Université de Lille
Faculté des Sciences et Technologies
Département de Mathématiques
Master I MAS

TRAVAIL ENCADRÉ DE RECHERCHE

Sujet :

La température annuelle moyenne au Maroc

Réalisé par :

- OUFASKA *Abdelmonssif*
- OUDGHIRI IDRISSI *Noura*

Encadré par :

- *Prof.* DERMOUNE *Azzouz*

Membres de Jury :

- *Prof.* ....
- *Prof.* ....
- *Prof.* ....

Année Universitaire : 2018/2019

Remerciement

Au terme de ce travail, nous adressons nos remerciements les plus sincères à notre encadrant Monsieur DERMOUNE Azzouz pour sa disponibilité, son aide, ses conseils précieux, ses critiques constructives, ses explications et suggestions pertinentes ainsi que pour ses qualités humaines et morales que nous avons toujours appréciées.

Table des matières

Introduction

Dans le cadre de notre projet de recherche durant le Master I ingénierie Statistique et Numérique, nous nous sommes intéressés à l'évolution de la température annuelle moyenne au Maroc entre 1900 et 2018. Étant donné que ne disposons que d'une table contenant les températures annuelles moyennes au Maroc entre 1900 et 2018, notre premier réflexe a été d'effectuer une statistique descriptive sur cette base de donnée à l'aide du langage R (moyennes,médianes, variances, boîtes à moustaches). Ensuite, nous nous sommes intéressés particulièrement à l'évolution de ces températures sur cette même période et nous avons essayer de voir si ces données suivent une loi usuelle, éventuellement une loi normale. Pour ceci, on va utiliser un QQ-plot et en suite on va appliquer le test de Shapiro-Wilk pour vérifier la normalité des données.Si c'est le cas, on effectuerait alors une régression linéaire, afin de déterminer les coefficients qui nous permettraient de prédire la température pour l'année suivante. Afin de prédire la température pour l'année suivante, nous appliquons la méthode d'interpolation à noyau, nous nous intéressons essentiellement à la matrice du bruit gaussien fractionnaire d'indice de Hurst qu'on va essayer de déterminer de façon à ce que la prévision calculée colle le plus possible avec la réalité...

Chapitre 1

Analyse du Tableau des Températures moyennes annuelles

1.1 Introduction

Dans ce chapitre, dans un premier temps on s'intéresse à l'analyse descriptive de nos données. A l'aide de la fonction *summary* du langage R, on obtient la moyenne des températures annuelles au Maroc, ainsi que la médiane, Min, Max...ceci nous donne une première idée sur cette moyenne, ensuite on fait un graphe afin de bien visualiser l'évolution de la température entre 1900 et 2018. La boite à moustache nous permet également d'avoir une idée sur la valeur de la température autour de laquelle se concentrent les températures des cents dix-huit années.

Ensuite , étant donnée notre base de données, on s'est demandé , si elle suit une loi normale. Pour vérifier ceci ; on utilise un $QQ-plot$, suivi d'un test de Shapiro-Wilk pour confirmer ou rejeter cette hypothèse.

Dans un second temps, on fait une régression linéaire, et on regarde le coefficient de détermination R^2 afin de voir si éventuellement on pouvait compter sur ce modèle pour prédire la température de l'année suivante.

1.2 Table de Données

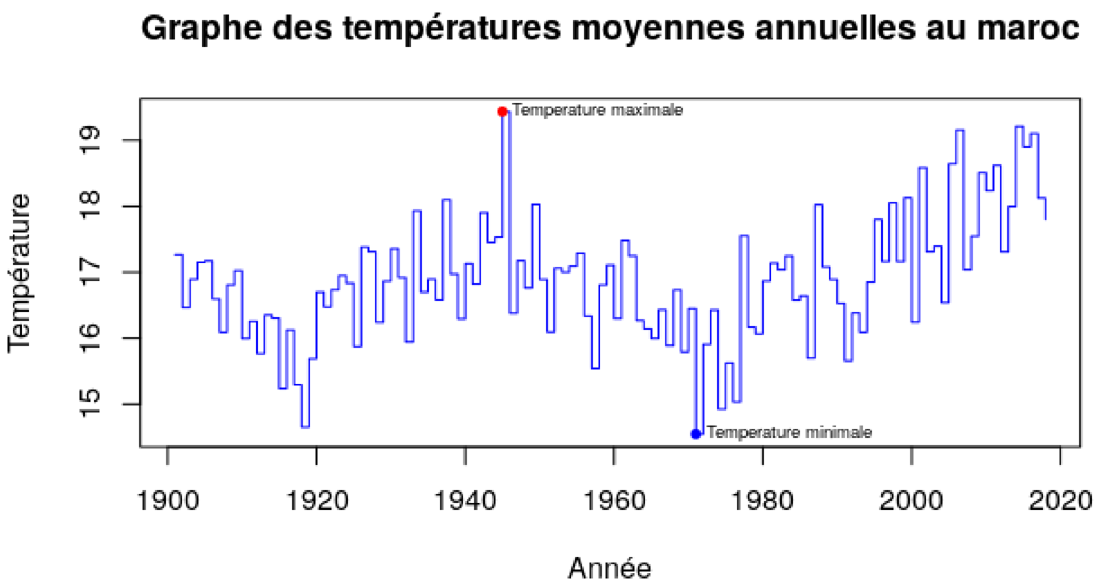
	Année	Température
1	1901	17.26595
2	1902	16.46445
3	1903	16.89110
4	1904	17.15045
5	1905	17.17115
6	1906	16.59410
7	1907	16.09055
8	1908	16.80720
9	1909	17.02400
10	1910	15.99565
11	1911	16.25630
12	1912	15.76695
13	1913	16.35055
14	1914	16.30280
15	1915	15.23885
16	1916	16.12285
17	1917	15.29395
18	1918	14.65460
19	1919	15.68985
20	1920	16.69855
21	1921	16.47455
22	1922	16.73505
23	1923	16.94875
24	1924	16.83405
25	1925	15.87310
26	1926	17.37765
27	1927	17.31120
28	1928	16.24045
29	1929	16.86560
30	1930	17.35415
31	1931	16.92065
32	1932	15.94330
33	1933	17.92855
34	1934	16.69940
35	1935	16.89310
36	1936	16.57905
37	1937	18.09945
38	1938	16.97220
39	1939	16.29280
40	1940	17.12830
41	1941	16.82185
42	1942	17.90155
43	1943	17.45370
44	1944	17.53185
45	1945	19.43390
46	1946	16.37885
47	1947	17.17460
48	1948	16.76200
49	1949	18.02585
50	1950	16.88955
51	1951	16.09340
52	1952	17.06150
53	1953	17.00045
54	1954	17.09255
55	1955	17.28915
56	1956	16.33480
57	1957	15.54125
58	1958	16.80745
59	1959	17.10650
60	1960	16.29705
61	1961	17.47970
62	1962	17.24790
63	1963	16.26875
64	1964	16.14225
65	1965	15.99710
66	1966	16.42955
67	1967	15.88720
68	1968	16.73295
69	1969	15.79380
70	1970	16.44775
71	1971	14.54600
72	1972	15.90555
73	1973	16.42725
74	1974	14.92625
75	1975	15.62540
76	1976	15.03310
77	1977	17.55335
78	1978	16.16550
79	1979	16.06435
80	1980	16.86770
81	1981	17.13790
82	1982	17.03770
83	1983	17.24490
84	1984	16.57530
85	1985	16.64185
86	1986	15.69875
87	1987	18.02020
88	1988	17.08010
89	1989	16.88845
90	1990	16.52590
91	1991	15.65780
92	1992	16.38310
93	1993	16.09055
94	1994	16.84880
95	1995	17.79865
96	1996	17.16175
97	1997	18.04950
98	1998	17.16170
99	1999	18.12820
100	2000	16.24315
101	2001	18.58385
102	2002	17.31125
103	2003	17.39330
104	2004	16.53990
105	2005	18.64455
106	2006	19.15395
107	2007	17.03765
108	2008	17.54505
109	2009	18.51130
110	2010	18.23800
111	2011	18.61795
112	2012	17.30885
113	2013	17.99565
114	2014	19.20845
115	2015	18.90080
116	2016	19.10102
117	2017	18.12502
118	2018	17.80022

1.3 Statistiques

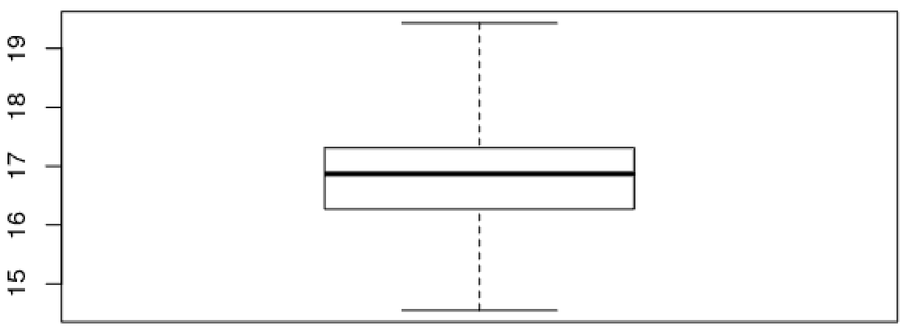
On effectue une analyse descriptive uni-variée pour les températures, on obtient les résultats suivants :

```
> View(Morocco)
> t=Morocco$Température
> summary(t)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  14.55   16.27   16.87   16.88   17.31   19.43
> 
```

