Facultad de Ciencias

Estadística de valores extremos no estacionarios.

(Statistics of non stationary extreme values.)

Trabajo de Fin de Máster
para acceder al

MÁSTER EN DATA SCIENCE

Autor: Alejandro González Ganzábal

Director\es: Fernando J. Méndez Incera, Nicolás Ripoll Cabarga

Septiembre - 2019

Resumen

Con el objetivo de obtener un modelo capaz de obtener los parámetros que definen una distribución de valores extremos no estacionarios, se ha desarrollado utilizando herramientas para Python 3 un algoritmo del tipo Shuffled Complex Evolution para encontrar el mínimo de la función logaritmo de máxima verosimilitud. Para comprobar la estacionalidad de los valores, los tres parámetros iniciales de una distribución GEV se han construido en forma de ondas sinusoidales mediante la combinación lineal de senos y cosenos, con lo que el modelo más complejo contaría con la búsqueda del mínimo para una función de nueve variables. Los datos a evaluar constituyen máximos de altura de ola mensual entre los años 1979 y 2014.

A la par que el algoritmo, se han desarrollado herramientas para testear esos resultados, fundamentalmente empleando la matriz de información de Fisher, así como los tests estadísticos *AIC* y χ^2.

A la vista de los resultados obtenidos, se concluye que el modelo que devuelve unos mejores parámetros es un modelo de nueve dimensiones, completamente no estacionario.

Para comprobar la robustez del algoritmo, se utilizó un segundo dataset mucho más amplio y variado con máximos de temperatura. En este caso, se usaron también técnicas de normalización. De nuevo se obtuvieron resultados satisfactorios optando por un modelo de nueve parámetros completamente no estacionario.

Abstract

With the objective of obtaining a model capable of determine the parameters that define a non-stationary generalised extreme value distribution, a Shuffled Complex Evolution algorithm was developed using Python 3. It was used to find the parameters that minimise the value of the log likelihood function for GEV distributions. To take seasonality into account, the parameters were built as a linear combination of sinusoidal terms to form waves, being the most complex model one with nine variables. The dataset consists of the maximum wave height per month during the years 1979-2014.

In parallel with the algorithm, some tools for testing were also developed, mainly, the Fisher information matrix and the statistical tests *AIC* and χ^2.

According to the results, the model that returns the better parameters for the GEV distribution was the one with nine parameters, that is, fully non-stationary.

In order to check how robust the algorithm is, a second, bigger and more varied dataset with temperature maxima was used. In this case, normalisation techniques were also used. The results were again satisfactory, opting again for a fully non-stationary model with 9 parameters.

Dedicado a mis padres.

Contents

Chapter 1

Introduction

1.1 Main purpose

As the study of the distribution of extreme values is extremely relevant in some fields of science and engineering, specially hydrology and coastal modelling, new approaches should be taken into account in order to properly solve and analyse problems to come. In 1.2 some of these approaches will be mentioned, as well as the conclusions extracted by means of the proper management of the distribution. After that, a brief insight of the distribution will be provided (1.3), including the principle of maximum Likelihood (1.3.1).

Since the search for the minimum of these functions can be difficult, a SCE algorithm for Python 3 has been developed in order to work with these distribution (1.5).

Although the majority of the project was made focusing on coastal measures (the heights of waves and its time dependency) the algorithm is capable of dealing with other kind of data regardless of its nature, as it will be shown.

1.2 Modern approaches in climatology of the GEV model

In climatology and some of its branches, the Generalised Extreme Value distribution is often used for datasets composed of the maxima or minima values for a certain event. While some results can be obtained by means of rather simple datasets (such as AMM[*],

[*] Annual Maximum Method

used in hydrology for instance), the lack of data is remarkable. Thus more methods were developed to avoid the scarcity of data, such as monthly analysis with time dependency[1]. These time dependant models were evaluated with the maxima for each month, since there is a difference in the value over time for a single year.

The utility of these analysis is akin to several fields: from civil engineering to maintenance of coastal environment and the analysis and prediction of massive wave heights, as it was found that there exists a correlation between the wave heights and the value of climatological indexes such as NINO3 (El Niño Southern Oscilation phenomenon) and the North Atlantic Oscilation (NAO)[2].

1.2.1 An example: the heat wave of 2003.

One of the most extreme cases in recent history of climate was the heat wave and drought of 2003. To properly measure the effects of climatic events several points of view are evaluated, from agriculture to the water flow, affecting even other areas of science such as human health[3].

The extreme climate effects of events of this kind is specially noteworthy when used standard statistical measures. For instance, the temperatures in Europe (average) were 2 times the standard deviation compared to the usual temperatures, reaching even 5σ in local areas. The lack of rainfall was responsible for extense drought, specially in Central Europe, to the point of not being able to navigate through the Rhine river.

1.3 The Generalised Extreme Value Distribution

While in most fields the main approach towards statistics is to estimate values such as the average or the standard deviation, in some other applications the current approach is focused on finding the extreme values, both maxima and minima. Thus, these values will be noted as M_n, leading to $M_n = max\{X_1, X_2, ...X_n\}$, being X_n a set of independent random variables that follow a common distribution function F that may be known or not. This section will feature some of the basic characteristics of these distribution functions F.

The classic extreme values statistical approach mainly focused on picking which family should be used in order to evaluate a model, thus approximating F. These families are the Weibull, Gumbel and Fréchet distributions. This comes directly from the extremal types theorem[4].

Theorem 1 *If there exist sequences of constants $a_n > 0$ and b_n such that*

$$P((M_n - b_n)/a_n \leq z) \to G(z)$$

being G a non-degenerative distribution function, then G must belong to one of the aforementioned families.

The Weibull distribution follows the following expression:

$$pdf(x;\zeta,\sigma) = \begin{cases} \frac{\zeta}{\sigma}\Big(\frac{x}{\sigma}\Big)^{\zeta-1}\exp\big(-(x/\sigma)^{\zeta}\big) & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{1.1}$$

being σ the scale parameter (the larger the scale parameter, the more spreaded out is the distribution) and ζ the shape parameter, which directly affects the shape of the distribution.

The Gumbel distribution follows the equation

$$pdf(x;\mu,\sigma) = \frac{1}{\sigma}e^{-(z+e^{-z})} \tag{1.2}$$

being z in 1.2 $z = \frac{x-\mu}{\sigma}$. Again, σ refers to the scale parameter while μ is the location parameter, which formally is identified as $f_{\mu} = f(x-\mu)$ being f a probability density function. It should be noted that some standard parameters used in probabilistic distributions not related to extreme values, such as the mean, follow this expression, being the most obvious one the mean value of a Gaussian distribution.

Finally, the Fréchet probability density function expression can be seen in eq.(1.3).

$$pdf(x;\zeta,\mu,\sigma) = \frac{\zeta}{\sigma}\left(\frac{x-\mu}{\sigma}\right)^{-1-\zeta} e^{-\left(\frac{x-\mu}{\sigma}\right)^{-\zeta}} \tag{1.3}$$

These families can be combined in the so called Generalised Extreme Values, following the probability density function seen in eq.(1.4).

$$pdf(x;\zeta,\mu,\sigma) = \frac{1}{\sigma}t(x)^{\zeta+1}e^{-t(x)} \tag{1.4}$$

being $t(x)$ defined as in eq.(1.5).

$$t(x) = \begin{cases} \left(1+\zeta\left(\frac{x-\mu}{\sigma}\right)\right)^{-1/\zeta} & \zeta \neq 0 \\ e^{-(x-\mu)/\sigma} & \zeta = 0 \end{cases} \tag{1.5}$$

Whether a probability density function defined by these parameters follows one distribution or the other (and thus matching with the Generalised Extreme Values distribution) is given by the value of the shape parameter. Thus, if $\zeta > 0$, it will follow a Fréchet distribution. If $\zeta = 0$, then it will correspond to a Gumbel distribution. Lastly, if $\zeta < 0$ it will follow a Weibull distribution.

To appreciate how the probability density function is modified by these parameters, several plots have been made in Fig.1.4.

