Projet fraudes covid

LUCEA Lenny - VALDEYRON Mathieu - CHERY Fanny

2022-05-29

Sommaire

I - Introduction . . . 2
II - Loi de Benford - Newcomb . . . 2
II.1 - Histoire mathématique de la loi de Benford . . . 2
II.2 - Loi de Benford continue . . . 3
II.3 - Loi de Benford discrète (ou simple) . . . 4
II.3.1 - Loi de Benford sur le premier chiffre significatif . . . 4
II.3.2 - Loi de Benford sur le deuxième chiffre significatif . . . 5
II.3.3 - Loi de Benford sur une bloc de k chiffres . . . 5
III - Les différents test . . . 5
III.1 - Test d'ajustement . . . 5
III.1.1 - Test du χ^2 [2] . . . 6
III.1.2 - Test de Kolmogorov-Smirnov . . . 7
III.1.3 - Test du T_k . . . 8
IV - Tests sur le premier chiffre significatif sur les données Covid des différents pays du monde . . . 8
IV.1 - Analyse des données . . . 9
V - Test sur le bloc des deux premiers chiffres significatifs . . . 11
V.1 - Préambule : Cadre Théorique . . . 12
V.2 - Construction des tests lisses T_K : . . . 13
V.3 - Comparaison des puissances avec d'autres tests : . . . 16
V.4 - Mise en pratique sur les données . . . 19
VI - Conclusion : . . . 21
Annexe : . . . 23
. . . 55

I - Introduction

Aujourd'hui connue sous le nom de loi de Benford, la loi de probabilité Benford - Newcomb a été découverte deux fois. Une première fois en 1881 par Simon Newcomb ($1835 - 1909$), mathématicien, statisticien et astronome américain d'origine canadienne du $19^{\text{ème}}$ siècle, puis par Frank Benford ($1883 - 1948$) en 1938 qui est un ingénieur physicien américain. Simon Newcomb remarqua en 1881 une détérioration des tables de logarithmes bien plus importante pour les pages des nombres commençant par 1 pour les pages des nombres commençant par 9. Suite à quelques travaux dessus, il découvrit cette loi de probabilité et il publia un article sur ce sujet dans le "Journal of Mathematics" qui passa inaperçu. Mais 57 ans plus tard, Franck Benford remarqua la même usure sur les pages des tables de logarithmes , et redécouvrit cette loi de probabilité qui aujourd'hui porte son nom.

La loi de Benford - Newcomb est une loi qui porte sur l'apparition des chiffres dans la nature, plus précisément l'apparition du premier chiffre significatif, c'est-à-dire qu'elle donne la probabilité d'un chiffre de 1 à 9 soit le premier chiffre significatif d'un nombre. Le premier chiffre significatif d'un nombre est le premier chiffre partant de gauche qui est différent de 0, par exemple le premier chiffre significatif de 3759 est 3, celui de $0,0821$ et 8. Maintenant que nous avons compris ce qu'est le premier chiffre significatif d'un nombre, passons à la loi de Benford - Newcomb qui nous donne la répartition de l'apparition de ces chiffres.

Contre-intuitivement cette loi n'est pas une loi uniforme sur $[1, 9]$ sur l'échelle additive, nous pourrions logiquement penser que la probabilité que $d \in \{1, 2, \cdots, 9\}$ soit le premier chiffre significatif d'un nombre X est égale à $\frac{1}{9} \approx 0.111$. Et bien non, c'est ce qu'on découvert Newcomb et Benford avec cette loi de probabilité, cette dernière est bien une loi uniforme sur $[1, 9]$ mais sur l'échelle logarithmique, *ie* multiplicative, ce qui donne que la probabilité que le premier chiffre significatif soit d est égale à $log_{10}\left(1 + \frac{1}{d}\right)$.

La distribution de cette loi de probabilité qui modélise l'apparition naturelle ces chiffes dans la nature, est aujourd'hui utilisée pour les données du génome, électorales, macroéconomiques; la détection de fraudes fiscales ou de données frauduleuses en économie, en sciences, etc .. ou encore pour la prédiction des numéros au Loto.

II - Loi de Benford - Newcomb

Posons quelles notations : soit r une nombre réel, on note $\{r\}$ sa partie fractionnaire, et $[r]$ sa partie entière. Nous avons donc $\{r\} = r - [r]$.

II.1 - Histoire mathématique de la loi de Benford

Une partie des informations ci-dessous est tirée de [4], et [5].

Partons de la loi de l'étalement uniforme de la partie fractionnaire d'un nombre réel, qui indique que les nombres d'une série dont on enlève ce qui précède la virgule (8.235 devient 0.235; 143.488 devient 0.488) se répartissent à peu près uniformément dans l'intervalle $[0, 1]$.

En voici l'énoncé plus explicite : si l'on choisit au hasard des nombres réels sur une plage large de plusieurs unités (par exemple entre 1 et 20), et que la loi donnant la probabilité de tomber sur une des valeurs possibles est assez régulière, alors la partie fractionnaire des nombres qu'on trouvera sera, à peu de chose près, uniformément répartie entre 0 et 1.

Plus généralement, si l'on se donne deux nombres a et b compris entre 0 et 1 tel que $a < b$, alors la proportion de réel dont la partie fractionnaire est comprise entre a et b vaut $a - b$, ie la longueur de l'intervalle $[a, b]$.

L'explication de cette loi est que, sauf cas particuliers, les parties fractionnaires des nombres ne seront pas concentrées sur la même zone de l'intervalle $[0, 1]$. Et si cas échéant, les irrégularités possibles de densité sur les 20 intervalles possibles entre deux entiers consécutifs se compenseront plus ou moins, ce qui uniformisera la série des parties fractionnaires, que nous pouvons voir comme une sorte de moyenne de ce qui se passe sur chacun des 20 intervalles entre deux entiers.

Évidence de la loi de Benford

La loi de l'étalement uniforme permet de déduire la loi de Benford. Grâce à cette loi d'étalement à la fois intuitive et formalisable, nous allons obtenir une justification naturelle et simple de la loi de Benford. L'idée consiste simplement à appliquer un logarithme décimal à la loi précédente et à creuser un peu.

Reprenons l'énoncé précédent et appliquons-le non pas aux nombres r de la série considérée, mais à leur logarithme décimal, $log_{10}(r)$. Si l'on choisit des nombres réels r au hasard sur une large plage couvrant plusieurs ordres de grandeur (par exemple entre 1 et 10^{20}), et que la loi qui indique la probabilité de tomber sur une des valeurs possibles est assez **régulière** et **étalée**, alors les parties fractionnaires des logarithmes décimaux des nombres, c'est-à-dire les $\{log_{10}(r)\}$, seront, à peu de chose près, uniformément réparties entre 0 et 1. Pour aller plus loin, ou pour plus d'informations sur les notions de **régularité et d'étalement** ([5]):

Ce que nous venons d'énoncer est la loi de Benford (ou plus exactement une loi dénommée «loi de Benford continue»). En effet, affirmer que c est le premier chiffre significatif du nombre r équivaut à énoncer que $log_{10}(c) \leq \{log_{10}(r)\} \leq log_{10}(c+1)$.

En effet, soit r un réel, il se décompose de manière unique comme $r = q * 10^{\alpha}$ avec $q \in \{1, 2, \cdots, 9\}$ et $\alpha \in \mathbb{Z}$. Notons $c = [q]$ la partie entière de q. Nous avons donc $log_{10}(r) = log_{10}(q * 10^{\alpha}) = log_{10}(q) + log_{10}(10^{\alpha}) = log_{10}(q) + \alpha$ avec $\alpha \in \mathbb{Z}$.

Par croissance de la fonction log_{10}, nous obtenons $1 \leq q < 10 \Rightarrow 0 \leq log_{10}(q) < 1$ et donc $\{log_{10}(r)\} = log_{10}(q)$. De plus, par définition $[q] \leq q < [q] + 1 \Leftrightarrow c \leq q < c + 1$ et par croissance du log_{10} nous obtenons $log_{10}(c) \leq log_{10}(q) < log_{10}(c+1)$.

Les parties fractionnaires des images par log_{10} des nombres r dont le premier chiffre significatifs est c occupent donc dans l'intervalle $[0, 1]$ un intervalle de longueur $log_{10}(c+1) - log_{10}(c)$. Cela signifie, si l'on admet l'uniforme répartition, que leur proportion est $log_{10}(c+1) - log_{10}(c) = log_{10}(\frac{c+1}{c}) = log_{10}(1 + \frac{1}{c})$. C'est exactement ce qu'exprime la loi de Benford formulée au sujet du premier chiffre significatif en base décimale.

II.2 - Loi de Benford continue

Nous obtenons la mantisse d'un réel x strictement positif en déplaçant la virgule après le premier chiffre significatif. Donc la mantisse d'un nombre appartient à l'intervalle $[0, 10[$ et est obtenue à partie de la formule $mantisse(x) = 10^{\{log_{10}(x)\}}$ (rappel : $\{.\}$ désigne la partie fractionnaire du nombre x).

Exemple : La mantisse du nombre 0.0581 est 5.81, et celle du nombre 326.41 est 3.2641.

La loi de Benford continue est donnée par la définition suivante :

La mantisse $X \in [1; 10[$ suit la loi de Benford continue si pour tout $[a, b[\subset [1, 10[$, la probabilité que la mantisse appartienne à $[a, b[$ vaut $log(b) - log(a)$.

Sa fonction de répartition est définie par :

$$F_x(d) = \mathbb{1}_{[1;10[} log_{10}(d) + \mathbb{1}_{[10;+\infty[}$$

De plus, la loi de Benford est invariante par changement d'échelle ou d'unité.

II.3 - Loi de Benford discrète (ou simple)

Tout ceci se généralise bien sûr en base quelconque. Nous choisissons, communément la base $c = 10$ comme base de référence pour le logarithme. Nous avons choisi la base 10 pour le logarithme dans la suite.

II.3.1 - Loi de Benford sur le premier chiffre significatif

Mathématiquement la loi de Benford - Newcomb, que nous allons noter LNB, est donnée par la formule suivante :

Nous notons PCS le premier chiffre significatif d'un nombre.

Soit X une variable aléatoire continue et positive, alors $D = PCS(X)$ a pour probabilité :

$$\mathbb{P}(D = d) = log_{10}\left(1 + \frac{1}{d}\right) \quad \forall d \in \{1, 2, \cdots, 9\}$$

Cette formule nous donne le tableau de probabilité suivant :

d=	1	2	3	4	5	6	7	8	9
$\mathbb{P}(D = d)$	0.301	0.176	0.125	0.097	0.079	0.067	0.058	0.051	0.046

Figure 1: **Tableau de probabilité d'appartion du premier chiffre significatif**

Ci-dessous, nous pouvons observer une représentation graphique de la loi de Benford discrète :

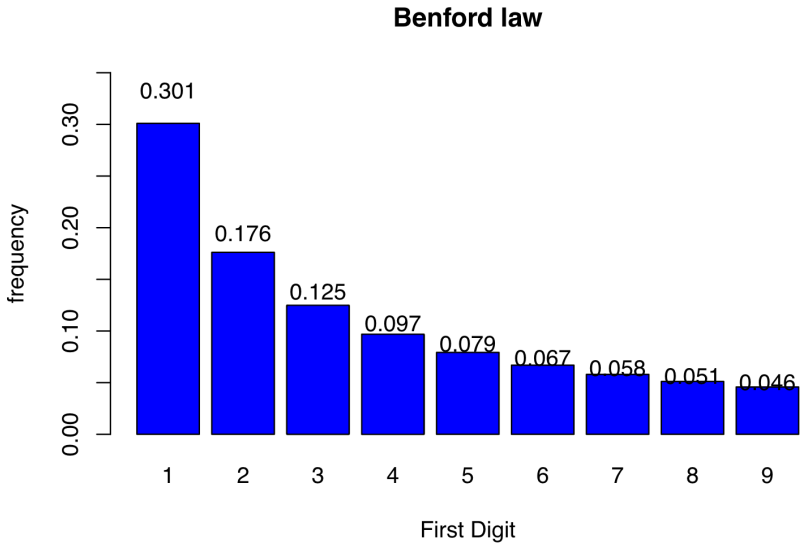

Figure 2: **Répartition théorique selon la loi de Benford**

II.3.2 - Loi de Benford sur le deuxième chiffre significatif

Sur le même principe nous construisons la loi de Benford sur le second chiffre significatif, tout en notant que le support de cette loi n'est plus $d \in \{1, 2, \cdots, 9\}$, mais $d \in \{0, 1, \cdots, 9\}$.