1.4 Graphe des températures moyennes annuelles

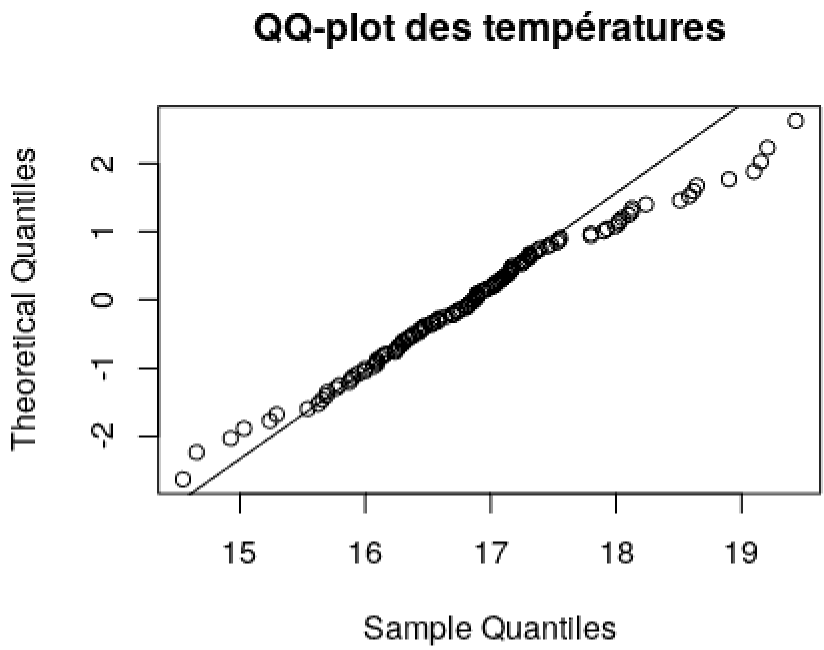


1.5 Boite à moustache



La boîte à moustaches nous aide à obtenir rapidement un aperçu des données. Ainsi, on peut voir directement que la médiane est égale à 16,8 et que 25% des données sont inférieurs à 16.3 degrés avec une valeur minimale de 14.5 degrés, et que également 25% des données sont supérieur à 17.31 degrés avec une valeur maximale de 19.43 degrés.

1.6 QQ-plot



On remarque que les points sont presque tous alignés sur la droite, ceci nous montre que nos données pourraient suivre une loi normale $\mathcal{N}(\mu, \sigma^2)$, pour vérifier cette hypothèse on va appliquer le test de Shapiro-Wilk.

1.7 Test de Shapiro-Wilk

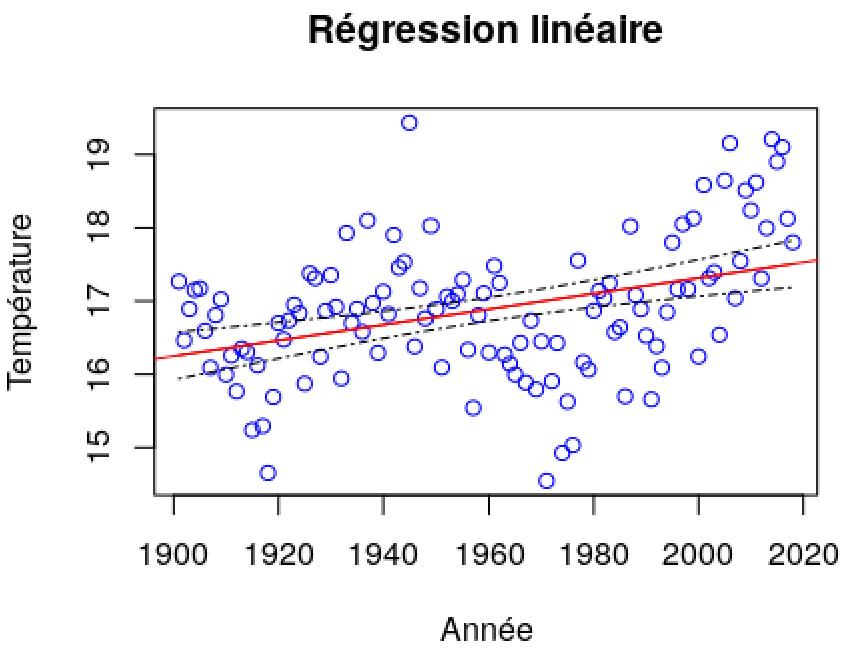
Le test de Shapiro-Wilk est l'un des tests permettant de vérifier la normalité d'une variable X. Il utilise le rapport de deux estimations de la variance. Dans le cas d'une variable normale, ces deux estimations coïncident et le rapport est voisin de 1, alors que si la variable n'est pas normale le rapport est plus petit que 1.

```
> ######## Test de shapiro-Wilk ########
> t=Morocco$Température
> shapiro.test(t)

        Shapiro-Wilk normality test

data:  t
W = 0.98357, p-value = 0.1603
```

Le test de Shapiro-Wilk donne une probabilité de dépassement de 0.1683, supérieure à 0.05, donc l'hypothèse de normalité est vérifiée.

1.8 Régression linéaire

La régression linéaire va nous permettre de calculer la tendance linéaire contenue dans la température annuelle moyenne. Elle permet également de représenter la température comme la somme d'une partie linéaire et d'un résidu i.e. sous la forme $T_j = \beta_0 + \beta_1 j + e_j$.



Résultats :

```
Residuals:
     Min       1Q   Median       3Q      Max
-2.45990 -0.58599  0.01351  0.54076  2.70515

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.004072   4.671512  -0.857    0.393
annee        0.010660   0.002384   4.472 1.82e-05
```

$\beta_0 = -4.004072$
$\beta_1 = 0.010660$
$\boldsymbol{R}^2 = 0.147$
$\boldsymbol{R}^2 ajust = 0.1397$

On remarque que le $\beta_1 = 0,01 > 0$, ce qui montre bien un échauffement dans les années qui suivent.
Cependant, on remarque que le coefficient de détermination $R^2 = 0.14$ qui est proche de 0 et la droite de régression ne détermine dans ce cas que 14% de la distribution des points. Alors on ne peut pas compter sur cette méthode pour prédire la température moyenne annuelle au Maroc pour l'année 2019.

Chapitre 2

Interpolation et Prévision

2.1 Introduction

L'objet de cette partie est de présenter les techniques permettant de prédire la température moyenne annuelle au Maroc.

L'interpolation et la prédiction à partir d'une base de données interviennent dans diverses applications. L'interpolation à noyau (Kernel) et la prédiction gaussienne sont équivalentes mais elles ont des interprétations différentes. Les deux approches sont basées sur la connaissance du noyau ou de la matrice de covariance. Dans l'approche stochastique, la matrice de covariance et la tendance sont estimées en utilisant la méthode du maximum de vraisemblance ou la méthode Bayésienne. Dans notre TER nous allons utiliser les matrices de covariance des bruits gaussiens fractionnaires paramétrées avec un indice $H \in]0,1[$, qu'on appelle indice de Hurst.

2.2 Motivation

La formulation mathématique du problème est la suivante :
Soit X un ensemble non vide, $\{x_1, \ldots, x_n, x_{n+1}\} \subset X$. On se donne une fonction $f : X \to \mathbb{R}$ avec $f(x_1), \ldots, f(x_n)$ connues. On se propose de prédire la valeur $f(x_{n+1})$.

Dans ce TER, on se propose de prédire la température moyenne annuelle au Maroc de l'année suivante en utilisant la modélisation stockastique.

2.3 Modélisation stochastique

2.3.1 Modèle centré

On pose $y_i = f(x_i)$ pour $i = 1, \ldots, n+1$.

On suppose que (y_1, …, y_{n+1}) est une réalisation d'un vecteur gaussien aléatoire centré $(Y_1, Y_2, ...Y_{n+1})$ de matrice de covariance $K = [k(i,j) : i,j = 1, \ldots, n+1]$ connue.

Théorème

$\mathbb{E}[Y_{n+1}|Y_1, \ldots, Y_n] = \sum_{i=1}^{n} w_i Y_i$, $w_i, \ldots, w_n$ sont uniques.