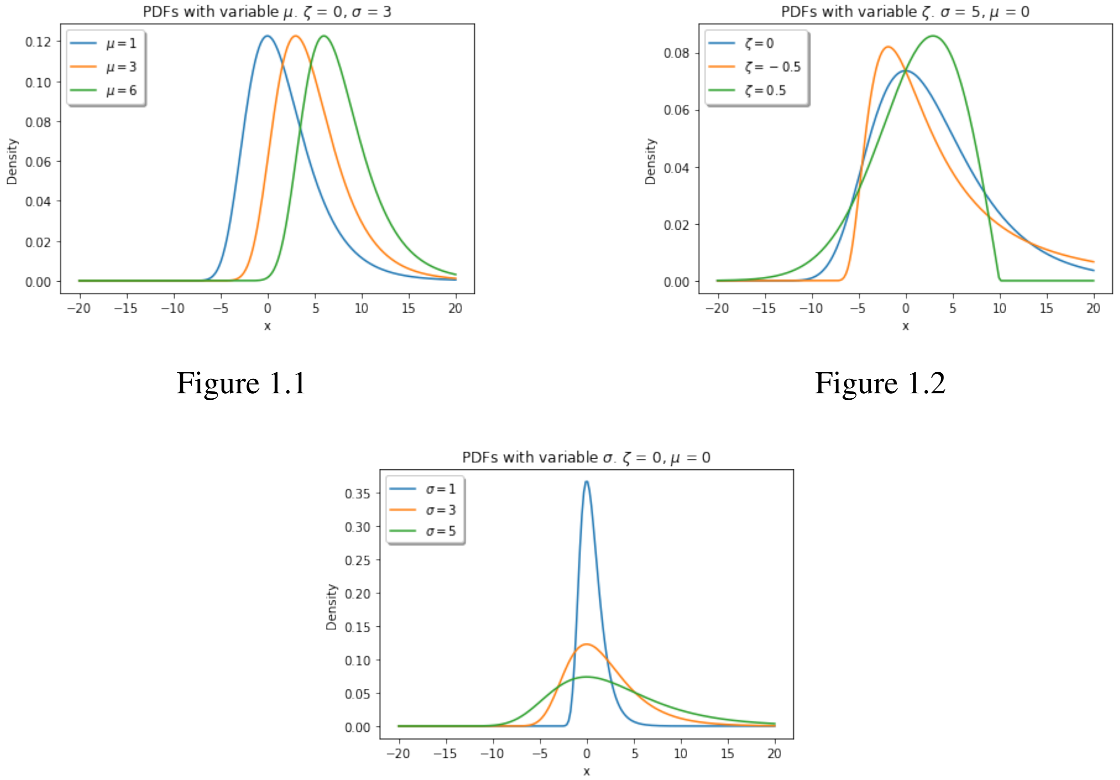


Figure 1.1

Figure 1.2

Figure 1.3

Figure 1.4: Figure shows how the shapes of the distribution for a GEV changes according to the defined parameters by means of fixing two of them and modifying the other one in each plot as detailed in the legend of each subplot.

1.3.1 The Maximum Likelihood

The Maximum Likelihood is a technique used to find the coefficients for a probability distribution, based on finding estimations for θ_1, θ_2, ..., θ_n so that the predicted probability $\bar{p}(v_i)$ for each individual value of the set is as close as possible to the probability of that value. The starting point is a sample of data $v_1, v_2, ..., v_n$ extracted from a probability distribution given by an unknown set of parameters θ such as $P(v_1, v_2, ..., v_n|\theta)$. Provided that θ is unknown, but the set of values v_n given by $P(\theta)$ is known, an estimation of θ can be given as the likelihood of these parameters given the sample $\ell(\theta|v_1, v_2, ..., v_n)$. So the most likely values for θ would be the ones that maximise ℓ [5].

If the function that describes the likelihood ℓ is known, it can be done by finding the maximum via the derivative. This is an easy task if the function only depends on one variable. Given that case, the only problem results on finding the absolute maximum of the probability function instead of confusing it with a local maximum. For more dimensions, not only the aforementioned problem persists, but also the computational cost of the calculations is much greater, even if the derivative is inserted in its analytical form. This is the case that is going to be used, as the expression for the log likelihood function for the

Generalised Extreme Values probability density function (eq.(1.4)) can be written as in eq.(1.6)[6].

$$\begin{aligned} \ell(\zeta,\mu,\sigma) = &- m\log(\sigma) - \left(1+\frac{1}{\zeta}\right)\sum_{i=1}^{m}\log\left[1+\zeta\left(\frac{x_i-\mu}{\sigma}\right)\right] \\ &- \sum_{i=1}^{m}\left[1+\zeta\left(\frac{x_i-\mu}{\sigma}\right)\right]^{-\frac{1}{\zeta}} \end{aligned} \tag{1.6}$$

being m the number of samples, provided that the condition in eq.(1.7) is fulfilled.

$$1+\zeta(t)\left(\frac{x_i-\mu}{\sigma}\right) > 0 \tag{1.7}$$

If this condition is not satisfied, x_i are considered outliers of the distribution and will not be taken into account for the calculations. It should also be noted that for $\zeta = 0$ the expression in 1.6 takes the form of eq.(1.8), as a Gumbel distribution.

$$\ell(x;\sigma,\mu) = -m\log(\sigma) - \sum_{i=1}^{m}\left(\frac{x_i-\mu}{\sigma}\right) - \sum_{i=1}^{m}\exp\left[-\left(\frac{x_i-\mu}{\sigma}\right)\right] \tag{1.8}$$

Here it has been used the log Likelihood function. The mathematical advantage of using the logarithm instead of using the function directly is to ensure the concavity of the function[†]. It's also more convenient in matters of computational effort. Now the objective is to obtain the coefficients $\hat{\mu}$, $\hat{\sigma}$, $\hat{\zeta}$ that minimise the log Likelihood function. While before it has been stated that the objective would be to maximise, since most of the existing algorithms and libraries work with minimising techniques, a simple change of sign is performed. This also will be convenient to avoid conflict with existing libraries that return the log likelihood with this convention (as will be displayed in (3)).

1.3.2 Retrieving results value

As mentioned in [8], the return value for these statistical models, assuming that they are time dependant (as will be explained in 2.4) is the correspondent to the 0.95 quantile. Mathematically it can be expressed as in eq.(1.9).

$$z(t,\mu,\sigma,\zeta) = \begin{cases} \mu(t) - \dfrac{\sigma(t)}{\zeta(t)}\left[1-(-log(1-q))^{\zeta(t)}\right] & \zeta \neq 0 \\ \mu(t) - \sigma(t)log(-log(1-q)) & \zeta = 0 \end{cases} \tag{1.9}$$

[†] More on this can be found in [7].

which is valid for both the GEV model and the Gumbel model. This equation has been written using time dependency, which will be properly explained in 2.4, as the quantiles for non-stationary models are time-dependant during the year[8].

1.4 Evaluating the results

Once the results for the log likelihood and the value for the parameters were obtained, these results were analysed with the aid of the following techniques in this section. Before acquiring a proper algorithm, only comparisons between algorithms were used, as will be provided in Table 3.2 for instance.

1.4.1 The Fisher information matrix

The Fisher information matrix is a method used in information sciences to obtain estimations for the covariation of certain variables. Since the main focus here is the GEVD, no other forms will be presented. For a common GEV distribution, the Fisher Matrix takes the form seen in eq.(1.10).

$$\mathbf{I}_\theta(\zeta,\mu,\sigma) = \begin{pmatrix} -E\frac{\partial^2\ell}{\partial\zeta^2} & -E\frac{\partial^2\ell}{\partial\zeta\partial\sigma} & -E\frac{\partial^2\ell}{\partial\mu\partial\zeta} \\ -E\frac{\partial^2\ell}{\partial\zeta\partial\sigma} & -E\frac{\partial^2\ell}{\partial\sigma^2} & -E\frac{\partial^2\ell}{\partial\mu\partial\sigma} \\ -E\frac{\partial^2\ell}{\partial\mu\partial\zeta} & -E\frac{\partial^2\ell}{\partial\mu\partial\sigma} & -E\frac{\partial^2\ell}{\partial\mu^2} \end{pmatrix} \tag{1.10}$$

For a Gumbel distribution, the Fisher information matrix takes the analytical form of eq.(1.11).

$$\mathbf{I}_\theta(\mu,\sigma) = \begin{pmatrix} \frac{m}{\sigma^2} & \frac{n(\gamma-1)}{\sigma^2} \\ \frac{n(\gamma-1)}{\sigma^2} & \frac{n(6-12\gamma+6\gamma^2+m^2)}{6\sigma^2} \end{pmatrix} \tag{1.11}$$

where γ is the Euler constant. Once obtained the Fisher matrix, the asymptotic variances and covariances can be obtained as in eq.(1.12).

$$\mathbf{\Sigma}_\theta = \mathbf{I}_\theta^{-1} \tag{1.12}$$

Lastly, if instead of θ the values given are $\hat{\theta}$, then an estimation of the covariance matrix is obtained, as in (1.13)[9].

$$\mathbf{\Sigma}_{\hat{\theta}} = \mathbf{I}_{\hat{\theta}}^{-1} \tag{1.13}$$

1.4.2 The *AIC* and χ^2 tests

To make a proper comparison between models according to the number of parameters used in determining the log likelihood, two test are proposed, one being a test using the χ^2 statistics and the other being the *AIC* test (as suggested in [8]).