Ce qui nous donne le tableau de probabilité suivante :

d=	0	1	2	3	4	5	6	7	8	9
$\mathbb{P}(D = d)$	0.12	0.114	0.109	0.104	0.10	0.097	ř.093	0.09	0.088	0.085

Figure 3: **Tableau de probabilité d'appartion du second chiffre significatif**

II.3.3 - Loi de Benford sur une bloc de k chiffres

En utilisant le loi de Benford continue ou la loi de Benford en base 10^k, nous obtenons que la probabilité benfordienne que l'écriture décimale d'un nombre réel commence par $d \in [\![10^{k-1}, 10^k[\![$ vaut :

$$log(d+1) - log(d) = log\left(1 + \frac{1}{d}\right)$$

Exemple : La probabilité qu'un nombre commence par 314, comme $3, 14159 \cdots, 314285, 7 \cdots$ ou $0, 00314465 \cdots$ vaut $log(314+1) - log(314) = log\left(1 + \frac{1}{314}\right) \approx 0.00138 = 0.138\%$ avec ici, $n = 314$ et $k = 3$.

III - Les différents test

Nous allons dans cette partie définir les différents tests que nous allons utiliser dans la suite de cette analyse.

Tout d'abord définissons ce qu'est un test lisse, un test d'ajustement.

III.1 - Test d'ajustement

On appelle test d'ajustement tout test visant à vérifier si les données observées sont compatibles avec un modèle théorique. C'est un problème de la forme suivante : [6]

Soit $(x_1, x_2, \cdots, x_N$ un échantillon de variables aléatoires indépendantes identiquement distribuées de fonction de répartition F inconnue. Nous testons l'hypothèse (19) $\mathcal{H}_0 : F = F_0$, contre $\mathcal{H}_1 : F \neq F_0$, où F_0 est la fonction de répartition de la loi testée choisie.

Nous pouvons modifier l'énoncé dans le cas de données discrètes, avec une fonction de répartition d'une loi discrète et les fréquences d'apparition des valeurs du support de cette loi discrète.

III.1.1 - Test du χ^2 [2]

Le test du χ^2 à été proposé par le statisticien Karl Pearson en 1900. Ce test du χ^2 nous permet de vérifier si un échantillon d'une variable aléatoire X nous donne des observations comparables à une loi de probabilité $\mathbb{P}$, ici la loi de Benford. C'est-à-dire, il nous permet de vérifier si les fréquences observées dans un échantillon correspondent aux fréquences attendues données par la loi de probabilité choisie.

Le test du χ^2 se définit comme ceci :

Soit un échantillon de données $(x_1, x_2, \cdots, x_N)$ d'une variable aléatoire X qui prend un nombre fini J de valeurs $(v_1, v_2, \cdots, v_J)$. Nous voulons tester la loi $\mathbb{P}$ avec $\forall j \in \{1, \cdots, J\}$, $\mathbb{P}(Y = v_j) = p_j$.

Nous testons donc :

- $\mathcal{H}_0$: l'échantillon $(x_1, x_2, \cdots, x_N)$ suit la loi de $\mathbb{P}$
- $\mathcal{H}_1$: l'échantillon $(x_1, x_2, \cdots, x_N)$ ne suit pas la loi de $\mathbb{P}$

Plus précisément nous testons $\mathcal{H}_0$: La probabilité que X prenne la valeur v_j vaut p_j, pour $j \in \{1, \cdots, J\}$, avec $\sum_{j=1}^{J} p_j = 1$.

On appelle $\hat{p}_j$ la probabilité empirique que X prenne la valeur v_j. C'est-à-dire le nombre d'observations x_i qui prennent la valeur v_j dans l'échantillon divisé par le nombre total N d'observations :

$$\hat{p}_j = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{\{x_i = v_j\}}$$

Nous pouvons alors définir la statistique de test du χ^2 :

$$T = \sum_{j=1}^{J} \frac{(N\hat{p}_j - Np_j)^2}{Np_j} = \sum_{j=1}^{J} \frac{(nj - Np_j)^2}{Np_j}, \ avec\ n_j = N\hat{p}_j = \sum_{i=1}^{N} \mathbb{1}_{\{x_i = v_j\}}$$

Sous $\mathcal{H}_0$, c'est à dire l'hypothèse nulle, cette statistique suit une loi du χ^2 à $(J-1)$ degrés de liberté.

Et nous pouvons donc construire un test de niveau α en rejetant l'hypothèse nulle lorsque la statistique de test est plus grande que le quantile d'ordre $1\breve{}\alpha$ de la loi du χ^2 à $(J\breve{}1)$ degrés de liberté :

$T \geq F^{-1}_{\chi^2(J-1)}(1-\alpha)$ avec $F^{-1}_{\chi^2(J-1)}(1-\alpha)$ le quantile d'ordre $(1-\alpha)$ de la loi du χ^2 à $(J-1)$ degrés de liberté.

Pour la loi de probabilité de Benford avec un risque d'erreur à $\alpha = 0.05$

Le test du χ^2 pour la loi de probabilité de Benford s'écrit sous la forme :

Soit $(x_1, x_2, \cdots, x_N)$ un échantillon de données d'une variable aléatoire X, dans notre cas le premier chiffre significatif des données du Covid, qui prend un nombre fini $J = 9$ de valeurs $(v_1 = 1, v_2 = 2, \cdots, v_9 = 9)$.

Nous voulons donc tester :

- $\mathcal{H}_0$: l'échantillon $(x_1, x_2, \cdots, x_N)$ suit la loi de Benford
- $\mathcal{H}_1$: l'échantillon $(x_1, x_2, \cdots, x_N)$ ne suit pas la loi de Benford

Nous avons la probabilité théorique qu'une variable aléatoire suivant la loi de Benford prenne la valeur v_j vaut p_j, nous avons vu le tableau des probabilités plus haut. Et la probabilité que X prennent la valeur v_j qui vaut $\hat{p}_j$.

Nous obtenons la statistique de test du χ^2 qui vaut :

$$T = \sum_{j=1}^{9} \frac{(N\hat{p}_j - Np_j)^2}{Np_j} = N \sum_{j=1}^{9} \frac{(\hat{p}_j - p_j)^2}{p_j}$$

Sous $\mathcal{H}_0$, c'est à dire l'hypothèse nulle, cette statistique suit une loi du χ^2 à $J - 1 = 8$ degrés de liberté.

Nous obtenons donc un test à $\alpha = 0.05$, ie à 5%, en rejetant l'hypothèse nulle lorsque $T \geq F^{-1}_{\chi^2(8)}(0.95) = 2.73$.

III.1.2 - Test de Kolmogorov-Smirnov

Ce test porte le nom du mathématicien russe Andréi Nikoláyevich Kolmogorov qui établit l'axiomatique des probabilités en 1933.

Sa principale différence avec le test du χ^2 est qu'il est fondé sur les fonctions de répartition plutôt que sur les densités.

Le test de Kolmogorov-Smirnov est un test d'ajustement, qui compare la distribution observée d'un échantillon à une distribution théorique choisie. Ce test mesure l'**écart maximum** qui existe entre la fonction de répartition empirique et la fonction de répartition théorique de la loi probabilité choisie. Il s'adapte aux échelles ordinales ce qui est un avantage, mais son principal défaut est de ne pas être très efficace dans les queues de distribution.

Le test de Kolmogorov-Smirnov se définit comme ceci :

Soit $(X_1, X_2, \cdots, X_N)$ un échantillon d'une variable aléatoire X de loi inconnue $\mathbb{P}$. Nous voulons tester si la loi de $\mathbb{P}$ a pour fonction de répartition F, avec F la fonction de répartition d'une loi de probabilité choisie.

De plus, notons $\hat{F} : \mathbb{R} \longrightarrow [0, 1]$, la fonction de répartition empirique de l'échantillon $(X_1, X_2, \cdots, X_N)$. Commençons par trier par ordre croissant les valeurs des X_i de l'échantillon, traditionnellement appelé les *statistiques d'ordre* de l'échantillon.

La fonction de répartition empirique est définie par : [1]

$$\hat{F}(x) = \begin{cases} 0 & si \;\; x < X_{(1)} \\ \frac{i}{N} & si \;\; X_{(i)} \leq x \leq X_{(i+1)} \\ 1 & si \;\; x > X_{(N)} \end{cases}$$

Nous estimons donc $F(x) = \mathbb{P}(X \leq x)$ au moyen de la proportion $\hat{F}(x)$ d'éléments de l'échantillon qui sont inférieurs ou égaux à x.

Nous testons donc :

- $\mathcal{H}_0$: la loi $\mathbb{P}$ a la même fonction de répartition F qu'une loi continue donnée
- $\mathcal{H}_1$: la loi $\mathbb{P}$ n'a pas pour fonction de répartition F

Nous mesurons l'adéquation de la fonction de répartition empirique à la fonction F par la distance de Kolmogorov-Smirnov, qui est la distance de la norme uniforme entre fonctions de répartitions. Pour la calculer, il suffit d'évaluer la différence entre $\hat{F}$ et F aux points $X_{(i)}$.

$$D_{KS}(F,\hat{F}) = \max_{i=1,\cdots,N}\left\{|F(X_{(i)}) - \hat{F}(X_{(i)})|, |F(X_{(i)}) - \hat{F}(X_{(i-1)})|\right\} \max_{i=1,\cdots,N}\left\{|F(X_{(i)}) - \frac{i}{N}|, |F(X_{(i)}) - \frac{i-1}{N}|\right\}$$

Voici une représentation du calcul de la distance de Kolmogorov-Smirnov ([1]) . Graphiquement, c'est le plus grand écart vertical en valeur absolue entre la valeur empirique et la valeur théorique.

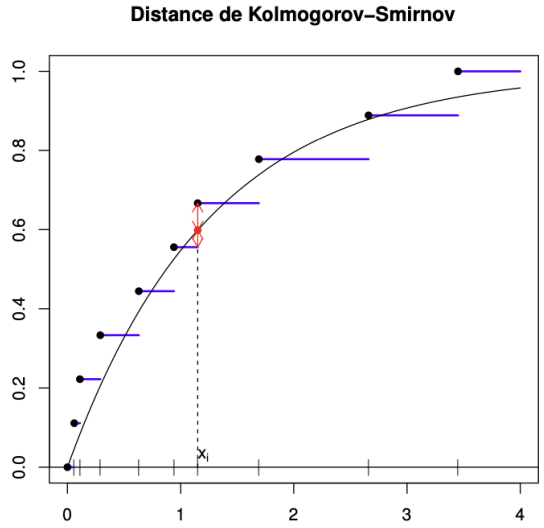

Figure 4: **Distance de Kolmogorov-Smirnov**

Sous l'hypothèse $\mathcal{H}_0$, la fonction de répartition $\hat{F}$ est très proche de F, et la loi de la variable de décision $D_{KS}(F,\hat{F})$ ne dépend pas de F. Nous comparons la valeur obtenue à une valeur critique $D_\alpha(N)$ fournie par les tables de Kolmogorov-Smirnov, le test est unilatéral. Si $D_{KS} > D_\alpha(N)$, nous rejetons $\mathcal{H}_0$ avec un risque α.

Remarque : Le test de Kolmogorov-Smirnov est préféré au test du χ^2 lorsque le caractère observé prend des valeurs continues. Ici, les données du Covid des pays du monde sont des données à caractère discret, donc nous allons pas utiliser ce test.

III.1.3 - Test du T_k

Théorème : Soit $X_1, \ldots, X_n$ des copies indépendantes d'une variable aléatoire X de densité $f(.)$ par rapport à une mesure v. Soit $\{h_0(.) := 1, h_k(.), k = 1, 2, ...\}$ une suite de fonctions orthonormales par rapport à $f(.)$; plus précisément, $\int h_k(x)h_{k'}(x)f(x)dv(x) = \delta_{kk'}$, la fonction delta de Kronecker. Soit $U_k = n^{-1/2}\sum_{i=1}^n h_k(X_i)$ et pour un entier $K \geq 1$, soir $T_k = \sum_{k=1}^K U_k^2$. Alors sous H_0, $T_K \xrightarrow{L} \chi^2_K$, la loi khi-deux à K degrés de liberté, et un test de niveau asymptotique α rejette H_0 si la valeur observée de T_K dépasse $x^2_{K,1-\alpha}$, le quantile d'ordre $1-\alpha$ de cette loi χ^2_K.