On rappelle que l'espérance conditionnelle de Y_{n+1} sachant $Y_1, Y_2, ..., Y_n$ est :

$$\mathbb{E}[Y_{n+1}|Y_1, Y_2, ...Y_n] = (k(n+1,1), ..., k(n+1,n))[k(i,j) : i,j = 1, ..., n]^{-1} \begin{pmatrix} Y_1 \\ Y_2 \\ . \\ . \\ Y_n \end{pmatrix}$$

$$= (w_1, ..., w_n) \begin{pmatrix} Y_1 \\ Y_2 \\ . \\ . \\ Y_n \end{pmatrix}$$

Preuve :

$\mathbb{E}[Y_{n+1}|Y_1, \ldots, Y_n] = \sum_{i=1}^{n} w_i Y_i$

$Y_j \mathbb{E}[Y_{n+1}|Y_1, \ldots, Y_n] = \sum_{i=1}^{n} w_i Y_j Y_i$ pour $j = 1, \ldots, n$

$\mathbb{E}[Y_j \mathbb{E}[Y_{n+1}|Y_1, \ldots, Y_n]] = \sum_{i=1}^{n} w_i \mathbb{E}[Y_j Y_i]$ pour $j = 1, \ldots, n$

Comme Les Y_i sont centrés, donc $\mathbb{E}[Y_i] = 0 \quad \forall i \in \{1, \ldots, n+1\}$

Alors $\mathbb{E}[Y_j Y_i] = cov(Y_j, Y_i) = K(i,j) \quad i,j = 1, \ldots, n+1$

On a $\mathbb{E}[Y_j \mathbb{E}[Y_{n+1}|Y_1,\ldots,Y_n]] = \mathbb{E}[\mathbb{E}[Y_j Y_{n+1}|Y_1,\ldots,Y_n]] \quad$ pour $\forall j = 1,\ldots,n$

Selon la propriété suivante : Si Y est G-mesurable alors, $\quad Y\mathbb{E}[X|G] = \mathbb{E}[YX|G]$.

$\mathbb{E}[\mathbb{E}[Y_j Y_{n+1}|Y_1,\ldots,Y_n]] = E[Y_j Y_{n+1}] = \sum_{i=1}^{n} w_i K(i,j) \quad$ pour $\forall$j=1,. . .,n

$K(i,j) = \sum_{i=1}^{n} w_i K(i,j) \quad$ pour $j = 1,\ldots,n$

$(w_1,\ldots,w_n) = (k(n+1,1),...,k(n+1,n))[k(i,j) : i,j = 1,...,n]^{-1}$

Quelles sont les meilleurs constantes $a_1, a_2, ..., a_n$ qui minimisent l'erreur quadratique $\min\limits_{a_1,a_2,\ldots,a_n} \mathbb{E}[|Y_{n+1} - \sum_{i=1}^{n} a_i Y_i|^2]$, au sens L^2 ?

$\mathbb{E}[(Y_{n+1} - \sum_{i=1}^{n} a_i Y_i)^2] = \mathbb{E}[(Y_{n+1})^2 - 2y_{n+1}\sum_{i=1}^{n} a_i Y_i + (\sum_{i=1}^{n} a_i Y_i)^2]$

$= \mathbb{E}[Y_{n+1}^2 - 2\sum_{i=1}^{n} a_i Y_{n+1} Y_i + \sum_{i=1}^{n}\sum_{j=1}^{n} a_i Y_i a_j Y_j]$

$= \mathbb{E}[Y_{n+1}^2] - 2\sum_{i=1}^{n} a_i \mathbb{E}[Y_{n+1} Y_i] + \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \mathbb{E}[Y_i Y_j]$

$= K(n+1,n+1) - 2\sum_{i=1}^{n} a_i K(n+1,i) + \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j K(i,j)$

$= K(n+1,n+1) - 2\sum_{i=1}^{n} a_i K(n+1,i) + \sum_{i=1}^{n} a_i^2 k_(i,i) + 2\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} a_i a_j K(i,j)$

$\frac{\partial \mathbb{E}[(Y_{n+1} - \sum_{i=1}^{n} a_i Y_i)^2]}{\partial a_i} = -2K(n+1,i) + 2a_i K(i,i) + 2\sum_{j=1,i\neq j}^{n} a_j K(i,j),\ i = 1,..,n$

$\frac{\partial \mathbb{E}[(Y_{n+1} - \sum_{i=1}^{n} a_i Y_i)^2]}{\partial a_i} = -2K(n+1,i) + 2\sum_{j=1}^{n} a_j K(i,j),\ i = 1,..,n$

$\frac{\partial \mathbb{E}[(Y_{n+1} - \sum_{i=1}^{n} a_i Y_i)^2]}{\partial a_i} = 0,\ i = 1,..,n, \Longrightarrow 2K(n+1,i) + 2\sum_{j=1}^{n} a_j K(i,j) = 0,\ i = 1,..,n,$

$\Longrightarrow K(n+1,i) = \sum_{a_j}^{n} a_j K(i,j),\ i = 1,..,n,$

$\Longrightarrow (a_1,\ldots,a_n) = (k(n+1,1),...,k(n+1,n))[k(i,j) : i,j = 1,...,n]^{-1} = (w_1,\ldots,w_n)$

Alors les poids $w_1, w_2, ...w_n$ sont les meilleurs constantes qui minimisent l'erreur quadratique $\min\limits_{a_1,a_2,\ldots,a_n} \mathbb{E}[|Y_{n+1} - \sum_{i=1}^{n} a_i Y_i|^2]$, au sens L^2.

D'après ceci, on peut donner un meilleur estimateur pour Y_{n+1} défini par :

$$\hat{Y}_{n+1} = \sum_{i=1}^{n} \omega_i Y_i \quad (1)$$

2.3.2 Modèle avec Tendance

On suppose que $\forall i \in \{1, \ldots, n+1\}$, la moyenne $\mathbb{E}(Y_i) = \sum_{k=1}^{p} \beta_k f_k(i)$, avec $f_1, \ldots, f_p$ sont des fonctions données et $\beta_1, \ldots, \beta_p$ sont des paramètres inconnus.

Nous cherchons la prévision $\sum_{i=1}^{n} w_i Y_i$ de Y_{n+1} sans biais et qui minimise l'erreur quadratique $\mathbb{E}[|Y_{n+1} - \sum_{i=1}^{n} a_i Y_i|^2]$.

Définition

$\sum_{i=1}^{n} w_i Y_i$ est une approximation sans biais de Y_{n+1} si $\mathbb{E}(\sum_{i=1}^{n} w_i Y_i) = \mathbb{E}(Y_{n+1})$.

Proposition

Si la moyenne $\mathbb{E}(Y_i) = \sum_{k=1}^{p} \beta_k f_k(i) \forall i \in 1, ..., n+1$, La contrainte sans biais est équivalent à dire que :

$$\sum_{i=1}^{n} w_i (\sum_{k=1}^{p} \beta_k f_k(i)) = \sum_{k=1}^{p} \beta_k f_k(n+1)$$

Cette égalité a lieu pour tous les paramètres $\beta_1, \ldots, \beta_p$.

Finalement, nous obtenons les contraintes :

$$\sum_{i=1}^{n} w_i f_k(i) = f_k(n+1), \ k = 1, \ldots, p.$$

Le problème $\inf_{a_1,\ldots,a_n} \{\mathbb{E}[|Y_{n+1} - \sum_{i=1}^{n} a_i Y_i|^2]\}$ sous les contraintes $\sum_{i=1}^{n} a_i f_k(i) = f_k(n+1), \ k = 1, \ldots, p,$ est équivalent d'après notre cours d'optimisation au système linéaire (2) :

$$(2) \begin{cases} \sum_{i=1}^{n} a_i K(i,j) + \sum_{k=1}^{p} \lambda_k f_k(j) = K(n+1,j), & j = 1, \ldots, n, \\ \sum_{i=1}^{n} a_i f_k(i) = f_k(n+1), & k = 1, \ldots, p. \end{cases}$$

Avec $\lambda_1, \ldots, \lambda_n$ sont les multiplicateurs de Lagrange.

On résout le système et on note $w_1, \ldots, w_n, \lambda_1, \ldots, \lambda_n$ sa solution. La prévision, est alors donnée par : $\sum_{i=1}^{n} w_i y_i$.

2.4 Application : en utilisant le bruit gaussien factionnaire

2.4.1 Le bruit gaussien fractionnaire

Définition :

Un vecteur gaussien $(X_1, ...X_{n+1})$centré de matrice de covariance K_H est dit bruit gaussien fractionnaire d'indice de Hurst H si :

$$\boxed{cov(X_i, X_j) = \frac{\sigma^2}{2}[|i-j+1|^{2H} - 2|i-j|^{2H} + |i-j-1|^{2H}] := \gamma(|i-j|), \quad i, j = 1, \ldots, n+1}$$

NB : Pour $H = \frac{1}{2}$, $K_H = \mathbf{I}_{n+1}$ matrice de covariance de bruit blanc.

Nous allons utiliser cette matrice dans tous les calculs qui suivent (regarder l'appendice pour savoir un peu plus sur son histoire).

Simulation du bruit gaussien fractionnaire :

Théorème :

Soit $Z = (Z_1, \ldots, Z_d)^\top$ un vecteur gaussien. On note $m = \mathbb{E}(Z)$ et $\Sigma = \mathrm{Var}(Z)$, on a pour toute matrice A possédant d colonnes et pour tout vecteur $b \in \mathbb{R}^d$
$AZ + b \sim \mathcal{N}\left(Am + b, A\Sigma A^\top\right)$.

