	0.27	0.26	0.26	0.25	0.26	100	mle	0.2	0.2	0.2	0.2	0.21
10	pwm	0.94	0.95	0.96	0.94	0.96	10	pwm	0.71	0.71	0.68	0.69	0.74
20	pwm	0.64	0.6	0.67	0.61	0.63	20	pwm	0.49	0.47	0.49	0.52	0.5
50	pwm	0.4	0.39	0.39	0.4	0.4	50	pwm	0.32	0.3	0.31	0.31	0.32
100	pwm	0.28	0.27	0.27	0.27	0.27	100	pwm	0.21	0.22	0.21	0.21	0.22

TABLE 3.3 – Matrice des biais et RMSE relatifs des estimateurs MLE et PWM pour le paramètre de forme σ pour 100 000 simulations, dans le cas i.i.d.

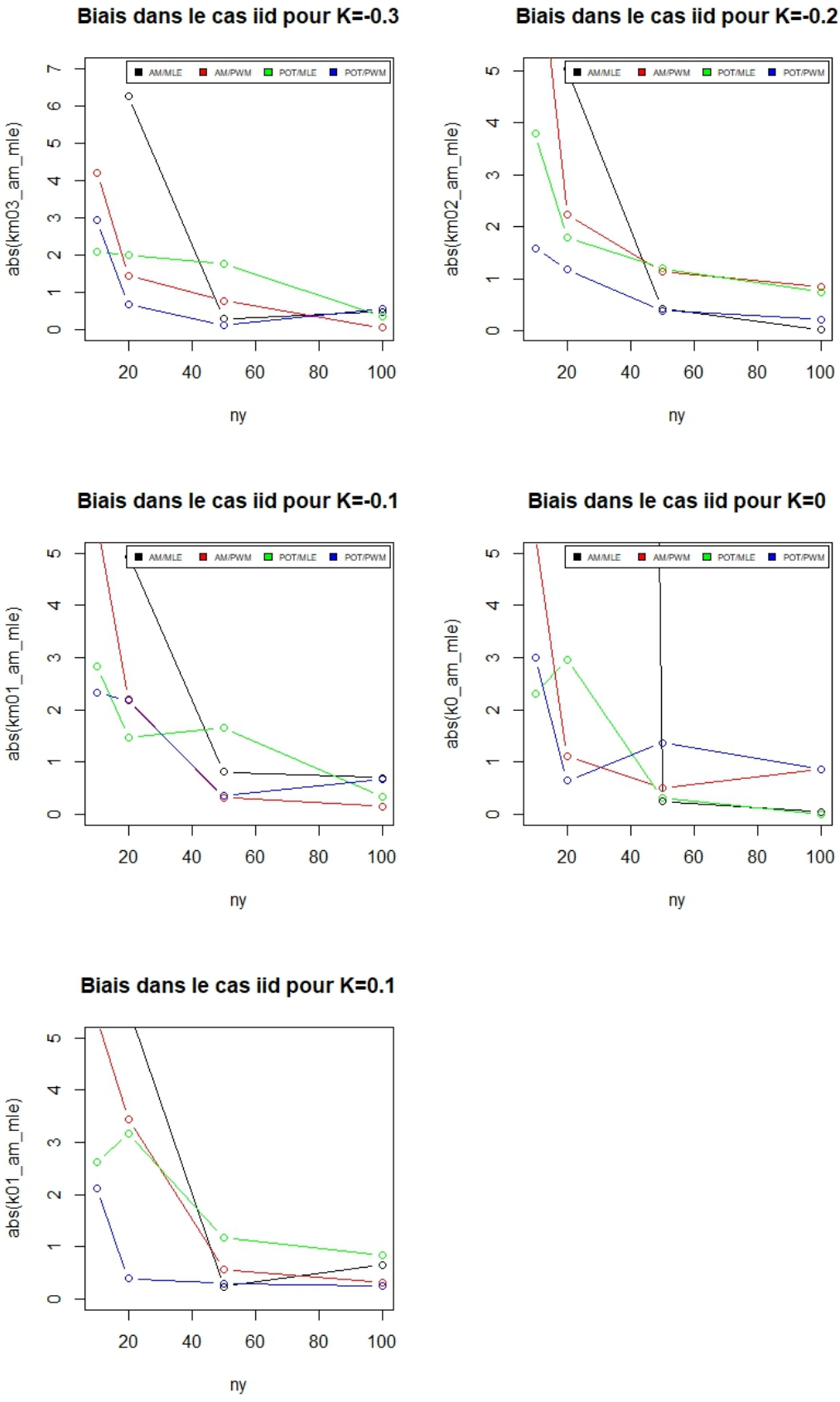

FIGURE 3.1 – Graphiques des biais (en valeur absolue) du niveau de retour à 100 dans l'analyse du cas iid

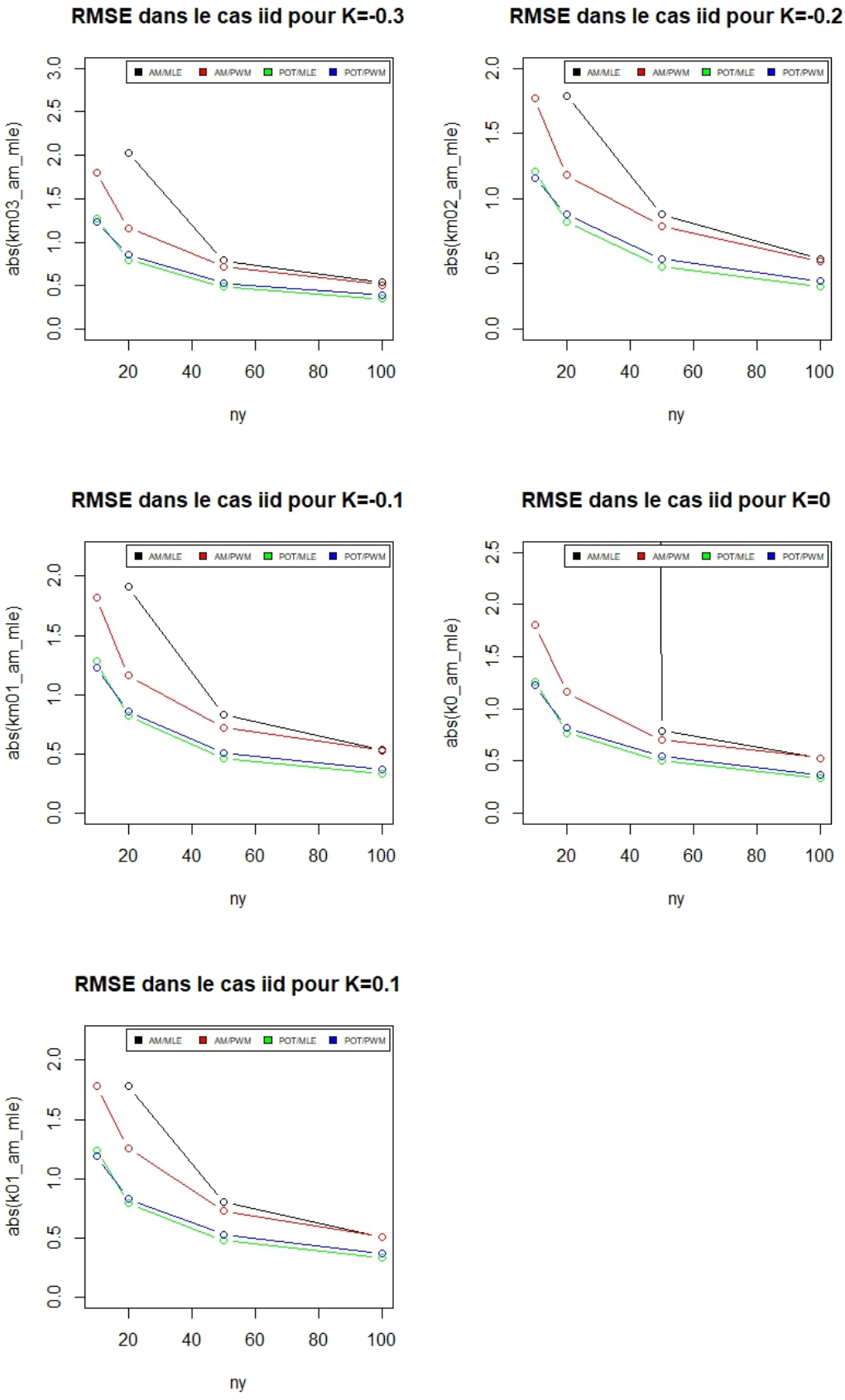

FIGURE 3.2 – Graphiques des RMSE du niveau de retour à 100 dans l'analyse du cas iid

Chapitre 4

Comparaison des méthodes POT et AM dans le cas de données dépendantes

Sommaire

Contrairement à la partie précedente, nous traitons le cas de données dépendantes entre elles. Nous avons étudié plus particulièrement les données réelles dont nous disposons, à savoir la vitesse du vent et la hauteur significative, en Angola et en Bretagne. Dans la partie précédente, du fait de l'indépendance des données, nous avions juste à simuler directement des échantillons de loi GEV et GPD. Il s'agissait d'échantillons que l'on aurait obtenus en appliquant la méthode POT et AM, c'est-à-dire que l'on disposait de données correspondant à des maxima annuels de nos données initiales pour la méthode AM, et des données au-dessus d'un certain seuil pour la méthode POT. Dans ce cas présent, nous ne pouvons plus procéder de cette manière, étant donnée la dépendance temporelle de notre jeu de données. Notre méthode est donc la suivante : trouver un modèle correspond à nos données intiales. A partir de ce modèle, nous pouvons générer des simulations sur un nombre d'années suffisamment grand afin d'en obtenir les valeurs théoriques de κ, q_{100}, σ ,$\widehat{\sigma}$. Nous pouvons également générer ce modèle sur un nombre d'années n_y, bien inférieur, et d'appliquer les méthodes usuelles POT et AM afin d'en estimer les paramètres. Comme précédemment, une fois ces paramètres estimés pour simulation de Monte Carlo, nous en déduisons le biais et le RMSE. Dans ce qui suit, nous détaillerons la méthode afin de trouver un modèle correspondant à nos données réelles. Au préalable, nous devons rappeler

les notions théoriques sur les séries temporelles (ou chronologiques).

4.1 Notions théoriques sur les séries temporelles

Considérons une suite de variables aléatoires $(X_t)_{t\in\mathbb{Z}}$ Il est courant de décomposer en trois termes une série temporelle $(x_t)_{t\in\mathbb{N}}$:

— tendance, notée m_t
— saisonnalité, notée s_t
— reste (terme aléatoire), ,noté u_t

Transformation préalable

Avant même d'étudier ces 3 termes d'une série temporelle, il est courant d'effectuer une transformation préalable, afin de stabiliser la variance. Une transformation courante est l'application de la fonction logarithme. Pour définir ces notions nous utiliserons l'exemple du traffic aérien mensuel international entre 1949 et 1960.

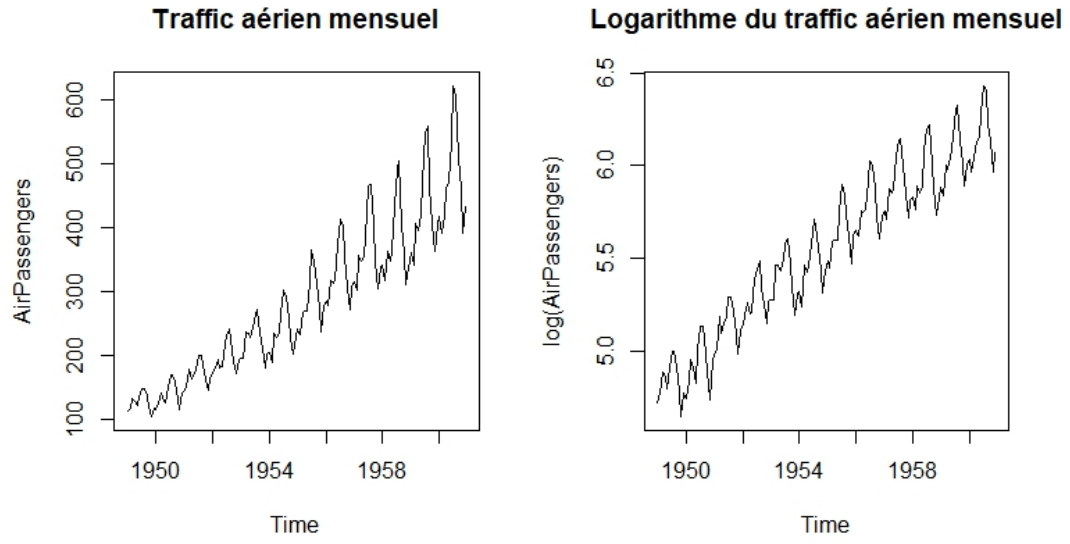

FIGURE 4.1 – Effet du logarithme sur la série temporelle

Nous remarquons, sur la figure 4.1, que la transformation a bien l'effet souhaité : la variance semble désormais à peu près constante.

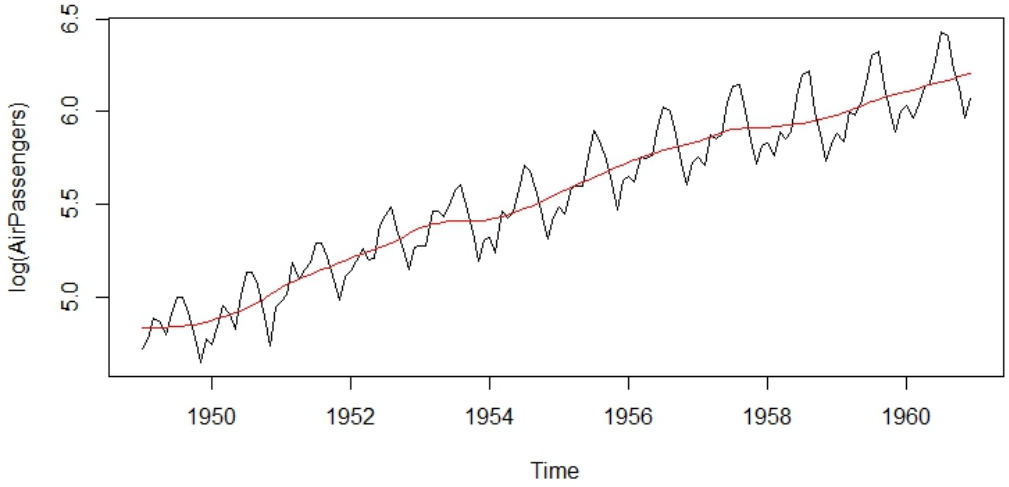

FIGURE 4.2 – Tendance d'une série temporelle

Tendance d'une série temporelle

Définissons à présent la tendance. Il s'agit en général de la première chose à détecter lorsque l'on analyse une série temporelle. La tendance peut être définie comme l'orientation générale d'une série chronologique, à la hausse ou à la baisse sur une période de temps assez longue. Si aucune orientation n'est réellement constatée dans les observations, on dit qu'il n'y a pas de tendance. Sans la définir de manière mathématique précise, nous pouvons remarquer graphiquement les tendances, notamment sur l'exemple du traffic aérien. (figure 4.2)

Nous voyons clairement sur la figure 4.2, une tendance qui se dégage au cours du temps, et celle-ci est représentée en rouge.

A présent, il nous faut savoir comment calculer une tendance lorsque l'on dispose d'une série temporelle. Une première méthode utilisée, est le choix d'une famille de fonctions correspondant à la tendance, comme un polynôme, une droite ou la fonction exponentielle. Ainsi, par les mêmes outils qu'en régression linéaire avec notamment la méthode des moindres carrés, il est possible de déterminer la tendance de la forme souhaitée qui ajuste le mieux notre série temporelle.

Une autre méthode couramment utilisée s'apelle la moyenne mobile (MM). La technique consiste à remplacer une valeur de la série initiale par une moyenne arithmétique calculée entre cette observation et celles qui se situent autour d'elle. En d'autres termes, la valeur estimée de la tendance à un instant dépend de quelques valeurs aux instants précédents et aux instants suivants. Il reste alors à choisir le nombre de valeurs à considérer, avant et

après chaque observation. Pour le calcul de tendance, les moyennes mobiles doivent être centrées, c'est-à-dire que le calcul est réalisé sur le même nombre de valeurs avant et après une observation. Par exemple, si on choisit de dire que la tendance à un instant t dépend des deux valeurs avant (en t-1 et t-2) et des deux valeurs après (en t+1 et t+2), alors la moyenne mobile sera la moyenne arithmétique des valeurs du processus X observé, en t-2,t-1, t, t+1, t+2 : $MM_t = \frac{X_{t-2}+X_{t-1}+X_t+X_{t+1}+X_{t+2}}{5}$ La largeur de la fenêtre doit être choisie en fonction de notre objectif. Ici, nous souhaitons filtrer la saisonnalité, il est alors courant de choisir une taille de fenêtre égale à la périodicité.

Discutons désormais de la notion de saisonnalité. Celle-ci caractérise l'évolution périodique de la série intiale. La tendance et la saisonnalité peuvent être liées, ce qui rend leur extraction difficile. On suppose par la suite que cette extraction est possible, et que l'on peut écrire notre série temporelle initiale $(x_t)_{tin\mathbb{N}}$ sous la forme : $x_t = m_t + s_t + u_t$

Expliquons comment il est possible de l'estimer. Une méthode consiste à filtrer la composante aléatoire encore présente, en partant de la série détendancée. Pour cela, on considère la série détendancée, à savoir : $D_t = x_t - m_t$. Nous appliquons un filtre MM soigneusement choisi, afin de lisser et d'éliminer la composante aléatoire, sans toutefois perdre trop d'informations. A partir de ceci, nous pouvons calculer une estimation de la saisonnalité. Nous détaillons le processus sur l'exemple précédent du traffic aérien. L'objectif dans cet exemple est d'estimer la saisonnalité mensuelle, c'est-à-dire que nous devons estimer une valeur saisonnière pour chaque mois de l'année. Pour cela, nous avons calculé la moyenne associée à chaque mois de la série détendancée filtrée. D_t correspond à la série désaisonnalisée, et $FD = (FD)_t$ à la série désaisonnalisée filtrée. Notons, $FD^{(i)}$, la série extraite de FD pour laquelle nous avons conservé uniquement les données relatives au mois i, $1 \leq i \leq 12$. Nous pouvons estimer la saisonnalité s_i d'un mois i, par une simple moyenne pondérée de FD_i : $s_i = \frac{1}{n_{annees}} \sum_{k=1}^{n_{annees}} FD_i(k)$, où n_{annees}, correspond au nombre d'années disponibles, soit 12 ans pour les données de traffic aérien.

Finalement nous obtenons la décomposition souhaitée, avec notre série temporelle en 3 termes : tendance, saisonnalité et reste. Ce que nous pouvons voir sur la figure 4.3.