For the first, the comparison of two models i, j is based on eq.(1.14).

$$\ell_i - \ell_j > \frac{1}{2}\chi^2(1-p) \tag{1.14}$$

where p indicates the p-value, in this case, 0.05 and χ^2 is the test corresponding to both models i, j. If the condition is fulfilled, then the model i is considered superior to the model j.

The *AIC* test, on the other hand, can be expressed as in eq.1.15.

$$AIC = -2\ell + 2c \tag{1.15}$$

where c indicates the number of parameters. Using our notation, the closest the *AIC* score is to zero results in the better model.

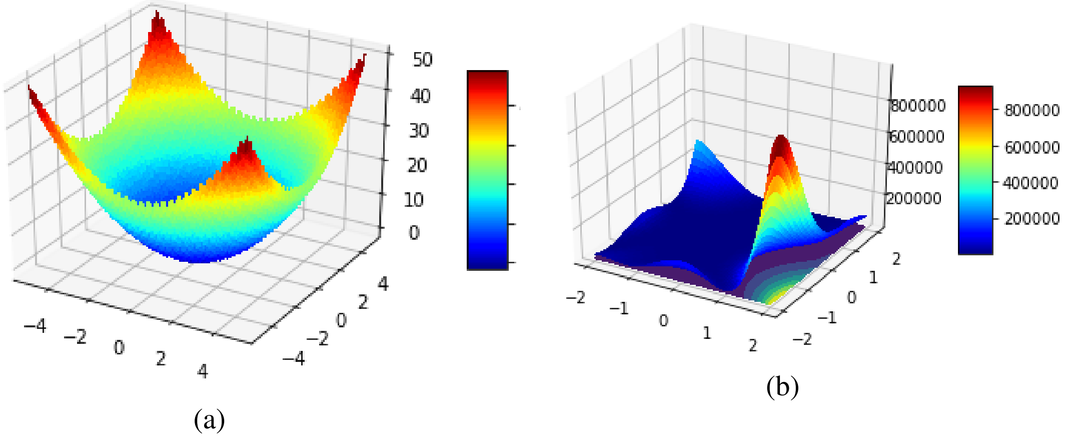
1.5 The SCE algorithm

While dealing with a minimisation problem several hindrances may rise[10]:

- There might be several regions where the search algorithm may get stuck.
- Each of the aforementioned regions may contain local minima quite different from the absolute minimum.
- The objective function may contain irregularities, such as being discontinuous in several regions.
- The interaction between the parameters may be non-linear and rather complex.
- Non-convexity near the wanted minimum.

The SCE (Shuffle Complex Evolution) algorithm was created at University of Arizona in order to deal with optimisation problems that fall in the categories listed above[10]. It's based on several concepts that were used previously in other algorithms, such as clustering and the combination of random and deterministic approaches.

In Figure 1.5 the algorithm was used to analyse the Rastrigin and the Goldstein-Price functions, often used to determine the usefullness of minimisation algorithms.



(a)

(b)

Figure 1.5: Some basic examples of the SCE algorithm evaluating the Rastrigin function (1.5a) and the Goldstein-Price function (1.5b). A brief insight on these functions can be found in [11] and [12] respectively.

The SCE algorithm for a n dimensional problem works as follows:

1. To start the process, $p \geq 1$ and $m \geq (n+1)$ must be selected, being p the number of complexes and m the number of points in each complex.

2. For a sample x with size pm formed by s points $(x_1, x_2, ..., x_s)$ in a given space $\Omega \in \Re$, compute the value of the objective function f.

3. Store the evaluated points with their function value in a vector D in increasing order, such as $\{x_i, f(x_i); ...; x_s, f(x_s)\}$, being $f(x_{i+1}) > f(x_i)$.

4. Part D into p complexes containing each m points, and evolve each p via the Competitive Complex Evolution algorithm.

5. Repeat the process replacing the values in D until either the number of iterations is reached or the system has reached the desired threshold for the value of $f(x_i) - f(x_{i+1})$, being i a given iteration.

A flow diagram for the Competitive Complex Evolution mentioned above can be found in Fig. 1.6.