Théorème :

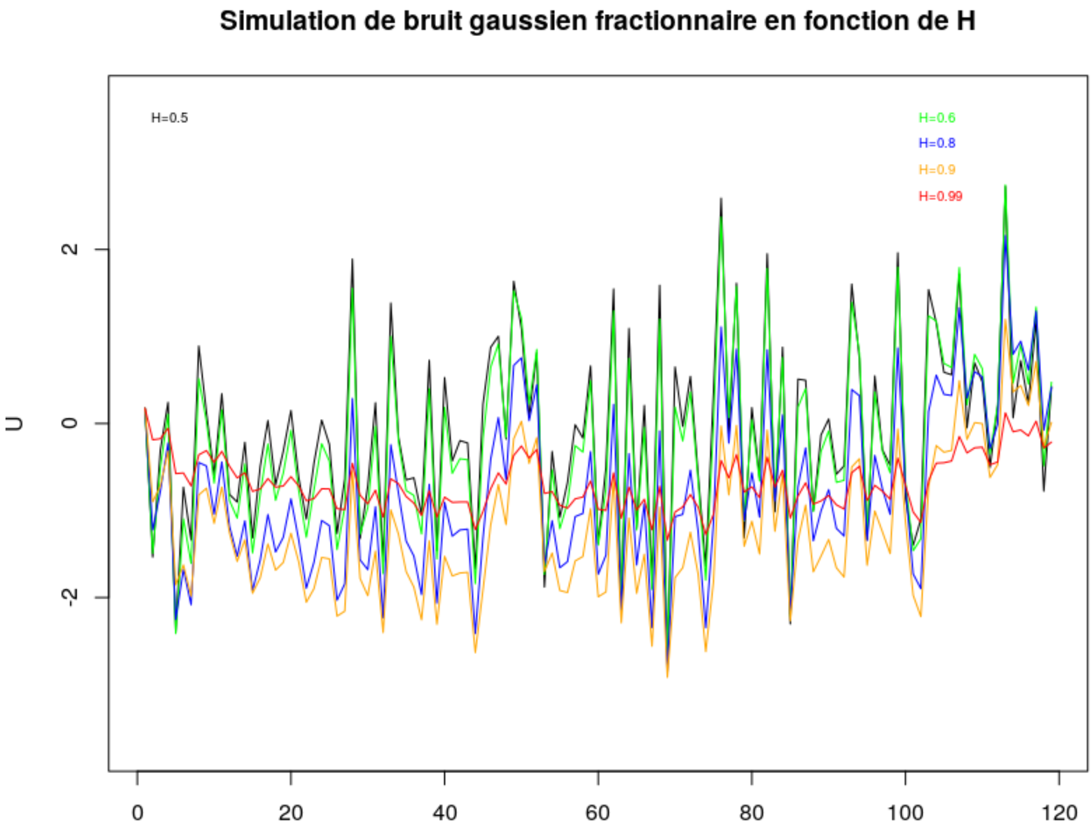
Soit $Z = (Z_1, ..., Z_{n+1})\top$ un vecteur gaussien centré réduit et K une matrice de covariance. Soit $K = LL^\top$ la décomposition de Cholesky de K.
Alors $LZ \sim \mathcal{N}\,(0, K)$

Soit $(z_1, .., z_{n+1})\top$ une réalisation de vecteur gaussien centré réduit $Z = (Z_1, ..., Z_{n+1})\top$. Et soit K_H la matrice de covariance $(n+1) \times (n+1)$ de bruit gaussien fractionnaire définie par :

$$K_H\,(i,j) = \tfrac{1}{2}\left(|i-j+1|^{2H} - 2|i-j|^{2H} + |i-j-1|^{2H}\right), \quad i, j = 1, ...n+1.$$

On effectue une décomposition de Cholesky à la matrice K_H, i.e., $K_H = LL^\top$ avec L une matrice triangulaire inférieur, on a alors $LZ \sim \mathcal{N}\left(0, L\mathbb{I}_d L^\top\right) \Rightarrow LZ \sim \mathcal{N}\,(0, K_H)$

$(z_1^*, ..z_{n+1}^*) = L(z_1, .., z_{n+1})$ est une réalisation de vecteur bruit gaussien fractionnaire.



2.4.2 Applications à la prévision en utilisant les données de la température annuelle moyenne au Maroc :

En ayant les températures moyennes annuelles de n années ; $t_1, ..., t_n$, la corrélation entre t_i et t_j et donnée par la composante (i, j) de la matrice de covariance du bruit gaussien fractionnaire $\boldsymbol{K}_H$.
En appliquant le résultat (1) on obtient :

$$\hat{t}_{n+1} = \sum_{i=1}^{n} \omega_i t_i$$

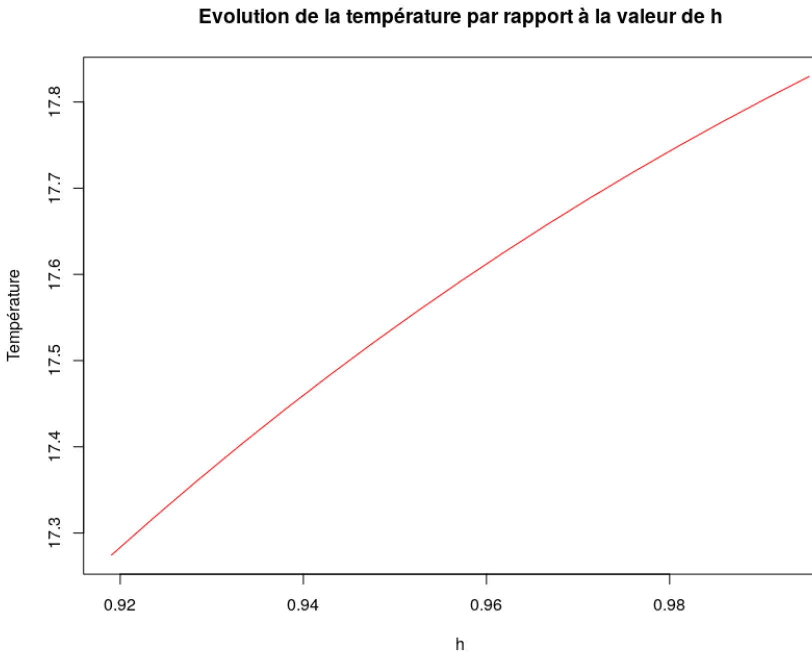
où $(\omega_1, \ldots, \omega_n)$ est la solution du système suivant :

$$[k_H(i,j) : i, j = 1, \ldots, n] \begin{pmatrix} \omega_1 \\ \omega_2 \\ . \\ . \\ \omega_n \end{pmatrix} = \begin{pmatrix} K_H(n+1,1) \\ K_H(n+1,2) \\ . \\ . \\ K_H(n+1,n) \end{pmatrix}$$

Application Numérique :

En résolvant le système au dessus avec le logiciel R, on a pu déterminer les poids ω_i $i = 1, \dots, n$ et prédire $\hat{t}_{n+1}$ pour différentes valeurs de H proches de 1. Alors on a trouvé les résultats suivants :

```
Pour h= 0.9190476        Température = 17.27445
Pour h= 0.9238095        Température = 17.31905
Pour h= 0.9285714        Température = 17.36217
Pour h= 0.9333333        Température = 17.40385
Pour h= 0.9380952        Température = 17.44412
Pour h= 0.9428571        Température = 17.48303
Pour h= 0.947619         Température = 17.52059
Pour h= 0.952381         Température = 17.55685
Pour h= 0.9571429        Température = 17.59183
Pour h= 0.9619048        Température = 17.62558
Pour h= 0.9666667        Température = 17.65811
Pour h= 0.9714286        Température = 17.68947
Pour h= 0.9761905        Température = 17.71968
Pour h= 0.9809524        Température = 17.74878
Pour h= 0.9857143        Température = 17.77678
Pour h= 0.9904762        Température = 17.80372
Pour h= 0.9952381        Température = 17.82962
```




```
Pour h= 0.7375    Température = 14.0108
Pour h= 0.75      Température = 14.3652
Pour h= 0.7625    Température = 14.69532
Pour h= 0.775     Température = 15.0026
Pour h= 0.7875    Température = 15.28838
Pour h= 0.8       Température = 15.55391
Pour h= 0.8125    Température = 15.80041
Pour h= 0.825     Température = 16.02899
Pour h= 0.8375    Température = 16.24074
Pour h= 0.85      Température = 16.43667
Pour h= 0.8625    Température = 16.61773
Pour h= 0.86875          Température = 16.70296
Pour h= 0.875     Température = 16.78482
Pour h= 0.88125          Température = 16.8634
Pour h= 0.8875    Température = 16.9388
Pour h= 0.925     Température = 17.32997
Pour h= 0.9375    Température = 17.43917
```

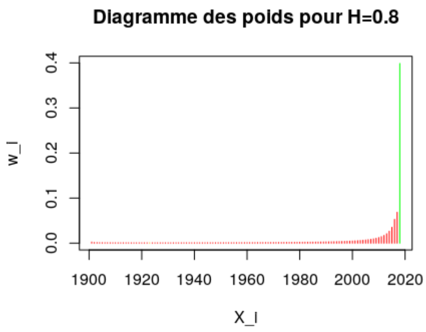
Pour :

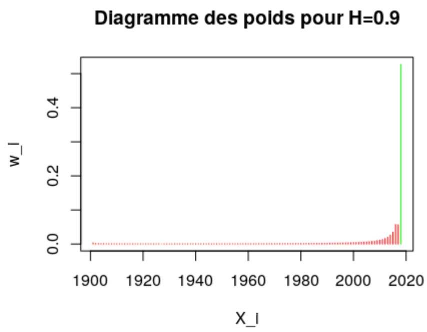
$H_1 = 0.8375$ (premier quartile) on a la prévision : $Q_1 \simeq 16.24074$