IV - Tests sur le premier chiffre significatif sur les données Covid des différents pays du monde

Comme observé précédemment il existe de nombreux tests d'adéquation pour la détection de fraudes via la loi de Newcomb-Benford. Par soucis de performance on décidera pour la suite d'appliquer le test du T_2 considéré comme l'un des plus puissants parmi les tests d'adéquations lisses pour la loi de Newcomb-Benford ([**?**]). Pour ce faire on aura recours au package benfordSmoothTest ([7]) .

Nous décidons alors dans un premier temps d'appliquer le test du T_2 sur la fréquence de distribution du premier chiffre significatif des nouveaux cas quotidiens de COVID-19 rapportés par 215 pays (pour plus d'informations sur les données veuillez consulter le github du CSSE ([3]) basés notamment sur les données de l'OMS, de l'ECDC etc ...). Puis, dans un second temps nous procéderons à l'estimation des p-values "ajustées" à l'aide de la méthode de correction de Benjamini, Hochberg & Yekutieli. Plus précisément à l'aide de la commande "p.adjust" ([11]) sur R.

En effet, lorsque des comparaisons multiples sont réalisées, le risque de se tromper n'est plus contrôlé.

Dans notre situation il semble donc nécessaire d'ajuster les p-values obtenues pour chacun des multiples tests, afin que le risque de se tromper pour l'ensemble de ces tests, soit à nouveau contrôlé, à niveau 5%.

IV.1 - Analyse des données

Après analyse, on observe qu'on rejette $\mathcal{H}_0$: *l'échantillon suit une loi de Newcomb-Benford contre* $\mathcal{H}_1$: *l'échantillon ne suis pas une loi de Newcomb-Benford* pour 33 pays. On remarque alors qu'on ne rejette pas $\mathcal{H}_0$ pour les 182 autres pays avec un risque d'erreur $\alpha = 5\%$.

Cette étude nous permet de lever un drapeau d'alerte face aux pays qui ne passent pas le test.

Nous listerons ces pays dans le tableau ci-dessous indiquant pour chaque pays la statistique du test du T_2 (T_2) et la p-value ajustée associée (P_value).

Pays	T_2	P_value	Pays	T_2	P_value
Africa	14.371	0.032	Latvia	27.235	0
Asia	99.288	0	Libya	28.853	0
Australia	16.357	0.013	Malaysia	21.142	0
Austria	14.602	0.029	Moldova	16.864	0.01
Belarus	19.117	0.006	Morocco	17.035	0.01
Cook Islands	17.467	0.01	Myanmar	19.548	0.006
Cuba	39.368	0	Oceania	31.464	0
Cyprus	16.007	0.013	Poland	18.842	0.006
Djibouti	16.094	0.01	Qatar	22.296	0
Egypt	81.358	0	Russia	24.994	0
Estonia	18.337	0.006	Serbia	52.819	0
Gibraltar	15.726	0.017	Syria	14.114	0.035
Greece	16.425	0.013	Taiwan	18.496	0.006
Iran	24.989	0	Trinidad and Tobago	41.511	0
Ireland	21.69	0	United Kingdom	28.201	0
Japan	33.564	0	Vietnam	56.115	0
Kuwait	24.557	0			

Figure 5: **Liste des pays pour lesquels on rejette $\mathcal{H}_0$**

Parmi eux on retrouve notamment Cuba, le Qatar, la Russie, le Japon, le Royaume-Uni ou encore Taïwan.

Ci-dessous nous décidons alors de représenter la distribution des premiers chiffres significatifs allant de 1 à 9 des données quotidiennes sur les nouveaux cas provenant de ces pays qui ne sont manifestement pas conformes à la loi de Newcomb-Benford.

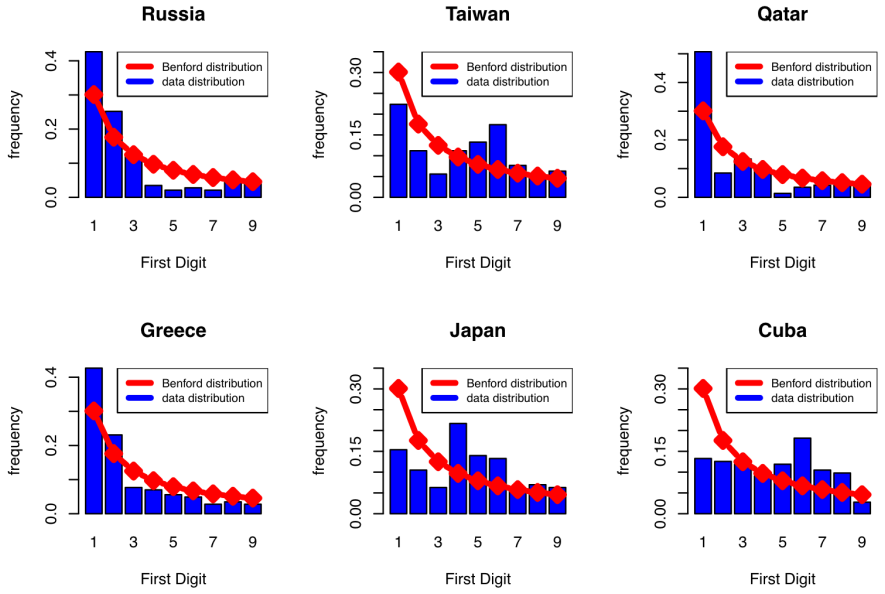

Figure 6: **Distribution des premiers chiffres significatifs**

Nous remarquons facilement une différence significative avec la distribution des premiers chiffres significatifs allant de 1 à 9 des données quotidiennes sur les nouveaux cas provenant des pays pour lesquels nous ne rejettons pas l'hypothèse selon laquelle ils seraient conformes à la loi de Benford :

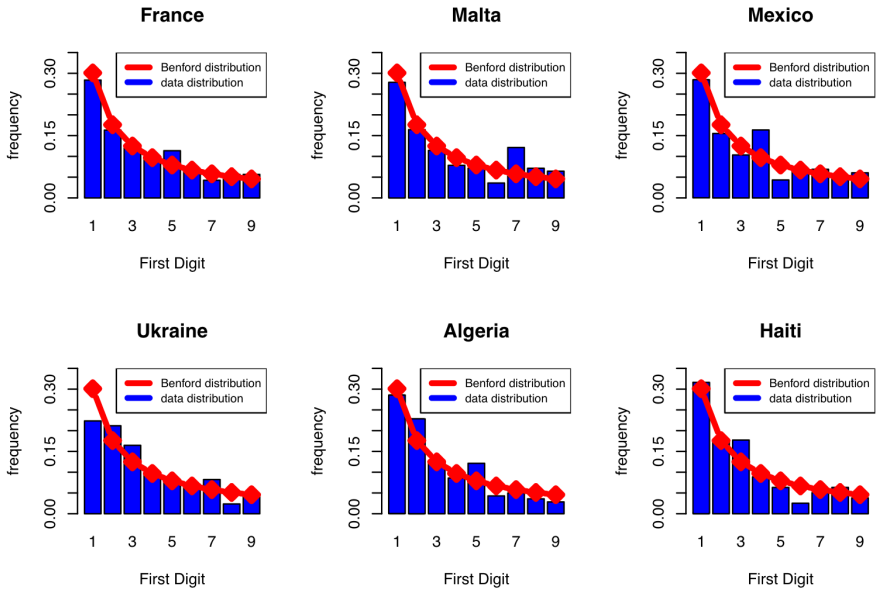

Figure 7: **Distribution des premiers chiffres significatifs**

Voir tableau complet en annexe à la figure (10).

Nous observons effectivement des discordances non négligeables entre les répartitions théoriques des premiers chiffres significatifs suivant la loi de Newcomb-Benford et la répartition des premiers chiffres significatifs des échantillons observés. Ceci nous réconforte dans l'hypothèse de rejet de $\mathcal{H}_0$.

Nous observons tout de même la présence de pays pour lesquels nous ne pouvons pas conclure.

Il serait intéressant de se questionner sur la véracité des données de nouveaux cas de COVID-19 des pays comme le Vatican ou encore Wallis et Futuna qui n'ont répertorié aucun cas depuis décembre 2021.

V - Test sur le bloc des deux premiers chiffres significatifs

L'objectif de cette partie est de reprendre la méthodologie de l'article [8] sur la construction des tests lisses d'adéquation, pour l'appliquer à la loi de Benford sur le bloc des deux premiers chiffres significatifs.

Nous allons ainsi étendre les tests T_K utilisés précédemment sur le premier chiffre significatif, pour les appliquer sur le bloc des deux premiers chiffres, et ainsi voire si les conclusions de rejet parmi les pays restent les mêmes.

Dans un premier temps nous construirons donc ces nouveaux tests. Puis dans un second nous observerons leurs courbes de puissance par rapport à deux autres tests : χ^2 et Freedman-Watson (U^2), sur deux alternatives : Rodriguez et Pietronero. Cela s'inspire complètement de [8].

V.1 - Préambule : Cadre Théorique

1) Loi de Benford sur le bloc des deux premiers chiffres significatifs :

Soit $X \in [1;10[$ la mantisse de notre nombre aléatoire.

On suppose que $\mathbb{P}(X \leq d) = \log_{10}(d)$ pour tout $d \in [1;10[$. C'est-à-dire X suit la loi de Benford continue.

Considérons $Y \in [\![10;99]\!]$ les deux premiers chiffres significatifs de X. Alors par un raisonnement analogue à la construction de la loi de Benford sur le premier chiffre :

$$\begin{aligned}\forall y \in [\![10;99]\!], \quad \mathbb{P}(Y=y) &= \mathbb{P}\left(\frac{y}{10} \leq X < \frac{y}{10} + \frac{1}{10}\right) \\ &= \log_{10}\left(\frac{y}{10} + \frac{1}{10}\right) - \log_{10}\left(\frac{y}{10}\right) \\ &= \log_{10}\left(\frac{y+1}{y}\right) \\ &= \log_{10}\left(1 + \frac{1}{y}\right)\end{aligned}$$

2) Théorèmes (issus de [8]) :

Tout repose sur le théorème suivant. Il explique comment construire une famille de tests lisses d'adéquation, indexée par l'entier K.

Théorème 1 : Soit G une loi cible, ayant pour densité f par rapport à une mesure dominante ν. Soit $(X_1, X_2, \ldots, X_n)$ un n-échantillon issu de $X \sim G_A$. On veut tester $\mathcal{H}_0 : G_A = G$, contre $\mathcal{H}_1 : G_A \neq G$.

Soit $\{h_0 \equiv 1, h_k, k \geq 1\}$ une famille de fonctions orthonormales par rapport à f, c'est-à-dire $\displaystyle\int h_k(x)h_{k'}(x)f(x)\,\mathrm{d}\nu(x) = \delta_{kk'}$, la fonction delta de Kronecker.

Soit $U_k = n^{-1/2}\sum_{i=1}^{n} h_k(X_i)$, et soit pour un entier $K \geq 1$, $T_K = \sum_{k=1}^{K} U_k^2$.

Alors sous H_0, $T_K \xrightarrow[n\to+\infty]{\mathcal{L}} \chi^2_K$, et un test de niveau asymptotique α rejette $\mathcal{H}_0$ au profit de $\mathcal{H}_1$ si la valeur observée de T_K dépasse $x_{1-\alpha}$ le quantile d'ordre $1-\alpha$ d'une loi χ^2_K.

Nous allons maintenant spécialiser ce théorème au cas où G est la loi de Benford précédente sur le bloc des deux premiers chiffres significatifs. Nous obtiendrons ainsi plusieurs tests d'adéquation T_K pour G.