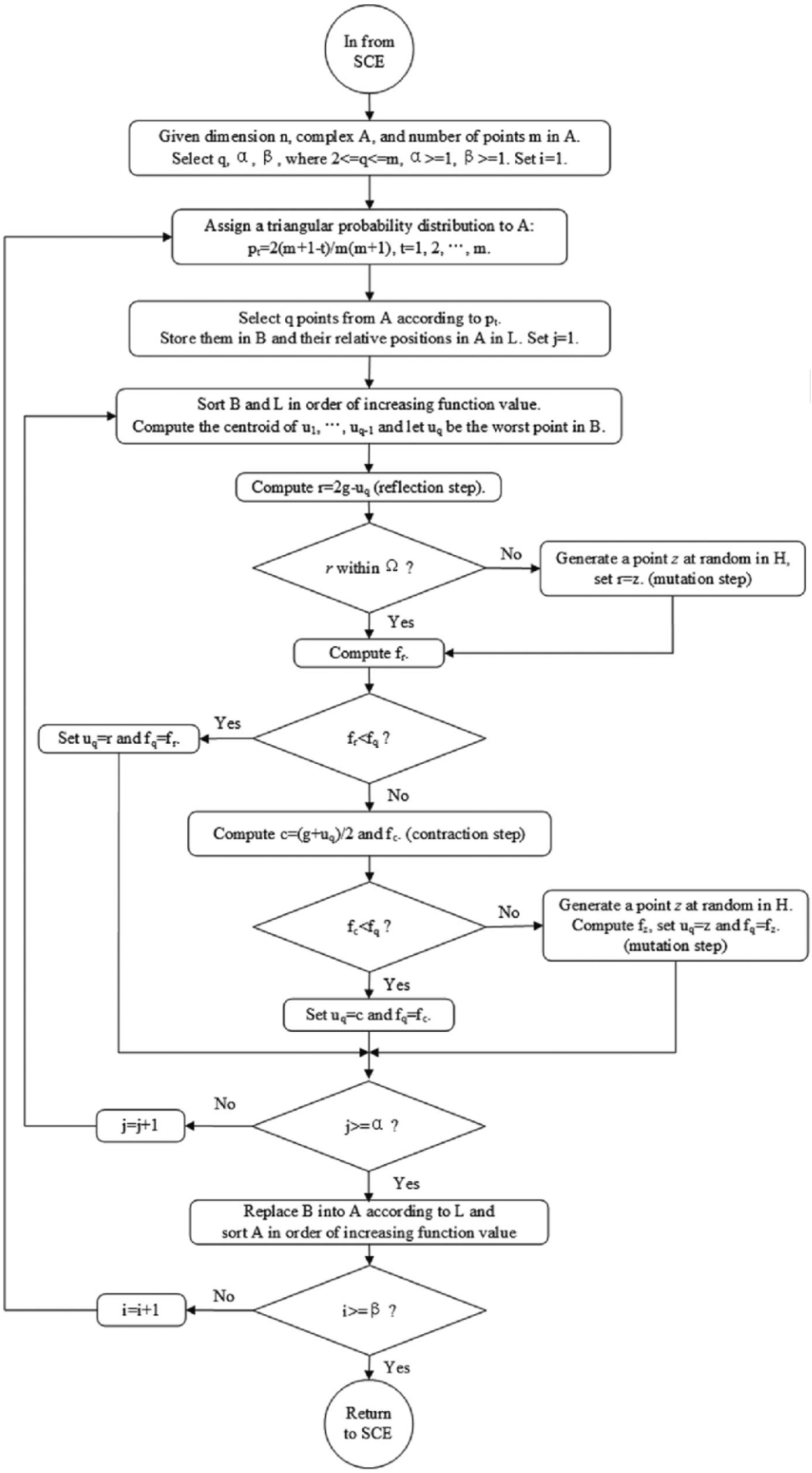

In from SCE
Given dimension n, complex A, and number of points m in A. Select q, α, β, where 2<=q<=m, α>=1, β>=1. Set i=1.
Assign a triangular probability distribution to A: p_t=2(m+1-t)/m(m+1), t=1, 2, ⋯, m.
Select q points from A according to p_t. Store them in B and their relative positions in A in L. Set j=1.
Sort B and L in order of increasing function value. Compute the centroid of u_1, ⋯, u_{q-1} and let u_q be the worst point in B.
Compute r=2g-u_q (reflection step).
r within Ω ?
No
Generate a point *z* at random in H, set r=z. (mutation step)
Yes
Compute f_r.
f_r<f_q ?
Yes
Set u_q=r and f_q=f_r.
No
Compute c=(g+u_q)/2 and f_c. (contraction step)
f_c<f_q ?
No
Generate a point *z* at random in H. Compute f_z, set u_q=z and f_q=f_z. (mutation step)
Yes
Set u_q=c and f_q=f_c.
j>=α ?
No
j=j+1
Yes
Replace B into A according to L and sort A in order of increasing function value
i>=β ?
No
i=i+1
Yes
Return to SCE

Figure 1.6: Flow diagram for the CCE algorithm following the steps of the original article. Flow chart extracted from [13].

Chapter 2

The library

2.1 Log likelihood function

The main concern, as mentioned in 1.1 was to write a Python library that would be able to deal with statistics for extreme values. The first step was to build a function that would calculate the log likelihood (eqs. (1.6) and (1.8)) for a Generalised Extreme Values distribution. The function takes a tuple *theta* which contains the parameters for a GEV model and the dataset to be evaluated (logLikelihood(theta, data)). Returns the value for the log likelihood as a float. To ensure that this function worked properly, the results were compared to the output for the function nnlf from the SciPy package[14]. The example datasets were taken from [9]*.

At first, the function logLikelihood was used in a greedy algorithm (greedy(n)) that will evaluate the function for a dataset over a grid of n^3 divisions, corresponding to n values for ζ, n values for σ and n values for μ and return the combination of parameters ζ, σ and μ that corresponds to the lowest value of the log likelihood. This is in no way recommended (although it returned good results, specially for the location parameter) due to it's high computational need (for $n = 40$ it can take up to several minutes). Also it should be noted that at this point only three parameters were tested, and in order to obtain a decent grid for the testing, a basic knowledge of the parameters is needed beforehand.

*It should be noted that what nnlf really does is simply $-\sum \log(pdf(\theta, x))$.

2.2 Normalisation

Since the objective functions require a range for the parameters to be searched, a simple approximation can be made by means of using normalised values for the parameters, thus reducing the space searched and converting the values back to normal. This conversion was made as in eq.(2.1).

$$\begin{cases} \zeta = \zeta' \\ \mu = \mu' std(data) + mean(data) \\ \sigma = \sigma' std(data) \end{cases} \tag{2.1}$$

It should also be noted that the normalisation may not be necessary, as it depends on several factors, such as how spreaded out the data is.

Other studies focus mostly on the use of power normalisation (PGEV) and only in certain scenarios the properties of a PGEV can be used in a GEV distribution[15]. Thus, the library will only evaluate the data in the original space, and the parameters will be given also in this space. Not using this technique directly also gives the possibility of using the library for more extense datasets, as the conditions in [15] don't necessarily have to be fulfilled.

2.3 The Datasets

As mentioned above, during the first testings for the logLikelihood function, two basic datasets were used, both taken from [9], named Waves and Flood. As the library progressed, a more complex dataset was used based on the detected heights of waves during the years 1979-2014 (for the Marshall island in Australia, according to CSIRO). This will be referred as the Heights dataset.

The presented dataset contained values for the time (up to hours), the mean period, the peak period, the height of the wave and the direction. An example can be found in Table 2.1.

Year	Month	day	hour	$h(m)$	$T_m(s)$	$T_p(s)$	direction(°)
1979	2	1	0	3.243	7.3487	1.786	25.4823
1979	2	1	1	3.2673	7.363	17.718	21.1644

Table 2.1: Table shows an example of the provided dataset, including the first two rows.

After eliminating the data and obtain the Julian day (as t for seasonality), an example of the dataset can be found in Table 2.2.

Year	Month	$h(m)$	Julian day
1979	2	4.771753	42
1979	3	3.696231	68

Table 2.2: Table shows the result of the given dataset after the proper data curation, containing only the maxima per month of each year, from 1979 to 2014.

Finally, in order to do the same and also to prove that higher amount of data can also be analysed in a similar way, a temperature dataset (taken from AEMET) was also used, containing, like its predecessor, the maximum per day. Once curated (in almost the same fashion as the Heights dataset), the structure was the same as in Table 2.2. This was also used to show that the algorithm was compatible with the normalisation presented in eq.(2.2).

2.4 Evaluating seasonality

For a better understanding of the phenomena, a season-based analysis was taken into account as some articles suggested[8]. This seasonal model focused on the use of sinusoidal functions to parameterise the values of the GEV distribution's parameters. Thus, the parameters will have the following form, depicted in eq.(2.2).

$$\begin{cases} \zeta(t) = \zeta_0 + \zeta_1 \cos \frac{2\pi t}{T} + \zeta_2 \sin \frac{2\pi t}{T} \\ \sigma(t) = \sigma_0 + \sigma_1 \cos \frac{2\pi t}{T} + \sigma_2 \sin \frac{2\pi t}{T} \\ \mu(t) = \mu_0 + \mu_1 \cos \frac{2\pi t}{T} + \mu_2 \sin \frac{2\pi t}{T} \end{cases} \tag{2.2}$$

being $T = 365.25$ (counting also leap-years) and t the day of the present maximum.

The need for a seasonal model comes from the fact that over the year the maximum values for the heights of waves varies significantly for each season. A simple example of this can be found in the plot in Fig. 2.1. These datasets and extreme, non stationary variables were already analysed in previous works, for instance, in [16].

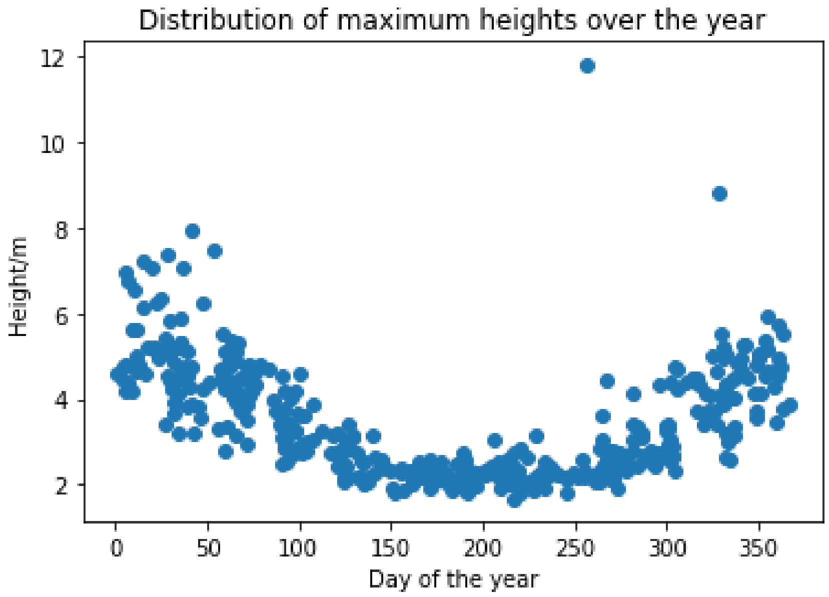


Figure 2.1: Figure shows the maximum height per month over the years 1979-2014 in meters of the waves correlated to the Julian day. It should also be noted that the plot contains at the very least two obvious outliers that were not discarded.