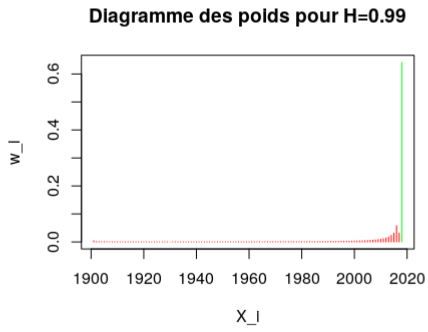
$H_2 = 0.88125$ (la médiane) on a la prévision : $M \simeq 16.8634$

$H_3 = 0.925$ (troisième quartile) on a la prévision : $Q_3 \simeq 17.32997$

Diagramme des poids :



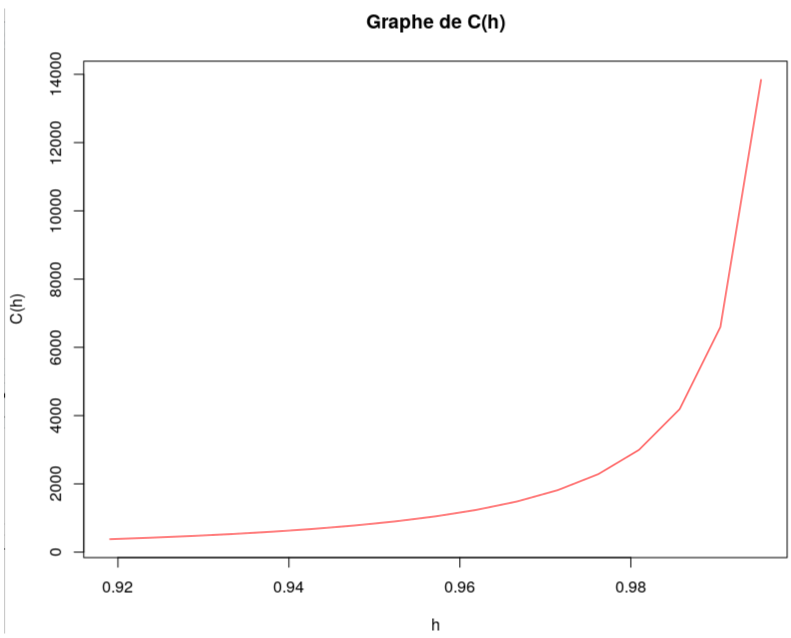




À partir des diagrammes des poids entre 1900 et 2018 pour les trois différentes valeurs de H (H=0.8 et H=0.9 et H=0.99), on peut remarquer que les poids associés aux années 1900 à 2000 diminuent lorsque H devient plus grand et ils deviennent presque égaux à zéro. Par contre le poids associé à l'année 2018 devient beaucoup plus important lorsque l'indice H augemente.
On peut en déduire que avec cette méthode et lorsque H est proche de 1, la prévision de la température t_{n+1} est surtout basées sur les températures moyennes annuelles des quelques dernières années passées et non sur la totalité .

On va étudier le cas où H est très proche de 1 :

Soit $C(H)$ le conditionnement de la matrice de covariance $\boldsymbol{K}_H$ définit par : $C(H) = \frac{\lambda^H_{max}}{\lambda^H_{min}}$ avec λ^H_{max}, λ^H_{min} la plus grande et la plus petite valeur propre de $\boldsymbol{K}_H$.



Problème : On remarque ici que lorsque l'indice H devient proche de 1 la matrice devient de plus en plus **mal conditionnée**, plus précisément la matrice K_H converge vers la matrice A telle que $A(i,j) = 1 \; \forall i,j \in \{1,...,n\}$ quand H tend vers 1 .

C'est pour cette raison, on va appliquer le système (2) sous contraintes :

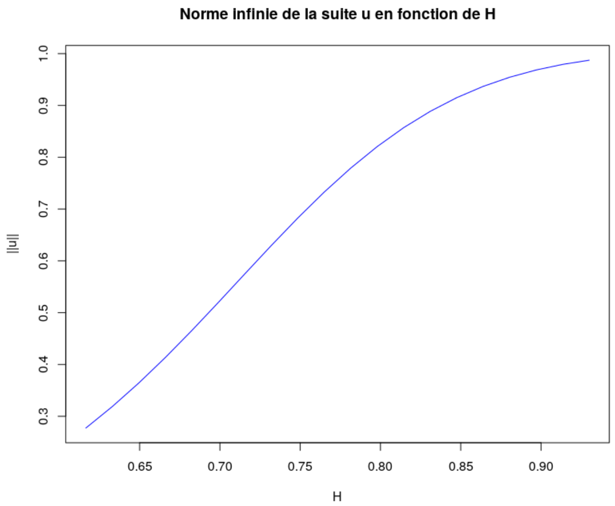
On note par $K_H(.,j)$ la $j^{\text{ème}}$ colonne de K_H.

Soit $(u_j)_{j\in\mathbb{N}}$ la suite définie par $u_j = \frac{\sum_{i=1}^{n+1} K_H(i,j)}{\sqrt{n+1}\sqrt{\sum_{i=1}^{n+1} |K_H(i,j)|^2}}, \quad j = 1, \ldots, n+1.$

On rappelle que le terme u_j de la suite correspond au cosinus de l'angle entre la $j^{\text{ème}}$ colonne de K_H et le vecteur 1_{n+1}.

Remarque :

Quand H est proche de 1 les termes de la suite $(u_j)_{j\in\mathbb{N}}$ s'approchent de 1 aussi.

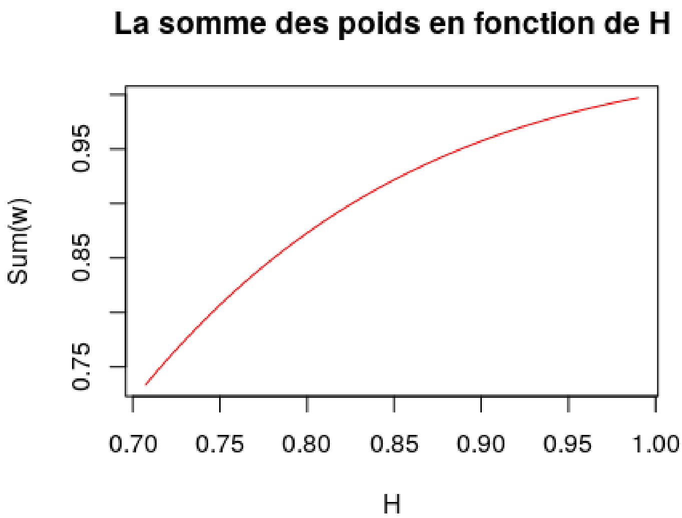


Ce résultat est compatible avec le résultat précédent et montre que les colonnes de K_H tendent vers le vecteur 1_{n+1} quand H tend vers 1.

Remarque :

Si $\begin{pmatrix} y_1 \\ . \\ . \\ y_{n+1} \end{pmatrix} \in ev(K(.,1),\ldots,K(.,n))$ alors $y_{n+1} = \sum_{i=1}^{n} w_i y_i$

Alors les w_i estiment sans erreur les n-ièmes premières composantes de y .



On ordonne la suite $(u_j)_{j\in\mathbb{N}}$ dans l'ordre décroissant, cela signifie qu'on range dans l'ordre décroissant les cosinus entre chaque vecteur colonnes de la matrice K_H et le vecteur 1_{n+1}, (i.e, on range dans l'ordre décroissant les vecteurs qui s'approchent le plus possible du vecteur 1_{n+1}), afin de déterminer les fonctions f_k qui satisfont les contraintes du système (2) $(\sum_{i=1}^{n} w_i f_k(i) = f_k(n+1),\ k = 1,\ldots,n.\ )$

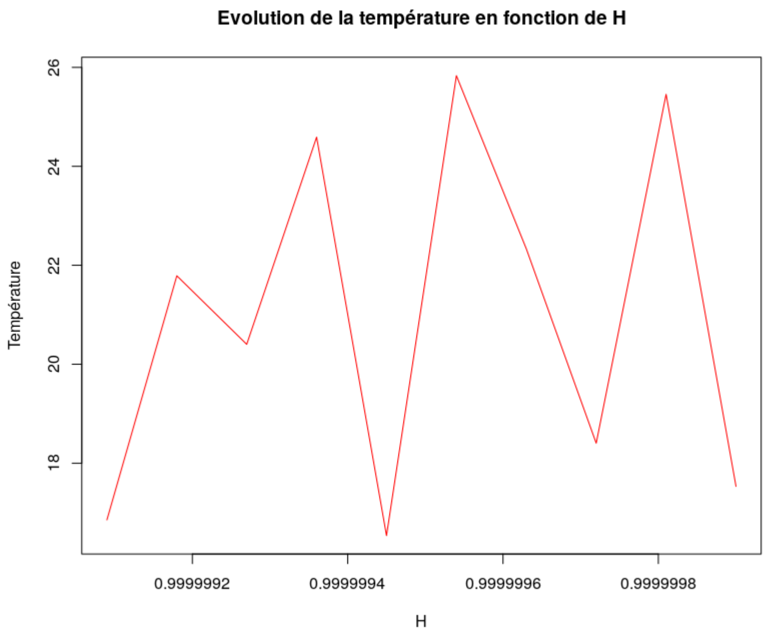
On a $u_{\sigma(1)} > \ldots > u_{\sigma(n+1)}$ avec σ une permutation de $\{1,\ldots,n+1\}$ dans $\{1,\ldots,n+1\}$. D'après la remarque précedente on peut poser $f_k = u_{\sigma(k)}$,Alors la moyenne $\mathbb{E}(Y_i) = \sum_{k=1}^{n} \beta_k u_{\sigma(k)}(i) \quad i = 1,\ldots,n+1$.