Nous allons à présent définir des notions essentielles à l'étude des séries temporelles : $(X_t)_{t\in\mathbb{Z}}$ est un **processus stationnaire** au sens large si et seulement si :

- $\mathbb{E}(X_t) = \mu \quad \forall t \in \mathbb{Z}$
- X_t est de carrée intégrable pour tout $t \in \mathbb{Z} \quad : \mathbb{E}(X_t) < \infty$
- $Cov(X_s, X_t) = Cov(X_{s-1}, X_{s-1+t}) = ... = Cov(X_0, X_t) \quad \forall t, s \in \mathbb{Z}$

Soit $(X_t)_{t\in\mathbb{Z}}$ un processus stationnaire, la **fonction d'auto-covariance** est la fonction γ définie par

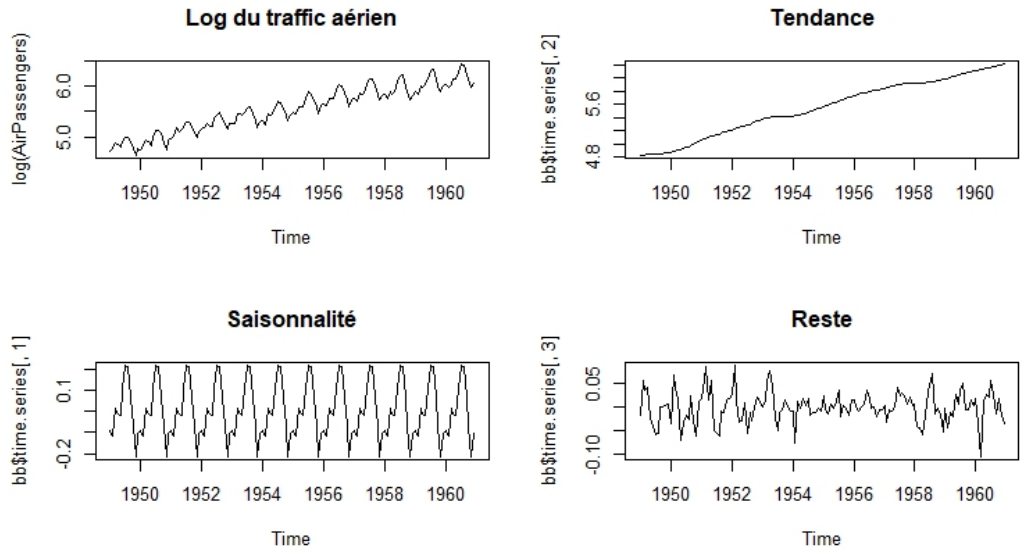

FIGURE 4.3 – Décomposition d'une série temporelle : tendance, saisonnalité, reste

$$\forall t, h \in \mathbb{Z} \quad \gamma(h) = Cov(X_t, X_{t+h})$$

En normalisant le fonction d'auto-covariance, on obtient alors la **la fonction d'auto-corrélation** ρ définie par :

$$\forall h \in \mathbb{Z} \quad \rho(h) = Corr(X_t, X_{t+h}) = \frac{\gamma(h)}{\gamma(0)}$$

L'opérateur retard B qui associe à tout processus $X = (X_t)_{t\in\mathbb{Z}}$ le processus $(Y_t)_{t\in\mathbb{Z}}$ est définie par :

$$\forall t \in \mathbb{Z}, \quad Y_t = BX_t = X_{t-1}$$

Un bruit blanc est un processus $\epsilon = (\epsilon_t)_{t\in\mathbb{Z}}$ tel que :

$$\mathbb{E}[\epsilon_t] = 0 \quad et \mathbb{E}[\epsilon_t \epsilon_t^{'}] = \sigma^2 \delta_{tt'} \quad \forall t, t^{'} \in \mathbb{Z}$$

Avec $\delta_{tt'}$ le symbole de Kronecker :

$$\delta_{tt'} = \begin{cases} 1 & \text{si } t = t' \\ 0 & \text{sinon} \end{cases}$$

$\sigma^2 > 0$ est appelée variance du bruit blanc ϵ_t

Un processus linéaire est un processus stochastique $(X_t)_{t\in\mathbb{Z}} \subset \mathcal{L}^2(\Omega, \mathcal{F}, P)$ formé d'une combinaison linéaire finie ou non de bruit blanc $(\epsilon_t)_{t\in\mathbb{Z}}$. Un processus $(X_t)_{t\in\mathbb{Z}}$ est linéaire de moyenne μ s'il peut s'écrire sous la forme :

$$X_t = \mu + \sum_{k=-\infty}^{+\infty} b_k \epsilon_{t-k} \quad avec \quad \sum_{k=-\infty}^{+\infty} b_k^2 < \infty$$

Nous définissons à présent quelques modèles de séries temporelles qui sont liés les uns aux autres :

Modèle AR Les modèles auto-régressifs forment une classe flexible de modèle pour de nombreux phénomènes observés. Ils sont construits à partir de l'idée que l'observation au temps t s'explique linéairement par les observations passées ; ils sont donc définis implicitement par la relation :

$$X_t = \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + ... + \alpha_p X_{t-p} + \epsilon_t \quad \forall t \in \mathbb{Z}, \quad p \geq 1. \tag{4.1}$$

Avec $\alpha 1....\alpha_p$ des réels fixés , $X_t)_{t\in\mathbb{Z}}$ un processus stationnaire, $(\epsilon_t)_{t\in\mathbb{Z}}$ un bruit blanc de variance σ^2. Le processus $(X_t)_{t\in\mathbb{Z}}$ est alors appelé processus auto-régressif d'ordre p $AR(p)$.

En posant $\Gamma(B) = I - \alpha_1 B - \alpha_2 B^2 - ... - \alpha_p B^p$, on obtient alors la relation suivante :

$$\Gamma(B) X_t = \epsilon_t \quad t \in \mathbb{Z}$$

Modèle MA Les processus moyennes mobiles forment une classe flexible de modèles pour de nombreux phénomènes observés. Ils sont construits à partir de l'idée que l'observation au temps t s'explique linéairement par les observations d'un bruit blanc ; ils sont

donc définis par la relation :

$$X_t = \epsilon_t - \theta_1\epsilon_{t-1} - \theta_2\epsilon_{t-2} - ... - \theta_q\epsilon_{t-q} \quad t \in \mathbb{Z}. \tag{4.2}$$

Où les $\theta 1, ..., \theta_q$ sont des réels fixés, $(\epsilon_t)_{t\in\mathbb{Z}}$ un bruit blanc de variance σ^2. Le processus $X_t)_{t\in\mathbb{Z}}$ est alors appelé processus en moyenne mobile d'ordre q : $MA(q)$.

De même, En posant $\Psi(B) = I - \theta_1 B - \theta_2 B^2 - ... - \theta_q B^q$, on obtient alors la relation suivante :

$$X_t = \Psi(B)\epsilon_t \quad , \textit{ avec } B \text{ l'opérateur } \textit{de retard.}$$

Le modèle ARMA Les séries ARMA sont des combinaisons des deux premiers modèles. Elles présentent les avantages d'être plus souples à l'utilisation et de fournir en principe de bonnes approximations des séries réelles , avec moins de paramètres que les modèles purs. Un processus stationnaire $X_t)_{t\in\mathbb{Z}}$ est un processus auto-régressif et moyenne mobile d'ordre (p, q) $(ARMA(p, q))$, s'il vérifie l'équation suivante :

$$X_t - \alpha_1 X_{t-1} - \alpha_2 X_{t-2} - ... - \alpha_p X_{t-p} = \epsilon_t - \theta_1\epsilon_{t-1} - \theta_2\epsilon_{t-2} - ... - \theta_q\epsilon_{t-q} \quad t \in \mathbb{Z}.$$

Avec $\alpha_1....\alpha_p$, $\theta_1....\theta_q$ des réels fixés et $\epsilon_t)_{t\in\mathbb{Z}}$ un bruit blanc de variance σ^2. En utilisant les fonctions définies précédemment, on obtient la relation :

$$\Gamma(B)X_t = \Psi(B)\epsilon_t \quad \forall t \in \mathbb{Z}.$$

Le modèle ARIMA On dit que que $X_t)_{t\in\mathbb{Z}}$ est un processus auto-régressif et moyenne mobile intégré ARIMA(p,d,q) s'il satisfait une équation du type :

$$\Gamma(B)(I - B)^d X_t = \Psi(B)\epsilon_t \tag{4.3}$$

où :

$$\begin{cases} \Gamma(B) = I + \alpha_1 B + \alpha_2 B^2 + ... + \alpha_p B^p \\ \Psi(B) = I + \theta_1 B + \theta_2 B^2 + ... + \theta_q B^q \end{cases}$$

$\Gamma(B)$ *et* $\Psi(B)$ sont des polynômes dont les racines sont de module supérieur à 1, ϵ_t est un bruit blanc de variance σ^2, $(I-B)^d X_t$ est un polynôme stationnaire. Et B l'opérateur retard.

Détermination des coefficients d'un modèle ARMA : méthode de Box et Jenkins

La méthode Box-Jenkins a été proposée par George Box et Gwilym Jenkins dans leur ouvrage de 1970 intitulé «Time Series Analysis : Forecasting and Control» .

L'approche commence par l'hypothèse que le processus qui a généré la série chronologique peut être approximé en utilisant un modèle ARMA s'il est stationnaire ou un modèle ARIMA s'il est non stationnaire. Cette méthode peut donc être présentée en trois parties :

— **Identification** : qui consiste à utiliser les données et toutes les informations connexes pour permettre de sélectionner une sous-classe de modèles qui résume le mieux les données. Cette étape peut être à son tour décomposée en deux sous parties à savoir :

. Différenciation : qui consiste à évaluer si la série temporelle est stationnaire et, dans le cas contraire, combien de différences sont nécessaires pour la rendre stationnaire. En général, on utilise le test statistique de racine unitaire sur la série chronologique pour déterminer si elle est stationnaire ou non. et on le répéte après chaque tour de différenciation. A noter que différencier la série temporelle plus que nécessaire peut entraîner l'ajout d'une série corrélation supplémentaire et d'une complexité supplémentaire.

. Choix du modèle AR et MA : les deux diagrammes qui peuvent être utilisés pour choisir les paramètres p et q de l'ARMA ou de l'ARIMA sont :

- la fonction d'autocorrélation (ACF), qui résume la corrélation d'une observation avec des valeurs de retard. L'axe des abscisses montre le décalage et l'axe des y montre le coefficient de corrélation entre -1 et 1 pour la corrélation négative et positive.

- la fonction d'autocorrélation partielle (PACF), dont la courbe résume les corrélations pour une observation avec des valeurs de retard qui ne sont pas prises en compte par les observations retardées antérieures.

On a donc :
Le modèle est AR si l'ACF se détache après un décalage et a une coupure dans le PACF après un retard. Ce décalage est pris comme valeur pour p .
Le modèle est MA si le PACF se détache après un décalage et a une coupure

dans l'ACF après le décalage. Cette valeur de décalage est prise comme la valeur de q .
Le modèle est un mélange de RA et de MA si l'ACF et le PACF sont tous deux présents.

— **Estimation** : qui permet d'utiliser les données pour former les paramètres du modèle (c'est-à-dire les coefficients).
— **Vérification diagnostique** : Évaluer le modèle ajusté dans le contexte des données disponibles et vérifier les zones où le modèle peut être amélioré.

4.2 Comment trouver un modèle afin de simuler nos données réelles ?

Dans cette partie notre objectif est de trouver un modèle de sorte que ses caractéristiques soient semblables à notre jeu de données initial. Une fois ce modèle mis en place, nous pourrons alors comparer les méthodes POT et AM, ainsi que les différents estimateurs, via la méthode de Monte Carlo que nous détaillerons par la suite.

4.2.1 Le jeu de données

Les données dont nous disposons dans le cadre de ce bureau d'étude décrivent l'évolution de la hauteur significative des vagues (HS) et de la vitesse du vent (WSPD) dans la région de Bretagne et dans le Sud de l'Angola.
HS est définie comme la moyenne des hauteurs crète à creux d'un tiers des plus fortes vagues observées pendant un intervalle de temps. Ceci permet de ne pas tenir compte des vagues très faibles, et de moyenner les vagues les plus fortes, afin qu'une vague exceptionnelle ne prenne pas trop d'importance dans le résultat final. Cette définition est expliquée sur la figure 4.4

Nous avons pour le jeu de données en Bretagne une observation toutes les heures, de 1994 à 2016, soit un total de 201624 observations pour HS et pour le jeu de données du Sud de l'Angola, nous avons une observation toutes les six heures soit 65744 observations, de 1957 à 2002.
Les données de vent sont en mètres par seconde (m/s) et les hauteurs significatives en mètres (m). Nous avons représenté sur la figure 4.5 les boîtes à moustache associées à chaque mois de nos données. Cette représentation graphique de données statistiques nous donne la valeur du premier et du troisième quartile, ainsi que la médiane. Sur la figure 4.6 nous avons affiché l'évolution de nos données au cours du temps.

Nous pouvons d'ores et déjà remarquer une forte saisonnalité dans nos données, ainsi

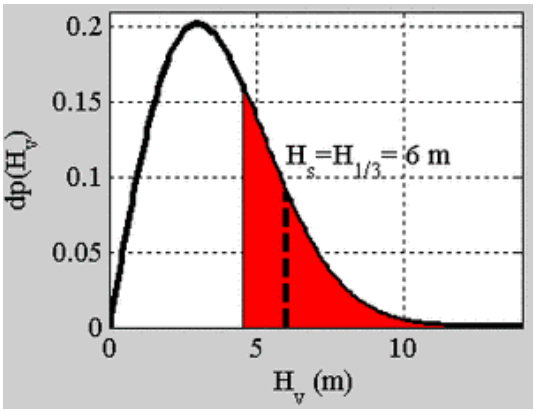

FIGURE 4.4 – Définition de HS : hauteur signicative d'une vague

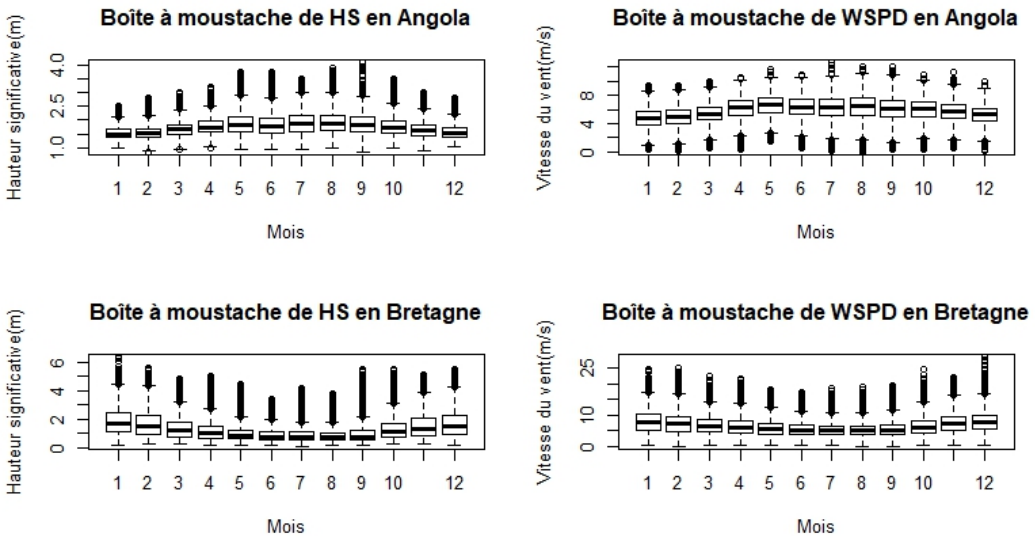

FIGURE 4.5 – Boxplot de nos données

qu'une corrélation importante. En effet, les données environnementales sont très dependantes entre elles, d'autant plus ici étant donné que l'écart entre chaque observation est de 1 à 6h. Il semble raisonnable de penser que la vitesse du vent à un instant est extrêmement liée à sa vitesse quelques heures auparavant. Il en est de même pour la hauteur significative des vagues. Ceci peut se confirmer graphiquement à l'aide d'un scatter plot entre les données $X(t)$ et les données décalées $X(t+1)$ ou du tracé de la fonction d'autocorrélation, définie précédemment.

Sur les graphiques de la figure 4.7 nous pouvons en effet voir que les nuages de points $(X(t),X(t+1))$ se situent autour de la droite $y = x$, ce qui montre également la forte corrélation entre les données que nous avons à notre disposition.

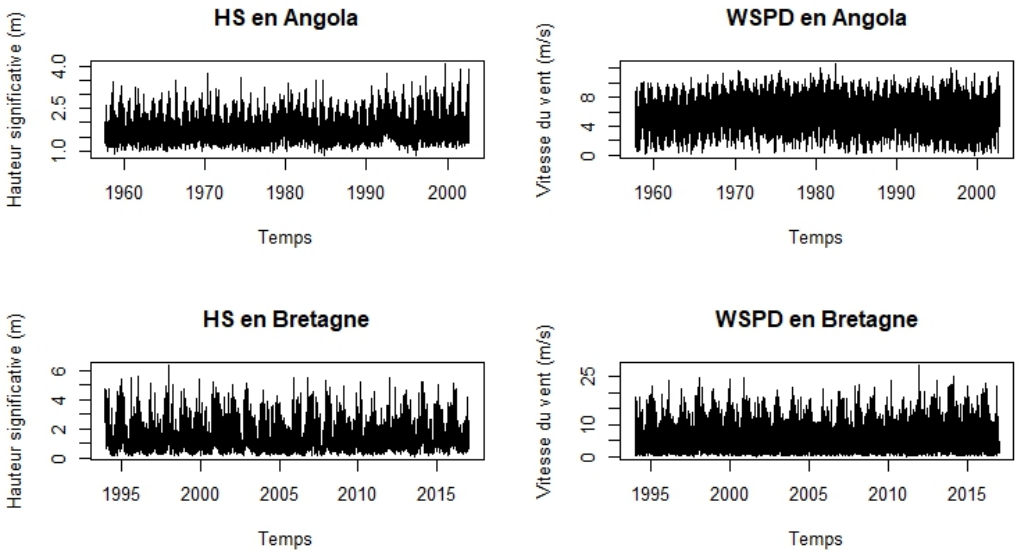

FIGURE 4.6 – Evolution de nos données au cours du temps

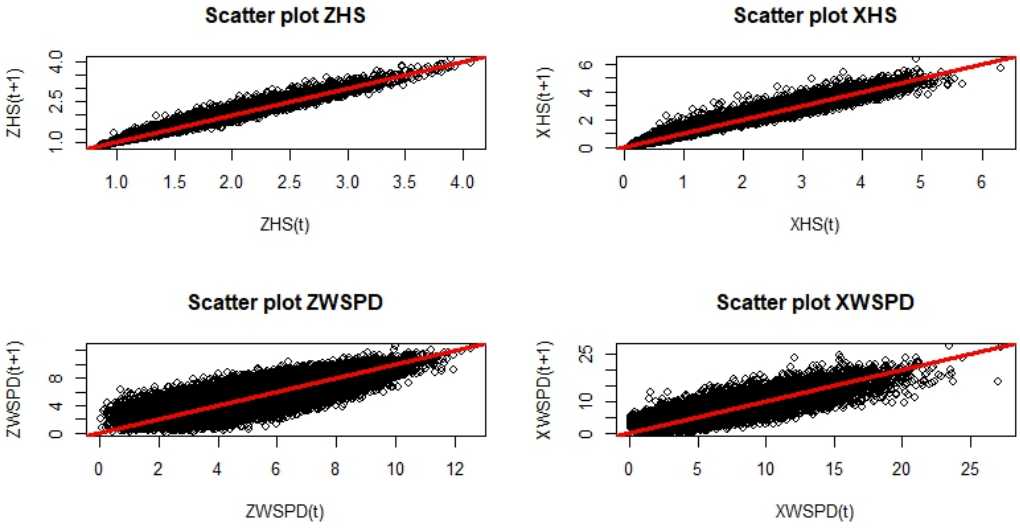

FIGURE 4.7 – Scatter plot de nos données

Cette corrélation et cette saisonnalité joueront un rôle majeur dans notre étude, et nous devrons en tenir compte. En effet, la plupart des modèles statistiques reposent sur une hypothèse de données i.i.d, ce qui ne sera pas vérifiée ici.

4.2.2 Première modélisation

Comme expliqué précédemment, nous voulons trouver un modèle fidèle à notre jeu de données. Dans cette première approche, nous tenterons de trouver un modèle qui représente de manière la plus proche possible nos données, contrairement à la partie suivante, où nous réaliserons un choix de loi au préalable, qui ne sera pas forcément similaire à la loi suivie par nos données.
La méthodologie que nous utiliserons pour étudier la série temporelle à notre disposition sera la suivante : - transformation adéquate de nos données pour les rendre gaussiennes
- désaisonnalisation de nos données (+ éventuellement lissage de la saisonnalité)
- modèle ARMA de notre série transformée,désaisonnalisée
- transformation inverse et ajout de la saisonnalité : modèle final

La première étape concerne la transformation de nos données. Pour cela, nous utilisons la transformation de Box Cox. L'idée est la suivante : à partir de données intiales, l'objectif est de trouver un coefficient λ afin de rendre ses données gaussiennes. C'est exactement ce dont nous avons besoin ici, car le modèle ARMA qu'on souhaite mettre en place nécessite des données gaussiennes. La transformée de Box Cox se présente sous la forme suivante :

$$B(x,\lambda) = \begin{cases} \frac{x^\lambda - 1}{\lambda} & \text{si } \lambda \neq 0 \\ \log(x) & \text{si } \lambda = 0 \end{cases}$$

Pour les données HS en Angola, nous avons appliqué la transformation logarithmique : $\lambda = 0$ (fonction couramment utilisée pour des données environnementales). Pour les données de vent en Angola, aucune transformation nécéssaire au préalable, $\lambda = 1$. Pour les données de HS en Bretagne, nous n'avons pas trouvé de coefficients permettant de rendre la série gaussienne. Nous avons décidé de ne pas s'occuper de celle-ci et de se concentrer sur les autres données à notre disposition Pour les données de WSPD en Bretagne, la transformation racine carrée a été utilisée : $\lambda = 0.5$

Les graphiques suivants (figure 4.8 et 4.9) résument les opérations réalisées et montrent que les données après transformations sont en effet proches d'une distribution gaussienne.