2.5 The SCE_UA library

The SCE algorithm is contained in three different modules. These modules have been adapted to Python 3 using the original libraries from [17], which is a direct Python adaptation from the original algorithm presented in [10].

The modules are as follows:

- SCE_python.py, which contains the proper algorithm for a given function and given parameters.
- SCE_functioncall.py, containing the mathematical expressions of the functions given to SCE_python.py.
- test_sce_implem.py, containing the initial parameters as well as the bounds for each function. This is the one that has to be executed. After running this module, it will ask a number depending of which function the user wants to evaluate. The functions available are:
  1. The Goldstein-Price function.
  2. The Rosenbrock function.
  3. The Six-hump Camelback Function.
  4. The Rastrigin function.

5. The Griewank Function[†].
6. The logLikelihood functions mentioned in (2.1).

2.6 The Fisher library

A module called Fisher.py was also developed, containing the function GEV_ACOV(theta,x), which takes the tuple *theta* and a dataset x and returns the covariation matrix (1.4.1). If seasonality is taken into account as in eq.(2.2), instead of a 3x3 matrix it will return a dxd matrix depending of the kind of seasonality used, up until 9x9 (completely non-stationary model).

Thus, GEV_ACOV evaluates the Fisher information matrix using partial derivatives. That is, directly as in eq.(1.10). For each element of the matrix, an infinitesimal change in the value of the function is evaluated. In order to simplify this and make it easier to use, the logLikelihood function was cut from the SCE algorithm, so only the values that are going to be evaluated are passed into the function logLike(theta).

In order to put it all together, Figure 2.2 links all the modules used in the process:

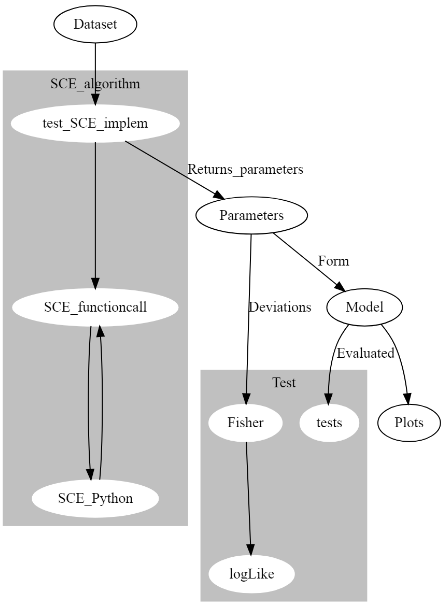


Figure 2.2: The diagram shows all the modules used during the process. To avoid over-complicating it, the functions inside each module are not mentioned.

[†] The code for this one has been slightly modified for the parameters to fit compared to the original script in [17], otherwise, it would be too heavy to compute

Chapter 3

Results

3.1 First results for the LogLikelihood

To test whether or not logLikelihood worked, it was tested with the Waves and Flood datasets mentioned above. As a first test, the results obtained were very satisfactory. Not only it was compared with the results of the book (which were cut up to just one digit), but also with the nnlf method. The results were as follows in Table 3.1.

Dataset	LogLikelihood (nnlf)	LogLikelihood (own)
Wave	164.84079365	164.95225721
Flood	215.78044042	217.04044113

Table 3.1: Table shows the different values obtained for the Log Likelihood of the two datasets using the two different functions. In both cases the values for the parameters were obtained with the fit function from SciPy.

Combined with the greedy function, it was possible to obtain the three parameters that returned the minimum value for the log Likelihood. The main issue with this was the time needed. For just 3 parameters and a 40 x 40 x 40 was exceedingly high (circa 200 seconds per run) and required a very closed grid, that is, knowledge regarding the possible value of the parameters. Thus it was completely discarded.

Other optimisation tools provided by SciPy were also used. For instance, the minimize function. Nevertheless, while some of the algorithms worked, others did not. CG could not even find a result, while COBYLA provided wrong parameters. Nelder-Mead and SLSQP were the only ones that provided a solid result.

3.2 Using the SCE algorithm

This section will emphasise on the use of the SCE algorithm, initially comparing it to the other used algorithms, and later, several trials will be displayed.

3.2.1 Comparison with other results

Once the library was finished, again the Waves dataset was tested again to check if the algorithm worked. In order to compare, a summary of the results for this dataset can be seen in Table 3.2 below.

Method	Log Likelihood	ζ	μ	σ
nnlf	164.84079365	0.02387417644	11.373634935	5.6464844917
greedy	164.88059828	0.0	11.18181818	5.63636363
minimize (CG)	166.40789883	0.	10.	5.
minimize (SLSQP)	164.840793708	-0.02383159	11.37363173	5.64643095
minimize (COBYLA)	7074.163337185	1.41127761	12.23879876	3.69586601
minimize (Nelder-Mead)	164.8407936447	-0.02386649	11.37364401	5.64646295
SCE algorithm	164.840794	-0.02386639	11.37362137	5.64647953

Table 3.2: Table displays the different results for the Waves dataset obtained using different algorithms and methods.

Aside from the SCE algorithm, the best results were given by nnlf and Nelder-Mead. It should be noteworthy that Nelder-Mead served as an inspiration in the creation of the original SCE algorithm[10].

3.3 Tuning up the models

For the Heights dataset, the first models considered were simply stationary models, using the Gumbel distribution and the GEV distribution with just 3 parameters. After that, non-stationary values were used to parameterise the location parameter. The same was done with the scale parameter and finally, with the shape parameter, ending up with the 9 parameter model seen in 2.4. At this point it should be noted that comparisons made with nnlf are nearly useless, since its internal calculations do not evaluate the log likelihood in the analytical way, so seasonality can't be analysed. The data obtained, alongside the deviation obtained using the Fisher information matrix (1.4.1) and the results for the χ^2 and AIC tests can be found in Table 3.3.

N	2	3	5	7	9
ℓ	662.8095	650.5278	440.6001	378.561	367.212
AIC test	-1321.619	-1295.0556	-871.2002	-743.122	-716.424
$\ell_i - \ell_j$	–	12.2817	209.9276	62.039	11.348
$\frac{1}{2}\chi^2(1-p)$	–	0.1101	47.5103	4.829	0.167
μ_0/m	2.874 ± 0.0024	2.733 ± 0.003	3.0548 ± 0.0010	3.0398 ± 0.0009	3.0625 ± 0.0011
μ_1/m	–	–	1.077 ± 0.003	1.0282 ± 0.0016	1.030 ± 0.002
μ_2/m	–	–	0.3589 ± 0.0014	0.2984 ± 0.0014	0.3286 ± 0.0018
σ_0/m	0.9557	0.8167 ± 0.0019	0.5805 ± 0.0005	0.5435 ± 0.0002	0.5580 ± 0.0006
σ_1/m	–	–	–	0.32461 ± 0.00015	0.3512 ± 0.0010
σ_2/m	–	–	–	0.0513 ± 0.0009	0.0361 ± 0.0009
ζ_0	0	0.2955 ± 0.004	0.0436 ± 0.0007	0.0505 ± 0.0006	-0.0279 ± 0.0009
ζ_1	–	–	–	–	0.090 ± 0.003
ζ_2	–	–	–	–	-0.216 ± 0.002

Table 3.3: Table shows the different results obtained for the parameters for each type of model. For the $\ell_i - \ell_j$, ℓ_i refers to the prior model compared to the evaluated.

As an easy way to see how the algorithm works for 9 variables, Figure 3.1 displays a typical run for the Heights dataset.