On propose le système sous contraintes suivant :

$$\begin{cases} \sum_{i=1}^{n} w_i + \sum_{i=1}^{n} \lambda_i K_H(i,\sigma(l)) = 1, & l = 1,\ldots,n, \\ \sum_{i=1}^{n} w_i K_H(i,\sigma(l)) = k_H(n+1,\sigma(l)), & l = 1,\ldots,n. \end{cases}$$

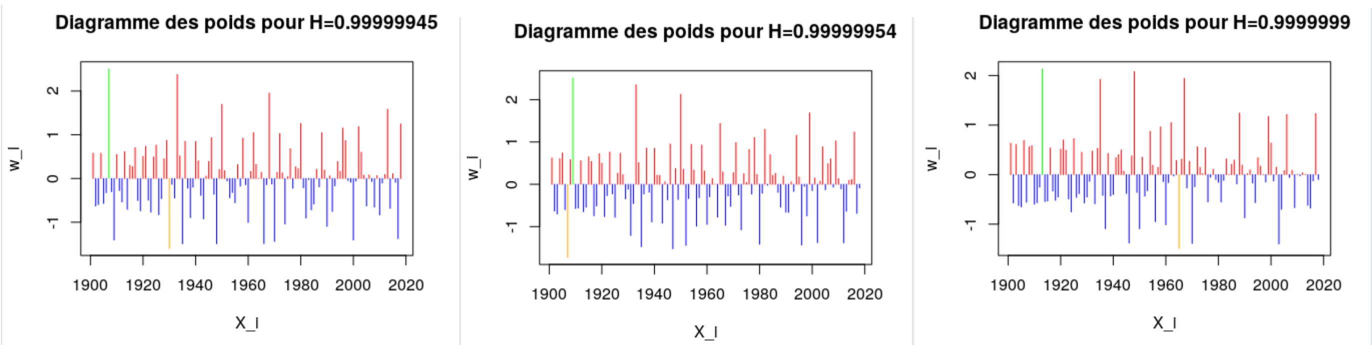
Résultats Numérique :

```
Pour H= 0.9999991        Temperature = 16.85411
Pour H= 0.9999992        Temperature = 21.78776
Pour H= 0.9999993        Temperature = 20.40221
Pour H= 0.9999994        Temperature = 24.58988
Pour H= 0.99999945       Temperature = 16.53675
Pour H= 0.99999954       Temperature = 25.83237
Pour H= 0.9999996        Temperature = 22.33339
Pour H= 0.9999997        Temperature = 18.4044
Pour H= 0.9999998        Temperature = 25.45494
Pour H= 0.9999999        Temperature = 17.53225
```



Les prévisions que nous venons d'obtenir pour les différentes valeurs de H très très proche de 1, restent bonnes étant donné que la médiane est de l'ordre de 16.8 degrés, même si il existe certaines qui ont dépassé la valeur de température annuelle moyenne maximale au Maroc (19.43 degrés) enregistrée jusqu'à maintenant (en 1945).

Diagramme des poids :



Les diagrammes ci-dessus représentent les poids associés aux années 1900 à 2018 pour trois valeurs de H très très proche de 1. On obtient alors des poids positifs (en rouge) ces derniers augmentent la température, et des poids négatifs (en bleu) qui la font baisser.

On remarque que cette méthode sous contraintes (pour H très proche de 1) nous donne des poids qui ne sont pas nuls, ce qui veut dire que la prévision de la température annuelle moyenne pour l'année 2019 au Maroc est ainsi basée sur toutes les températures annuelles moyennes observées au Maroc entre 1900 et 2018 contrairement à la méthode vue précédemment qui se base uniquement sur les observations enregistrées durant les quelques dernières années passées.

Conclusion

Dans ce Travail Encadré de Recherche, nous nous sommes intéressés à la prévision de la température annuelle moyenne au Maroc pour l'année 2019. Nous ne disposions que d'une table contenant les températures annuelles moyennes de ce pays entre 1900 et 2018, alors tout naturellement on a commencé par faire une analyse descriptive(boite a moustache, graphes...), voir également si ces températures pourraient suivre une loi usuelle (éventuellement la loi normale), alors nous avons fait un QQ-plot ainsi qu'un test de Shapiro-Wilk qui tous les deux nous ont confirmé la normalité de ces températures. Une fois la normalité vérifiée, nous avons effectué une régression linéaire afin de voir si l'on pouvait prévoir avec cette méthode la température annuelle moyenne au Maroc pour l'année en 2019, ce qui a été vite abandonné étant donnée que le coefficient de détermination R^2 était proche de 0 et ne nous pouvions donc compte sur ce modèle pour faire cette prévision.

Nous nous sommes alors retournés vers une idée très simple qui est "l'espérance conditionnelle", car nous étions sûrs qu'elle allait nous donner une très bonne prévision qui est en plus compatible avec la réalité. Cependant pour appliquer ceci, nous avions été contraints d'avoir une matrice de variance-covariance. Raison pour laquelle on a eu recours a la matrice du bruit gaussien fractionnaire, qui nous a donné de bonnes prévisions comme elles correspondaient bien aux températures entre la moyenne et le premier et troisième quartile.

Soucieux d'avoir une prévision encore mieux, nous nous sommes intéressés à une prévision sans biais, ce qui nous a imposé des contraintes. Pour trouver les bonnes contraintes, nous avons donc dû en essayer plusieurs mais ne trouvant pas de bonnes prévisions, nous avons pensé qu'en analysant d'une manière détaillée cette matrice de variance-covariance on pourrait avoir une idée sur les contraintes à imposer à notre système, et effectivement on a trouvé des conditions qui nous ont permis une fois ajoutées à notre système d'avoir de très bonnes prévisions pour cette température recherchée.

Appendice

2.4.3 mouvement brownien fractionnaire

Soit $\boldsymbol{B_H}$ un mouvement brownien fractionnaire de paramètre de ***Hurst H*** ($H \in]0,1[$) , défini par :

- $B_H(0) = 0$ p.s
- $B_H(k) = \sum_{j=0}^{k-1} (B_H(j+1) - B_H(j))$ pour $k \geq 1$

Le bruit gaussien fractionnaire est défini par :

$$X_j = B_H(j+1) - B_H(j), \text{ pour tout } j \in \mathbb{Z}$$

2.4.4 Processus gaussien stationnaire

Le processus centré (X_i) est stationnaire si :
$cov(X_i, X_j) = \gamma(|i-j|), \quad \forall\, i,j \in \mathbb{Z}$
Le processus gaussien stationnaire centré dont la fonction de covariance $\gamma(i-j) = 0, \quad i \neq j$ est appelé le bruit blanc gaussien.

Remarque :
Si le processus (X_i) est stationnaire, alors X_i a la même loi que X_1, mais X_i et X_1 sont corrélées. Leur coefficient de corrélation est $\rho(X_i, X_1) = \frac{\gamma(|i-1|)}{\gamma(0)}$.

2.4.5 Transformée de Fourier (TF)

La suite $(\gamma(k) : \quad k \in \mathbb{Z})$ définit la fonction périodique
$S(f) = \sum_{k\in} \gamma(k) \exp(i2kf\pi) = \gamma(0) + 2\sum_{k=1}^{+\infty} \gamma(k) \cos(2kf\pi) \geq 0, \quad f \in [0,1]$,

appelée la transformé de Fourier de γ. Sachant S, on peut retrouver γ par $\int_0^1 S(f) \exp(-2fni\pi) df = \gamma(n), \quad n \geq 0$.
Certains auteurs utilisent la TF

$S(\omega) = \frac{1}{2\pi} \sum_{k\in} \gamma(k) \exp(ik\omega), \quad \omega \in (-\pi, \pi)$,
et alors

$\gamma(n) = \int_{-\pi}^{\pi} S(\omega) \exp(in\omega) d\omega, \quad n \geq 0.$

La TF du bruit blanc gaussien est constante
$S(f) = S(0) = \gamma(0) = var(X_1), \quad S(\omega) = S(0) = \frac{var(X_1)}{2\pi}$

La TF du $\frac{S(f)}{S(0)}$ est uniforme sur $[0, 1]$, et la TF du $\frac{S(\omega)}{S(0)}$ est uniforme sur $(-\pi, \pi)$.