Une fois nos données transformées, il reste la saisonnalité, qu'il faut extraire. Cette étape aurait également pu être réalisée avant l'étape de normalisation des données mais les résultats étaient plus concluants en effectuant les étapes dans cet ordre ci. Pour trouver la saisonnalité dans notre jeu de données, nous utilisons la fonction 'seasadj' du package 'forecast' de R, qui utilise une méthode adaptée de ce qui a été présenté dans la partie précédente. Nous avons par la suite lissé notre saisonnalité, à l'aide d'une moyenne mobile. Notons qu'ici, nous ne possédons pas de tendance dans notre série. En effet, nos données sont supposées stationnaires, c'est-à-dire que la moyenne reste la même au cours du temps. On néglige donc ici les éventuels effets du changement climatique sur nos données de vent

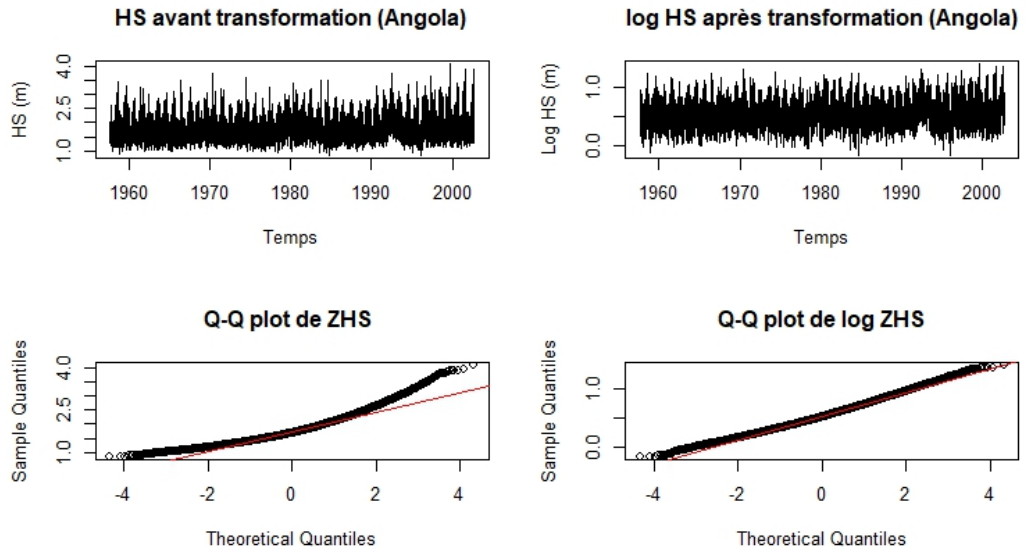

FIGURE 4.8 – Normalisation de nos données de hauteur significative en Angola

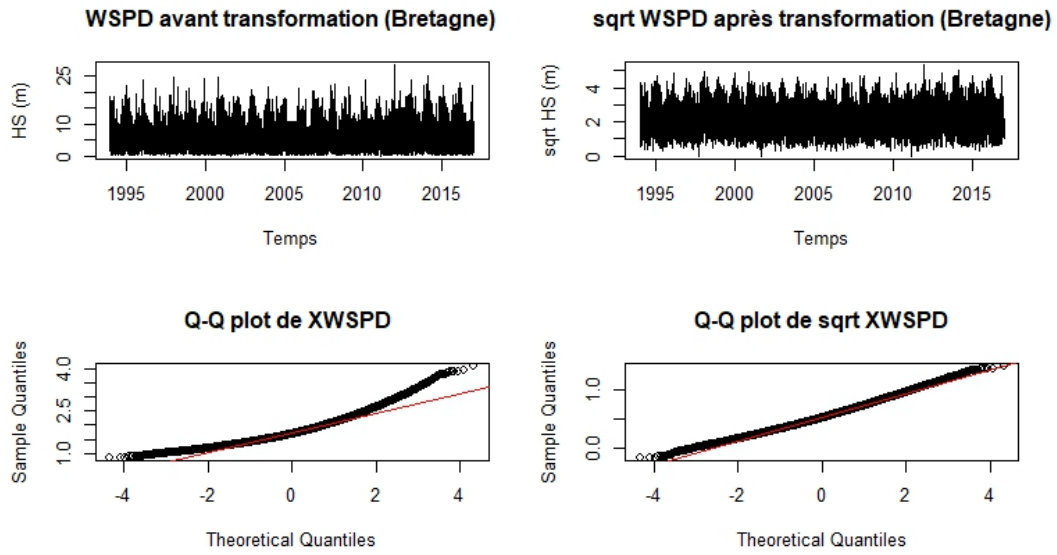

FIGURE 4.9 – Normalisation de nos données de vent en Bretagne

et de hauteur significative.

Les graphiques de la figure 4.10 résument cette étape d'extraction de la saisonnalité de nos données :

Nous avons à présent des données transformées, désaisonnalisées, qui sont approximativement gaussiennes. Nous avons ainsi la décomposition détaillée dans la partie précédente,

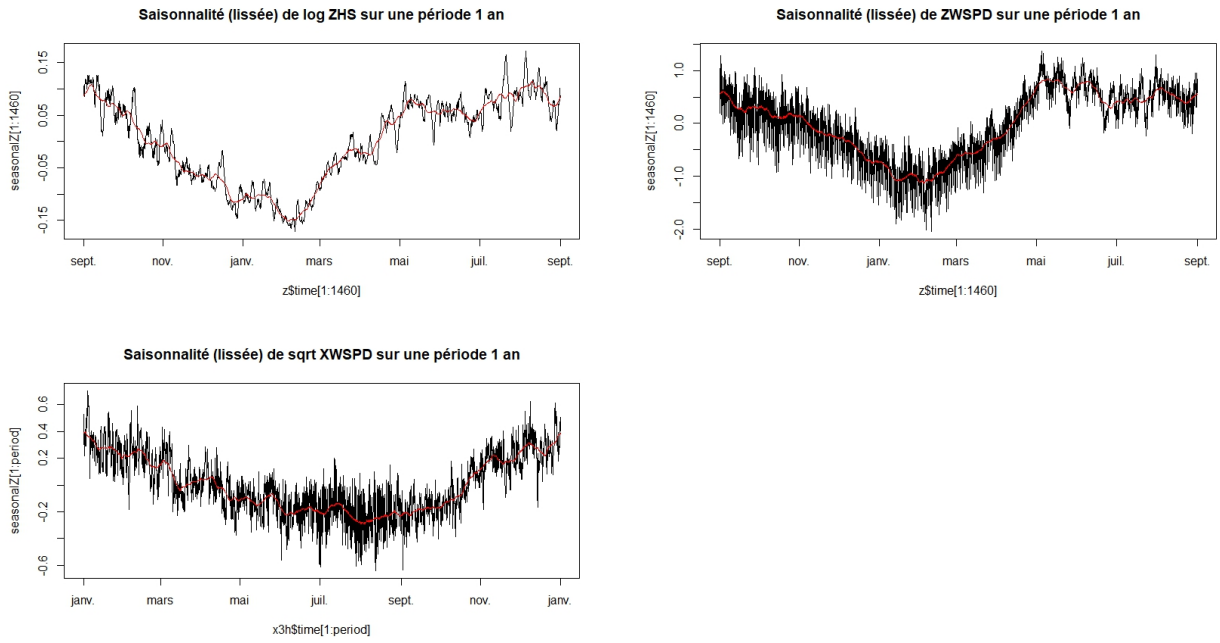

FIGURE 4.10 – Calcul de la saisonnalité de nos jeux de données

à savoir : tendance (nulle ici), saisonnalité et reste. Nos étapes de décomposition sont résumées sur les figures 4.11, 4.12 et 4.13.

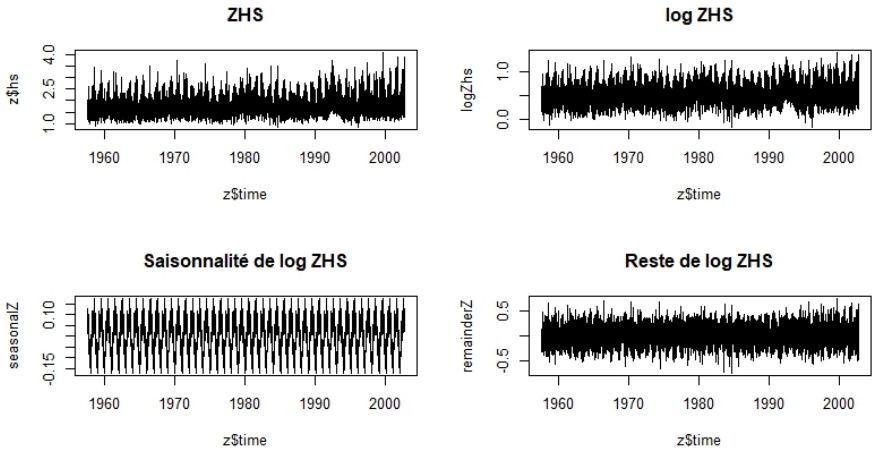

FIGURE 4.11 – Décomposition de la série ZHS

A présent, il est alors possible de trouver un modèle ARMA qui approxime nos données. Pour cela, nous avons appliqué la méthode de Box et Jenkins vue précédemment.

Les résultats obtenus via cette méthode sont résumés dans les tableaux des figures

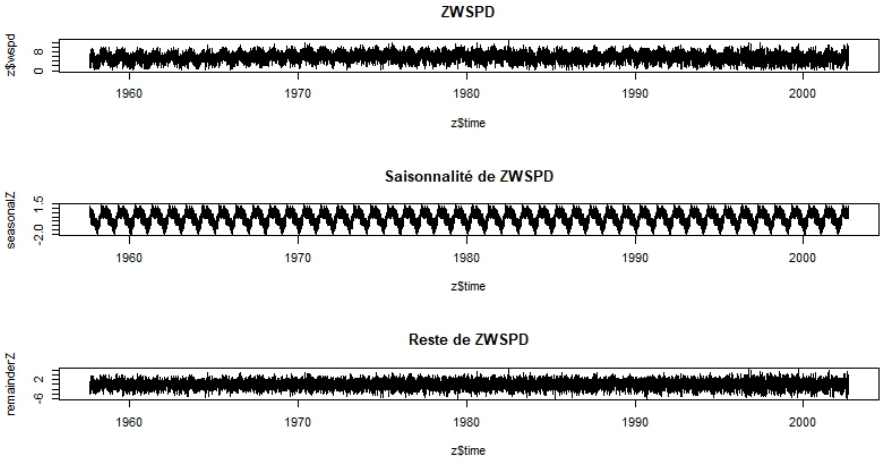

FIGURE 4.12 – Décomposition de la série ZWSPD

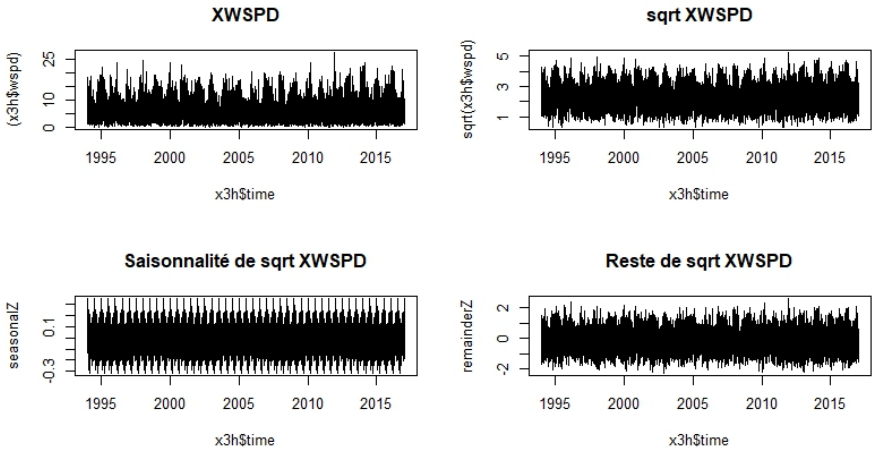

FIGURE 4.13 – Décomposition de la série XWSPD

4.14, 4.15 et 4.16. Ceux-ci contiennent les ordres p et q des modèles ARMA ajustés, coefficients α_i, β_i associés, l'écart-type σ du bruit restant, ainsi que les critères AIC, BIC, AICc permettant d'obtenir le modèle reflétant au mieux notre modèle.

```
ARIMA(2,0,5) with non-zero mean

Coefficients:
         ar1      ar2      ma1      ma2     ma3     ma4     ma5   mean
      1.6264  -0.6665  -0.1751  -0.0615  0.1187  0.1474  -0.037  0.535
s.e.  0.0122   0.0113   0.0129   0.0073  0.0049  0.0054   0.006  0.003

sigma^2 estimated as 0.0009665:  log likelihood=134905.9
AIC=-269793.8   AICc=-269793.8   BIC=-269712

Training set error measures:
                      ME       RMSE        MAE       MPE    MAPE      MASE          ACF1
Training set 8.655832e-07 0.03108724 0.02141728 -0.28246 5.45535 0.7396963 -4.968244e-05
```

FIGURE 4.14 – Résultats de la méthode Box-Jenkins sur ZHS

```
ARIMA(5,0,4) with non-zero mean

Coefficients:
         ar1     ar2      ar3     ar4      ar5      ma1      ma2     ma3      ma4     mean
      0.7841  0.1597  -0.2177  0.4683  -0.3195  -0.0758  -0.2693  0.2181  -0.1391  5.8143
s.e.  0.0218  0.0240   0.0249  0.0229   0.0163   0.0223   0.0209  0.0166   0.0195  0.0201

sigma^2 estimated as 0.7678:  log likelihood=-84595.53
AIC=169213   AICc=169213.1   BIC=169313.1

Training set error measures:
                      ME      RMSE       MAE       MPE    MAPE      MASE          ACF1
Training set 2.450277e-05 0.8761592 0.6875889 -4.026103 15.0144 0.9010524 -0.002292874
```

FIGURE 4.15 – Résultats de la méthode Box-Jenkins sur ZWSPD

```
ARIMA(1,0,5) with non-zero mean

Coefficients:
         ar1     ma1      ma2      ma3      ma4      ma5    mean
      0.9132  0.0249  -0.1181  -0.1368  -0.0556  -0.0710  2.4481
s.e.  0.0036  0.0054   0.0054   0.0050   0.0043   0.0045  0.0086

sigma^2 estimated as 0.09096:  log likelihood=-14801.84
AIC=29619.69   AICc=29619.69   BIC=29692.61

Training set error measures:
                       ME      RMSE       MAE       MPE     MAPE      MASE         ACF1
Training set -2.170443e-05 0.3015833 0.2285307 -2.132558 10.73226 0.9523406 0.001416483
```

FIGURE 4.16 – Résultats de la méthode Box-Jenkins sur XWSPD

Maintenant que notre modèle ARMA est défini, nous devons réaliser les étapes précédentes dans l'autre sens afin de revenir à nos données initiales. En effet, le modèle trouvé correspondait à nos données transformées, désaisonnalisées. Notre idée est alors maintenant d'ajouter cette saisonnalité, ainsi que de réaliser la transformation inverse (à savoir passage à l'exponentielle pour ZHS, au carré pour XWSPD). Nous obtenons alors le modèle final, qui doit être proche de notre jeu de données initial. Nous avons superposé nos données initiales avec une simulation de notre modèle final sur les figures 4.17, 4.18 et 4.19. Nous voulions vérifier que graphiquement les simulations de notre modèle sont en adéquation avec les données initiales, ce qui semble être le cas ici.

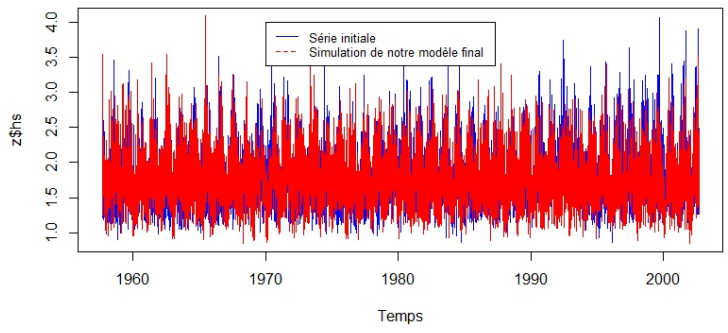

FIGURE 4.17 – Superposition données réelles ZHS et simulation de notre modèle

Cependant, nous observons à l'aide d'un QQplot (c.f figure 4.20), une différence notable au niveau des queues de notre modèle. Ce dernier a tendance à sous-estimer les valeurs

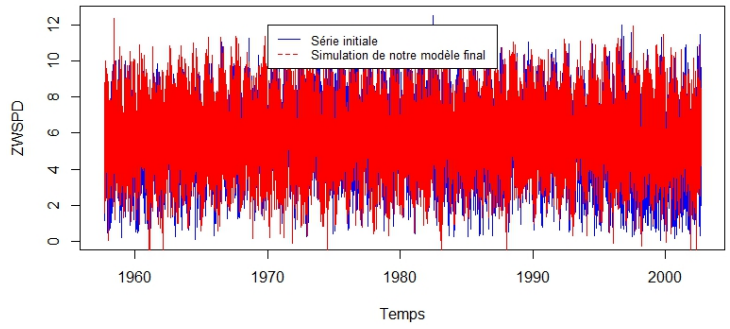

FIGURE 4.18 – Superposition données réelles ZWSPD simulation de notre modèle

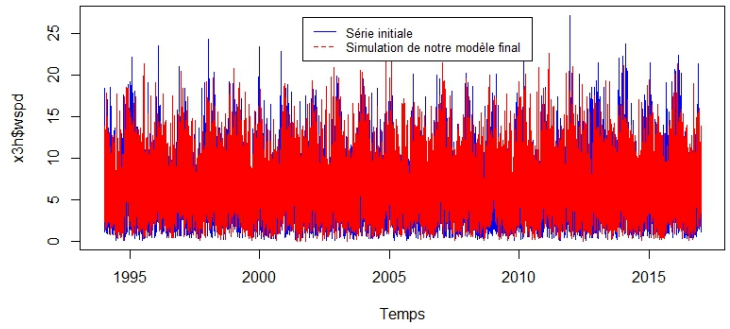

FIGURE 4.19 – Superposition données réelles XWSPD et simulation de notre modèle

élevées. Ceci peut être très dangereux, car une sous-estimation de notre niveau de retour à 100 ans conduit à une mauvaise estimation (à la baisse) du risque encouru.

Finalement, nous avons décidé de ne pas retenir ce modèle final mais d'adapter la méthodologie utilisée. Cette seconde méthode reposera sur le choix de loi avant la modélisation ce qui nous permettra de mieux contrôler les paramètres de forme et d'échelle des distributions GEV limite associées à notre modèle. Ceci est détaillé dans la partie suivante.

4.2.3 Deuxième modélisation

Dans cette deuxième approche, nous reprenons les étapes de la première méthode, que nous adaptons par la suite. Nous voulons toujours prendre en compte la dépendance temporelle de notre série. Notre objectif est de trouver un modèle ARMA (Auto Regressive Moving Average) correspondant à notre série de données désaisonnalisées, i.e. en enlevant la partie saisonnière de la série (il s'agit d'une série périodique de période 1 an). Cependant,

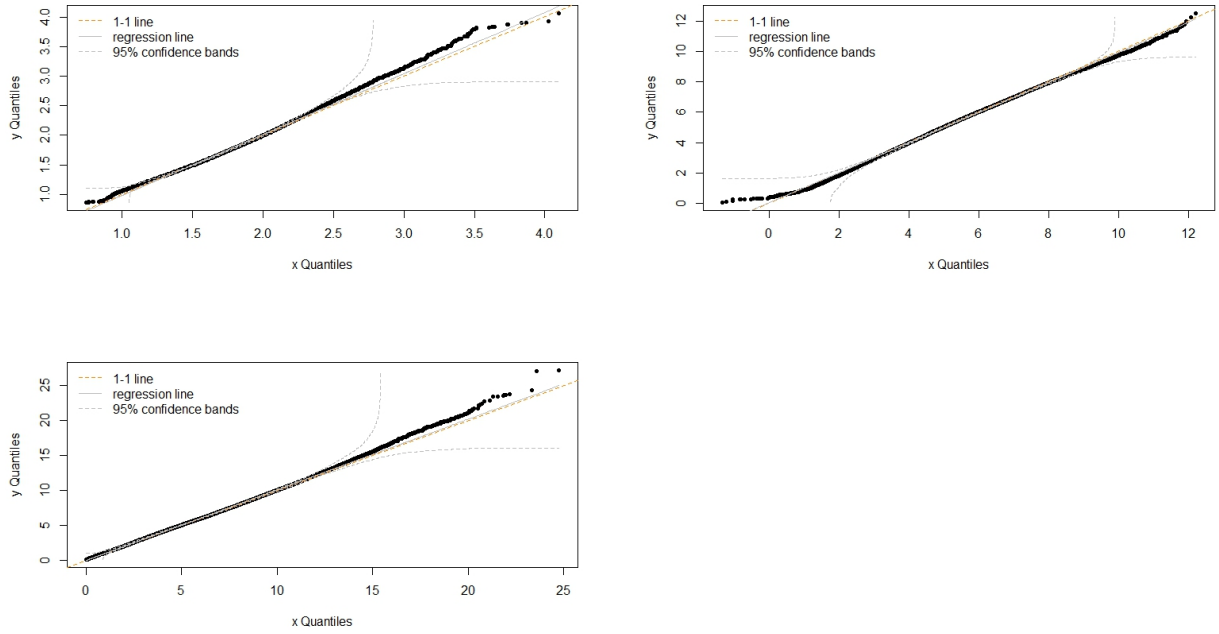

FIGURE 4.20 – QQplot de nos modèles finaux de ZHS,ZWSPD et XWSPD

nous devons au préalable effectuer une transformation afin de rendre nos données gaussiennes (car un modèle ARMA impose la normalité des données). La transformation usuelle pour les données environnementales est l'application de la fonction logarithme ou racine carrée aux données initiales. Une fois la transformation et la désaisonnalisation effectuées, il est possible de trouver un modèle ARMA. Pour les données de hauteurs significatives, nous avons les modèles suivants :

ARMA			MA(q)					AR(p)				
Coefficients	p	q	β_1	β_2	β_3	β_4	β_5	α_1	α_2	α_3	α_4	α_5
zhs	3	4	-0.18	-0.06	0.12	0.15	-0.04	1.63	-0.67			
xhs	3	3	0.78	0.88	0.42			0.91	-0.73	0.76		
zwspd	4	5	-0.08	-0.27	0.22	-0.14		0.78	0.16	-0.22	0.47	-0.32
xwspd	5	1	0.025	-0.12	-0.14	-0.06	-0.07	0.91				

TABLE 4.1 – Résultats des modèles ARIMA ajustés pour nos jeux de données

L'odre p du modèle signifie que la valeur du processus à un instant donné est une combinaison linéaire des observations 1 à p ; et l'ordre q signifie qu'il faut ajouter une combinaison linéaire de q+1 bruits blancs gaussiens (erreur aléatoire). Une fois le modèle ARMA ajusté, nous disposons d'un modèle décrivant la dépendance temporelle de notre

jeu de données. Ce processus ARMA, centré réduit (noté X_t) suit une loi gaussienne : $X_t \sim N(0,1)$. Nous pouvons donc lui appliquer la fonction de répartition ϕ de loi normale centrée réduite et nous obtenons finalement $U_t = \phi(X_t)$ qui suit une loi uniforme sur l'intervalle [0,1] : $U_t \sim N(0,1)$. Il est important de noter que les échantillons de loi U_t ne sont pas indépendants ; en effet, l'objectif de la mise en place du modèle ARMA était de conserver la dépendance temporelle de nos données. A partir de cette loi uniforme, nous pouvons générer n'importe quelle loi, à condition que l'on connaisse sa fonction répartition (et son inverse généralisée). En effet, si Y est une variable aléatoire de fonction de répartition F alors la variable $F^{-1}(U)$ a la même loi que Y : $Y \sim F^{-1}(U)$ (où $U \sim U([0,1])$ est uniforme sur [0,1]).