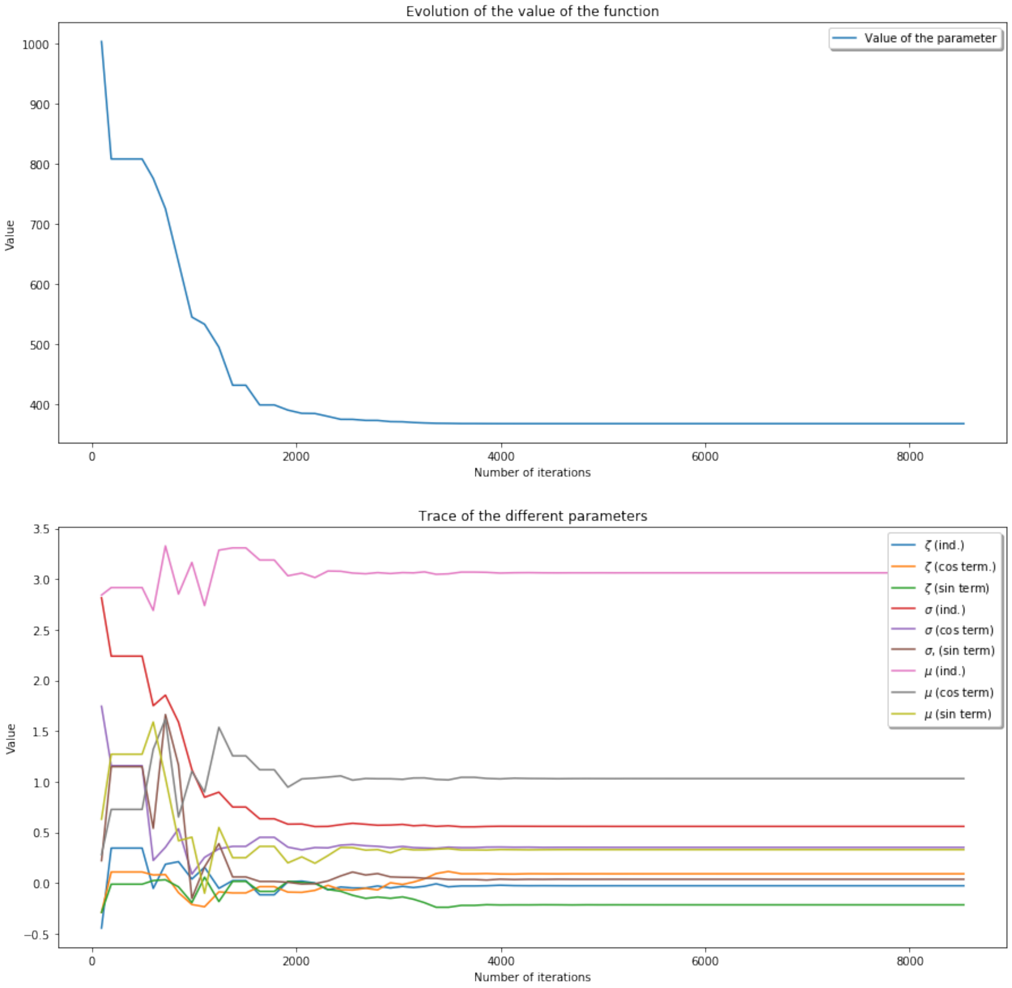


Figure 3.1: Figure shows the evolution of values with the iteration for executing the SCE algorithm for 9 variables. The upper side shows the evolution for the value of the log likelihood, while the lower part represents the evolution of the parameters.

Once these results were obtained, several plots can be made using these results, which will also combine the plot in Figure 2.1 in order to make it easier to follow. The return value has also been calculated using eq.(1.9), as seen in in figure 3.2.

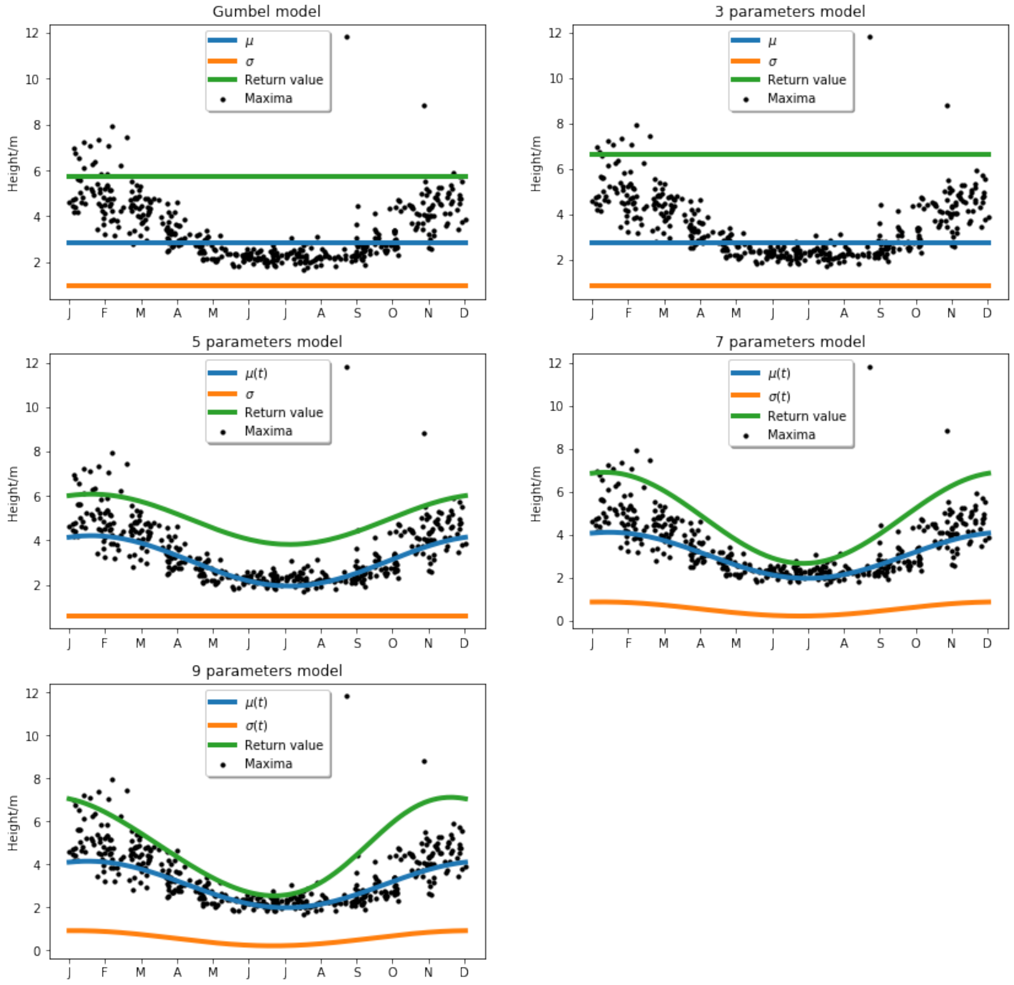
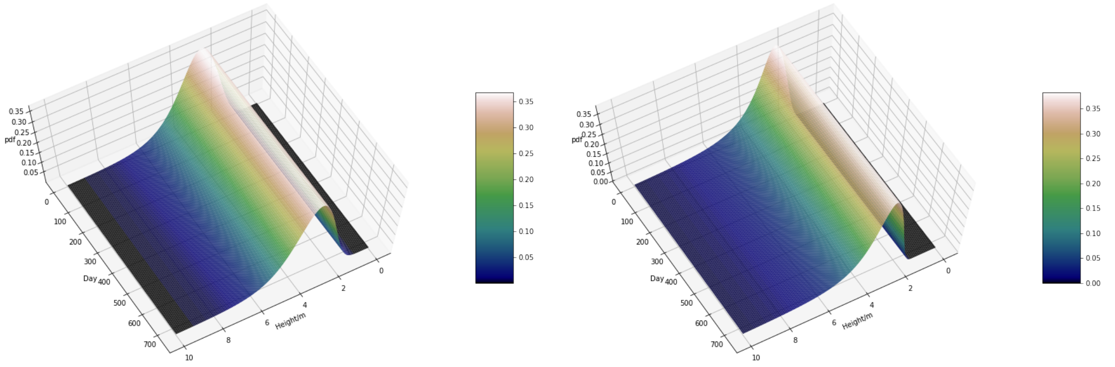