2.4.6 Bruit gaussien fractionnaire

Introduction :

Le mouvement brownien fractionnaire (fBm) de paramètre de ***Hurst*** H tel que $0 < H < 1$ est un modèle non stationnaire de signaux fractales stochastiques .Les incréments de ce processus, nommés bruits gaussiens fractionnaires (fGn)(On appelle bruit toute variation imprévisible d'une quantité dans le temps), sont stationnaires. Ces deux modèles ont tout d'abord été utilisés pour modéliser des signaux issus de phénomènes physiques tels que des bruits en $1/f$, des séries économiques, et plus récemment des données de trafic Ethernet. En deux dimensions, le paramètre H permet de quantifier la notion intuitive de rugosité d'une image. Il est alors employé pour caractériser des textures ou pour analyser des surfaces.Lors de l'application du modèle fBm à un cas concret, il convient dans un premier temps de se soucier du caractère fractale des signaux expérimentaux (signaux discrets de longueur finie). Des méthodes ont déjà été proposées pour s'assurer de l'adéquation entre le modèle fBm et des signaux réels.Dans un deuxième temps, il est nécessaire d'estimer aussi précisément que possible le paramètre H. Il existe de nombreuses méthodes qui permettent de réaliser cette tâche et il est difficile d'en choisir une a priori. Théoriquement, même si des résultats d'efficacité asymptotique sont démontrés pour certains estimateurs, rien ne permet de connaître leur comportement dans toutes les circonstances (signaux de longueur réduite, sensibilité au bruit...). Il est donc nécessaire de mener des études expérimentales pour mieux cerner les potentialités des estimateurs.Ces études reviennent à tester ces diverses méthodes sur des signaux fBm de synthèse. Dans la première référence,une analyse croisée synthèse-analyse n'a pas permis de tirer de conclusions définitives. En effet, certaines méthodes de synthèse ne sont qu'approximatives et peuvent interférer avec l'analyse.Dans la deuxième étude, seuls quelques estimateurs ont été évalués uniquement pour $H > 0.5$. Pour mener une étude générale,il faut au préalable disposer d'une méthode de synthèse générant de vrais signaux fBm. La synthèse par décomposition de Cholesky de la matrice de covariance des incréments du fBm est théoriquement exacte. En effet, les statistiques d'ordre 2 des incréments de tels signaux sont celles des incréments gaussiens du fBm. L'appréciation de la qualité d'un estimateur revient à comparer la valeur du paramètre H mesurée sur ces signaux à celle injectée lors de la synthèse.

Hurst (1965) a proposé une méthode appelée analyse des étendues normalisées, ou Rescaled Range Analysis (R/S Analysis). Notons que l'objectif initial de l'auteur était de modéliser la série temporelle de la hauteur des crues du Nil, de l'antiquité à nos jours. Selon cette méthode, une série chaotique peut être caractérisée par un exposant (noté H), qui représente la probabilité pour qu'un événement soit suivi par un événement similaire. C'est donc les aspects de persistance qui sont principalement visés par cette analyse. Le mode de calcul de l'exposant de Hurst n'est pas encore fixé, et certaines différences peuvent apparaître en fonction des détails des algorithmes utilisés. Nous exposons ici la méthode originale de l'auteur.

Propriété :

Soit $(X_j)_{j\in\mathbb{Z}}$ est un processus gaussien centré stationnaire de covariance : $cov(X_i, X_j) = \frac{\sigma^2}{2}[|i-j+1|^{2H} - 2|i-j|^{2H} + |i-j-1|^{2H}] := \gamma(|i-j|),\ \ i, j \in \mathbb{Z}$

1) La fonction $x \in (0, +\infty) \to |x|^\alpha$ est convexe lorsque $\alpha \geq 1$ ou bien $\alpha < 0$. Elle est concave lorsque $\alpha \in (0, 1)$.

2) Si $2H \geq 1$, alors $\gamma(k) > 0, \quad k \geq 1$. De plus la suite $k \to \gamma(k)$ est décroissante, convexe et $\gamma(k) \sim H(2H-1)k^{2H-2}\sigma^2$ lorsque $k \to +\infty$.

3) Si $0 < 2H < 1$, alors $\gamma(k) < 0, \quad k \geq 1$. De plus $\gamma(k) \sim H(2H-1)k^{2H-2}\sigma^2$ lorsque $k \to +\infty$.

Corollaire :

La transformée de Fourier du bruit fractionnaire vérifie

$S(f) \sim f^{-\alpha}$ lorsque $f \to 0$, où $\alpha = 1 - (2 - 2H) = 2H - 1$.

On dit que le processus suit la loi $f^{-\alpha}$. Lorsque $H = 1$ on obtient $\alpha = 1$ et donc le processus suit la loi $1/f$.

Théorème de probabilité (cours du M1 Mas)

Soit $(y_1, \ldots, y_{n+1})$ une réalisation d'un vecteur aléatoire gaussien $(Y_1, Y_2, \ldots Y_{n+1})$ alors l'espérance conditionnelle de Y_{n+1} sachant $Y_1, Y_2, \ldots, Y_n$ est égale à :

$$\mathbb{E}[Y_{n+1}|Y_1, Y_2, \ldots, Y_n] = \sum_{i=1}^{n} w_i y_i$$

Bibliographie

A. Dermoune, Cours processus stochastique M2, 2018.

H.E. Hurst. Long-term stocrage capacity of reservoirs. Transactions of the Ame- rican Society of Civil Engineers, 116 :770–799, 1951.

B . Mandelbrot and J. Van Ness. Fractional brownian motion, fractional noises and application. SIAM Review, 10 :422–437, 1968.