Nous allons à présent mettre en place le modèle final, pour cela nous devons choisir la fonction de répartition F_t. Ce choix n'est pas totalement arbitraire, nous devons choisir F_t de sorte que la loi associée ait un paramètre de queue κ fixé, une certaine moyenne m_t et un certain écart-type $\S_t$ déterminé par le mois dans lequel la mesure est prise. Le seul choix qu'il nous reste à faire est la loi considérée, le choix des paramètres étant imposé par la méthode des moments (de sorte que la moyenne et l'écart type du mois soient respectés). Nous avons choisi, comme dans la référence 1 les lois : GEV, Gamma, Beta I, Beta II. Pour la loi GEV nous pouvons directement contrôler le paramètre de queue, pour la loi gamma, $\kappa = 0$ (loi de type I), et pour la beta I et beta II il est possible de fixer κ également. Pour une loi Beta I, κ est strictement négatif , et pour une loi Beta II, κ est strictement positif. Ainsi pour trouver le modèle final, il nous faut calculer la moyenne et l'écart type de chaque mois de nos données. Nous avons détaillé les résultats pour la hauteur significative en Angola, et ceci est résumé dans le tableau suivant :

Mois	Moyenne (m_t)	Ecart type (S_t)
Janvier	1.53	0.24
Février	1.54	0.26
Mars	1.68	0.26
Avril	1.78	0.31
Mai	1.87	0.39
Juin	1.83	0.40
Juillet	1.91	0.41
Août	1.92	0.42
Septembre	1.87	0.40
Octobre	1.75	0.33
Novembre	1.64	0.29
Décembre	1.57	0.26

TABLE 4.2 – Moyenne et écart-type des données d'hauteurs significatives en Angola

Il faut bien noter que l'on a utilisé des séries temporelles pour créer des séries avec des caractéristiques « réalistes » ; bien que les séries créées en soi ne sont pas réalistes du fait qu'elles ne représentent en fait aucune donnée réelle. Idéalement il faudrait créer chaque série en analysant chaque comportement des variables d'intérêt dans différentes régions du monde, ce qui est trop complexe pour notre étude ici et pas forcément nécessaire. Mais notre étude, en incorporant des valeurs proches de la réalité pour kappa et des caractéristiques des séries temporelles que l'on souhaite étudier, devrait donner des conclusions plutôt générales. A la suite de cette étude, nous pourrons alors faire des recommandations à toute personne possédant des données environnementales et souhaitant calculer des niveaux de retour à 100 ans. En effet, nous pourrons lui indiquer quelle méthode choisir entre la méthode du dépassement de seuil et la méthode des maxima annuels ; ainsi qu'entre l'estimateur du maximum de vraisemblance et l'estimateur PWM, suivant le nombre d'années de données dont il dispose.

Pour résumé, nous avons réalisé les étapes suivantes :
1) Données initiales (WSPD ou HS) : Y_t
2) On « gaussiannise » nos données (transformation adéquate) : $Zt = logY_t$ ou $\sqrt{Y_t}$
3) Désaisonnalisation : $X_t = Z_t -$ saisonnalité
4) On centre le processus (moyenne nulle) : $X_t < -X_t - mean(X_t)$
5) On ajuste un modèle ARMA à X_t : A_t (qui suit une loi normale $N(0, S^2)$)
6) On rend le modèle N(0,1) : A_t <- A_t/S
7) On rend la loi uniforme : $U_t = \phi(A_t)$ où phi est la fonction de répartion de la loi $N(0, 1)$
8) On choisit une loi F_t pour chaque mois, de sorte que la variance et la moyenne des données initiales soient respectées, et on la génère via : $Vt = Ft^{-1}(Ut)$ (en imposant un paramètre de forme κ de la loi GEV limite)
Ainsi, la série temporelle finale a des propriétés similaires aux données initiales : même moyenne et écart-type pour chaque mois ainsi qu'une dépendance temporelle conservée.

Dans la méthode décrite précedemment, nous devons réaliser un choix de la loi de F_t pour chaque mois de l'année. Nous devons choisir des lois telles qu'on ait la possibilité de choisir un paramètre de forme κ et possédant deux paramètres afin de les relier à la moyenne et l'écart de données initiales sur chaque mois via la méthode des moments. Notre objectif est d'étudier des valeurs de kappa entre -0.3 et 0.1. Pour les valeurs négatives de kappa, on se trouve avec des lois de type II , pour un kappa nul, nous avons des lois de type I et enfin avec un kappa strictement positif, une loi de type III. Ceci est résumé dans le tableau suivant :

Dans un premier temps, nous avons choisi une loi GEV pour chaque mois de l'année. Ceci est le cas le plus simple, car on peut contrôler directement kappa qui est un paramètre de la loi et on a aussi la possibilité de choisir à la fois des valeurs positives, nulles ou négatives. Ensuite, nous avons choisi d'autres lois suivant les valeurs de kappa : Pour une

TABLE 4.3 – Loi d'extrêmum généralisée : GEV

Kappa	Type	Domaine d'attraction	Queue
<0	III	Weibull	bornée
0	I	Gumbel	normale
>0	II	Fréchet	lourde

loi de type III ($\kappa < 0$), nous avons choisi la loi Beta I, pour une loi de type I ($\kappa = 0$), la loi Gamma et enfin pour une loi de type II, ($\kappa > 0$), nous avons opté pour une loi Beta II. Ces choix ont été réalisés à l'aide de la référence. Le détail des calculs pour ces lois est détaillé dans la partie suivante. Sur la figure 4.21, nous avons superposé une simulation obtenue via notre modèle (à partir des 4 lois ci-dessus) et nos données réelles.

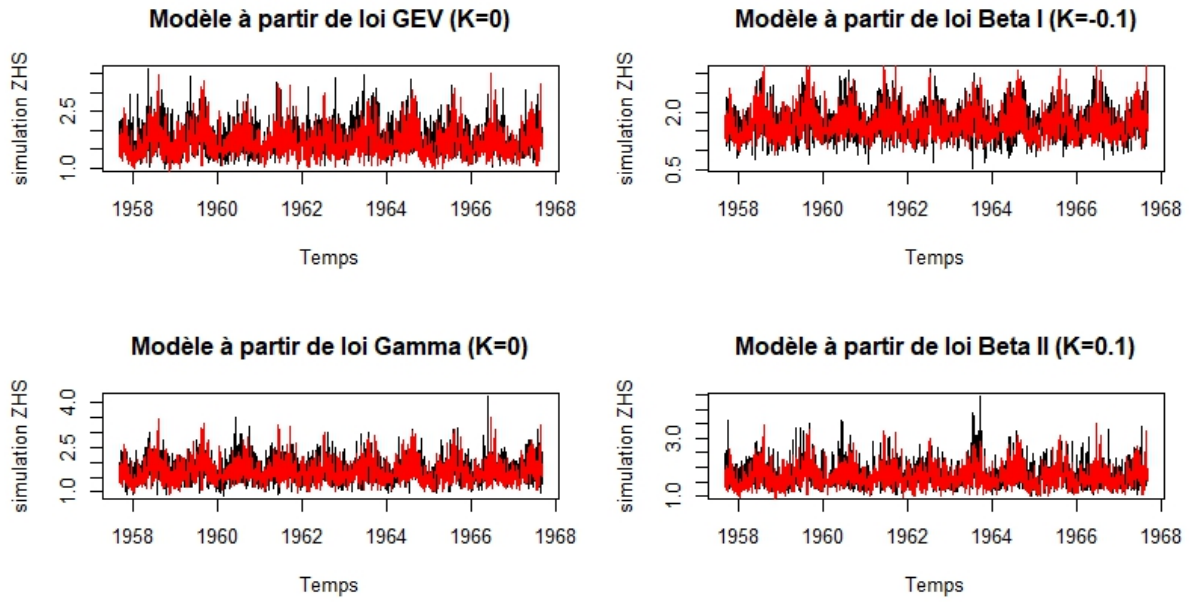

FIGURE 4.21 – Superposition de nos données réelles et d'une simulation de notre deuxième modèle à partir des lois GEV,Beta I, Gamma et Beta II

4.3 Résultats des différentes études réalisées auparavant dans le cas de données dépendantes

Les propriétés des estimateurs MLE et PWM des lois GEV et GPD ont également déjà été étudiées dans le cas de données dépendantes.
En 2005, van den Krink et al. ont réalisé une étude sur un modèle de tempêtes à un endroit bien spécifique dans la mer du Nord, avec 7540 années de données. (c.f référence 16) Ils ont enregistré 65 fois 116 années de données et ont analysé les résultats des méthodes AM et POT, en utilisant l'estimateur MLE. L'objectif était l'estimation du niveau de retour à 10 000 ans. La conclusion était que si le nombre de clusters est inférieur à 4, le choix du seuil est difficile et la méthode AM/GEV est préférable.

Une autre étude réalisée par de Valk en 1993 avec des données de vagues, (de type I : $\kappa = 0$) et 10, 25 et 100 ans de taille d'échantillon (c.f référence 11). Il a étudié la performance de la méthode POT/GPD en considérant trois seuils fixés : 4,5 et 6 mètres. Sa conclusion était, qu'à moins que l'échantillon soit de taille 100, le niveau de retour estimé via GPD surestime la valeur théorique.

4.4 Interprétation des résultats

4.4.1 Mise en place des méthodes AM et POT

Comme dans le cas des données indépendantes, nous réalisons une méthode de Monte Carlo afin de déterminer, entre POT et AM, la méthode plus performante pour l'estimation du niveau de retour à 100 ans. Pour se faire nous utilisons le modèle défini dans la partie précédente. En le simulant sur un grand nombre d'années, nous pouvons obtenir les valeurs "théoriques" puis nous les comparons aux résultats obtenus à partir de simulations de taille réduite. Comme les données ne sont pas iid, nous devons utiliser les résultats de la partie 2.3 pour assurer l'indépendance de nos échantillons POT et AM. Rappelons que dans le cas de la méthode AM, nous devons juste vérifier que le maximum d'une année donnée n'est pas pas à la même tempête que le maximum de l année suivante. Concernant la méthode POT, nous mettons en place des clusters. En gardant les notations de la partie 2.3.2, nous devons choisir les valeurs de u, u_{min} et d_{clust}. Pour u, nous avons choisi le quantile d'ordre 0.95 de la simulation considérée. Ceci permet d'automatiser ce choix du seuil et de ne pas faire une étude graphique pour chaque simulation de Monte Carlo. Pour u_{min}, nous avons décidé de le fixer au quantile d'ordre 0.9. Enfin, pour d_{clust}, nous avons considéré une valeur de 3 jours, ce qui correspond à la durée moyenne d'une tempête.

4.4.2 Analyse dans le cas de la loi GEV

Les tableaux 4.4 et 4.5 présentent les résultats des biais et RMSE relatifs des estimateurs ML et PWM pour le niveau de retour à 100 ans et le paramètre de forme kappa pour 100 simulations générées via la loi GEV.

— **Analyse du taux d'erreur(du MLE)** : comme dans le cas iid et pour les mêmes raisons, le taux d'erreur est plus faible pour la loi GPD que pour la loi GEV. Nous notons évidemment que le taux d'erreur diminue avec l'augmentation de la taille de l'échantillon. Nous pouvons aussi remarquer que le taux d'erreur semble plus conséquent surtout lorsque le nombre d'années n_y est inférieur à 50.

— En terme de RMSE, pour le niveau de retour et le paramètre de forme kappa, la méthode POT semble toujours plus performante que la méthode AM et ceci quelque soit le paramètre de forme et la taille de l'échantillon n_y. Concernant les estimateurs MLE et PWM, l'écart entre les deux méthodes semble léger bien qu'on observe plus de précision avec PWM.

— En terme de biais, pour le paramètre de forme et niveau de retour à 100 ans, les méthodes POT et AM semblent donner des résultats similaires bien que POT paraît meilleur. Pour ce qui est des estimateurs, on peut observer que l'écart de précision entre MLE et PWM est léger.

méthode AM/GEV							méthode POT/GPD						
kappa		-0.3	-0.2	-0.1	0	0.1	kappa		-0.3	-0.2	-0.1	0	0.1
n_y	est						n_y	est					
10	mle	5.01	5.02	-6.2	8.7	-15.01	10	mle	6.49	-5.48	-5.22	3.34	-7.02
20	mle	3.82	3.83	3.39	-7.38	7.93	20	mle	3.25	3.17	2.84	1.82	4.66
50	mle	-2.34	-4.2	2.19	1.14	2.07	50	mle	1.57	2.07	1.82	1.48	0.19
100	mle	1.5	2.05	-1.15	-0.79	-0.22	100	mle	-0.76	-0.86	0.9	-0.65	-0.5
10	pwm	4.16	-8.36	-4.66	7.08	-8.23	10	pwm	5.37	3.56	2.01	2.94	-6.17
20	pwm	-3.48	3.26	3.56	6.5	5.57	20	pwm	-2.27	-2.81	1.65	-1.38	3.14
50	pwm	1.81	-2.35	1.73	1.04	1.31	50	pwm	1.64	1.45	1.26	-1.12	-0.17
100	pwm	0.23	-0.09	-0.57	0.09	-0.11	100	pwm	-0.26	0.87	0.12	-0.15	-0.08
ny	est						ny	est					
10	mle	-4.9	-7.84	-15.07	1.7	18.21	10	mle	-1.8	-2.74	-5.5	0.54	6.1
20	mle	-2.33	-3.51	-7.36	0.76	8.06	20	mle	-1.06	-1.74	-3.48	0.37	3.8
50	mle	-1.13	-1.77	-3.52	0.37	4.04	50	mle	-0.59	-0.96	-1.81	0.22	2.3
100	mle	-0.71	-1.12	-2.27	0.24	2.52	100	mle	-0.4	-0.64	-1.36	0.15	1.6
10	pwm	-3.12	-4.95	-9.4	0.92	9.22	10	pwm	-2.06	-2.89	-5.65	0.53	5.6
20	pwm	-1.98	-2.91	-5.87	0.58	6.35	20	pwm	-1.41	-2.08	-3.86	0.38	3.7
50	pwm	-1.17	-1.67	-3.26	0.35	3.82	50	pwm	-0.9	-1.31	-2.32	0.23	2.3
100	pwm	-0.76	-1.15	-2.25	0.24	2.58	100	pwm	-0.62	-0.89	-1.75	0.17	1.6

TABLE 4.4 – Matrice des biais et RMSE relatifs des estimateurs MLE et PWM pour le niveau de retour à 100 ans pour 100 simulations, dans le cas non i.i.d, générées via la loi GEV.

méthode AM/GEV							méthode POT/GPD						
quantile		2.49	3.01	3.69	4.6	5.84	quantile		2.82	3.56	4.63	6.21	8.62
kappa		-0.3	-0.2	-0.1	0	0.1	kappa		-0.3	-0.2	-0.1	0	0.1
n_y	est			Biais			n_y	est			Biais		
10	mle	*	*	*	*	*	10	mle	-6.49	-5.48	-5.22	-2.34	-1.02
20	mle	-3.82	-3.83	3.39	7.38	10.93	20	mle	-3.25	-3.17	-2.84	-1.82	-1.66
50	mle	-3.64	-4.2	-1.19	0.14	1.07	50	mle	-1.57	-2.07	-0.82	-1.48	-0.19
100	mle	-2.5	-2.05	-1.15	-0.79	0.22	100	mle	-0.76	-0.86	-0.9	-0.65	-0.5
10	pwm	8.16	8.86	4.66	6.08	2.23	10	pwm	1.37	3.56	2.01	3.94	1.17
20	pwm	3.48	1.26	3.56	3.05	-0.57	20	pwm	1.27	2.81	1.65	1.38	0.14
50	pwm	0.81	-0.35	1.73	1.04	-0.31	50	pwm	0.64	-0.45	1.26	-0.12	0.72
100	pwm	0.23	0.09	0.57	0.09	-0.11	100	pwm	0.96	0.87	-0.12	-0.15	-0.08
n_y	est			RMSE			n_y	est			RMSE		
10	mle	*	*	*	*	*	10	mle	0.53	0.67	0.95	1.19	1.59
20	mle	0.83	1.07	2.19	1.89	2.66	20	mle	0.34	0.47	0.62	0.82	1.01
50	mle	0.42	0.51	0.65	0.81	1.04	50	mle	0.19	0.28	0.35	0.49	0.61
100	mle	0.26	0.34	0.42	0.51	0.62	100	mle	0.13	0.19	0.25	0.33	0.42
10	pwm	1.25	1.44	1.52	1.78	1.98	10	pwm	0.78	0.87	1.07	1.21	1.4
20	pwm	0.79	0.85	1.08	1.18	1.39	20	pwm	0.55	0.66	0.75	0.86	0.97
50	pwm	0.47	0.51	0.63	0.74	0.89	50	pwm	0.35	0.41	0.45	0.53	0.61
100	pwm	0.3	0.36	0.43	0.5	0.61	100	pwm	0.25	0.29	0.33	0.37	0.43

TABLE 4.5 – Matrice des biais et RMSE relatifs des estimateurs MLE et PWM pour le paramètre de forme κ pour 100 simulations, dans le cas non i.i.d, générées via la loi GEV.