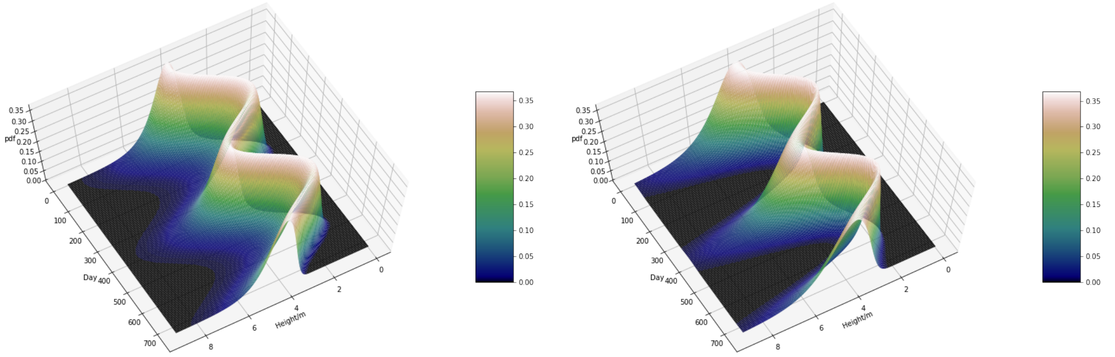
Figure 3.2: Figure shows the location parameter, the return value for the 0.95 quantile and the scale parameter using each of the models featured in Table 3.3

Once this was achieved, a final set of 3D plots can be made, using the probability density function for GEV distributions (eq.(1.4)) in a 3 dimensional plot that shows the pdf for any day of the year, along with the height of waves for each of the models obtained above.



(a) (b)

Figure 3.3: The plots display the probability distribution function over 2 years for the stationary models with two (left, Gumbel model) and three (right) parameters, using eq.(1.4) with the results from Table 3.3.



(a) (b)

Figure 3.4: Again, the plots display the probability distribution function over 2 years for the models with 5 (left) and 7 (right) parameters, using eq.(1.4).

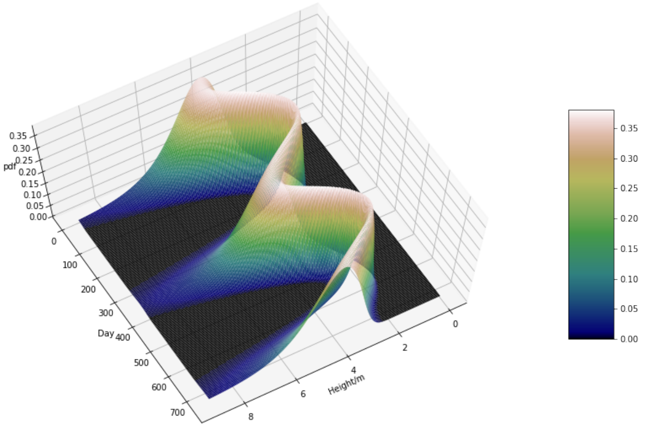


Figure 3.5: Finally, figure shows the pdf plot for the model with 9 variables, similar to how it was obtained for the plots in Figures 3.3 and 3.4.

Given the results for the tests presented in Table 3.3, the conclusion is that the model with 9 parameters (Figure 3.5) is the one that better suits the dataset provided, that is, a fully non-stationary model. The stationary models (Figure 3.3) can be discarded, not only due to the poor results obtained in the tests, but more simply, because for a non stationary data distribution (Figure 3.2), a non stationary probability function is expected. Obtaining the same value for the probability density during a year for the dataset is simply not acceptable.

Between the non stationary models the differences are harder to spot. The influence of a non-stationary value for the scale parameter can be appreciated by comparing the plots in Figure 3.4, but the influence of a non-stationary shape parameter is much subtle. This is also coherent with the difference between the scores in the tests. These differences are continuously decreasing as the number of parameters go higher.

3.4 Temperature analysis

So far, in choosing the model only one dataset (Heights) has been used. Other extreme values distributions for climatic phenomena are suitable for this task. Given the case, the monthly maxima temperatures were also analysed in a similar way. In this case, since the range of temperatures registered ranged from 120 to nearly 400, the normalisation procedure explained in 2.2 is almost mandatory.

After using the algorithm with the temperatures dataset, used the same methodology from the Heights dataset, the parameters obtained, as well as the results for the log likelihood and the proposed tests can be found in Table (3.4). The shown results are in the normalised space, as trying to use the algorithm in the Fisher matrix with the original data based on the output for the normalised space does not have any meaning. Also, further variable changes may lead to outliers (see eq.(1.7)).

N	2	3	5	7	9
ℓ	7963.643	7189.859	2607.158	2596.823	2592.078
AIC test	-15923.287	-14373.720	-5204.317	-5179.648	-5166.158
$\ell_i - \ell_j$	–	773.783	4582.701	10.334	4.745
$\frac{1}{2}\chi^2(1-p)$	–	41.637	4027.593	0.0205	$4.3 * 10^{-3}$
μ_0/m	0*	-0.351 ± 0.015	-0.214 ± 0.006	-0.214 ± 0.006	-0.2135 ± 0.0010
μ_1/m	–	–	-1.208 ± 0.008	-1.223 ± 0.009	-1.217 ± 0.002
μ_2/m	–	–	-0.424 ± 0.008	-0.417 ± 0.006	-0.423 ± 0.009
σ_0/m	1.113*	0.982 ± 0.011	0.5805 ± 0.0005	0.3839 ± 0.0008	0.3838 ± 0.0005
σ_1/m	–	–	–	-0.01732 ± 0.00012	-0.021 ± 0.003
σ_2/m	–	–	–	0.020 ± 0.006	0.016 ± 0.008
ζ_0	0	-0.289 ± 0.010	-0.184 ± 0.007	-0.186 ± 0.007	0.1865 ± 0.0003
ζ_1	–	–	–	–	-0.026 ± 0.003
ζ_2	–	–	–	–	0.040 ± 0.006

Table 3.4: Table shows the different results obtained for the parameters in the normalised space for each type of model for the temperature dataset. Again, for the $\ell_i - \ell_j$, ℓ_i refers to the prior model compared to the evaluated. The two values marked with * returned an error value so low that it was neglected.

The parameters, as well as the corresponding expected result were plotted for each model as in Figure 3.6, this time, back in the original space after applying the requited transformations.