Pour ce faire il nous faut calculer des fonctions orthonormales h_k associées à G comme dans le théorème ci-dessus. Nous utilisons le résultat suivant (qui est général) :

Théorème 2 : Soit $X \sim G$, et $\mu_k = \mathbb{E}(X^k)$, pour $k \geq 0$. Soit aussi la matrice $M_k = [\mu_{i+j}]_{i,j=0,\ldots,k-1}$, le vecteur $\lambda_k = (\mu_k, \mu_{k+1}, \ldots, \mu_{2k-1})^T$, et la constante $c_k = \mu_{2k} - \lambda_k^T M_k^{-1}\lambda_k$. Alors les polynômes :

$$h_k(x) = c_k^{-1/2}(x^k - (1, x, x^2, \ldots, x^{k-1})M_k^{-1}\lambda_k)$$

satisfont le théorème précédent.

Heureusement pour nous, la loi de Benford à deux chiffres est bornée p.s., et donc admet un moment pour tous les ordres. Nous avons donc tous les éléments de "la recette de cuisine."

V.2 - Construction des tests lisses T_K :

Soit X suivant la loi de Benford sur le bloc des deux premiers chiffres significatifs, $\forall x \in [\![10;99]\!]$, $\mathbb{P}(X = x) = \log_{10}\left(1 + \frac{1}{x}\right)$. On note $X \sim LB(2)$.

Cette partie de construction a été effectuée sur R, et toutes les commandes se trouvent en annexe.

Nous commençons par calculer les fonctions orthonormales h_k du théorème numéro 2) ci-dessus, associées à la $LB(2)$. On obtient :

- $h_1(x) = -1.54751771 + 0.04010177x$
- $h_2(x) = 2.795867953 - 0.167441591x + 0.001736457x^2$
- $h_3(x) = -5.18781 + 4.869054e^{-01}x - 1.155519e^{-02}x^2 + 7.630402e^{-05}x^3$
- $h_4(x) = 9.725235 - 1.237520x + 4.769440e^{-02}x^2 - 6.950207e^{-04}x^3 + 3.371880e^{-06}x^4$

Avec ces quatre fonctions nous pouvons donc construire T_K pour $1 \leq K \leq 4$. Le choix de K est important : s'il est trop grand alors le test sera très peu puissant, et s'il est trop petit alors il le sera également sur certaines alternatives.

Dans le package associé à [8], on peut calculer les tests pour K entre 1 et 7. Pour des raisons pratiques ici, nous n'avons pas pu calculer les h_k pour $k \geq 5$. En effet la $LB(2)$ a un support entre 10 et 99, et ses moments deviennent très rapidement élevés. Or dans le théorème numéro 2), nous calculons l'inverse de M_k, ce qui en fait une matrice avec de très petits coefficients. Petits à tel point que le logiciel "plante" car il "ne peut pas effectuer de division par zéro" lors du calcul de l'inverse. Autrement dit, R considère M_k comme une matrice non-inversible, ce qui n'est pourtant pas le cas.

Cependant, cela n'est pas gênant, car comme nous pourrons le voir plus tard lors du calcul des courbes de puissances, le test préféré est celui du T_2, surpassant T_3 et T_4. Ce qui fait des T_K, $K \geq 5$, de mauvais candidats.

Nous pouvons donc maintenant calculer la statistique T_K pour $1 \leq K \leq 4$, et effectuer le test proposé dans le théorème numéro 1) : On rejette $\mathcal{H}_0$ si T_K dépasse le quantile d'ordre $1 - \alpha$ de sa loi sous $\mathcal{H}_0$.

Pour une observation $T_K(\omega) = t$, nous calculons la p.value associée : $\mathbb{P}_{H_0}(t \leq T_K)$. Si n, le nombre de X_i, est grand, alors on peut approximer la p.value asymptotiquement par $\mathbb{P}(t \leq \chi^2_K)$. Dans la pratique, si $n \leq 100$, on fait l'approximation par Monte-Carlo, et sinon on la fait par approximation asymptotique du χ^2_K.

Justification du choix $n > 100$:

Ci-dessous, voici en couleur les fonctions de répartition empirique issues d'échantillons de taille 10 000 de T_K (pour $n = 101$, et $n = 102$ données), et générés sous $\mathcal{H}_0$. En noir la fonction de répartition théorique d'un χ^2_K. Un graphique pour chaque $1 \leq K \leq 4$.

On constate que pour n > 100, l'ecdf de T_K est quasiment égale à la fonction de répartition théorique d'une χ^2_K. À tel point qu'il est extrêmement difficile de distinguer cette dernière, c'est-à-dire la courbe noire, et cela même en zoomant (elle existe bien sur chaque graphique !).

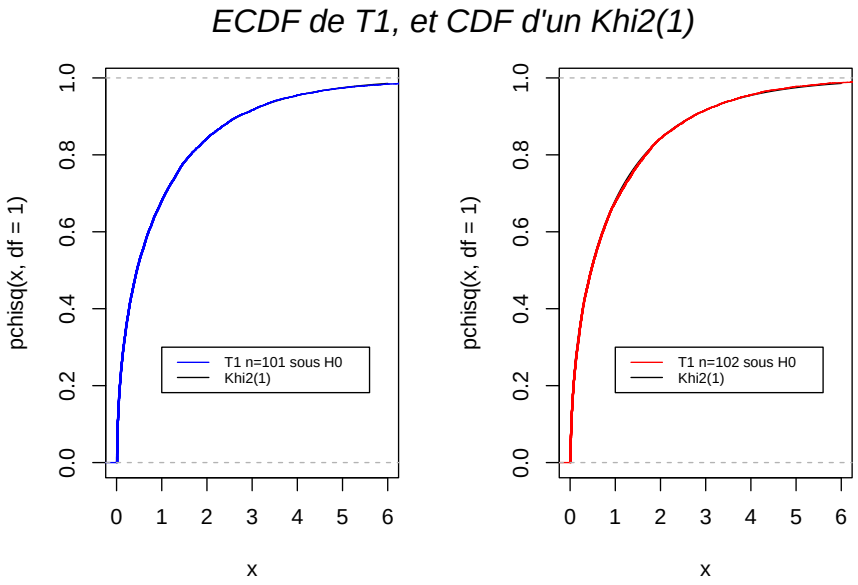

Figure A

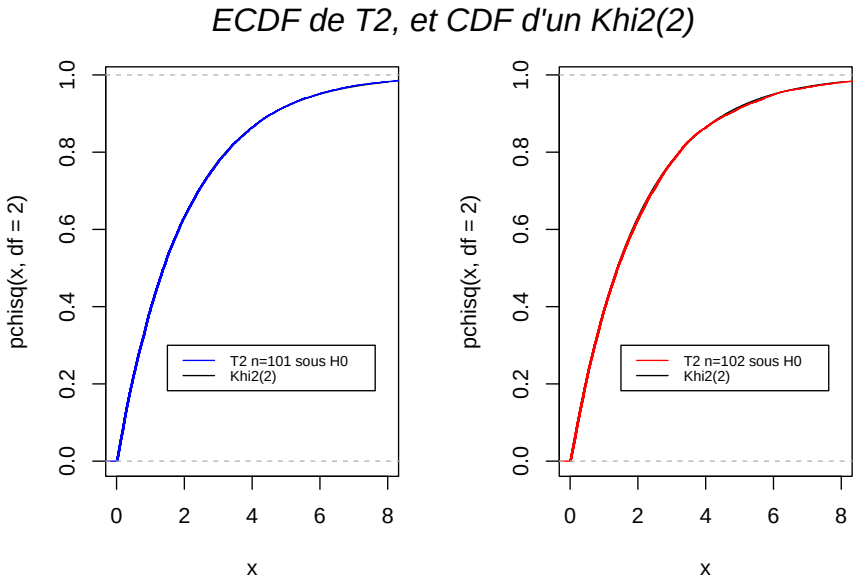

Figure B

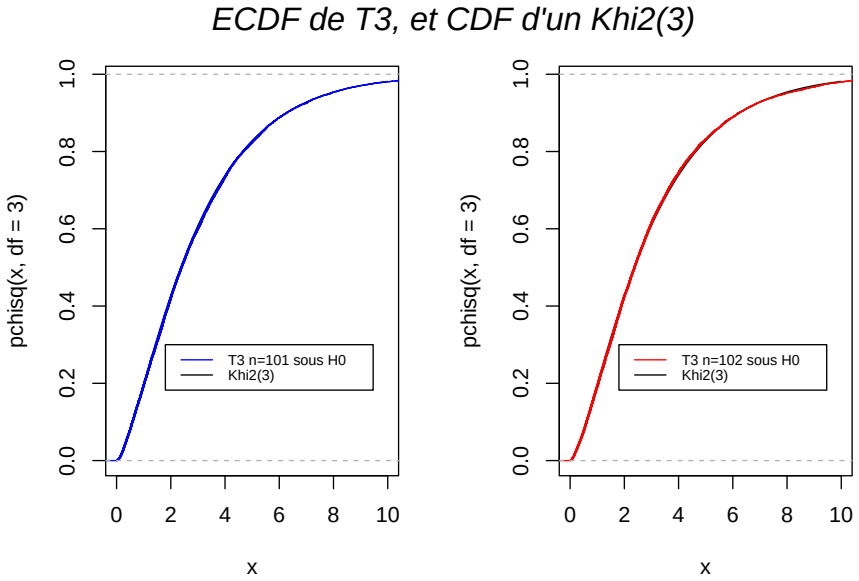

Figure C

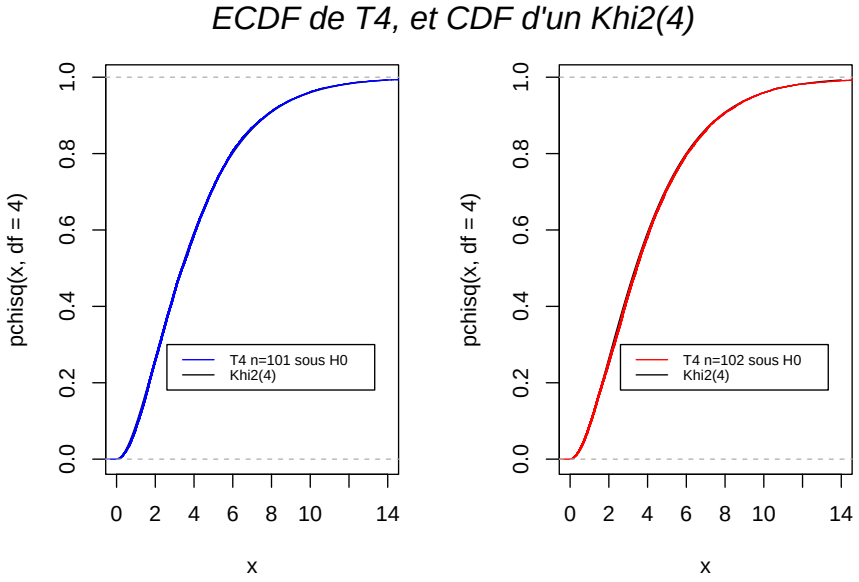

Figure D

Nous avons même recommencé l'opération, toujours avec des échantillons de 10 000 T_k, mais cette fois-ci pour $n = 50$. En bleu la fonction de répartition empirique de T_k, et en noir la fonction de répartition théorique d'un χ^2_K.

On constate toujours une extrême proximité des courbes bleus et noires. C'est pourquoi le choix de $n > 100$ est même peut-être trop exigeant.

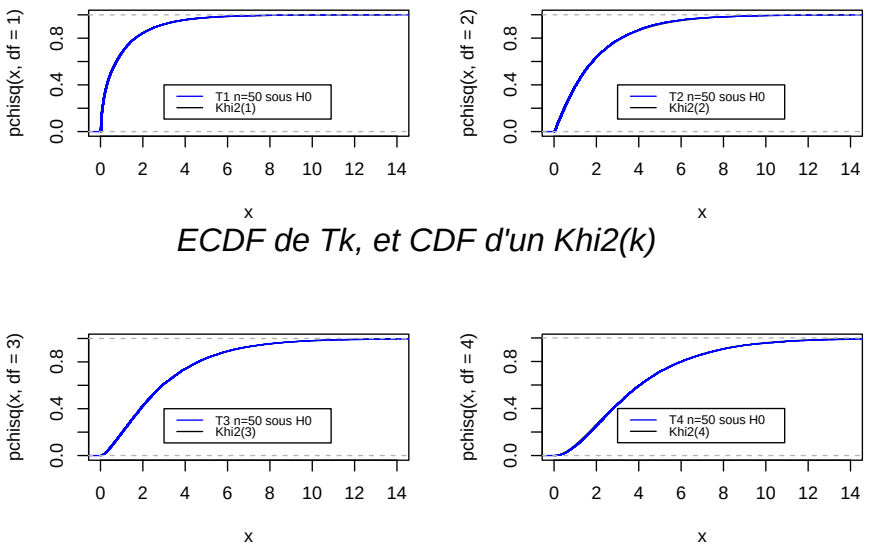

Figure E

V.3 - Comparaison des puissances avec d'autres tests :

Nous allons présenter les courbes de puissance des quatre tests T_k construits ci-dessus, du test du χ^2, et de celui de Freedman-Watson (U^2), sur deux alternatives à la $LB(2)$: La Rodriguez et la Pietronero bivarié.