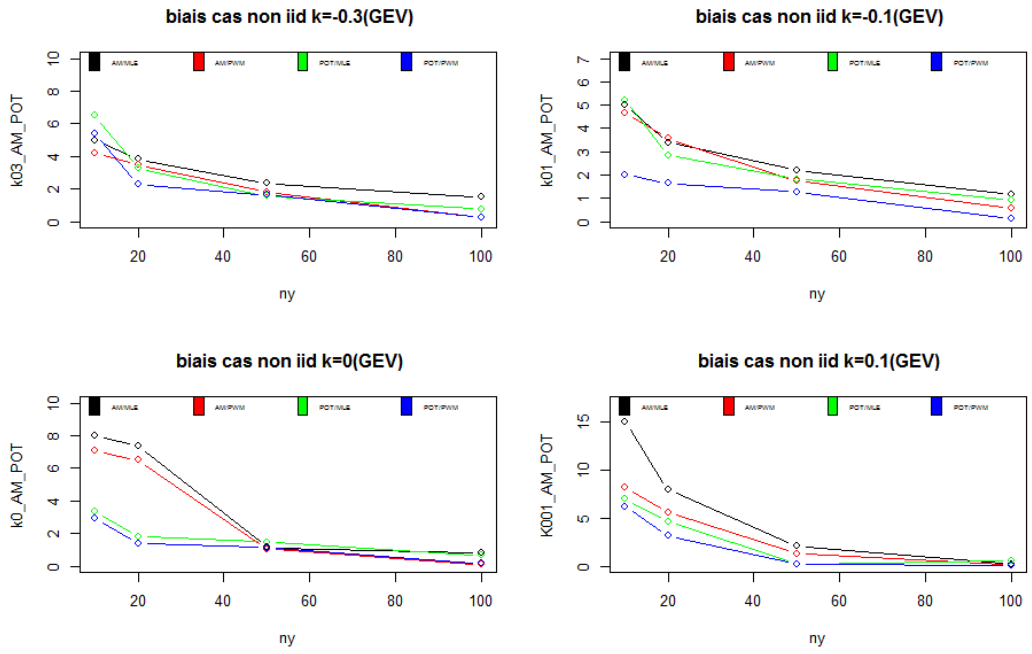

FIGURE 4.22 – Graphiques des biais (en valeur absolue) du niveau de retour à 100 ans pour 100 simulations dans l'analyse du cas non iid générées via la loi GEV

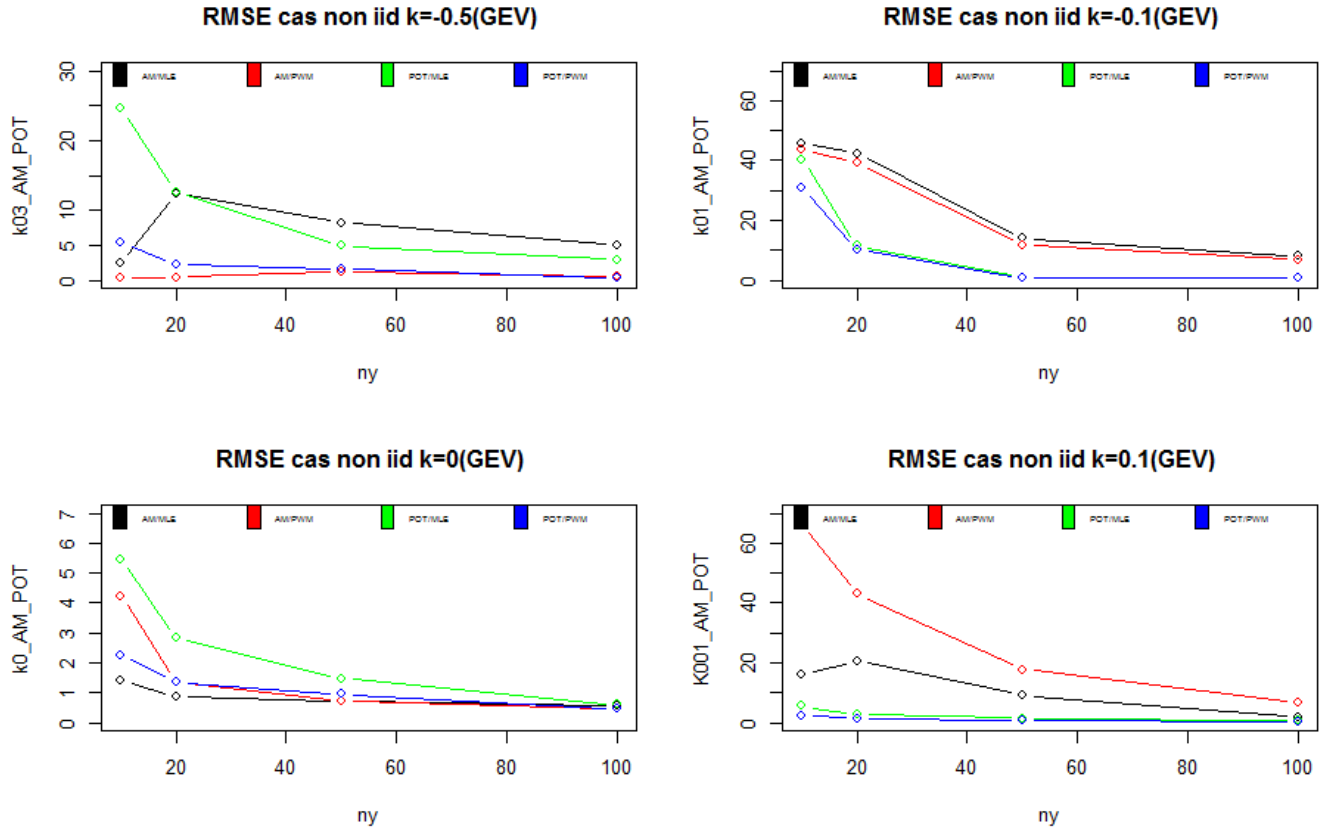

FIGURE 4.23 – Graphiques des RMSE du niveau de retour à 100 ans dans l'analyse du cas non iid pour 100 simulations générées via la loi GEV

4.4.3 Analyse dans le cas des lois Beta I, Gamma et Beta II

Les tableaux 4.6 et 4.7 présentent les résultats des biais et RMSE relatifs des estimateurs MLE et PWM pour le niveau de retour à 100 ans et le paramètre de forme kappa pour 100 simulations générées via les lois gamma, beta I et beta II.
Les conclusions de la section précédente concernant les séries chronologiques avec distributions marginales GEV s'appliquent également ici. Cependant, dans ce cas, l'approche GEV / AM se compare légèrement moins bien avec l'approche POT / GPD. Ceci n'est pas surprenant compte tenu du fait que les maxima annuels d'un processus avec des distributions marginales de GEV avec le même indice de queue sont plus proches d'une distribution GEV limite des extrêmes que les maxima annuels d'un processus avec des distributions marginales gamma ou bêta. C'est en réalité dans le but d'illustrer ce phénomène que nous avons choisi des distributions marginales autres que le GEV.

Méthode AM/GEV						Méthode POT/GPD					
kappa		-0.3	-0.2	-0.1	0	kappa		-0.3	-0.2	-0.1	0
Loi			Beta I		Gam	Loi			Beta I		Gam
ny	est		Biais relatif			ny	est		Biais relatif		
10	mle	-15	-25	-27	-9	10	mle	-18	-22	-22	-8
20	mle	-27	-22	-24	-15	20	mle	-20	-19	-21	-4.7
50	mle	-20	-21	-21	-11	50	mle	-17	-15	-15	-2.3
100	mle	-18	-20	-19	-7	100	mle	-12	-16	-15	-2.5
10	pwm	-18	-23	-22	-11	10	pwm	-10	-13	-12	-4
20	pwm	-24	-19	-21	-12	20	pwm	-12	-13	-10	1.4
50	pwm	-19	-20	-20	-10	50	pwm	-10	-7	-7	3.0
100	pwm	-19	-21	-19	-7	100	pwm	-3	-7	-5	2.6
ny	est		RMSE			ny	est		RMSE		
10	mle	7.84	5.05	6.27	4.87	10	mle	3.79	4.72	4.98	1.57
20	mle	3.65	3.26	3.44	2.88	20	mle	5.22	4.49	4.42	1.11
50	mle	2.28	2.28	2.34	1.61	50	mle	5.27	6	4.42	0.65
100	mle	1.98	2.09	2.07	0.95	100	mle	5.6	8.22	6.23	0.51
10	pwm	3.38	3.45	3.57	2.72	10	pwm	2.74	3.46	3.21	1.65
20	pwm	3.17	2.78	2.79	2.29	20	pwm	3.72	3.42	2.56	0.94
50	pwm	2.16	2.33	2.26	1.49	50	pwm	3.55	3.17	2.53	0.74
100	pwm	2.01	2.19	2	0.93	100	pwm	2.01	3.71	2.29	0.52

TABLE 4.6 – Matrice des biais et RMSE relatifs des estimateurs MLE et PWM pour le paramètre de forme κ pour 100 simulations, dans le cas non i.i.d, générées via les lois Beta I, Gamma et Beta II.

Méthode AM/GEV						Méthode POT/GPD				
quantile		2.49	3.01	3.69	4.6	quantile		2.82	3.56	4.63	6.21
kappa		-0.3	-0.2	-0.1	0	kappa		-0.3	-0.2	-0.1	0
ny	est		Biais relatif			ny	est		Biais relatif		
10	mle	22.7	-11.4	18.1	25.01	10	mle	11.2	7.73	-10.48	11.21
20	mle	20.8	8.6	14.3	13.5	20	mle	9.7	6.1	7.37	10.3
50	mle	17.3	-8.24	9.5	9.4	50	mle	6.3	4.9	4.77	7.21
100	mle	-11.3	7.4	4.4	5.7	100	mle	4.6	2.53	3.52	5.86
10	pwm	16.6	-8.35	-16.11	-22.1	10	pwm	9.2	5.3	9.66	9.6
20	pwm	-10.6	-6.36	11.33	12.5	20	pwm		-5.8	6.5	8.01
50	pwm	9.1	-5.9	-6.96	-7.21	50	pwm	22.3	-3.6	3.1	5.2
100	pwm	6.9	- 4.3	3.49	-5.86	100	pwm	28.3	2.8	2.15	3.86
ny	est		Biais relatif			ny	est		Biais relatif		
10	mle	*	19.42	*	-7.01	10	mle	9.29	11.1	9.82	3.11
20	mle	16.48	-15.98	-13.26	5.81	20	mle	6.02	9.04	7.52	1
50	mle	11.48	5.86	9.16	-2.48	50	mle	4.96	5.3	4.76	0.83
100	mle	-16.49	4.86	6.3	1.29	100	mle	-3.32	3.84	-3.34	-0.61
10	pwm	20.81	16.05	-12.53	-6.01	10	pwm	6.35	8.1	-5.86	-2.61
20	pwm	10.53	-13.03	9.65	3.76	20	pwm	-5.34	6.02	3.76	-1.6
50	pwm	9.53	6.89	5.03	-1.48	50	pwm	3.94	-4.7	-1.64	1.54
100	pwm	5.5	-5.85	- 3.17	-1.1	100	pwm	2.7	2.65	-1.18	-0.98

TABLE 4.7 – Matrice des biais et RMSE relatifs des estimateurs MLE et PWM pour le niveau de retour à 100 ans pour 100 simulations, dans le cas non i.i.d, générées via les lois Beta I et Gamma .

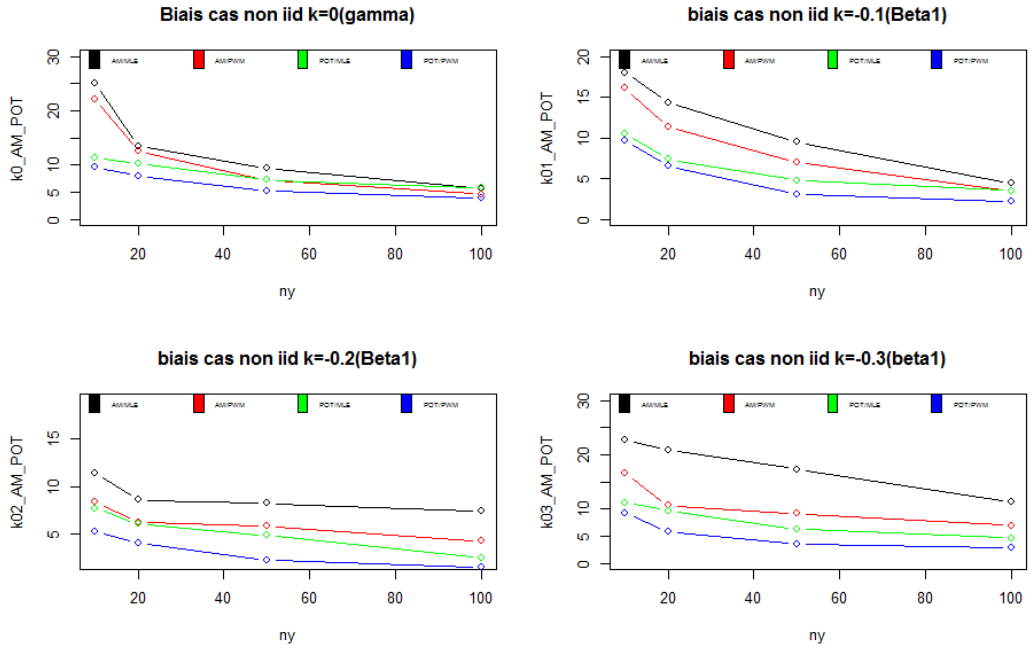

FIGURE 4.24 – Graphiques des biais (en valeur absolue) du niveau de retour à 100 ans dans l'analyse du cas non iid pour 100 simulations générées via les loi gamma et beta

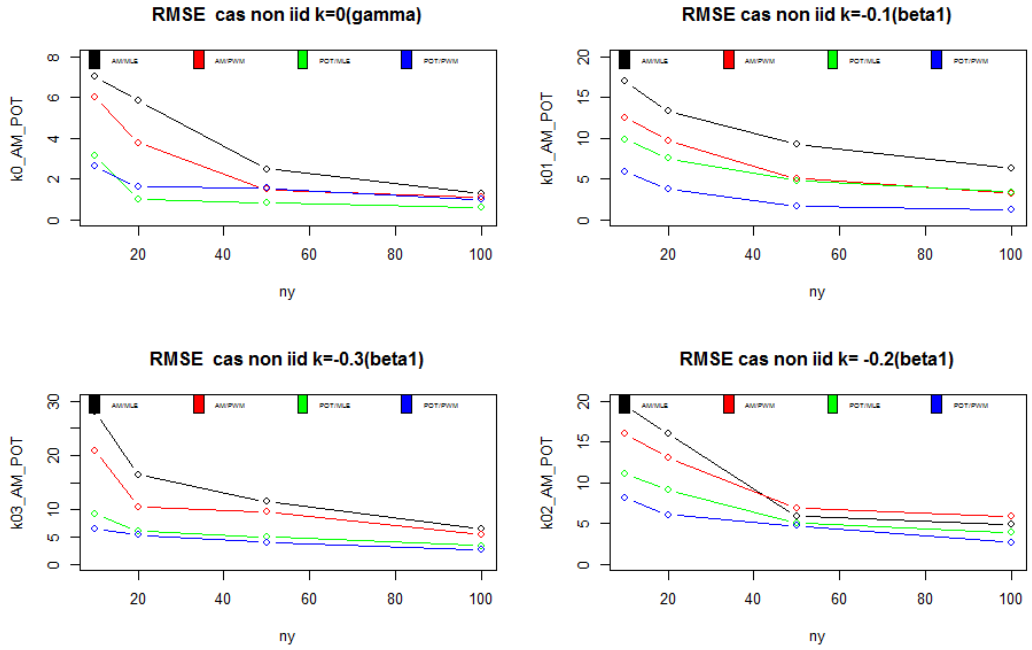

FIGURE 4.25 – Graphiques des RMSE du niveau de retour à 100 ans dans l'analyse du cas non iid générés via les loi gamma et beta

Conclusion et perspectives

L'objectif principal de notre étude était de fournir une comparaison entre deux méthodes d'estimation de quantiles extrêmes (POT et AM), à partir de données fictives ayant des caractéristiques similaires à celles de données environnementales réelles. Nous avons donc dans un premier temps mis au point un modèle, à l'aide de la théorie des séries temporelles et des modèles ARIMA, qui reflète les jeux de données à notre disposition, à savoir des données de vitesses de vent et de hauteurs significatives en Angola et en Bretagne, en y intégrant la dépendance temporelle. A partir de ces modèles, deux études ont été réalisées.

La première portait sur les propriétés intrinsèques des méthodes AM et POT, en analysant le niveau de retour à 100 ans sur des échantillons i.i.d de GEV et de GPD. Nous avons fait varier la longueur des échantillons entre 10 et 100 ans et le paramètre de forme entre -0.3 et 0.1. Cette étude a montré que si les échantillons de GPD sont plus de deux fois plus grands que les échantillons GEV, la méthode POT est la plus efficace. En outre, l'estimateur PWM est le plus performant pour les tailles d'échantillons inférieures à 50 ans et le MLE semble devenir légèrement meilleur pour un nombre plus conséquent d'années, ce que nous avons rarement en pratique dans le cas de données environnementales .

La deuxième étude utilisait le modèle trouvé précédemment afin de comparer les méthodes POT et AM dans le cas de données dépendantes cette fois-ci. Etant donné que les algorithmes étaient éxécutés sur nos ordinateurs, nous n'avons pas pu réaliser plus de 100 simulations de Monte Carlo, ce qui génère une grande incertitude sur la qualité des résultats présentés. Pour une plus grande précision, il faudrait pouvoir augmenter ce nombre de simulations, en utilisant par exemple des ordinateurs plus puissants. Néanmoins, les résultats que l'on a obtenus semblent indiquer que les conclusions dans le cas de données dépendantes sont similaires au cas iid, avec des erreurs bien plus importantes. Notons de plus que le choix du seuil dans la méthode POT a été choisi de manière automatique. En étudiant le graphique de la durée de vie résiduelle, un meilleur choix de seuil peut être effectué, ce qui devrait améliorer sensiblement les performances de la méthode POT. La référence 10 illustre une méthode d'automatisation de ce choix de seuil.

Nos conclusions générales sont alors les suivantes : quelque soit la taille de l'échantillon de données environnementales (de vitesse de vent ou de hauteur significative) à disposition, la méthode POT/GPD, combinée à l'estimateur PWM, est globalement la plus performante

pour le calcul du niveau de retour à 100 ans.

Annexe A

Description des lois choisies pour la modélisation de nos données partie 4.2

A.1 Loi GEV

Cette loi a déjà été définie et étudiée dans une partie précédente. Avec cette méthode, nous attendons de meilleurs résultats qu'avec les lois suivantes. Cela vient du fait que notre loi se trouve déjà dans le domaine d'attraction considéré et l'approximation de la loi GEV limite sera alors plus précise.

Après avoir choisi le paramètre de forme κ, pour estimer les paramètres de position μ et de dispersion σ, nous utilisons la méthode des moments. Rappelons bien que l'on définit une loi GEV pour chaque mois, avec les paramètres précédents reliés à la moyenne et l'écart type de chaque mois. En reprenant les moyennes m_i et écarts types S_i du tableau de la partie précédente pour chaque mois i,$1 \leq i \leq 12$, on peut trouver les paramètres de la loi GEV correspondants. Soit μ_i et σ_i ces paramètres, et κ fixé au préalable. Appliquons la méthode des moments à $X_i \sim GEV(\mu_i, \sigma_i, \kappa)$ de sorte à ce que X est une moyenne m_i et un écart-type S_i : On trouve les relations suivantes :

$\sigma_i = \sqrt{\frac{S_i.\kappa^2}{\Gamma(1-2\kappa)-2\Gamma(1-\kappa)}}$

$\mu_i = m_i - \frac{\sigma_i}{\kappa}\Gamma(1-\kappa)$

Enfin, à partir de U_t simulé précédemment, on génère notre modèle final via : $V_t = F_i^{-1}(U_t; \kappa, \mu, \sigma)$ où F_i est la fonction de répartition de la loi GEV correspondant au mois dans lequel on se trouve.

A.2 Loi Beta I

La loi Beta I la plus répandue dans la littérature se trouve sous la forme suivante :

$$X \sim BetaI(a,b) \Leftrightarrow f(x;\alpha,\beta) = \begin{cases} \dfrac{x^{\alpha-1}(1-x)^{\beta-1}}{\int_0^1 u^{\alpha-1}(1-u)^{\beta-1}\,du} & \text{pour } x \in [0,1] \\ 0 & \text{sinon} \end{cases}$$

$= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\, x^{\alpha-1}(1-x)^{\beta-1}\ \mathbb{1}_{[0,1]}(x) = \frac{1}{\mathrm{B}(\alpha,\beta)}\, x^{\alpha-1}(1-x)^{\beta-1}\ \mathbb{1}_{[0,1]}(x)$ avec $\mathrm{B}(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} = \int_0^1 y^{\alpha-1}(1-y)^{\beta-1}dy$ la fonction Beta.
α est alors un paramètre d'échelle et β un paramètre de forme.
On a alors la fonction de répartition suivante : $F(x;\alpha,\beta) = \frac{\mathrm{B}_x(\alpha,\beta)}{\mathrm{B}(\alpha,\beta)} = I_x(\alpha,\beta)$
où $\mathrm{B}_x(\alpha,\beta)$ est la fonction bêta incomplète et $I_x(\alpha,\beta)$ est la fonction bêta incomplète régularisée.
On peut adapter cette loi, et ajouter un nouveau paramètre de forme $\kappa < 0$ qui correspond au paramètre de forme de la GEV et de la GPD limite : $f(x;\alpha,\beta,\kappa) = \frac{-\kappa\alpha(-\kappa\alpha x)^{\beta-1}(1+\kappa\alpha x)^{\frac{1}{\kappa}-1}}{B(-\kappa^{-1},\beta)}\mathbb{I}_{[0,-(\kappa\alpha)^{-1}]}$
On peut remarquer deux cas particuliers intéressants :
- lorsque $t \to -(\alpha\kappa)^{-1}$, la fonction de répartition tend vers une loi GPD avec paramètre de forme $\kappa < 0$.
-lorsque $\kappa \to 0$, la densité de la loi beta I tend vers la densité de la loi gamma. Ainsi la GPD est approximativement une distribution exponentielle et la GEV est approximativement une distribution de Gumbel lorsque κ est proche de 0.
Appliquons à présent la méthode des moments, en imposant comme moyenne et variance m_i et S_i pour le mois i. On obtient après calcul :

$$\beta_i = \frac{1}{2}\left\{(\frac{1}{\kappa}-1) + \sqrt{(\frac{1}{\kappa}-1)^2 - \frac{4M_i^2}{\kappa S_i^2}}\right\}$$

$$\alpha_i = -\frac{\beta_i}{\kappa(\beta_i-\kappa^{-1})}M_i$$

Ainsi, la variable uniforme U_t générée auparavant tombant dans le mois i, peut-être transformée en loi Beta I, de paramètre de forme κ, de moyenne mensuelle m_i, et d'écart type mensuel S_i par la transformation : $V_t = F_t^{-1}(U_t;\alpha_i,\beta_i,\kappa)$
où F_i est la fonction de répartition de la loi Beta I correspondant au mois dans lequel on se trouve.