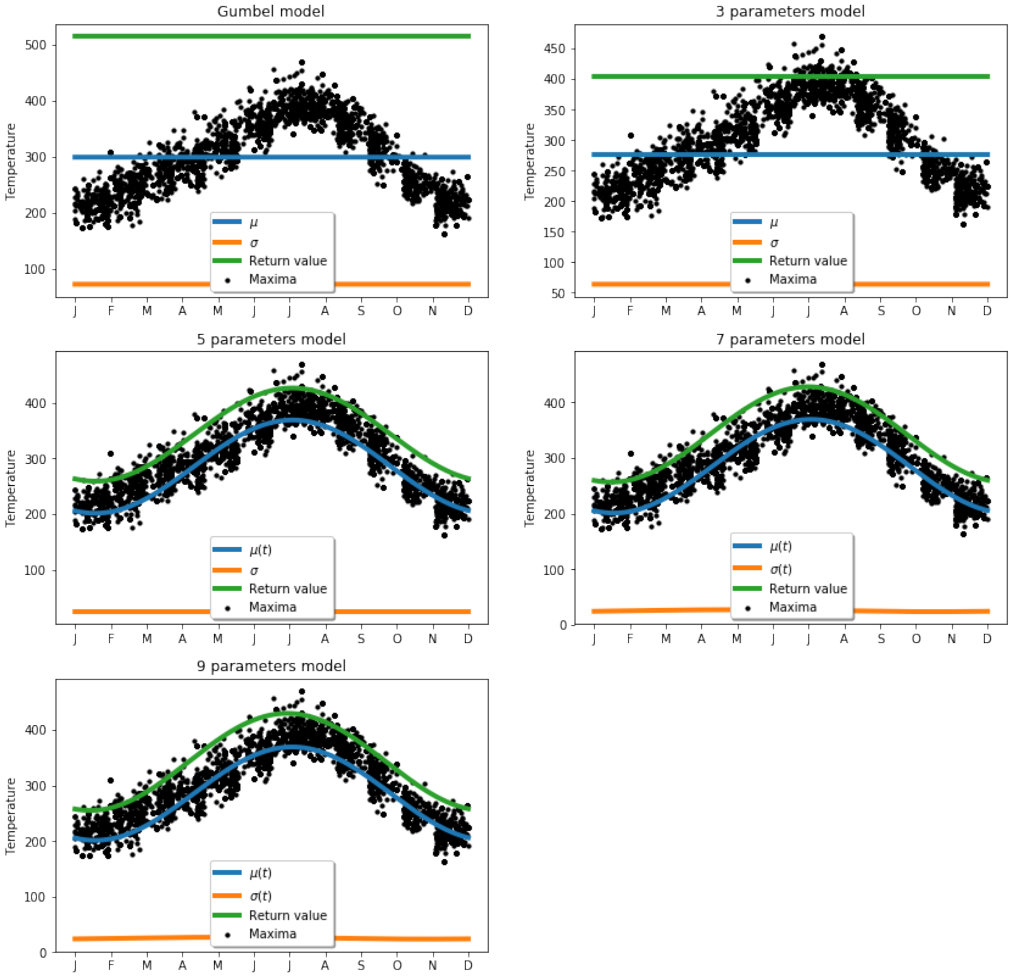


Figure 3.6: Figure shows the results for the temperature datasets, in the same fashion as before. Even if the scale parameters seem to be flat, it should be noted that their value is way below the other values. Thus, from 7 and 9 variables $\sigma(t)$ also obeys a sinusoidal distribution.

The pdf for 2 years, as done before, and with the best result, again, with nine variables (as seen in Table (3.4)) can be found in Figure 3.7. Again, it's easy to see the need for a non-stationary model for these results.

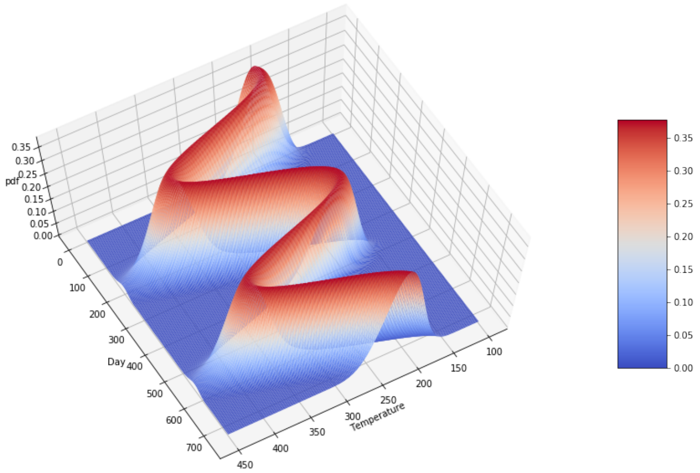


Figure 3.7: Figure shows the pdf for the GEV distribution using the best parameters found using the SCE algorithm with the temperature dataset.

Chapter 4

Final remarks and acknowledgements

4.1 Conclusion

After providing the insights regarding extreme values distributions, the SCE algorithm was introduced in order to obtain the best parameters that suit the distribution for a given dataset, with very good results in several datasets.

As mentioned in Table 3.2, the algorithm is way more robust than any other basic Python libraries for even simple datasets, which some of them not being able to return results, let alone more complex functions or datasets. The use of normalisation is not a problem for the algorithm as seen when the Temperature dataset was evaluated, albeit it may lead to confusion when evaluating the results using the Fisher information matrix.

The algorithm was proven to be way flexible, as the addition of new terms, large sets of data and even sinusoidal expressions were not a problem to compute.

All in all, the SCE algorithm is quite a powerful tool in the analysis of statistical functions and should be taken into account even for simpler cases, since the only thing that is required is to modify the needed function and the dataset.

4.2 Annexed content

The items that can be found are the following:

- The proper SCE algorithm, consisting on the modules mentioned in (2.5). Other algorithms mentioned are omitted, either because of their simplicity (greedy) or simply because they are already built in Python via SciPy.

- All the datasets mentioned above in non-proprietary format (.csv and .txt), both raw and curated. The final versions are in .npy format.
- The scripts used to curate the data in order to make them reproducible.
- A Jupyter Notebook showing some examples of the algorithm.
- The modules containing the Fisher information matrix and the tests.

In other words, every file in 2.2 is available in the final version (CD) as annexed content.

4.3 Acknowledgements.

Thanks to Fernando Méndez for his labour as director of this work, as well as Nicolás Ripoll for his work as codirector and for all the programming advice he gave in hard moments, which weren't few, to say the least.

Thanks to Pablo Gómez, who actually took the effort to present this and save me quite the trip.

And of course, all my family and friends, always present during this last year.

Bibliography

[1] Fernando J Méndez, Melisa Menéndez, Alberto Luceño, and Inigo J Losada. Analyzing monthly extreme sea levels with a time-dependent gev model. *Journal of Atmospheric and Oceanic Technology*, 24(5):894–911, 2007.

[2] Cristina Izaguirre, Fernando J Méndez, Melisa Menéndez, and Inigo J Losada. Global extreme wave height variability based on satellite data. *Geophysical Research Letters*, 38(10), 2011.

[3] Jacob Schewe, Simon N Gosling, Christopher Reyer, Fang Zhao, Philippe Ciais, Joshua Elliott, Louis Francois, Veronika Huber, Heike K Lotze, Sonia I Seneviratne, et al. State-of-the-art global models underestimate impacts from climate extremes. *Nature communications*, 10(1):1005, 2019.

[4] Stuart Coles. *An introduction to statistical modeling of extreme values*. Springer Series in Statistics. Springer-Verlag, London, 2001.

[5] Joel Grus. *Data Science from Scratch: First Principles with Python*. O'Reilly, 1 edition, 2015.

[6] Trevor Hastie Robert Tibshirani Gareth James, Daniela Witten. *An Introduction to Statistical Learning with Applications in R*. Springer US, 1 edition, 2013.

[7] Paul W. Vos Robert E. Kass. *Geometrical Foundation of Asymptotic Inference*. Wiley Interscience, 1 edition, 1997.

[8] Melisa Menéndez, Fernando J Méndez, Cristina Izaguirre, Alberto Luceño, and Inigo J Losada. The influence of seasonality on estimating return values of significant wave height. *Coastal Engineering*, 56(3):211–219, 2009.

[9] N. Balakrishnan Jose M. Sarabia Enrique Castillo, Ali S. Hadi. *Extreme Value and Related Models with Applications in Engineering and Science*. Springer US, 1 edition, 2004.

[10] QY Duan, Vijai K Gupta, and Soroosh Sorooshian. Shuffled complex evolution approach for effective and efficient global minimization. *Journal of optimization theory and applications*, 76(3):501–521, 1993.

[11] Wikipedia the Free Encyclopedia. Rastrigin function. `https://en.wikipedia.org/wiki/Rastrigin_function`, 2018. [Online; accessed 20-May-2019].

[12] Sonja Surjanovic and Derek Bingham. Virtual Library of Simulation Experiments: Test Functions and Datasets - Goldstein-Price function. `https://www.sfu.ca/~ssurjano/goldpr.html`, 2013. [Online; accessed 20-May-2019].

[13] Tianjie Lei et all Guangyuan Kan. A multi-core CPU and many-core GPU based fast parallel shuffled complex evolution global optimization approach. `https://www.researchgate.net/figure/Flow-chart-of-the-CCE-algorithm_fig2_303982322`, 2016. [Online; accessed 19-May-2019].

[14] SciPy.org. scipy.stats.rv_continuous.nnlf. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.rv_continuous.nnlf.html`, 2019. [Online; accessed 2-May-2019].

[15] Ali Saeb. A note on power generalized extreme value distribution and its properties, 2017.

[16] Melisa Menéndez. *Metodología para el análisis estadístico no estacionario de valores extremos de variables geofísicas.* PhD thesis, University of Cantabria, Departamento de ciencias y técnicas del agua y del medio ambiente, 2 2008.

[17] Stijn Van Hoey. Optimization algorithm SCE. `https://github.com/stijnvanhoey/Optimization_SCE`, 2011. [Online; accessed 17-Apr-2019].