→ Le test du U^{2} étant moins connu que celui du χ^{2}, on rappelle sa statistique :

Pour $d \in [\![10; 99]\!]$, on pose $\hat{p_d} = n_d/n$ la proportion de d dans l'échantillon $(X_1, X_2, \ldots, X_n)$, et π_d les probabilités théoriques de la $LB(2)$. Soit aussi $S_d = \sum_{i=10}^{d} \hat{p_i}$ et $S_d^* = \sum_{i=10}^{d} \pi_i$. Posons $Z_d = S_d - S_d^*$ l'écart entre la fonction de répartition empirique et la fonction de répartition théorique de la $LB(2)$. Posons également $t_d = (\pi_d + \pi_{d+1})/2$ pour $d = 10, 11, \ldots, 98$, et $t_{99} = (\pi_{99} + \pi_{10})/2$. Alors :

$$U^2 = \frac{1}{n} \sum_{d=10}^{99} (Z_d - \bar{Z})^2 t_d$$

→ On présente également les alternatives à la $LB(2)$:

1) Rodriguez bivarié : ($\gamma = -1$: on retrouve la $LB(2)$)

$$\mathbb{P}(X = d) = \begin{cases} \log_{10}(1 + \frac{1}{d}) & \text{si } \gamma = -1 \\ \frac{1}{810}\left(9 + 19\ln(10) + 9d\ln(d) - 9(d+1)\ln(d+1)\right) & \text{si } \gamma = 0 \\ \frac{\gamma+1}{90\gamma} - \frac{(d+1)^{\gamma+1} - d^{\gamma+1}}{\gamma(100^{\gamma+1} - 10^{\gamma+1})} & \text{sinon} \end{cases}$$

2) Pietronero bivarié : ($\gamma = 0$: on retrouve la $LB(2)$)

$$\mathbb{P}(X=d)=\begin{cases}\log_{10}(1+\dfrac{1}{y}) & \text{si } \gamma=0\\ \dfrac{d^{-\gamma}-(d+1)^{-\gamma}}{10^{-\gamma}-100^{-\gamma}} & \text{sinon}\end{cases}$$

Dans [8], voici les courbes de puissances obtenues pour les tests ci-dessus (T^2, χ^2, U^2) portant sur l'adéquation du premier chiffre significatif avec la $LB(1)$ (loi de Benford simple), et pour les alternatives correspondantes univariées :

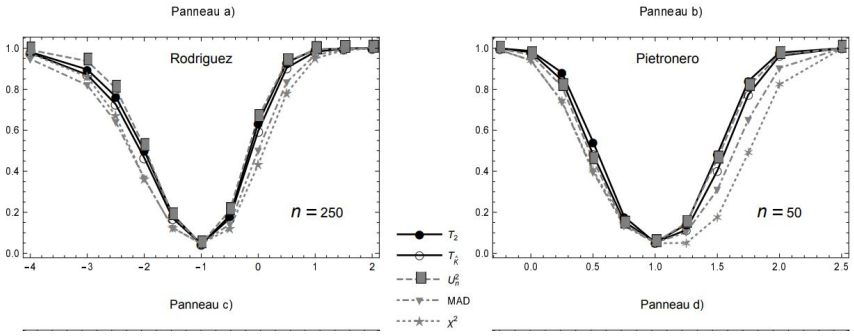

Figure F

Les tests du T^2 et du U^2 sont très bons. Tandis que celui du χ^2 est très mauvais. Voyons voir si nous avons équivalence avec le cas bivarié à deux chiffres significatifs.

Nous allons effectuer des simulations pour trois choix de n : 50, 100, et 150. Ces ordres de grandeur correspondent à la taille moyenne des échantillons que l'on peut rencontrer sur les données Covid des pays.

Notons également que chaque test sera effectué au niveau 5%.

Dans un premier temps, pour chaque choix de n, nous approximons les quantiles de référence de chacun des tests par les quantiles empiriques d'ordres 95% des statistiques précédentes sous $\mathcal{H}_0 : X \sim LB(2)$. Cela dans le but d'être sûr de bien comparer la puissance des différents tests à 5%. Nous effectuons 100 000 réplications pour cela.

Ensuite pour chaque alternative, chaque choix de n, nous allons générer 10 000 échantillons de taille n, pour un γ fixé, et appliquer les tests à chacun de ces échantillons, puis compter le nombre de rejet parmi le nombre total afin d'approximer par Monte-Carlo $\mathbb{P}_\gamma(\Phi = 1)$. Nous allons parcourir une centaine de γ sur un intervalle $[-5; 5]$.

Notons que nous obtenons une erreur d'approximation des probabilités qui ne dépasse jamais les 0.01 pour Monte-Carlo (au niveau de confiance 95%).

Nous avons les graphiques suivants :

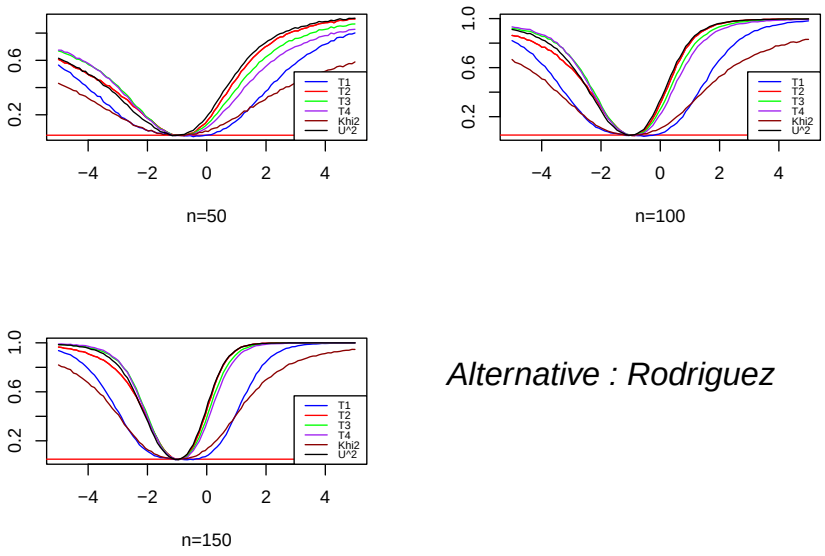

Figure G

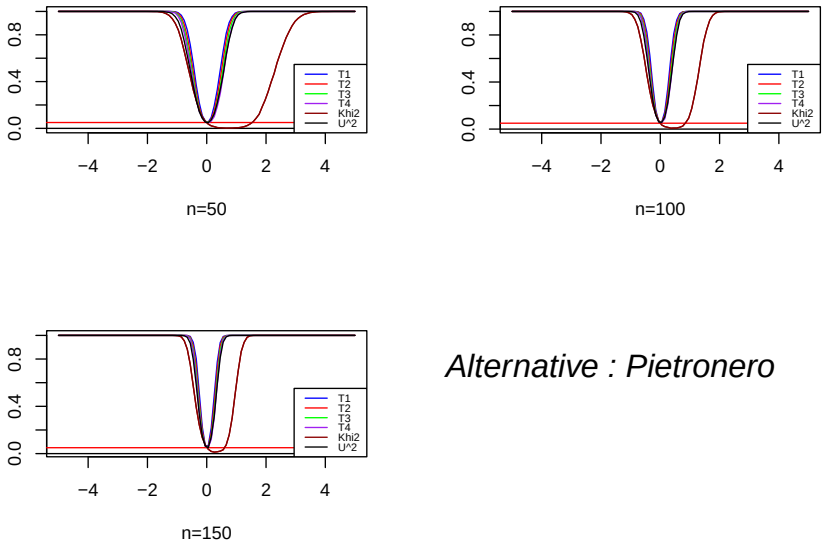

Figure H

1) Pour l'alternative Rodriguez tout d'abord. On retrouve le test du T_2 et celui du U^2parmi les meilleurs. Le U^2 est même légèrement mieux que le T_2, en particulier pour des $\gamma < -1$. Exactement comme pour le cas univarié. On retrouve aussi un test du χ^2 très mauvais. Cependant on constate que les tests T_3 et T_4 sont les meilleurs parmi ceux proposés, pour des $\gamma < -1$. On pourra même préférer le T_3 au T_4 car pour des $\gamma > -1$, il est meilleur. Par conséquent, si l'on suspecte une alternative Rodriguez avec des $\gamma < -1$ ou bien des $\gamma > -1$, on pourra privilégier le T_3, ou bien le U^2. Le problème c'est qu'une telle chose est difficilement suspectable.

2) Pour l'alternative Pietronero. Le test du χ^2 est toujours très mauvais. Sa courbe vient même toucher $y = 0$, pour x entre 0 et 1, ce qui est surprenant... Peut-être une erreur lors de la simulation, mais

je n'ai pas pu la détecter. Pour les autres tests, le U^2 est cette fois-ci assez mauvais par rapport aux autres. Comme dans le cas univarié, il est moins bon que le T_2. Ce dernier T_2 reste toujours en tête du classement, mais il est dépassé de peu par le test du T_1. Malheureusement le T_1 était assez mauvais pour l'alternative Rodriguez.

On retrouve tout de même des résultats proches de ceux du cas univarié. On peut même suspecter la même chose pour les alternatives que nous n'avons pas testée, et qui sont présentés dans [8]. Dans ces dernières alternatives, le test du U^2 s'effondrait, et le test du T_2 restait bon. De plus, juste en considérant les deux alternatives Rodriguez et Pietronero que nous avons présenté, ne sachant pas dans quelle situation on se situe dans la pratique, le test du T_2 est un bon candidat pour les deux alternatives. Nous allons d'ailleurs choisir ce test pour effectuer les analyses des données Covid par la suite.

V.4 - Mise en pratique sur les données

Après cette première analyse, nous décidons d'approfondir la recherche en analysant les deux premiers chiffres significatifs afin de repérer d'autres potentiels fraudeurs.

Nous obtenons alors la table suivante représentative des individus pour lesquels on rejette $\mathcal{H}_0$ (voir tableau complet figure (11) en annexe):

Pays	T_2	P_value	Pays	T_2	P_value
Africa	15.738	0.016	Latvia	29.333	0
Asia	96.327	0	Libya	27.72	0
Australia	17.671	0.007	Malaysia	21.719	0.002
Austria	13.305	0.044	Maldives	12.916	0.05
Belarus	13.935	0.034	Moldova	18.425	0.006
Cambodia	14.102	0.033	Montserrat	17.314	0.009
Comoros	13.558	0.04	Morocco	14.808	0.024
Cook Islands	18.139	0.006	Myanmar	16.104	0.014
Cuba	39.159	0	Oceania	34.985	0
Cyprus	20.681	0.003	Poland	19.698	0.004
Djibouti	31.834	0	Qatar	18.294	0.006
Egypt	72.478	0	Russia	20.552	0.003
Eritrea	17.751	0.007	Sao Tome and Principe	17.639	0.007
Estonia	17.796	0.007	Serbia	47.553	0
Falkland Islands	14.012	0.05	South Korea	14.889	0.023
Gibraltar	18.043	0.007	Sri Lanka	13.263	0.044
Guinea-Bissau	12.897	0.05	Taiwan	16.282	0.013
Iran	23.575	0.001	Trinidad and Tobago	41.847	0
Ireland	20.235	0.003	United Kingdom	27.431	0
Japan	30.688	0	Vietnam	45.682	0
Kuwait	27.21	0			

Figure 8: **Distribution des deux premiers chiffres significatifs**

Nous remarquons que pour nos deux analyses nous retrouvons 31 pays identiques dont la Russie, le Qatar ou encore le Royaume-Uni ce qui permet de confirmer le doute.