Afin de réaliser le calcul numérique de F^{-1}, il est utile de remarquer que : $\forall\ 0 < u < 1$, $F^{-1}(u;\alpha,\beta,\kappa) = (-\kappa\alpha)^{-1}I_u^{-1}(\beta,-\frac{1}{\kappa})$, avec I la fonction bêta incomplète régularisée, définie précédemment. Ainsi, la seule difficulté pour trouver F^{-1} est le calcul de l'inverse de cette fonction I.

A.3 Loi Gamma

Une loi Gamma de paramètres (a,b), avec $a > 0$ et $b > 0$, a une densité de la forme : $\forall x > 0, f_{gamma}(x; a, b) = \frac{a^b x^{b-1} e^{-ax}}{\Gamma(b)}$, où a est le paramètre de forme et b est le paramètre d'intensité.
Si $X \sim Gam(a, b)$, alors on a :
$\mathbb{E}[X] = a/b$
$\mathbb{V}[X] = a/b^2$
Alors on obtient via la méthode des moments, les paramètres a_i et b_i de la loi de chaque mois, en fonction de m_i et S_i (respectivement la moyenne et l'écart-type du mois i considéré) :
$a_i = S_i^2/m_i$
$b_i = m_i^2/S_i^2$
Ainsi, notant F_t la fonction répartition de la loi gamma associée au mois considéré, on génère le modèle final via : $V_t = F_t^{-1}(U_t; a_i, b_i)$

A.4 Loi Beta II

Une autre adaptation de la loi Beta permet d'ajuster un paramètre de forme $\kappa > 0$ pour la loi GEV limite. On obtient alors une distribution appartenant au domaine d'attraction d'une loi de Fréchet (loi de type II). La densité associée est de la forme :
$f(x; \alpha, \beta, \kappa) = \frac{\kappa\alpha(\kappa\alpha x)^{\beta-1}}{B(-\kappa^{-1},\beta)(1+\kappa\alpha x)^{1/\kappa-\beta)}} \mathbb{I}_{(x>0)}$

On peut remarquer deux cas particuliers intéressants :
- lorsque $t \to \infty$, la fonction de répartition tend vers une loi GPD avec paramètre de forme $\kappa > 0$.
-lorsque $\kappa \to 0$, la densité de la loi beta II tend vers la densité de la loi gamma. Ainsi la GPD est approximativement une distribution exponentielle et la GEV est approximativement une distribution de Gumbel lorsque κ est proche de 0.

Appliquons à présent la méthode des moments, en imposant comme moyenne et variance m_i et S_i pour le mois i. On obtient après calcul :

$$\beta_i = \left(\frac{S_i^2+m_i^2}{m_i^2}\frac{1-2\kappa}{1-\kappa} - 1\right)^{-1}$$

$$\alpha_i = -\frac{\beta_i}{m_i(1-\kappa)}$$

Ainsi, la variable uniforme U_t générée auparavant tombant dans le mois i, peut-être transformée en loi Beta II, de paramètre de forme $\kappa > 0$, de moyenne mensuelle m_i, et d'écart type mensuel S_i par la transformation : $V_t = F_t^{-1}(U_t; \alpha_i, \beta_i, \kappa)$

où F_i est la fonction de répartition de la loi Beta II correspondant au mois dans lequel on se trouve.

Afin de réaliser le calcul de F^{-1}, il est utile de remarquer que : $\forall\ 0 < u < 1$, $F^{-1}(u;\alpha,\beta,\kappa) = (\kappa\alpha)^{-1}(\frac{1}{I^{-1}_{1-u}(\frac{1}{\kappa},\beta)} - 1)$, avec I la fonction bêta incomplète régularisée, définie précédemment.