Annexe

```
Morocco <- read_excel("Morocco.xlsx")
##########Graphe des Temp ratures ###################################
plot(Morocco$Annee,Morocco$Temperature,type="s",col="blue",xlab="Annee",ylab="Temperature"
     ,main="Graphe des temperatures moyennes annuelles au maroc")
t=Morocco$Temperature
annee=Morocco$Ann e
points(1971,14.546,col="blue",pch=20)
points(1945,19.4339,col="red",pch=20)
text(c(1970,1944),c(14.54,19.43),c("Temperature minimale","Temperature maximale"),cex = 0.6,pos = 4)
###### ############################Boite    mustaches###################
boxplot(t, range=0,sub="Une pr sentation sous forme boite    moustaches")
summary(t)

################# QQ-plot ######################
qqnorm(t, datax=TRUE, main="QQ-plot des temp ratures")
qqline(t,datax=TRUE)
############### Test de shapiro-Wilk #####################
t=Morocco$Temp rature
shapiro.test(t)
############### La regression lineaire #######################
plot(Morocco$Ann e,Morocco$Temp rature,col="blue",xlab="Ann e",ylab="Temp rature",main="R gression lin aire")
t=Morocco$Temp rature
annee=Morocco$Ann e
Temp.lm<-lm(t~annee)
abline(Temp.lm,col="red")
summary(Temp.lm)
ICconf <- predict(Temp.lm, interval = "confidence", level = 0.95)
matlines(annee, ICconf, lty = c(1, 4, 4), col = c(2, 9,9 ))
  #intervale de prediction#
ICpred <- predict(Temp.lm, interval = "prediction", level = 0.95)
matlines(annee, ICpred, lty = c(1, 4, 4), col = c(2, 3, 3),ylab="interval")
######## simulation de bruit gaussien fractionnaire############
n=118
U=rnorm(n+1,0,1) # bruit blanc
plot(U,type = "l",col="black",ylim=c(-3.7,3.7),main = "Simulation de bruit gaussien fractionnaire en fonction de H")
U_bruit=vector("double",n)
H=vector("double",4)
H[1]=0.6
H[2]=0.8
H[3]=0.9
H[4]=0.99
Couleur=c("green","blue","orange","red")

for(k in seq(1,4)){
```

```
h=H[k]
K_h=matrix(nrow=n+1,ncol=n+1)
x=seq(1,n+1)
for(i in seq(1,n+1)){
  for (j in seq(1,n+1)){

    K_h[x[i],x[j]]=(1/2)*(((abs(x[i]-x[j]+1))^(2*h))-2*((abs(x[i]-x[j]))^(2*h))+((abs
        (x[i]-x[j]-1))^(2*h)))
  }
}
L=t(chol(K_h))
U_bruit=L%*%U
lines(U_bruit, col=Couleur[k], type="l")
}
text(0,3.5,"H=0.5",col="black",cex = 0.6,pos = 4)
text(100,3.5,"H=0.6",col="green",cex = 0.6,pos = 4)
text(100,3.2,"H=0.8",col="blue",cex = 0.6,pos = 4)
text(100,2.9,"H=0.9",col="orange",cex = 0.6,pos = 4)
text(100,2.6,"H=0.99",col="red",cex = 0.6,pos = 4)
#################### R solution du Premier syst me ###############
n=118
vecteur_h=vector("double",15)
vecteur_t=vector("double",15)
for (l in seq(4,20))
{
  h=0.9+l/210
  vecteur_h[l-3]=h
#Remplissage de la matrice K_h de taille (n+1)*(n+1)
  K_h=matrix(nrow=n+1,ncol=n+1)
  x=seq(1,n+1)
  for(i in seq(1,n+1)){
    for (j in seq(1,n+1)){

      K_h[x[i],x[j]]=(1/2)*(((abs(x[i]-x[j]+1))^(2*h))-2*((abs(x[i]-x[j]))^(2*h))+((
          abs(x[i]-x[j]-1))^(2*h)))
    }
  }
#R solution
  M=matrix(nrow =n,ncol = n)
  M=K_h[1:n,1:n]
  alpha=solve(M,K_h[n+1,1:n])
#Evaluer
solution=sum(alpha*t)
cat("Pour h=",h,"\t","Temperature =",solution,"\n")
vecteur_t[l-3]=solution
}
########### Diagramme des poids ###################################
Couleur=vector( "character",n)
for(i in seq(1:n)){

  if( alpha[i]>0){Couleur[i]="red"}
  if(alpha[i]<0){Couleur[i]="blue"}
  if(alpha[i]==0){Couleur[i]="blac"}
  if(alpha[i]==max(alpha)){Couleur[i]="green"}
  if(alpha[i]==min(alpha)){Couleur[i]="orange"}
}
plot(annee,alpha,type="h",main="Diagramme des poids pour H=0.99",xlab ="X_i",ylab = "
    w_i", col=Couleur)
###### Graphe des temperatures par rapport a h ###################
plot(vecteur_h,vecteur_t,type="l",col="red",xlab="h",ylab="Temp rature",main="
     Evolution de la temp rature par rapport   la valeur de h")
########### La somme des poids tend vers 1 quand h tend vers 1 ##
```

```
n=118
sompoids=vector("double",40)
vecteur_h=vector("double",40)
for (l in seq(1,40))
{
  h=0.7+l*0.29/40
  vecteur_h[l]=h

  #Remplissage de la matrice K_h de taille (n+1)*(n+1)
  K_h=matrix(nrow=n+1,ncol=n+1)
  x=seq(1,n+1)
  for(i in seq(1,n+1)){
    for (j in seq(1,n+1)){

      K_h[x[i],x[j]]=(1/2)*(((abs(x[i]-x[j]+1))^(2*h))-2*((abs(x[i]-x[j]))^(2*h))+((abs(x[i]-x[j]-1))^(2*h)))
    }
  }
  #R solution
  M=matrix(nrow =n,ncol = n)
  M=K_h[1:n,1:n]
  alpha=solve(M,K_h[n+1,1:n])
sompoids[l]=sum(alpha)
}

plot(vecteur_h,sompoids,type="l",col="red",xlab = "H",ylab = "Sum(w)",main = "La somme des poids en fonction de H")


################ Conditionement de la matrice ################
n=118
vecteur_h=vector("double",17)
valeurpropre=vector("double",n)
c_h=vector("double",17)
for (l in seq(4,20))
{
  h=0.9+l/210
  vecteur_h[l-3]=h

#Remplissage de la matrice K_h de taille n*n
  K_h=matrix(nrow=n,ncol=n)
  x=seq(1,n)
  for(i in seq(1,n)){
    for (j in seq(1,n)){

      K_h[x[i],x[j]]=(1/2)*(((abs(x[i]-x[j]+1))^(2*h))-2*((abs(x[i]-x[j]))^(2*h))+((abs(x[i]-x[j]-1))^(2*h)))
    }
  }

  vect=eigen(K_h)
  valeurpropre=vect$values
  valeurpropre=-(sort(- valeurpropre)) #trier les valeur propres
  c_h[l-3]=valeurpropre[1]/valeurpropre[n]
}
plot(vecteur_h,c_h,type="l",col="red",xlab = "h",ylab = "C(h)",main = "Graphe de C(h)")

##################### Probleme avec contrainte #####################
# Graphe de ||u|| en fonction de H pour montrer que si H proche de 1 ||u|| et proche de 1 #
norme_u=vector("double",20)
```

```
vecteur_H=vector("double",20)
Vecteur_u=vector("double", length = n+1)
for(k in seq(1:20)){
h=0.6+k*(0.33/20)
vecteur_H[k]=h
n=118
K_h=matrix(nrow=n+1,ncol=n+1)
x=seq(1,n+1)
for(i in seq(1,n+1)){
  for (j in seq(1,n+1)){

    K_h[x[i],x[j]]=(1/2)*(((abs(x[i]-x[j]+1))^(2*h))-2*((abs(x[i]-x[j]))^(2*h))+((abs
        (x[i]-x[j]-1))^(2*h)))
  }
}

#la matrice k_h2 contient les carr s de chaque composante de K_h
K_h2=matrix(nrow=n+1,ncol=n+1)
for(i in seq(1,n+1)){
  for (j in seq(1,n+1)){
    K_h2[i,j] =(K_h[i,j])^2
  }
}

# on stocke les valeurs de la suite u
for(j in seq(1,n+1)){
  Vecteur_u[j]=colSums(K_h)[j]/((sqrt(n+1))*(sqrt(colSums(K_h2)[j])))
}
 #calcul norme infini de la suite
s=max(abs(Vecteur_u))
norme_u[k]=s
 }
plot(vecteur_H,norme_u,xlab = "H",ylab="||u||",main = "Norme infinie de la suite u en
     fonction de H",type ='l',col="blue")
# Partie pevision #

Vecteur_u=vector("double", length = n+1)
Vecteur_H=vector("double",10)
Veceteur_Temp rature=vector("double",10)
n=118
for(k in seq(1:10)){
 h=0.999999+k*(0.0000009/10)
  Vecteur_H[k]=h
# Remplissage de la matrice  K_h du syst me
K_h=matrix(nrow=n+1,ncol=n+1)
x=seq(1,n+1)
for(i in seq(1,n+1)){
  for (j in seq(1,n+1)){

    K_h[x[i],x[j]]=(1/2)*(((abs(x[i]-x[j]+1))^(2*h))-2*((abs(x[i]-x[j]))^(2*h))+((abs
        (x[i]-x[j]-1))^(2*h)))
  }
}

#la matrice k_h2 contient les carr s de chaque composante de K_h
K_h2=matrix(nrow=n+1,ncol=n+1)
for(i in seq(1,n+1)){
  for (j in seq(1,n+1)){
    K_h2[i,j] =(K_h[i,j])^2
  }
}
```

```
#on stocke les valeurs de la suite u
for(j in seq(1,n+1)){
  Vecteur_u[j]=colSums(K_h)[j]/((sqrt(n+1))*(sqrt(colSums(K_h2)[j])))
}
#on ordonne la suite u dans le sens d croissant
Vecteur_u_tri = vector("double", length = n+1)
Vecteur_u_tri= -(sort(- Vecteur_u))
#apres le tri , on prend uniquement les n premieres valeurs de la suite
Vecteur_u_tri_n=  vector("double", length = n)
Vecteur_u_tri_n= Vecteur_u_tri[1:n]

#r cup rer les n indices (I) qui correspondent a la position des Vecteur_u_tri dans le tableau initial Vecteur_u

#on commence d ja par prendre les n premieres composantes uniquement

Vecteur_u_n=  vector("double", length = n)
Vecteur_u_n= Vecteur_u[1:n]

#r cup ration des indices (I)
Vecteur_u_indice=order(- Vecteur_u_n)
# r soudre le syst me.. on le reecrit sous forme Ax=b

# on  crit  la matrice du systeme par blocs 1:n x 1:n, (n+1):2n x n, 1:n x (n+1):2n, (n+1):2n x (n+1):2n
Matrice_syst=matrix(nrow = 2*n, ncol=2*n)

# bloc 1:n x 1:n
for (i in seq(1,n) ) {
  for (j in seq(1,n)) {
    Matrice_syst[i,j]=1
  }
}
#bloc (n+1):2n x n
for (i in seq(n+1,2*n) ) {
  for (j in seq(1,n)) {
    Matrice_syst[i,j]=K_h[Vecteur_u_indice[j], i-n]
  }
}
#bloc 1:n x (n+1):2n
for (i in seq(1,n) ) {
  for (j in seq(n+1,2*n)) {
    Matrice_syst[i,j]=K_h[Vecteur_u_indice[j-n], i]
  }
}
#bloc (n+1):2n x (n+1):2n
for (i in seq(n+1,2*n) ) {
  for (j in seq(n+1,2*n)) {
    Matrice_syst[i,j]= 0
  }
}
# le second membre b
Vecteur_b=vector("double",2*n)
Vecteur_b[1:n]=rep(1, n)
for (i in seq(n+1,2*n)) {
  Vecteur_b[i]=K_h[n+1, Vecteur_u_indice[i-n]]
}

# solution du syst me , qui est un vecteur de dim 2n
Solution=solve(Matrice_syst, Vecteur_b)
#les premiers n composantes de la Solution correspondent a W
Solution_w = vector("double",n)
```

```
Solution_w=Solution[1:n]

#les derni res n composantes de la Solution correspondent a lambda
Solution_lambda = vector("double",n)
Solution_lambda=Solution[(1+n):(2*n)]

#calculer la prevision de la temp rature T_(n+1)

Temperature=sum(Solution_w*t)
Vecteur_Temp rature[k]=Temperature
cat("Pour H=",h,"\t","Temperature =",Temperature,"\n")
}

plot(Vecteur_H,Vecteur_Temp rature,type = "l",xlab = "H",ylab = "Temp rature",main
    = "Evolution de la temp rature en fonction de H",col="red")

########### Diagramme des poids (systeme avec contraintes) ##########################
Couleur=vector( "character",n)
for(i in seq(1:n)){

  if( Solution_w[i]>0){Couleur[i]="red"}
  if(Solution_w[i]<0){Couleur[i]="blue"}
  if(Solution_w[i]==0){Couleur[i]="blac"}
  if(Solution_w[i]==max(Solution_w)){Couleur[i]="green"}
  if(Solution_w[i]==min(Solution_w)){Couleur[i]="orange"}
}
plot(annee,Solution_w,type="h",main="Diagramme des poids pour H=0.9999999",xlab ="X_i
    ",ylab = "w_i", col=Couleur)
 ##################################################################
```