Par cette méthode on découvre également d'autres suspects que nous n'avions pas réussi à repérer précédemment qui sont :

- Le Cambodge
- Les Comores
- L'Érythrée
- Les îles Malouines
- La République de Guinée-Bissau
- Les Maldives
- Montserrat
- Sao Tomé-et-principe
- La Corée du Sud
- Le Sri Lanka

On ne peut que constater que pour ce test on ne rejette pas $\mathcal{H}_0$ ni pour la Grèce ni pour la Syrie qui étaient considérés comme suspects en vue des résultats des tests sur le premier chiffre significatif.

Nous décidons alors de pousser l'analyse sur ces deux pays afin d'essayer de comprendre la cause.

On décide alors d'observer la distribution de leurs deux premiers chiffres significatifs par rapport à la distribution théorique de Benford.

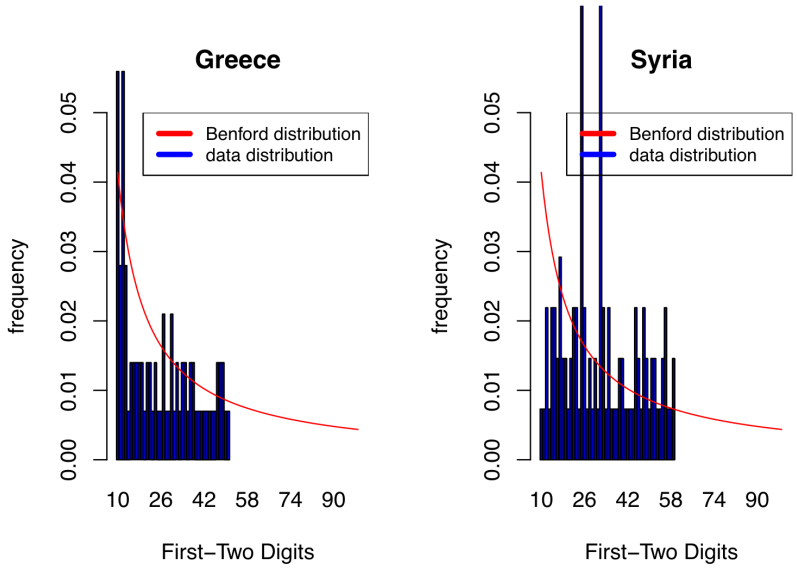

Figure 9: **Comparaison distribution échantillon et Benford**

Après observations approfondies on se rend compte que ceci peut être la résultante d'un manque de données car en effet notamment pour la Syrie on observe la présence de certaines fréquences très éloignées des fréquences théoriques associées à la distribution théorique de Benford.

Il serait donc quand même préférable de mener une enquête.

VI - Conclusion :

Les résultats présentés ici confirment la possibilité de corruption des données rapportées sur la COVID-19 dans de nombreux pays présentant des données douteuses comme Taiwan ou encore la Russie. Cependant, ces résultats nous permettent également de repérer des pays auxquels nous n'aurions pas pensé comme l'Australie ou encore les Maldives. Il serait donc fortement recommandé de mener une enquête concernant la véracité des données de ces pays car ces fraudes peuvent avoir un grand impact sur la prise de décision et sur la gestion de la pandémie de COVID-19 dans le monde entier

Cependant, notre étude n'est pas exempte de critique. En effet, il faut manier ce test avec précautions car il existe des fraudes qui restent indécelables avec cette analyse. Typiquement nous ne pouvons rien conclure sur les pays qui ne présentent que très peut de données.

References

[1] *Test de Kolmogorov-Smirnov*.

[2] *Test du khi-deux*.

[3] CSSEGI. *Données CSSE*, 2020.

[4] JEAN-PAUL DELAHAYE. *UNE EXPLICATION POUR LA LOI DE BENFORD*.

[5] JEAN-PAUL DELAHAYE and Nicolas GAUVRIT. *POURQUOI LA LOI DE BENFORD N'EST PAS MYSTÈRIEUSE*.

[6] B. Desgraupes. *Méthodes Statistiques : Tests d'adequation*.

[7] Gilles DUCHARME and Credo VOVOR-DASSU. *benfordSmoothTest*, 2020.

[8] Gilles R.DUCHARME and Credo VOVOR-DASSU. *Test d'adéquation loi de Benford*.

[9] Kolmogorov-Smirnov. *Fonction de repartition empirique*.

[10] Leonid I. GALTCHOUK. *STATISTIQUES TESTS D'HYPOTHÈSES*.

[11] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2022.

Annexe :

Tableau complet de l'application du T_2 au premier chiffre significatif :

Pays	T_2	P_value	Pays	T_2	P_value
Afghanistan	1.599	1	Costa Rica	3.946	1
Africa	14.371	0.032	Croatia	7.417	0.468
Albania	4.921	1	Cuba	39.368	0
Algeria	2.609	1	Curacao	2.041	1
Andorra	4.773	1	Cyprus	16.007	0.014
Angola	0.139	1	Czechia	8.062	0.356
Anguilla	0.32	1	Democratic Republic of Congo	0.757	1
Antigua and Barbuda	0.146	1	Denmark	8.889	0.258
Argentina	0.146	1	Djibouti	16.094	0.017
Armenia	10.703	0.134	Dominica	2.476	1
Aruba	0.308	1	Dominican Republic	0.162	1
Asia	99.288	0	Ecuador	7.15	0.526
Australia	16.357	0.014	Egypt	81.358	0
Austria	14.602	0.029	El Salvador	1.761	1
Azerbaijan	8.743	0.269	Equatorial Guinea	0.715	1
Bahamas	1.518	1	Eritrea	11.361	0.099
Bahrain	10.094	0.167	Estonia	18.337	0.006
Bangladesh	0.916	1	Eswatini	5.881	0.814
Barbados	2.554	1	Ethiopia	11.719	0.088
Belarus	19.117	0.006	Faeroe Islands	8.91	0.269
Belgium	4.993	1	Falkland Islands	6.476	0.675
Belize	0.499	1	Fiji	2.103	1
Benin	1.063	1	Finland	11.423	0.098
Bermuda	0.231	1	France	0.498	1
Bhutan	4.029	1	French Polynesia	2.449	1
Bolivia	3.386	1	Gabon	0.925	1
Bonaire Sint Eustatius and Saba	8.213	0.335	Gambia	0.265	1
Bosnia and Herzegovina	0.012	1	Georgia	1.13	1
Botswana	2.45	1	Germany	5.62	0.896
Brazil	1.498	1	Ghana	6.149	0.729
British Virgin Islands	1.036	1	Gibraltar	15.726	0.017
Brunei	2.121	1	Greece	16.425	0.014
Bulgaria	6.433	0.675	Greenland	2.612	1
Burkina Faso	3.034	1	Grenada	2.216	1
Burundi	1.25	1	Guatemala	3.755	1
Cambodia	12.878	0.052	Guinea	2.786	1
Cameroon	1.766	1	Guinea-Bissau	7.49	0.485
Canada	3.568	1	Guyana	9.24	0.226
Cape Verde	2.861	1	Haiti	0.629	1
Cayman Islands	4.113	1	Honduras	1.334	1
Central African Republic	0.237	1	Hong Kong	2.122	1
Chad	3.111	1	Hungary	1.511	1
Chile	9.327	0.223	Iceland	12.349	0.05
China	7.659	0.421	India	3.445	1
Colombia	10.34	0.155	Indonesia	1.135	1
Comoros	6.494	0.625	Iran	24.989	0
Congo	1.979	1	Iraq	9.375	0.223
Cook Islands	17.467	0.01	Ireland	21.69	0

Pays	T_2	P_value	Pays	T_2	P_value
Isle of Man	2.302	1	Nigeria	0.807	1
Israel	6.331	0.701	North America	2.79	1
Italy	3.645	1	North Macedonia	9.672	0.202
Jamaica	2.573	1	Norway	0.545	1
Japan	33.564	0	Oceania	31.464	0
Jordan	1.412	1	Oman	1.621	1
Kazakhstan	4.912	1	Pakistan	9.476	0.221
Kenya	9.038	0.245	Palau	9.095	0.223
Kiribati	6.385	0.749	Palestine	2.191	1
Kosovo	6.05	0.758	Panama	10.474	0.147
Kuwait	24.557	0	Papua New Guinea	3.836	1
Kyrgyzstan	1.525	1	Paraguay	1.241	1
Laos	2.36	1	Peru	0.566	1
Latvia	27.235	0	Philippines	3.937	1
Lebanon	1.522	1	Poland	18.842	0.006
Lesotho	0.974	1	Portugal	6.203	0.729
Liberia	0.933	1	Qatar	22.296	0
Libya	28.853	0	Romania	2.334	1
Liechtenstein	3.543	1	Russia	24.994	0
Lithuania	1.085	1	Rwanda	0.459	1
Low income	0.421	1	Saint Kitts and Nevis	2.961	1
Luxembourg	7.209	0.501	Saint Lucia	0.502	1
Madagascar	0.007	1	Saint Pierre and Miquelon	2.991	1
Malawi	3.885	1	Saint Vincent and the Grenadines	8.019	0.322
Malaysia	21.142	0	Samoa	0.981	1
Maldives	12.581	0.061	San Marino	1.639	1
Mali	0.207	1	Sao Tome and Principe	9.418	0.226
Malta	5.491	0.944	Saudi Arabia	0.537	1
Mauritania	1.435	1	Senegal	0.378	1
Mauritius	0.037	1	Serbia	52.819	0
Mexico	0.689	1	Seychelles	5.157	1
Moldova	16.864	0.01	Sierra Leone	2.598	1
Monaco	5.745	0.888	Singapore	0.873	1
Mongolia	8.426	0.31	Slovakia	3.415	1
Montenegro	6.489	0.675	Slovenia	7.191	0.501
Montserrat	8.068	0.366	Solomon Islands	0.628	1
Morocco	17.035	0.01	Somalia	1.297	1
Mozambique	2.451	1	South Africa	13.119	0.05
Myanmar	19.548	0.006	South America	4.335	1
Namibia	1.402	1	South Korea	13.121	0.05
Nepal	1.682	1	South Sudan	0.421	1
Netherlands	3.558	1	Spain	2.165	1
New Caledonia	1.92	1	Sri Lanka	12.911	0.052
New Zealand	5.151	1	Sudan	2.165	1
Nicaragua	6.559	0.675	Suriname	1.576	1
Niger	4.132	1	Sweden	5.79	0.841

Pays	T_2	P_value	Pays	T_2	P_value
Switzerland	2.149	1	Ukraine	1.83	1
Syria	14.114	0.035	United Arab Emirates	11.662	0.088
Taiwan	18.496	0.006	United Kingdom	28.201	0
Tajikistan	2.58	1	United States	3.034	1
Tanzania	0.664	1	Upper middle income	3.76	1
Thailand	0.948	1	Uruguay	4.307	1
Timor	0.827	1	Uzbekistan	10.269	0.157
Togo	0.688	1	Vanuatu	6.146	0.749
Tonga	1.519	1	Venezuela	1.206	1
Trinidad and Tobago	41.511	0	Vietnam	56.115	0
Tunisia	0.248	1	Yemen	0.271	1
Turkey	12.823	0.052	Zambia	4.704	1
Turks and Caicos Islands	3.674	1	Zimbabwe	1.974	1
Uganda	4.38	1			

Figure 10: **Tableau analyse premier chiffre significatif**

Tableau complet de l'application du T_2 aux deux premiers chiffres significatif :