Annexe B

Code R

```
### Packages a charger
library(forecast) ;library(TSA)   ;library(lubridate);library(evd)
library(POT);library(fExtremes);library(extRemes)
library(gPdtest)library(ismev) ;library(evir);library(xtable);library(xlsx)

##Importation des donnees
#importation de la base de donnees data
x=read.csv2('data.csv')  #dataframe 201624 lignes, 9 colonnes (dont le temps)
#importation de la base de donnees era40
z=read.csv2('era40.csv') #dataframe 65744 lignes, 6 colonnes (dont le temps)
z2=z
x$time <- as.POSIXct(as.character(x$time),format="%Y-%m-%d %H:%M:%S")
#conversion du temps
z$time <- as.POSIXct(as.character(z$time),format="%Y-%m-%d %H:%M:%S")

#Boxplot en fonction du mois
xwspd_kmh=x$wspd*3.6
zmonth=z2
zmonth$month=month(z2$time)
xmonth=x
xmonth$month=month(x$time)

#Series temporelles: trouver le modele ARIMA pour ZHS
#   1) transformation des donnees
logZhs=log(z2$hs)
#   2) Desaisonnaliser les donnees
freq=365*4 #frequence pour z: 4:nombre de valeurs par an
```

```
tsZ=ts(logZhs,frequency=freq)
stlZ=stl(tsZ,s.window='periodic') #decomposition de var=trend+seasonal+remainder
strVec=stlZ$time.series
seasonalZ=strVec[,1]  #saisonnalite
trendZ=strVec[,2]     #tendance
remainderZ=strVec[,3] #residus
deseasonalZ=seasadj(stlZ) #equivalent a trend+remainder (serie desaisonnalisee)
lagmax=8760*3 # choix du lagmax pour calculer l'ACF (fonction autocorrelation)
#et PCF (fonction d'autocorrelation partielle)
autoCorr=acf(deseasonalZ,lag.max=lagmax,type='correlation') #acf
pacfZ=acf(deseasonalZ,lag.max=lagmax,type='partial')  #pacf

#test de normalite de deseasonal
qqnorm(deseasonalZ)
qqline(deseasonalZ)

#   4) On ajuste un modele ARMA aux donnees log(z$hs)
#a l'aide de la fonction auto.arima:
nb=nrow(z2)
deseasonalZnb=deseasonalZ[1:nb]
autoArimaZ=auto.arima(deseasonalZnb,d=0,D=0)
#comme la serie est stationnaire on sait que d=0
plot(autoArimaZ$fitted)
lines(deseasonalZ[1:nb],col='red')
pAuto=autoArimaZ$arma[1]  #on recupere p  (trouve par la fonction auto.arima)
qAuto=autoArimaZ$arma[2]  #on recupere q              " "
dAuto=autoArimaZ$arma[3]  #on recupere d              " "
coeffAutoAR=autoArimaZ$coef[1:pAuto]
# coeffs AR(les alpha i) trouves par la fonction auto.arima
coeffAutoMA=autoArimaZ$coef[(pAuto+1):(pAuto+qAuto)]
# coeffs MA(les beta i) trouves par la fonction auto.arima
interceptAuto=tail(autoArimaZ$coef,1)
# coeff trouve par la fonction ARIMA (mu)
# que si d=0 (environ egale a mean(deseasonal)
if (interceptAuto==tail(coeffAutoMA,1)){interceptAuto=0}
sigma2Auto=autoArimaZ$sigma2  # variance du bruit blanc dans le modele ARIMA

#comparaison de nos donnees simulees de log(z$hs) avec log(z$hs)
qqplot(interceptAuto+autoArimaSimZ[1:nrow(z2)]+seasonalZ[1:nrow(z2)],deseasonalZ)
abline(0,1,col='red')
```

```
#plot de toute la serie et du modele sur log(z$hs)
plot(1:nrow(z2),log(z2$hs),col='red',type='l')
autoArimaSimZ2=arima.sim(model=list(ma=coeffAutoMA,ar=coeffAutoAR)
                         ,nrow(z2),sd=sqrt(sigma2Auto))
lines(interceptAuto+autoArimaSimZ2[1:nrow(z2)]
      +seasonalZ[1:nrow(z2)],col='green')
#plot de z$hs (on passe a l'exponentielle)
plot(z2$time,z2$hs,col='red',type='l')
lines(z2$time,exp(interceptAuto+autoArimaSimZ2[1:nrow(z2)]
                  +seasonalZ[1:nrow(z2)]),col='green')
#comparaison de nos donnees simulees de z$hs avec z$hs
qqplot(exp(interceptAuto+autoArimaSimZ2[1:nrow(z2)]
           +seasonalZ[1:nrow(z2)]),z2$hs)
abline(0,1)



## Calcul des moyennes et ecarts type de chaque mois pour X et Z
calculMonthMeanSD_Z=function(data=z$hs){
  z=read.csv2('era40.csv') #dataframe 65744 lignes, 6 colonnes (dont le temps)
  zmonth=z
  zmonth$month=month(z$time)
  meanMonth=NULL
  sdMonth=NULL
  for (i in 1:12){
    meanMonth[i]=mean(data[zmonth$month==i])
    sdMonth[i]=sd(data[zmonth$month==i])
  }
  return(list(mean=meanMonth,sd=sdMonth))
}
mean_sd_zwspd=calculMonthMeanSD_Z(z$wspd)
mean_sd_zwspd$mean;mean_sd_zwspd$sd

calculMonthMeanSD_X=function(data=x$hs){
  x=read.csv2('data.csv')
  xmonth=x
  xmonth$month=month(x$time)
  meanMonth=NULL
  sdMonth=NULL
```

```
  for (i in 1:12){
    meanMonth[i]=mean(data[xmonth$month==i])
    sdMonth[i]=sd(data[xmonth$month==i])
  }
  return(list(mean=meanMonth,sd=sdMonth))
}
mean_sd_xwspd=calculMonthMeanSD_X(x$wspd)
mean_sd_xwspd$mean;mean_sd_xwspd$sd


## Calcul des parametres des lois GEV,Gamma, Beta I et Beta II
#(methode des moments)

#permet de trouver les parametres mu et sigma d'une loi GEV
# a l'aide de la methode des moments (MM)
#a partir de la moyenne et de l'ecart type de notre echantillon
#qu'on souhaite modeliser
#on fixe de plus un kappa (passe en parametre)

#renvoie les parametres mu et sigma de la loi GEV correspondante
#(kappa etant fixe)
findGEVparam=function(mean,sd,kappa){
  if(kappa==0){
    euler=-digamma(1)
    return(list(mu=mean-sd*sqrt(6)*euler/pi,sig=sqrt(6)*sd/pi))
  }
  else{
    muu=mean-sd*gamma(1-kappa)*sqrt(gamma(1-2*kappa)-gamma(1-kappa)^2)
    sigg=sd*kappa*sqrt(gamma(1-2*kappa)-gamma(1-kappa)^2)
    return(list(mu=muu,sig=sigg))
  }
}

#renvoie les parametres alpha et beta de la loi Gamma correspondantes
findGAMparam=function(mean,sd){
  alpha=sd^2/mean
  beta=mean^2/sd^2
  return(list(alpha=alpha,beta=beta))
}
```

```
#renvoie les parametres alpha et beta de la loi Beta I correspondantes
findBeta1param=function(mean=3,sd=0.2,kappa=-0.1){
  beta=0.5*((1/kappa-1)+sqrt((1/kappa-1)^2-4*mean^2/(kappa*sd^2)))
  alpha=-beta/(mean*kappa*(beta-1/kappa))
  return(list(alpha=alpha,beta=beta))
}

#renvoie les parametres alpha et beta de la loi Beta II correspondantes
findBeta2param=function(mean=3,sd=0.2,kappa=0.1){
  beta=- 1 / ( (1-2*kappa)*(mean^2+sd^2)/(mean^2*(1-kappa))-1)
  alpha=beta/(mean*(1-kappa))
  return(list(alpha=alpha,beta=beta))
}

## Trouver un modele de nos donnees: ARMA :#les coeffs du modele ont ete stockes
#au prealable et on ete trouves grace a la fonction auto.arima

#simulationZHS renvoie un echantillon de taille 'annee'
#du modele ARMA correspondant (1460 donnees par an)
simulationZHS = function(annee=1000){
  #importation de la base de donnees era40
  z=read.csv2('era40.csv') #dataframe 65744 lignes, 6 colonnes (dont le temps)
  z$time <- as.POSIXct(as.character(z$time),format="%Y-%m-%d %H:%M:%S")
  z2=z
  logZhs=log(z2$hs)
  freq=365*4 #frequence pour z: nb de valeurs par an
  tsZ=ts(logZhs,frequency=freq)
  #mise sous serie temporelle de la variable consideree(colonne i de z)
  stlZ=stl(tsZ,s.window='periodic')
  #decomposition de var=trend+seasonal+remainder
  strVec=stlZ$time.series
  seasonalZ=strVec[,1]  #saisonnalite
  deseasonalZ=seasadj(stlZ)
  #equivalent a trend+remainder (serie desaisonnalisee)
  nb=annee*365*4
  interceptAutoZHS=0.5350055
  sigma2autoZHS=0.0009665339
  coeffAutoARZHS=c(1.6263951,-0.6664814)
  coeffAutoMAZHS
  =c(-0.17510746, -0.06150521, 0.11867406, 0.14736859, -0.03701617 )
```

```
  autoArimaSimZ
  =arima.sim(model=list(ma=coeffAutoMAZHS
                        ,ar=coeffAutoARZHS),nb,sd=sqrt(sigma2autoZHS))
  #modele de log z$hs desaisonnalise
  seasonalZ1period=seasonalZ[1:(365*4)]
  seasonalZtotal=rep_len(seasonalZ1period,length.out=nb)
  modelFinal=exp(interceptAutoZHS+autoArimaSimZ+seasonalZtotal)
  return(modelFinal)
}


#simulationZWSPD2 renvoie un echantillon de taille
#'annee' du modele trouve de ZWSPD (1460 donnees par an)
simulationZWSPD = function(annee=1000){
  Zwspd=z$wspd
  #   2) Desaisonnaliser les donnees
  freq=365*4 #frequence pour z: nb de valeurs par an
  tsZ=ts(Zwspd,frequency=freq)
  #mise sous serie temporelle de la variable consideree(colonne i de z)
  stlZ=stl(tsZ,s.window='periodic')
  #decomposition de var=trend+seasonal+remainder
  strVec=stlZ$time.series
  seasonalZ=strVec[,1]  #saisonnalite
  deseasonalZ=seasadj(stlZ)
  #equivalent a trend+remainder (serie desaisonnalisee)
  sigma2AutoZWSPD=0.7677717
  coeffAutoARZWSPD=c(0.7841023, 0.1597126, -0.2176757, 0.4682848, -0.3194570)
  coeffAutoMAZWSPD=c(-0.07578547, -0.26929985, 0.21805625, -0.13909350)
  interceptAutoZWSPD=5.814298
  nb=annee*365*4
  autoArimaSimZ
  =arima.sim(model=list(ma=coeffAutoMAZWSPD
                        ,ar=coeffAutoARZWSPD),nb,sd=sqrt(sigma2AutoZWSPD))
  #modele de z$wspd desaisonnalise
  seasonalZ1period=seasonalZ[1:(365*4)]
  seasonalZtotal=rep_len(seasonalZ1period,length.out=nb)
  modelFinal=interceptAutoZWSPD+autoArimaSimZ+seasonalZtotal
  return(modelFinal)
}
```

```
#simulationXWSPD renvoie un echantillon de taille
#'annee' du modele trouve de XWSPD (1460*2 donnees par an)
simulationXWSPD=function(annee=1000){
  seq1=c(TRUE,FALSE,FALSE)
  seqq=rep(seq1,nrow(x)/3)
  x3h=x[seqq,]
  freq=365*4*2 #frequence pour z: nb de valeurs par an
  tsZ=ts(sqrt(x3h$wspd),frequency=freq)
  #mise sous serie temporelle de la variable consideree(colonne i de z)
  stlZ=stl(tsZ,s.window='periodic')
  #decomposition de var=trend+seasonal+remainder
  strVec=stlZ$time.series
  seasonalZ=strVec[,1]  #saisonnalite
  deseasonalZ=seasadj(stlZ)
  #equivalent a trend+remainder (serie desaisonnalisee)
  coeffAutoMAXWSPD
  =c(0.02494598, -0.11808043, -0.13682580, -0.05556777, -0.07097323)
  coeffAutoARXWSPD=c(0.9132143)
  sigma2AutoXWSPD=0.09096195
  interceptAutoXWSPD=2.448135
  nb=annee*365*4*2
  autoArimaSimZ
  =arima.sim(model=list(ma=coeffAutoMAXWSPD
                        ,ar=coeffAutoARXWSPD),nb,sd=sqrt(sigma2AutoXWSPD))
  #modele de z$wspd desaisonnalise
  seasonalZ1period=seasonalZ[1:(365*4*2)]
  seasonalZtotal=rep_len(seasonalZ1period,length.out=nb)
  modelFinal=(interceptAutoXWSPD+autoArimaSimZ+seasonalZtotal)^2
}

## Generation de nos modeles a partir des lois: GEV,Gamma, Beta I, Beta II,
#a partir de Ft^(-1)(Ut) ; Ut=phi(At) ou At est le modele ARMA trouve
#ou Ut est uniforme et tient compte de la dependance temporelle des donnees

#renvoie Ut decrit plus haut, a partir de z$hs
loiUnifZHS = function(annee=1000){
  #importation de la base de donnees era40
  z=read.csv2('era40.csv') #dataframe 65744 lignes, 6 colonnes (dont le temps)
  z$time <- as.POSIXct(as.character(z$time),format="%Y-%m-%d %H:%M:%S")
  z2=z
```

```
  logZhs=log(z2$hs)
  freq=365*4 #frequence pour z: nb de valeurs par an
  tsZ=ts(logZhs,frequency=freq)
  #mise sous serie temporelle de la variable consideree(colonne i de z)
  stlZ=stl(tsZ,s.window='periodic')
  #decomposition de var=trend+seasonal+remainder
  strVec=stlZ$time.series
  seasonalZ=strVec[,1]
  #saisonnalite
  deseasonalZ=seasadj(stlZ)
  #equivalent a trend+remainder (serie desaisonnalisee)
  nb=annee*365*4
  interceptAutoZHS=0.5350055
  sigma2autoZHS=0.0009665339
  coeffAutoARZHS=c(1.6263951,-0.6664814)
  coeffAutoMAZHS=c(-0.17510746
                   , -0.06150521, 0.11867406, 0.14736859, -0.03701617 )
  autoArimaSimZ=arima.sim(model=list(ma=coeffAutoMAZHS
                                     ,ar=coeffAutoARZHS),nb
                          ,sd=sqrt(sigma2autoZHS))
  #modele de log z$hs desaisonnalise
  seasonalZ1period=seasonalZ[1:(365*4)]
  seasonalZtotal=rep_len(seasonalZ1period,length.out=nb)
  modelFinal=exp(interceptAutoZHS+autoArimaSimZ+seasonalZtotal)
  return(pnorm(autoArimaSimZ/sd(autoArimaSimZ)))
}

#renvoie le quantile d'ordre p d'une loi GEV
inverse_FR_gev=function(p=0.5,mu=0,sigma=1,kappa=0){
  return(calculQ_theoriqueGEV(mu,sigma,kappa,1/(1-p)))
}

#renvoie le quantile d'ordre p d'une loi Gamma
inverse_FR_gamma=function(p=0.5,alpha=0.5,beta=1){
  return(qgamma(p=p,shape=beta,scale=alpha))
}

#renvoie le quantile d'ordre p d'une loi Beta I
inverse_FR_beta1=function(p=0.5,alpha=1,beta=1,kappa=-0.1){
  ll=Rbeta.inv(p,beta,-1/kappa)
```

```
  res=1/(-kappa*alpha)*ll
  return(res)
}

#renvoie le quantile d'ordre p d'une loi Beta II
inverse_FR_beta2=function(p=0.5,alpha=1,beta=1,kappa=0.1){
  ll=Rbeta.inv(1-p,1/kappa,beta)
  return(1/(alpha*kappa)*(1/ll-1))
}

#renvoie une sequence de dates sur nbAnnees, espacees toutes les 6h (comme z)
generation_seq_date_Z=function(nbAnnees=10){
  date1=as.POSIXct(z$time[1],"%Y-%m-%d %H")
  seqq=seq(from=date1,by='6 hour',length.out = (nbAnnees*365*4))
  return(seqq)
}

#renvoie le modele final a partir de la loi GEV
generation_GEV_dependante=function(data=z$hs,nbAnnees=10,kappa=0){
  Ut=loiUnifZHS(nbAnnees)
  seqDate=generation_seq_date_Z(nbAnnees)
  vecFinal=rep(0,nbAnnees*1460)
  mean_sdZ=calculMonthMeanSD_Z(data)
  meanZ=mean_sdZ$mean
  sdZ=mean_sdZ$sd
  seqDate=generation_seq_date_Z(nbAnnees)
  for (mois in 1:12){
    param_gev=findGEVparam(meanZ[mois],sdZ[mois],kappa)
    mu_mois=param_gev$mu
    sig_mois=param_gev$sig
    Ut_mois=Ut[month(seqDate)==mois]
    Vt=inverse_FR_gev(Ut_mois,mu_mois,sig_mois,kappa)
    vecFinal[month(seqDate)==mois]=Vt
  }
  return(vecFinal)
}

#renvoie le modele final a partir de la loi Gamma
generation_gamma_dependante=function(data=z$hs,nbAnnees=10){
  Ut=loiUnifZHS(nbAnnees)
```

```
  seqDate=generation_seq_date_Z(nbAnnees)
  vecFinal=rep(0,nbAnnees*1460)
  mean_sdZ=calculMonthMeanSD_Z(data)
  meanZ=mean_sdZ$mean
  sdZ=mean_sdZ$sd
  seqDate=generation_seq_date_Z(nbAnnees)
  for (mois in 1:12){
    param_gamma=findGAMparam(meanZ[mois],sdZ[mois])
    alpha_mois=param_gamma$alpha
    beta_mois=param_gamma$beta
    Ut_mois=Ut[month(seqDate)==mois]
    Vt=inverse_FR_gamma(Ut_mois,alpha_mois,beta_mois)
    vecFinal[month(seqDate)==mois]=Vt
  }
  return(vecFinal)
}

#renvoie le modele final a partir de la loi Beta I
generation_beta1_dependante=function(data=z$hs,nbAnnees=10,kappa=-0.1){
  Ut=loiUnifZHS(nbAnnees)
  vecFinal=rep(0,nbAnnees*1460)
  mean_sdZ=calculMonthMeanSD_Z(data)
  meanZ=mean_sdZ$mean
  sdZ=mean_sdZ$sd
  seqDate=generation_seq_date_Z(nbAnnees)
  for (mois in 1:12){
    param_gamma=findBeta1param(meanZ[mois],sdZ[mois],kappa)
    alpha_mois=param_gamma$alpha
    beta_mois=param_gamma$beta
    Ut_mois=Ut[month(seqDate)==mois]
    Vt=inverse_FR_beta1(Ut_mois,alpha_mois,beta_mois,kappa)
    vecFinal[month(seqDate)==mois]=Vt
  }
  return(vecFinal)
}

#renvoie le modele final a partir de la loi Beta II
generation_beta2_dependante=function(data=z$hs,nbAnnees=10,kappa=0.1){
  Ut=loiUnifZHS(nbAnnees)
  vecFinal=rep(0,nbAnnees*1460)
```

```
  mean_sdZ=calculMonthMeanSD_Z(data)
  meanZ=mean_sdZ$mean
  sdZ=mean_sdZ$sd
  seqDate=generation_seq_date_Z(nbAnnees)
  for (mois in 1:12){
    param_gamma=findBeta2param(meanZ[mois],sdZ[mois],kappa)
    alpha_mois=param_gamma$alpha
    beta_mois=param_gamma$beta
    Ut_mois=Ut[month(seqDate)==mois]
    Vt=inverse_FR_beta2(Ut_mois,alpha_mois,beta_mois,kappa)
    vecFinal[month(seqDate)==mois]=Vt
  }
  return(vecFinal)
}

## Fonctions utiles pour le calcul de la matrice de biais / RMSE

#genere un echantillon i.i.d de taille nb d'une loi GEV
generateurGEV=function(nb=10,mu=0,sigma=1,kappa=0){
  return(revd(n=nb,loc=mu,scale=sigma,shape=kappa))
}

#genere un echantillon i.i.d de taille nb d'une loi GPD
generateurGPD=function(nb=10,sigma=1,kappa=0){
  return(revd(n=nb,loc=0,scale=sigma,shape=kappa,type='GP'))
}

#calcul du niveau de retour theorique d'une loi GEV
calculQ_theoriqueGEV=function(mu=0,sigma=1,kappa=0,returnLevel=100){
  aa=-log(1-1/returnLevel)
  if (kappa!=0){
    bb=1-aa^(-kappa)
    return(mu-sigma/kappa*bb)
  }
  if(kappa==0){
    return(mu-sigma*log(aa))
  }
}

#calcul du niveau de retour theorique d'une loi GPD
```

```
calculQ_theoriqueGPD=function(seuil=0,sigma=1,kappa=0,lambda=5,returnLevel=100){
  if (kappa!=0){
    return(seuil +sigma/kappa*((lambda*returnLevel)^kappa-1))
  }
  if (kappa==0){
    return(seuil+sigma*log(lambda*returnLevel))
  }
}

#permet a partir d'une matrice de sigma,quantile,kappa observes,
#de renvoyer le biais et RMSE par rapport aux valeurs theoriques
biais_RMSE=function(matrixx,qTheorique=3.970358
                    ,sigmaTheorique=0.1839776 ,kapTheorique=-0.03340718){
  nm=nrow(matrixx)
  biaisQ=mean(matrixx[,1]-qTheorique)
  relativeBiaisQ=biaisQ/qTheorique
  rmseQ=1/sqrt(nm)*sqrt(mean((matrixx[,1]-qTheorique)^2))
  relativeRmseQ=rmseQ/qTheorique # a verifier !!!
  biaisS=mean(matrixx[,2]-sigmaTheorique)
  relativeBiaisS=biaisS/sigmaTheorique
  rmseS=1/sqrt(nm)*sqrt(mean((matrixx[,2]-sigmaTheorique)^2))
  relativeRmseS=rmseS/sigmaTheorique
  biaisK=mean(matrixx[,3]-kapTheorique)
  relativeBiaisK=biaisK/kapTheorique
  rmseK=1/sqrt(nm)*sqrt(mean((matrixx[,3]-kapTheorique)^2))
  relativeRmseK=rmseK/kapTheorique
  return(100*data.frame(relativeBiaisQ,relativeRmseQ,relativeBiaisS,relativeRmseS,rel
}

##Methode de MC: calcul de la matrice de biais/RMSE

# 1) Cas i.i.d

#renvoie les biais/RMSE correpondants a des realisations i.i.d de GEV et GPD
simulation_iid_pot_am=function(nbMC=3,ny=100,lambda=5
                               ,sig=1,mu=0,kap=0,returnLevel=100){
  qtheoriqueGEV
  =calculQ_theoriqueGEV(mu=mu,sigma=sig,kappa=kap,returnLevel=returnLevel)
  qtheoriqueGPD
  =calculQ_theoriqueGPD(seuil=0
```

```
                      ,sigma=sig,kappa=kap
                      ,lambda=lambda,returnLevel = returnLevel)
#AM/GEV
#mle
listQuantiles_mle=NULL;listSig_mle=NULL;listKap_mle=NULL
#pwm
listQuantiles_pwm=NULL;listSig_pwm=NULL;listKap_pwm=NULL;

#POT/GPD
#mle
listQuantiles_mle_pot=NULL;listSig_mle_pot=NULL;listKap_mle_pot=NULL
#pwm
listQuantiles_pwm_pot=NULL;listSig_pwm_pot=NULL;listKap_pwm_pot=NULL
G=(kap==0)*1
#coeff pour eviter division par 0 pour calcul biais relatif et RMSE
k1=0;k2=0;k3=0;k4=0;
nbErr1=0;nbErr2=0;nbErr3=0;nbErr4=0;
matriceErreur1=NULL
for (i in 1:nbMC){
  #simulation d'une echantillon de taille ny d'une loi GEV(mu,sig,kapp)
  maxAnnuels=generateurGEV(nb=ny,mu=mu,sigma=sig,kappa = kap)
  #fit GEV avec method = MLE
  fitGEV_mle=try(gevFit(maxAnnuels,type='mle'),T)
  if (class(fitGEV_mle)=='try-error'){
    nbErr1=nbErr1+1
  }
  else{
    k1=k1+1
    parametersGEV_mle=fitGEV_mle@fit$par.ests
    mu_GEV_mle=parametersGEV_mle[2]
    sigGEV_mle=parametersGEV_mle[3];kapGEV_mle=parametersGEV_mle[1]
    #qGEV_mle=calculQuantileGEV(mu_GEV_mle,sigGEV_mle,kapGEV_mle,returnLevel)
    qGEV_mle
    =calculQ_theoriqueGEV(mu=mu_GEV_mle,sigma = sigGEV_mle
                          ,kap=kapGEV_mle,returnLevel=returnLevel)
    #qGEV_mle=gevrlevelPlot(fitGEV_mle,kBlocks=returnLevel)$v
    listQuantiles_mle[k1]=qGEV_mle
    listSig_mle[k1]=sigGEV_mle
    listKap_mle[k1]=kapGEV_mle+G
  }
```

```
#fit GEV avec method = PWM
fitGEV_pwm=try(gevFit(maxAnnuels,type='pwm'),T)
if (class(fitGEV_pwm)=='try-error'){
  nbErr2=nbErr2+1
}
else{
  k2=k2+1
  parametersGEV_pwm=fitGEV_pwm@fit$par.ests
  mu_GEV_pwm=parametersGEV_pwm[2];sigGEV_pwm=parametersGEV_pwm[3]
  kapGEV_pwm=parametersGEV_pwm[1]
  qGEV_pwm
  =calculQ_theoriqueGEV(mu=mu_GEV_pwm
                        ,sig = sigGEV_pwm
                        ,kap=kapGEV_pwm,returnLevel=returnLevel)
  listQuantiles_pwm[k2]=qGEV_pwm
  listSig_pwm[k2]=sigGEV_pwm
  listKap_pwm[k2]=kapGEV_pwm+G
}

#simulation d'une echantillon de taille ny d'une loi GPD(sig,kapp)
simulationGPD=generateurGPD(nb=lambda*ny,sigma=sig,kappa = kap)
#fit GPD avec method = MLE
fitGPD_mle_pot=try(gpdFit(simulationGPD,u=0,type='mle'),T)
if (class(fitGPD_mle_pot)=='try-error'){
  nbErr3=nbErr3+1
}
else{
  k3=k3+1
  parametersGPD_mle_pot=fitGPD_mle_pot@fit$par.ests
  sigGPD_mle_pot=parametersGPD_mle_pot[2]
  kapGPD_mle_pot=parametersGPD_mle_pot[1]
  qGPD_mle_pot
  =calculQ_theoriqueGPD(seuil=0,sigma=sigGPD_mle_pot
                        ,lambda=lambda
                        ,kappa=kapGPD_mle_pot,returnLevel = returnLevel)
  listQuantiles_mle_pot[k3]=qGPD_mle_pot
  listSig_mle_pot[k3]=sigGPD_mle_pot
  listKap_mle_pot[k3]=kapGPD_mle_pot+G
}
#fit GPD avec method = PWM
```

```
  fitGPD_pwm_pot=try(gpdFit(simulationGPD,u=0,type='pwm'),T)
  if (class(fitGPD_pwm_pot)=='try-error'){
    nbErr4=nbErr4+1
  }
  else{
    k4=k4+1
    parametersGPD_pwm_pot=fitGPD_pwm_pot@fit$par.ests
    sigGPD_pwm_pot=parametersGPD_pwm_pot[2]
    kapGPD_pwm_pot=parametersGPD_pwm_pot[1]
    qGPD_pwm_pot
    =calculQ_theoriqueGPD(seuil=0
                          ,sigma=sigGPD_pwm_pot
                          ,lambda=lambda,kappa=kapGPD_pwm_pot
                          ,returnLevel = returnLevel)
    listQuantiles_pwm_pot[k4]=qGPD_pwm_pot
    listSig_pwm_pot[k4]=sigGPD_pwm_pot
    listKap_pwm_pot[k4]=kapGPD_pwm_pot+G
  }
}
matrice_mle=matrix(c(listQuantiles_mle,listSig_mle,listKap_mle)
                   ,nrow=nbMC-nbErr1)
biais_rmse_gev_mle=biais_RMSE(matrice_mle
                              ,qTheorique =qtheoriqueGEV
                              ,sigmaTheorique = sig, kapTheorique = kap+G)
matrice_pwm=matrix(c(listQuantiles_pwm
                     ,listSig_pwm,listKap_pwm),nrow=nbMC-nbErr2)
biais_rmse_gev_pwm=biais_RMSE(matrice_pwm
                              ,qTheorique =qtheoriqueGEV
                              ,sigmaTheorique = sig, kapTheorique = kap+G)
matrice_mle_pot=matrix(c(listQuantiles_mle_pot
                         ,listSig_mle_pot,listKap_mle_pot),nrow=nbMC-nbErr3)
biais_rmse_gpd_mle_pot=biais_RMSE(matrice_mle_pot
                                  ,qTheorique =qtheoriqueGPD,sigmaTheorique = sig
                                  , kapTheorique = kap+G)
matrice_pwm_pot=matrix(c(listQuantiles_pwm_pot
                         ,listSig_pwm_pot,listKap_pwm_pot),nrow=nbMC-nbErr4)
biais_rmse_gpd_pwm_pot=biais_RMSE(matrice_pwm_pot
                                  ,qTheorique =qtheoriqueGPD,sigmaTheorique = sig
                                  , kapTheorique = kap+G)
```

```
  return(list(mleGEV=biais_rmse_gev_mle
              ,pwmGEV=biais_rmse_gev_pwm
              ,mleGPD=biais_rmse_gpd_mle_pot
              ,pwmGPD=biais_rmse_gpd_pwm_pot
              ,err=c(nbErr1/nbMC*100,nbErr2/nbMC*100
                     ,nbErr3/nbMC*100,nbErr4/nbMC*100)))
}

#renvoie la matrice de biais/RMSE propre, exactement sous la forme souhaitee
constructionMatriceBiaisRmse_iid_zhs=function(
  nbMC=100,range_ny=c(10,20,50,100,200),rangeKap=c(-0.3,-0.2,-0.1,0,0.1)){
  #Partie tableau quantiles
  n1=length(range_ny)
  matBiaisGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
  matBiaisGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
  matRMSEGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
  matRMSEGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
  errGEV=matrix(0,nrow=n1,ncol=length(rangeKap))
  errGPD=matrix(0,nrow=n1,ncol=length(rangeKap))
  j=0
  for (kappa in rangeKap){
    k=0
    j=j+1
    for (ny in range_ny){
      k=k+1
      simu=simulation_iid_pot_am(nbMC=nbMC,ny=ny)
      biais_gev=simu$mleGEV$relativeBiaisQ
      biais_gpd=simu$mleGPD$relativeBiaisQ
      rmse_gev=simu$mleGEV$relativeRmseQ
      rmse_gpd=simu$mleGPD$relativeRmseQ
      matBiaisGEV[k,j]=round(biais_gev,2)
      matBiaisGPD[k,j]=round(biais_gpd,2)
      matRMSEGPD[k,j]=round(rmse_gpd,2)
      matRMSEGEV[k,j]=round(rmse_gev,2)
      simu=simulation_iid_pot_am(nbMC=nbMC,ny=ny)
      biais_gev=simu$pwmGEV$relativeBiaisQ
      biais_gpd=simu$pwmGPD$relativeBiaisQ
      rmse_gev=simu$pwmGEV$relativeRmseQ
      rmse_gpd=simu$pwmGPD$relativeRmseQ
      matBiaisGEV[k+n1,j]=round(biais_gev,2)
```

```
    matBiaisGPD[k+n1,j]=round(biais_gpd,2)
    matRMSEGPD[k+n1,j]=round(rmse_gpd,2)
    matRMSEGEV[k+n1,j]=round(rmse_gev,2)
    errGEV[k,j]=simu$err[1]
    errGPD[k,j]=simu$err[3]
    #print(rmse_gpd)
  }
}
col1=c(range_ny,range_ny)
col2=c(rep('mle',n1),rep('pwm',n1))
col1
Biais=c('ny','est',rep('',length(rangeKap)))

aa=cbind(col1,col2,matBiaisGEV)
mat1=rbind(Biais,aa)

aa=cbind(col1,col2,matBiaisGPD)
mat2=rbind(Biais,aa)

matt=cbind(mat1,mat2)


RMSE=c('ny','est',rep('',length(rangeKap)))

aa=cbind(col1,col2,matRMSEGEV)
mat3=rbind(RMSE,aa)

aa=cbind(col1,col2,matRMSEGPD)
mat4=rbind(RMSE,aa)

matt2=cbind(mat3,mat4)
matFinal=rbind(matt,matt2)

l1=c("","",rep('GEV',length(rangeKap)),'','',rep('GPD',length(rangeKap)))
qtheoriqueGEV=round(calculQ_theoriqueGEV(kappa = rangeKap),2)
qtheoriqueGPD=round(calculQ_theoriqueGPD(kappa = rangeKap),2)
if(is.element(0,rangeKap)){
  ind=which(rangeKap==0)
  qtheoriqueGEV[ind]=round(calculQ_theoriqueGEV(kappa=0),2)
  qtheoriqueGPD[ind]=round(calculQ_theoriqueGPD(kappa=0),2)
```

```
}
l2=c('quantile','quantile',qtheoriqueGEV,"quantile",'quantile',qtheoriqueGPD)
l3=c('kappa','kappa',rangeKap,'kappa','kappa',rangeKap)
matFinal=rbind(l1,l2,l3,matFinal)

#partie tableau kappa
matBiaisGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matBiaisGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
j=0
for (kappa in rangeKap){
  k=0
  j=j+1
  for (ny in range_ny){
    k=k+1
    simu=simulation_iid_pot_am(nbMC=nbMC,ny=ny)
    biais_gev=simu$mleGEV$relativeBiaisK
    biais_gpd=simu$mleGPD$relativeBiaisK
    rmse_gev=simu$mleGEV$relativeRmseK
    rmse_gpd=simu$mleGPD$relativeRmseK
    matBiaisGEV[k,j]=round(biais_gev,2)
    matBiaisGPD[k,j]=round(biais_gpd,2)
    matRMSEGPD[k,j]=round(rmse_gpd,2)
    matRMSEGEV[k,j]=round(rmse_gev,2)
    simu=simulation_iid_pot_am(nbMC=nbMC,ny=ny)
    biais_gev=simu$pwmGEV$relativeBiaisK
    biais_gpd=simu$pwmGPD$relativeBiaisK
    rmse_gev=simu$pwmGEV$relativeRmseK
    rmse_gpd=simu$pwmGPD$relativeRmseK
    matBiaisGEV[k+n1,j]=round(biais_gev,2)
    matBiaisGPD[k+n1,j]=round(biais_gpd,2)
    matRMSEGPD[k+n1,j]=round(rmse_gpd,2)
    matRMSEGEV[k+n1,j]=round(rmse_gev,2)
    #print(rmse_gpd)
  }
}
col1=c(range_ny,range_ny)
col2=c(rep('mle',n1),rep('pwm',n1))
col1
```

```
Biais=c('ny','est',rep('',length(rangeKap)))

aa=cbind(col1,col2,matBiaisGEV)
mat1=rbind(Biais,aa)

aa=cbind(col1,col2,matBiaisGPD)
mat2=rbind(Biais,aa)

matt=cbind(mat1,mat2)


RMSE=c('ny','est',rep('',length(rangeKap)))

aa=cbind(col1,col2,matRMSEGEV)
mat3=rbind(RMSE,aa)

aa=cbind(col1,col2,matRMSEGPD)
mat4=rbind(RMSE,aa)

matt2=cbind(mat3,mat4)
matFinal2=rbind(matt,matt2)

l1=c("","",rep('GEV',length(rangeKap)),'','',rep('GPD',length(rangeKap)))

l3=c('kappa','kappa',rangeKap,'kappa','kappa',rangeKap)
matFinal2=rbind(l1,l3,matFinal2)

#Partie tableau sigma
matBiaisGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matBiaisGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
j=0
for (kappa in rangeKap){
  k=0
  j=j+1
  for (ny in range_ny){
    k=k+1
    simu=simulation_iid_pot_am(nbMC=nbMC,ny=ny)
    biais_gev=simu$mleGEV$relativeBiaisS
```

```
    biais_gpd=simu$mleGPD$relativeBiaisS
    rmse_gev=simu$mleGEV$relativeRmseS
    rmse_gpd=simu$mleGPD$relativeRmseS
    matBiaisGEV[k,j]=round(biais_gev,2)
    matBiaisGPD[k,j]=round(biais_gpd,2)
    matRMSEGPD[k,j]=round(rmse_gpd,2)
    matRMSEGEV[k,j]=round(rmse_gev,2)
    simu=simulation_iid_pot_am(nbMC=nbMC,ny=ny)
    biais_gev=simu$pwmGEV$relativeBiaisS
    biais_gpd=simu$pwmGPD$relativeBiaisS
    rmse_gev=simu$pwmGEV$relativeRmseS
    rmse_gpd=simu$pwmGPD$relativeRmseS
    matBiaisGEV[k+n1,j]=round(biais_gev,2)
    matBiaisGPD[k+n1,j]=round(biais_gpd,2)
    matRMSEGPD[k+n1,j]=round(rmse_gpd,2)
    matRMSEGEV[k+n1,j]=round(rmse_gev,2)
    #print(rmse_gpd)
  }
}
col1=c(range_ny,range_ny)
col2=c(rep('mle',n1),rep('pwm',n1))
col1
Biais=c('ny','est',rep('',length(rangeKap)))

aa=cbind(col1,col2,matBiaisGEV)
mat1=rbind(Biais,aa)

aa=cbind(col1,col2,matBiaisGPD)
mat2=rbind(Biais,aa)

matt=cbind(mat1,mat2)


RMSE=c('ny','est',rep('',length(rangeKap)))

aa=cbind(col1,col2,matRMSEGEV)
mat3=rbind(RMSE,aa)

aa=cbind(col1,col2,matRMSEGPD)
mat4=rbind(RMSE,aa)
```

```
  matt2=cbind(mat3,mat4)
  matFinal3=rbind(matt,matt2)
  l1=c("","",rep('GEV',length(rangeKap)),'','',rep('GPD',length(rangeKap)))
  l2=c('sigma','sigma',rep(1,length(rangeKap))
       ,'sigma','sigma',rep(1,length(rangeKap)))
  l3=c('kappa','kappa',rangeKap,'kappa','kappa',rangeKap)
  matFinal3=rbind(l1,l3,matFinal3)
  listF=NULL
  listF$matQ=matFinal
  listF$matK=matFinal2
  listF$matS=matFinal3
  return(listF)
}

# 2) Cas non i.i.d
simulation_non_iid_non_stationnaire_pot_am_beta2=
  function(data=z$hs,nbMC=3,ny=100,kap=0.1,returnLevel=100){
  bcpAnnees=1000 #objectif: augmenter ce nombre pour obtenir valeurs theoriques
  simu=generation_beta2_dependante(data,bcpAnnees,kap)
  mu_sig_kap_q=quantileAM(simu,bcpAnnees,returnLevel,'mle')
  sig=mu_sig_kap_q$sigma
  qtheorique=mu_sig_kap_q$quantile
  #AM/GEV
  #mle
  listQuantiles_mle=NULL;listSig_mle=NULL;listKap_mle=NULL
  #pwm
  listQuantiles_pwm=NULL;listSig_pwm=NULL;listKap_pwm=NULL;

  #POT/GPD
  #mle
  listQuantiles_mle_pot=NULL;listSig_mle_pot=NULL;listKap_mle_pot=NULL
  #pwm
  listQuantiles_pwm_pot=NULL;listSig_pwm_pot=NULL;listKap_pwm_pot=NULL
  kap=0
  G=(kap==0)
  #coeff pour eviter division par 0 pour calcul biais relatif et RMSE
  k1=0;k2=0;k3=0;k4=0;
  nbErr1=0;nbErr2=0;nbErr3=0;nbErr4=0;
  matriceErreur1=NULL
```

```
for (i in 1:nbMC){
  #simulation d'une echantillon de taille ny d'une loi GEV(mu,sig,kapp)
  simu=generation_beta2_dependante(data,ny,kappa)
  maxAnnuels=maxBlocs(simu,1460)
  #maxAnnuels=generateurGEV(nb=lambda*ny,mu=mu,sigma=sig,kappa = kap)
  #fit GEV avec method = MLE
  fitGEV_mle=try(gevFit(maxAnnuels,type='mle'),T)
  if (class(fitGEV_mle)=='try-error'){
    nbErr1=nbErr1+1
  }
  else{
    k1=k1+1
    parametersGEV_mle=fitGEV_mle@fit$par.ests
    mu_GEV_mle=parametersGEV_mle[2]
    sigGEV_mle=parametersGEV_mle[3]
    kapGEV_mle=parametersGEV_mle[1]
    qGEV_mle=calculQuantileGEV(mu_GEV_mle,sigGEV_mle,kapGEV_mle,returnLevel)
    listQuantiles_mle[k1]=qGEV_mle
    listSig_mle[k1]=sigGEV_mle
    listKap_mle[k1]=kapGEV_mle+G
  }
  #fit GEV avec method = PWM
  fitGEV_pwm=try(gevFit(maxAnnuels,type='pwm'),T)
  if (class(fitGEV_pwm)=='try-error'){
    nbErr2=nbErr2+1
  }
  else{
    k2=k2+1
    parametersGEV_pwm=fitGEV_pwm@fit$par.ests
    mu_GEV_pwm=parametersGEV_pwm[2]
    sigGEV_pwm=parametersGEV_pwm[3]
    kapGEV_pwm=parametersGEV_pwm[1]
    qGEV_pwm=calculQuantileGEV(mu_GEV_pwm,sigGEV_pwm,kapGEV_pwm,returnLevel)
    listQuantiles_pwm[k2]=qGEV_pwm
    listSig_pwm[k2]=sigGEV_pwm
    listKap_pwm[k2]=kapGEV_pwm+G
  }

  #simulation d'une echantillon de taille ny d'une loi GPD(sig,kapp)
  #simulationGPD=generateurGPD(nb=ny,sigma=sig,kappa = kap)
```

```
#fit GPD avec method = MLE
u=findThreshold(simu)
fitGPD_mle_pot=try(quantilePOT_Z(simu
                                 ,duree_tempete_jours = 3
                                 ,rlow_clust = 1
                                 ,ulow_clust = 2.4
                                 ,seuil=u,npp=1460
                                 ,levelReturn = returnLevel,method='mle'),T)
if (class(fitGPD_mle_pot)=='try-error'){
  nbErr3=nbErr3+1
}
else{
  k3=k3+1
  parametersGPD_mle_pot=fitGPD_mle_pot
  sigGPD_mle_pot=as.numeric(parametersGPD_mle_pot[2])
  kapGPD_mle_pot=as.numeric(parametersGPD_mle_pot[3])
  qGPD_mle_pot=as.numeric(parametersGPD_mle_pot[1])
  #qGEV_mle_pot=gevrlevelPlot(fitGEV_mle_pot,kBlocks=returnLevel)$u
  listQuantiles_mle_pot[k3]=qGPD_mle_pot
  listSig_mle_pot[k3]=sigGPD_mle_pot
  listKap_mle_pot[k3]=kapGPD_mle_pot+G
}
#fit GPD avec method = PWM
u=findThreshold(simu)
fitGPD_pwm_pot=try(quantilePOT_Z(simu,duree_tempete_jours = 3,
                                 rlow_clust = 1
                                 ,ulow_clust = 2.4,seuil=u
                                 ,npp=1460
                                 ,levelReturn = returnLevel
                                 ,method='pwm'),T)
if (class(fitGPD_pwm_pot)=='try-error'){
  nbErr4=nbErr4+1
}
else{
  k4=k4+1
  parametersGPD_pwm_pot=fitGPD_pwm_pot
  sigGPD_pwm_pot=as.numeric(parametersGPD_pwm_pot[2])
  kapGPD_pwm_pot=as.numeric(parametersGPD_pwm_pot[3])
  qGPD_pwm_pot=as.numeric(parametersGPD_pwm_pot[1])
```

```
      listQuantiles_pwm_pot[k4]=qGPD_pwm_pot
      listSig_pwm_pot[k4]=sigGPD_pwm_pot
      listKap_pwm_pot[k4]=kapGPD_pwm_pot+G
    }
  }
  matrice_mle=matrix(c(listQuantiles_mle,listSig_mle,listKap_mle)
                     ,nrow=nbMC-nbErr1)
  biais_rmse_gev_mle=biais_RMSE(matrice_mle
                                ,qTheorique =qtheorique
                                ,sigmaTheorique = sig, kapTheorique = kap+G)
  matrice_pwm=matrix(c(listQuantiles_pwm
                       ,listSig_pwm,listKap_pwm),nrow=nbMC-nbErr2)
  biais_rmse_gev_pwm=biais_RMSE(matrice_pwm
                                ,qTheorique =qtheorique
                                ,sigmaTheorique = sig, kapTheorique = kap+G)
  matrice_mle_pot=matrix(c(listQuantiles_mle_pot
                           ,listSig_mle_pot,listKap_mle_pot),nrow=nbMC-nbErr3)
  biais_rmse_gpd_mle_pot=biais_RMSE(matrice_mle_pot
                                    ,qTheorique =qtheorique
                                    ,sigmaTheorique = sig
                                    , kapTheorique = kap+G)
  matrice_pwm_pot=matrix(c(listQuantiles_pwm_pot
                           ,listSig_pwm_pot,listKap_pwm_pot),nrow=nbMC-nbErr4)
  biais_rmse_gpd_pwm_pot=biais_RMSE(matrice_pwm_pot
                                    ,qTheorique =qtheorique
                                    ,sigmaTheorique = sig, kapTheorique = kap+G)

  return(list(mleGEV=biais_rmse_gev_mle,
              pwmGEV=biais_rmse_gev_pwm
              ,mleGPD=biais_rmse_gpd_mle_pot
              ,pwmGPD=biais_rmse_gpd_pwm_pot
              ,err=c(nbErr1/nbMC*100,nbErr2/nbMC*100
                     ,nbErr3/nbMC*100,nbErr4/nbMC*100)))
}

#renvoie la matrice de biais/RMSE propre, exactement sous la forme souhaitee
constructionMatriceBiaisRmse_zhs_beta1_gamma_beta2=
  function(nbMC=100,range_ny=c(10,20,50,100),rangeKap=c(-0.3,-0.2,-0.1,0,0.1)){
  #Partie tableau quantiles
  n1=length(range_ny)
```

```
#quantile
matBiaisGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matBiaisGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGEV=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGPD=matrix(0,ncol=length(rangeKap),nrow=n1*2)
errGEV=matrix(0,nrow=n1,ncol=length(rangeKap))
errGPD=matrix(0,nrow=n1,ncol=length(rangeKap))
#kappa
matBiaisGEV2=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matBiaisGPD2=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGEV2=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGPD2=matrix(0,ncol=length(rangeKap),nrow=n1*2)

#sigma
matBiaisGEV3=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matBiaisGPD3=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGEV3=matrix(0,ncol=length(rangeKap),nrow=n1*2)
matRMSEGPD3=matrix(0,ncol=length(rangeKap),nrow=n1*2)

j=0
for (kappa in rangeKap){
  k=0
  j=j+1
  print(j)
  for (ny in range_ny){
    tt=Sys.time()
    k=k+1
    if (kappa<0){
      simu
      =simulation_non_iid_non_stationnaire_pot_am_beta1(nbMC=nbMC
                                                        ,ny=ny,kap=rangeKap[j])
    }
    if (kappa==0){
      simu=simulation_non_iid_non_stationnaire_pot_am_gamma(nbMC=nbMC,ny=ny)
    }
    if (kappa>0){
      simu
      =simulation_non_iid_non_stationnaire_pot_am_beta2(nbMC=nbMC
                                                        ,ny=ny,kap=rangeKap[j])
```

```
}
print(Sys.time()-tt)
print(k)

##quantile
biais_gev=simu$mleGEV$relativeBiaisQ
biais_gpd=simu$mleGPD$relativeBiaisQ
rmse_gev=simu$mleGEV$relativeRmseQ
rmse_gpd=simu$mleGPD$relativeRmseQ
matBiaisGEV[k,j]=round(biais_gev,2)
matBiaisGPD[k,j]=round(biais_gpd,2)
matRMSEGPD[k,j]=round(rmse_gpd,2)
matRMSEGEV[k,j]=round(rmse_gev,2)
biais_gev=simu$pwmGEV$relativeBiaisQ
biais_gpd=simu$pwmGPD$relativeBiaisQ
rmse_gev=simu$pwmGEV$relativeRmseQ
rmse_gpd=simu$pwmGPD$relativeRmseQ
matBiaisGEV[k+n1,j]=round(biais_gev,2)
matBiaisGPD[k+n1,j]=round(biais_gpd,2)
matRMSEGPD[k+n1,j]=round(rmse_gpd,2)
matRMSEGEV[k+n1,j]=round(rmse_gev,2)
errGEV[k,j]=simu$err[1]
errGPD[k,j]=simu$err[3]

##kappa
biais_gev2=simu$mleGEV$relativeBiaisK
biais_gpd2=simu$mleGPD$relativeBiaisK
rmse_gev2=simu$mleGEV$relativeRmseK
rmse_gpd2=simu$mleGPD$relativeRmseK
matBiaisGEV2[k,j]=round(biais_gev2,2)
matBiaisGPD2[k,j]=round(biais_gpd2,2)
matRMSEGPD2[k,j]=round(rmse_gpd2,2)
matRMSEGEV2[k,j]=round(rmse_gev2,2)
biais_gev2=simu$pwmGEV$relativeBiaisK
biais_gpd2=simu$pwmGPD$relativeBiaisK
rmse_gev2=simu$pwmGEV$relativeRmseK
rmse_gpd2=simu$pwmGPD$relativeRmseK
matBiaisGEV2[k+n1,j]=round(biais_gev2,2)
matBiaisGPD2[k+n1,j]=round(biais_gpd2,2)
matRMSEGPD2[k+n1,j]=round(rmse_gpd2,2)
```

```
    matRMSEGEV2[k+n1,j]=round(rmse_gev2,2)

    #sigma
    biais_gev3=simu$mleGEV$relativeBiaisS
    biais_gpd3=simu$mleGPD$relativeBiaisS
    rmse_gev3=simu$mleGEV$relativeRmseS
    rmse_gpd3=simu$mleGPD$relativeRmseS
    matBiaisGEV3[k,j]=round(biais_gev3,2)
    matBiaisGPD3[k,j]=round(biais_gpd3,2)
    matRMSEGPD3[k,j]=round(rmse_gpd3,2)
    matRMSEGEV3[k,j]=round(rmse_gev3,2)
    biais_gev3=simu$pwmGEV$relativeBiaisS
    biais_gpd3=simu$pwmGPD$relativeBiaisS
    rmse_gev3=simu$pwmGEV$relativeRmseS
    rmse_gpd3=simu$pwmGPD$relativeRmseS
    matBiaisGEV3[k+n1,j]=round(biais_gev3,2)
    matBiaisGPD3[k+n1,j]=round(biais_gpd3,2)
    matRMSEGPD3[k+n1,j]=round(rmse_gpd3,2)
    matRMSEGEV3[k+n1,j]=round(rmse_gev3,2)
  }
}
#quantile:
col1=c(range_ny,range_ny)
col2=c(rep('mle',n1),rep('pwm',n1))
col1
Biais=c('ny','est',rep('',length(rangeKap)))
aa=cbind(col1,col2,matBiaisGEV)
mat1=rbind(Biais,aa)
aa=cbind(col1,col2,matBiaisGPD)
mat2=rbind(Biais,aa)
matt=cbind(mat1,mat2)
RMSE=c('ny','est',rep('',length(rangeKap)))
aa=cbind(col1,col2,matRMSEGEV)
mat3=rbind(RMSE,aa)
aa=cbind(col1,col2,matRMSEGPD)
mat4=rbind(RMSE,aa)
matt2=cbind(mat3,mat4)
matFinal=rbind(matt,matt2)
l1=c("","",rep('GEV',length(rangeKap)),'','',rep('GPD',length(rangeKap)))
qtheoriqueGEV=round(calculQ_theoriqueGEV(kappa = rangeKap),2)
```

```
qtheoriqueGPD=round(calculQ_theoriqueGPD(kappa = rangeKap),2)
if(is.element(0,rangeKap)){
  ind=which(rangeKap==0)
  qtheoriqueGEV[ind]=round(calculQ_theoriqueGEV(kappa=0),2)
  qtheoriqueGPD[ind]=round(calculQ_theoriqueGPD(kappa=0),2)
}
l2=c('quantile','quantile',qtheoriqueGEV,"quantile",'quantile',qtheoriqueGPD)
l3=c('kappa','kappa',rangeKap,'kappa','kappa',rangeKap)
matFinal=rbind(l1,l2,l3,matFinal)

##kappa
col1=c(range_ny,range_ny)
col2=c(rep('mle',n1),rep('pwm',n1))
col1
Biais=c('ny','est',rep('',length(rangeKap)))
aa=cbind(col1,col2,matBiaisGEV2)
mat1=rbind(Biais,aa)
aa=cbind(col1,col2,matBiaisGPD2)
mat2=rbind(Biais,aa)
matt=cbind(mat1,mat2)
RMSE=c('ny','est',rep('',length(rangeKap)))
aa=cbind(col1,col2,matRMSEGEV2)
mat3=rbind(RMSE,aa)
aa=cbind(col1,col2,matRMSEGPD2)
mat4=rbind(RMSE,aa)
matt2=cbind(mat3,mat4)
matFinal2=rbind(matt,matt2)
l1=c("","",rep('GEV',length(rangeKap)),'','',rep('GPD',length(rangeKap)))
l3=c('kappa','kappa',rangeKap,'kappa','kappa',rangeKap)
matFinal2=rbind(l1,l3,matFinal2)

#sigma
col1=c(range_ny,range_ny)
col2=c(rep('mle',n1),rep('pwm',n1))
col1
Biais=c('ny','est',rep('',length(rangeKap)))
aa=cbind(col1,col2,matBiaisGEV3)
mat1=rbind(Biais,aa)
aa=cbind(col1,col2,matBiaisGPD3)
mat2=rbind(Biais,aa)
```

```
  matt=cbind(mat1,mat2)
  RMSE=c('ny','est',rep('',length(rangeKap)))
  aa=cbind(col1,col2,matRMSEGEV3)
  mat3=rbind(RMSE,aa)
  aa=cbind(col1,col2,matRMSEGPD3)
  mat4=rbind(RMSE,aa)
  matt2=cbind(mat3,mat4)
  matFinal3=rbind(matt,matt2)
  l1=c("","",rep('GEV',length(rangeKap)),'','',rep('GPD',length(rangeKap)))
  l2=c('sigma','sigma',rep(1,length(rangeKap)),'sigma','sigma',rep(1,length(rangeKap)
  l3=c('kappa','kappa',rangeKap,'kappa','kappa',rangeKap)
  matFinal3=rbind(l1,l3,matFinal3)

  #liste finale :
  listF=NULL
  listF$matQ=matFinal
  listF$matK=matFinal2
  listF$matS=matFinal3
  return(listF)
}
```