Pays	T_2	P_value	Pays	T_2	P_value
Afghanistan	2.76	1	Costa Rica	2.865	1
Africa	15.738	0.016	Croatia	7.733	0.377
Albania	5.533	0.914	Cuba	39.159	0
Algeria	4.481	1	Curacao	2.43	1
Andorra	4.321	1	Cyprus	20.681	0.003
Angola	0.409	1	Czechia	6.574	0.591
Anguilla	0.829	1	Democratic Republic of Congo	0.736	1
Antigua and Barbuda	0.935	1	Denmark	8.402	0.291
Argentina	0.23	1	Djibouti	31.834	0
Armenia	9.292	0.205	Dominica	3.419	1
Aruba	0.691	1	Dominican Republic	0.128	1
Asia	96.327	0	Ecuador	8.249	0.309
Australia	17.671	0.007	Egypt	72.478	0
Austria	13.305	0.044	El Salvador	1.771	1
Azerbaijan	9.386	0.203	Equatorial Guinea	0.542	1
Bahamas	2.653	1	Eritrea	17.751	0.007
Bahrain	9.963	0.16	Estonia	17.796	0.007
Bangladesh	1.793	1	Eswatini	6.524	0.598
Barbados	2.335	1	Ethiopia	10.483	0.133
Belarus	13.935	0.034	Faeroe Islands	7.174	0.472
Belgium	3.562	1	Falkland Islands	14.012	0.061
Belize	0.524	1	Fiji	1.382	1
Benin	1.584	1	Finland	10.758	0.12
Bermuda	1.371	1	France	0.344	1
Bhutan	4.76	1	French Polynesia	2.327	1
Bolivia	3.085	1	Gabon	0.958	1
Bonaire Sint Eustatius and Saba	8.93	0.237	Gambia	0.863	1
Bosnia and Herzegovina	0.069	1	Georgia	0.815	1
Botswana	2.381	1	Germany	6.086	0.726
Brazil	1.27	1	Ghana	7.807	0.369
British Virgin Islands	1.617	1	Gibraltar	18.043	0.007
Brunei	2.671	1	Greece	11.896	0.074
Bulgaria	5.477	0.929	Greenland	2.634	1
Burkina Faso	1.647	1	Grenada	3.012	1
Burundi	1.482	1	Guatemala	2.569	1
Cambodia	14.102	0.033	Guinea	3.714	1
Cameroon	1.9	1	Guinea-Bissau	12.897	0.051
Canada	3.652	1	Guyana	7.011	0.493
Cape Verde	10.343	0.14	Haiti	1.627	1
Cayman Islands	4.014	1	Honduras	0.925	1
Central African Republic	0.11	1	Hong Kong	0.93	1
Chad	3.254	1	Hungary	1.088	1
Chile	7.108	0.482	Iceland	11.553	0.086
China	9.363	0.203	India	3.648	1
Colombia	10.653	0.124	Indonesia	1.815	1
Comoros	13.558	0.04	Iran	23.575	0.001
Congo	1.072	1	Iraq	9.245	0.206
Cook Islands	18.139	0.014	Ireland	20.235	0.003

Pays	T_2	P_value	Pays	T_2	P_value
Isle of Man	2.414	1	Oceania	34.985	0
Israel	6.028	0.73	Oman	1.448	1
Italy	3.824	1	Pakistan	9.713	0.178
Jamaica	3.296	1	Palau	12.757	0.053
Japan	30.688	0	Palestine	1.565	1
Jordan	1.994	1	Panama	9.344	0.203
Kazakhstan	3.928	1	Papua New Guinea	3.81	1
Kenya	7.966	0.35	Paraguay	0.734	1
Kiribati	11.326	0.093	Peru	1.039	1
Kosovo	8.397	0.291	Philippines	4.843	1
Kuwait	27.21	0	Poland	19.698	0.004
Kyrgyzstan	0.844	1	Portugal	4.652	1
Laos	3.787	1	Qatar	18.294	0.006
Latvia	29.333	0	Romania	1.648	1
Lebanon	2.111	1	Russia	20.552	0.003
Lesotho	0.903	1	Rwanda	2.45	1
Liberia	0.474	1	Saint Kitts and Nevis	8.621	0.268
Libya	27.72	0	Saint Lucia	0.486	1
Liechtenstein	4.366	1	Saint Pierre and Miquelon	4.231	1
Lithuania	0.965	1	Saint Vincent and the Grenadines	8.818	0.247
Low income	0.43	1	Samoa	1.479	1
Luxembourg	6.061	0.727	San Marino	1.95	1
Madagascar	0.167	1	Sao Tome and Principe	17.639	0.007
Malawi	2.183	1	Saudi Arabia	1.29	1
Malaysia	21.719	0.002	Senegal	1.479	1
Maldives	12.916	0.051	Serbia	47.553	0
Mali	0.266	1	Seychelles	4.625	1
Malta	5.614	0.888	Sierra Leone	3.874	1
Mauritania	0.566	1	Singapore	0.804	1
Mauritius	0.419	1	Slovakia	2.725	1
Mexico	0.621	1	Slovenia	6.945	0.503
Moldova	18.425	0.006	Solomon Islands	0.302	1
Monaco	4.741	1	Somalia	1.014	1
Mongolia	7.451	0.422	South Africa	10.253	0.143
Montenegro	7.492	0.42	South America	4.301	1
Montserrat	17.314	0.017	South Korea	14.889	0.023
Morocco	14.808	0.024	South Sudan	1.799	1
Mozambique	1.227	1	Spain	2.094	1
Myanmar	16.104	0.014	Sri Lanka	13.263	0.044
Namibia	1.774	1	Sudan	1.681	1
Nepal	2.115	1	Suriname	1.007	1
Netherlands	4.046	1	Sweden	5.32	0.994
New Caledonia	2.12	1	Switzerland	1.432	1
New Zealand	6.571	0.591	Syria	11.324	0.093
Nicaragua	6.084	0.697	Taiwan	16.282	0.014
Niger	3.553	1	Tajikistan	1.7	1
Nigeria	0.461	1	Tanzania	0.756	1
North America	2.821	1	Thailand	1.614	1
North Macedonia	7.854	0.365	Timor	3.882	1
Norway	0.654	1	Togo	2.263	1

Pays	T_2	P_value	Pays	T_2	P_value
Tonga	1.573	1	Upper middle income	7.218	0.468
Trinidad and Tobago	41.847	0	Uruguay	4.306	1
Tunisia	0.454	1	Uzbekistan	10.147	0.148
Turkey	12.231	0.066	Vanuatu	6.967	0.493
Turks and Caicos Islands	1.603	1	Venezuela	2.048	1
Uganda	1.719	1	Vietnam	45.682	0
Ukraine	2.004	1	Yemen	0.679	1
United Arab Emirates	11.933	0.074	Zambia	4.548	1
United Kingdom	27.431	0	Zimbabwe	2.091	1
United States	3.162	1			