Bibliographie

[1] Caires S., *A comparative simulation study of the annual maxima and the peaks over-threshold methods* . Deltrares, 2009.

[2] Dombry C., *Maximum likelihood estimators for the extreme value index based on the block maxima method*. 2013.

[3] Gilleland E., W. Katz Richard, *extRemes 2.0 : An Extreme Value Analysis Package in R*. 2016.

[4] Jockovic J., *Quantile estimation for the Generalized Pareto Distribution with application to finance*. 2012.

[5] Ribereau P., Guillou A., Naveau P., *Estimating return levels from maxima of non-stationary random sequences using the Generalized PWM method*. 2008.

[6] Raillard N., *Modélisation du comportement extrême de processus spatio-temporels. Applications en océanographie et météorologie*. 2011.

[7] Hosking J.R.M, Wallis J.R, *Parameter and quantile estimation for the Generalized Pareto Distribution*. 1997.

[8] Caires S., *Extreme wave statistics. Methodology and applications to North Sea wave data*. 2007.

[9] P. Embrehts, C. Kluppelberg, and T. Mikosh *Modelling extremal events for insurance and finance*. 1997.

[10] B. Bader, J. Yan1, X. Zhan *Automated Threshold Selection for Extreme Value Analysis via Goodness-of-Fit Tests with Application to Batched Return Level Mapping*. 2016.

[11] C. de Valk *Estimation of marginals from measurements and hindcast data*. 1993.

[12] Hosking, J.R.M., J.R. Wallis, E.F. Wood *Estimation of the generalized extreme-value distribution by the method of probability-weighted moments*. 1985.

[13] Hosking, J.R.M. , J.R. Wallis *Parameter and quantile estimation for the Generalized Pareto Distribution*. 1987.

[14] Yevjevich, V. , V. Taesombut *Information on flood peaks in daily flow series. Proc. Int. Symp. on Risk and Reliability in Water Resources*. 1978.

[15] Cunnane C. *A particular comparison of annual maxima and partial duration series methods of flood frequency prediction.* 1973.

[16] van den Brink, H.W, G.P Können, J.D Opsteegh, *Uncertainties in extreme surge level estimates from observational records.* 2005.

[17] Raynal-Villasenor J. *Probability Weighted Moments Estimators for the GEV (minima) Distribution* 2013.

[18] Greenwood J.A., Landwher J.M., Matalas N.C., Wallis J.R., *Definition and relation to parameters of several distributions expressable in inverse form* 1979.

Résumé — Le calcul des niveaux de retour, ou quantiles extrêmaux, de données demeure capital dans de nombreux domaines, notamment en actuariat pour calculer une Value-At-Risk (ou VaR), qui est un quantile d'ordre 0.995 dans le cadre de Solvabilité II. Notre étude portait essentiellement sur des données environnementales, qui présente la particularité d'être très corrélées entre elles et d'être marquées par une forte saisonnalité. Ces caractéristiques ont une forte influence sur les méthodes de calcul de quantiles de niveau élevé, et nécessitent un traitement préalable. Notre objectif était de comparer les méthodes d'estimation dans la théorie des valeurs extrêmes à savoir le maxima par blocs et le dépassement de seuil ; ainsi que les différents estimateurs utilisés : PWM et MLE. Notre étude a conclu que lorsque peu de données sont à notre disposition, il est préférable d'utiliser la méthde GPD/POT avec l'estimateur PWM. Pour des jeux de données de taille assez conséquente, ce qui est rare dans le cadre de données environnementales, l'écart entre ces méthodes diminue, et l'estimateur du maximum de vraisemblance semble devenir meilleur, du fait de ces propriétés asymptotiques.

Mots clés : Valeurs extrêmes, loi de Pareto généralisée(GPD), loi d'extrêmum généralisée (GEV), méthode de dépassement de seuil (POT), maxima par blocs, méthode des maxima annuels(AM), estimateur du maximum de vraisemblance (MLE), méthode de probabilité pondérée des moments (PWM), séries temporelles, modèle ARMA.

EURIA
6 Avenue Victor le Gorgeu, 29200 Brest
29200 Brest