Figure 11: **Tableau analyse premier chiffre significatif**

```
#---------------------------- Test lisse first digit -------------------------
library(dplyr)
library(tidyr)
library(data.table)
library(BenfordSmoothTest)
library(BenfordTests)
library(kableExtra)

#----------------------- Importation des données ----------------------------
b <-read.csv("covid6.csv",sep=",")
b <- data.table(b)
b <- b[b$date > "2021-12-01",]
drop_na(b)
b <- b[b$location!="World",]
b <- b[b$location!="Europe",]
b <- b[b$location!="European Union",]
b <- b[b$location!="Lower middle income"]
b <- b[b$location!="High income"]

# on commence à partir de septembre 2021 on s'intéresses aux données récentes.

#------------------------------------------------------------------------------
#------------------------------ Fonction graphes ----------------------------
library(tidyr)
library(dplyr)
vec <- function(df){
  x <- c(df$new_cases)
  x <- x[x!=0]
  j=1
  for(i in 1:length(x)){
  if(is.na(x[j])){
    x <- x[-j]
  }
  else{
    j <- j +1
  }
  }
```

```
  return(x)
}
benf <- function(dd){
x <- vec(dd)
benlaw <- function(d) log10(1 + 1 / d)
digits <- 1:9
firstDigit <- function(x) substr(gsub('[0.]', '', x), 1, 1)
pctFirstDigit <- function(x) data.frame(table(firstDigit(x)) / length(x))
df <- pctFirstDigit(x)
if(length(firstDigit(x))<=10){
    return()
}
else{
baseBarplot <- barplot(df$Freq[1:9], names.arg = digits, xlab = "First Digit",ylab="frequency", ylim = 
lines(x = baseBarplot[,1], y = benlaw(digits), col = "red", lwd = 4,
      type = "b", pch = 23, cex = 1.5, bg = "red")
legend(x="topright", legend=c("Benford distribution","data distribution"),
       col=c("red","blue"), lty=1,lwd=4, cex=0.8)
}
}
#-------------------------------------------------------------------------------

#------------- Fonction nom de pays pouvant passer le test -------------------
# Récupère les noms des pays ayant minimum 5 données
library(BenfordTests)
givenames <- function(df){
  name <- distinct(df,df$location)
  w <- 0
  x <- as.vector(name$`df$location`)
  t=0
  for(i in 1:length(x)){
    d <- df[df$location==x[i],]
    v <- vec(d)
    dx <- firstDigit(v)
    if(length(dx)<=10){
      t=t
    }
    else{
      t=t+1
      w[t]=x[i]
    }
  }
  return(w)
}
#-------------------------------------------------------------------------------
#--------------------------- Test du T_2 first digit ---------------------------
# fonction T_2
library(BenfordSmoothTest)
T2 <- function(df){
  x <- vec(df)
  benf <- BenfordSmooth.test(x)
  return(benf)
}
```

```r
#---------------------------- Test appliqué aux données --------------------
twodigitst <- function(df,k){
  x<-vec(df)
  w <-signifd(x = x, digits = 2)
  if(length(w)<=50){
    y<-Test.lisse(k,w,texte=F,d=1)
  }
  else{
    y<-Test.lisse(k,w,texte=F,d=2)
  }

  return(y)
}
twodigitst2 <- function(df,k){
  x<-vec(df)
  w <-signifd(x = x, digits = 2)
  y<-Test.lisse(k,w,texte=F,d=2)

  return(y)
}
#--------------------------------------------------------------------------------
#------------------------------ Fonction p_adjust ---------------------------------
p_adjusted <- function(w,k){

  n <- givenames(w)
  pval = 0
  for( i in 1:length(n)){

    bq <- w[w$location==n[i],]
    Tk = twodigitst(bq,k)
    pval[i]= Tk[2]
  }
  padj <- p.adjust(pval,method="holm")
  return(padj)
}
#--------------------------------------------------------------------------------
#-----visualisation graphique de la distribution comparé à benford-------
name <- distinct(b,b$location)
x <- as.vector(name$`b$location`)
for(i in 1:length(x)){
  bq <- b[b$location==x[i],]
  benf(bq)
}
#---------------------------------------------------------------------------
#--------------------------- Creation tableau first digit ---------------
w = givenames(b)
pval = 0
test = 0
for(i in 1:length(w)){
  bq <- b[b$location==w[i],]
  T_k <- T2(bq)
  pval[i] <- as.numeric(T_k[3,2])
  test[i] <- as.numeric(T_k[2,2])
```

```
}
pvaladj = p.adjust(pval,method="BY")
tab <- as.data.frame(matrix(c(w,round(test,3),round(pvaladj,3)),ncol=3))
colnames(tab) = c("Pays","T_2","P_value")
benfo <- tab[as.numeric(tab$P_value)<=0.05,] # pays qui ne suivent pas une benford.
rownames(benfo) <- NULL
benfn <- tab[as.numeric(tab$P_value)>0.05,] # pays qui ne suivent pas une benford.
rownames(benfn) <- NULL
write.csv(benfn,"h0.csv")
write.csv(benfo,"h1.csv")
write.csv(tab,"tabadj.csv")
#-------------------------------------------------
#---------------- tableau two digits -----------------
w = givenames(b)
pval = 0
test = 0
for(i in 1:length(w)){
  bq <- b[b$location==w[i],]
  T_k <- twodigitst(bq,2)
  pval[i] <- T_k[2]
  test[i] <- T_k[1]
}
pvaladj = p.adjust(pval,method="BY")
tab <- as.data.frame(matrix(c(w,round(test,3),round(pvaladj,3)),ncol=3))
colnames(tab) = c("Pays","T_2","P_value")
benfo <- tab[as.numeric(tab$P_value)<=0.05,] # pays qui ne suivent pas une benford.
rownames(benfo) <- NULL
benfn <- tab[as.numeric(tab$P_value)>0.05,] # pays qui ne suivent pas une benford.
rownames(benfn) <- NULL
write.csv(benfn,"twdh0.csv")
write.csv(benfo,"twdh1.csv")
write.csv(tab,"twd.csv")

#---------------------- graphes -------------------------------
#----------------------- benford 1st digit ---------------------------
#modélisation benford
benlaw <- function(d) log10(1 + 1 / d)
digits <- 1:9
barres <-barplot(benlaw(digits), names.arg = digits, xlab = "First Digit",ylab="frequency", ylim = c(0,
text(x=barres[,1], y=benlaw(digits)*1.105,labels = round(benlaw(digits),3))

#----------------------- benford 2 digits --------------------------------
benlaw2 <- function(d) log10(1 + 1 / d)
digits2 <- 10:99
baseBarplot <- barplot(benlaw2(digits2), names.arg = digits2, xlab = "First-Two Digits",ylab="frequency
lines(x = baseBarplot[,1], y = benlaw2(digits2), col = "red", lwd = 1, pch = 23, cex = 1.5, bg = "red")

#----------------------- tab 1st digit -----------------------------------
```

```
#----------------------- PARTIE TEST BLOC DEUX PREMIERS CHIFFRES -------------------------------


#**A) Test deux chiffres significatifs.**
```

```
#Pour 1\<= k \<= 4.

#**I/ Fonctions h(k) orthonormales :**

Mu = rep(0,8)

for (k in c(1:8)) { # Calcul de Mu(k) :

  S = 0
  for (y in c(10:99)) {

    S = S + (y^k)*log(1 + 1/y)/log(10)

  }

  Mu[k] = S
}


for (k in c(1:4)) { # Calcul de M(k)^(-1) :

  A = matrix(1, ncol=k, nrow=k)

  for (i in c(0:(k-1)) ) {
    for (j in c(0:(k-1)) ) {

      if ( (i != 0) || (j != 0) ) {
        A[i+1, j+1] = Mu[i+j]
      }

    }
  }

  if (k == 1) { M1_ = solve(A) }
  if (k == 2) { M2_ = solve(A) }
  if (k == 3) { M3_ = solve(A) }
  if (k == 4) { M4_ = solve(A) }

}

M_ = list(M1_, M2_, M3_, M4_)

c_ = rep(0,4)

for (k in c(1:4)) { # Calcul de h(k) :

  Mu2 = matrix(0, nrow= k, ncol= 1)

  for (i in c(1:k)) {
    Mu2[i, 1] = Mu[ k-1+i ]
  }

  c_[k] = 1/(sqrt( Mu[2*k] - ( t(Mu2) %*% M_[[k]]  %*% Mu2 ) )) # 1/sqrt(c(k))
```

```
  A = -1*c_[k] * M_[[k]] %*% Mu2

  L = rep(0, k+1)
  for (i in c(1:k)) {
    L[i] = A[i]
  }
  L[k+1] = c_[k]

  print(L) # Les coéfficients de h(k)

}


###**II/ Statistique T(k) :**

# Fonctions h(k) :

h = function (k, x) {

  if (k == 1) {
    return( -1.54751771 + 0.04010177 *x )
  }

  if (k == 2) {
    return( 2.795867953 - 0.167441591 *x + 0.001736457 *x^2 )
  }

  if (k == 3) {
    return( -5.187810e+00  + 4.869054e-01 *x - 1.155519e-02 *x^2  + 7.630402e-05 *x^3 )
  }

  if (k == 4) {
    return( 9.725235e+00 - 1.237520e+00 *x + 4.769440e-02 *x^2 - 6.950207e-04 *x^3  + 3.371880e-06 *x^4
  }

}

# Statistique T(k) :      ( 1<= k <= 4)

T = function (k, data) {

  n = length(data)

  T_ = 0 # la statistique T(k)
  for (l in c(1:k)) {

    U = 0 # U(l)
    for (i in c(1:n)) {

      U = U + h(l, data[i])

    }
    U = U*(1/sqrt(n))
```

```
    T_ = T_ + U^2

  }

  return(T_)
}


#**III/ Calcul de la p.value :**


pvalue = function(t, n, k, d, w) { # d et w pour Monte-Carlo

   if ( ( (n <= 100) || (d == 1) ) && (d != 2) ) { ### 1) approximation Monte-Carlo ###

     M = 0 # Moyenne empirique = p.value
     L = rep(0, w) # Liste des Y(l)

     proba = rep(0, 90) # vecteur des probas sous H0 pour sample
     for (i in c(1:90)) {

       proba[i] = log(1 + 1/(i+9) )/log(10)

     }

     for (l in c(1:w)) {

       data = rep(0, n)
       for (i in c(1:n)) { # Génération des Xi sous H0 de taille n

         data[i] = sample(x= c(10:99), size=1, prob= proba )

       }

       a = T(k, data)

       if (a >= t) {
         M = M+1
         L[l] = 1
       }

     }
     M = M/w

     S = 0 # Variance empirique
     for (l in c(1:w)) {

       S = S + (L[l] - M)^2

     }
     S = S/w

     e = 1.96*sqrt(S/w) # erreur d'approximation (à 95%)
```

```
    return( c(M, e) )


  } else { ### 2) approximation Khi2 ###

    M = 1 - pchisq(t, df=k)

    return( c(M, 0) )

  }

}


#**IV / Fonction globale :**

# 1 <= k <= 4.
# data : sous forme de vecteur c(...).
# d = 1 : force le calcul de la p.value par Monte-Carlo.
# d = 2 : force le calcul de la p.value par approximation Khi2
# Si n <= 100 alors d=1 par défault, et si n > 100 alors d=2 par défault.
# w = taille de l'échantillon pour Monte-Carlo.
# texte = FALSE : n'affiche pas de texte, et return c(Tk, p.value, erreur M-C).

Test.lisse = function (k, data, d=0, w=10000, texte=TRUE) {

  t = T(k, data) # Valeur de la statistique T(k)

  n = length(data)
  A = pvalue(t, n, k, d, w) # On récupère la p.value associé

  if ( texte == TRUE ) {

    cat("\n")
    cat("k =", k, "\n")
    cat("Statistique Tk =", t, "\n \n")

    if ( ( (n <= 100) || (d == 1) ) && (d != 2) ) {
      cat("Approximation de la p.value par Monte-Carlo :", "\n \n")
      cat("--> p.value =", A[1], "\n")
      cat("Erreur d'approximation en valeur absolue (à 95%) <=", A[2])

    } else {
      cat("Approximation asymptotique de la p.value par une Khi2(k) :", "\n \n")
      cat("--> p.value =", A[1])
    }
    cat("\n")

  } else {

    return( c(t, A[1], A[2]) )
```

```
  }

}

##### TEST #####

data = c(11, 12, 50, 51, 53, 56, 96, 99, 12, 24, 26, 27)
#data = seq(1:101)

#T(4, data)

Test.lisse(1, data)


proba = rep(0, 90) # vecteur des probas sous H0 pour sample
for (i in c(1:90)) {

  proba[i] = log(1 + 1/(i+9) )/log(10)

}
data = sample(x= c(10:99), size=20, prob= proba , replace=TRUE)


Test.lisse(1, data)
Test.lisse(2, data)
Test.lisse(3, data)
Test.lisse(4, data)

## **B) Analyse du Test (ecdf + puissance).**

## **I/ Justificatif du choix n \> 100 pour la Khi2(k) :**

q = 1
B = matrix(0, nrow=8, ncol=w)

for (k in c(1:4)) {

  w = 10000
  A = matrix(0, nrow = 2, ncol = w)

  z = 0
  for (n in c(101:102)) {

    L = rep(0, w) # Liste des T(k)

    proba = rep(0, 90) # vecteur des probas sous H0 pour sample
    for (i in c(1:90)) {
    proba[i] = log(1 + 1/(i+9) )/log(10)
    }

    for (l in c(1:w)) {

      data = rep(0, n)
```

```
      for (i in c(1:n)) { # Génération des Xi sous H0 de taille n

        data[i] = sample(x= c(10:99), size=1, prob= proba )

      }

      L[l] = T(k, data)
    }

    z = z+1
    A[z,] = L

    print(q) # q = 8 au total.
    q = q+1
  }

  B[c( (1+ (k-1)*2):(k*2) ),] = A
}

op = par( mfrow=c(1,2) )

curve(pchisq(x,df=1), add=T, xlim=c(0,6))
lines(ecdf(B[1,]), col="blue")
legend(1, 0.3, legend=c("T1 n=101 sous H0", "Khi2(1)"), col=c("blue", "black"), lty=1, cex=0.7)

curve(pchisq(x,df=1), add=T, xlim=c(0,6))
lines(ecdf(B[2,]), col="red")
legend(1, 0.3, legend=c("T1 n=102 sous H0", "Khi2(1)"), col=c("red", "black"), lty=1, cex=0.7)

par(op)
mtext("ECDF de T1, et CDF d'un Khi2(1)", side=1, line=-18, font=3, cex=1.5, adj=0)

op = par(mfrow=c(1,2))

curve(pchisq(x,df=2), add=T, xlim=c(0,8))
lines(ecdf(B[3,]), col="blue")
legend(1.5, 0.3, legend=c("T2 n=101 sous H0", "Khi2(2)"), col=c("blue", "black"), lty=1, cex=0.7)

curve(pchisq(x,df=2), add=T, xlim=c(0,8))
lines(ecdf(B[4,]), col="red")
legend(1.5, 0.3, legend=c("T2 n=102 sous H0", "Khi2(2)"), col=c("red", "black"), lty=1, cex=0.7)

par(op)
mtext("ECDF de T2, et CDF d'un Khi2(2)", side=1, line=-18, font=3, cex=1.5, adj=0)

op = par(mfrow=c(1,2))

curve(pchisq(x,df=3), add=T, xlim=c(0,10))
lines(ecdf(B[5,]), col="blue")
legend(1.8, 0.3, legend=c("T3 n=101 sous H0", "Khi2(3)"), col=c("blue", "black"), lty=1, cex=0.7)

curve(pchisq(x,df=3), add=T, xlim=c(0,10))
lines(ecdf(B[6,]), col="red")
```

```
legend(1.8, 0.3, legend=c("T3 n=102 sous H0", "Khi2(3)"), col=c("red", "black"), lty=1, cex=0.7)

par(op)
mtext("ECDF de T3, et CDF d'un Khi2(3)", side=1, line=-18, font=3, cex=1.5, adj=0)

op = par(mfrow=c(1,2))

curve(pchisq(x,df=4), add=T, xlim=c(0,14))
lines(ecdf(B[7,]), col="blue")
legend(2.6, 0.3, legend=c("T4 n=101 sous H0", "Khi2(4)"), col=c("blue", "black"), lty=1, cex=0.7)

curve(pchisq(x,df=4), add=T, xlim=c(0,14))
lines(ecdf(B[8,]), col="red")
legend(2.6, 0.3, legend=c("T4 n=102 sous H0", "Khi2(4)"), col=c("red", "black"), lty=1, cex=0.7)

par(op)
mtext("ECDF de T4, et CDF d'un Khi2(4)", side=1, line=-18, font=3, cex=1.5, adj=0)

q = 1
B = matrix(0, nrow=4, ncol=w)

for (k in c(1:4)) {

  w = 10000
  A = matrix(0, nrow = 1, ncol = w)

  z = 0
  for (n in c(50)) {

    L = rep(0, w) # Liste des T(k)

    proba = rep(0, 90) # vecteur des probas sous H0 pour sample
    for (i in c(1:90)) {
    proba[i] = log(1 + 1/(i+9) )/log(10)
    }

    for (l in c(1:w)) {

      data = rep(0, n)
      for (i in c(1:n)) { # Génération des Xi sous H0 de taille n

        data[i] = sample(x= c(10:99), size=1, prob= proba )

      }

      L[l] = T(k, data)
    }

    z = z+1
    A[z,] = L

    print(q) # q = 4 au total.
    q = q+1
```

```
  }

  B[c( (1+ (k-1)*1):(k*1) ),] = A
}

# n = 50

op = par(mfrow=c(2,2))

curve(pchisq(x,df=1), add=T, xlim=c(0,14))
lines(ecdf(B[1,]), col="blue")
legend(3, 0.4, legend=c("T1 n=50 sous H0", "Khi2(1)"), col=c("blue", "black"), lty=1, cex=0.7)

curve(pchisq(x,df=2), add=T, xlim=c(0,14))
lines(ecdf(B[2,]), col="blue")
legend(3, 0.4, legend=c("T2 n=50 sous H0", "Khi2(2)"), col=c("blue", "black"), lty=1, cex=0.7)

curve(pchisq(x,df=3), add=T, xlim=c(0,14))
lines(ecdf(B[3,]), col="blue")
legend(3, 0.4, legend=c("T3 n=50 sous H0", "Khi2(3)"), col=c("blue", "black"), lty=1, cex=0.7)

curve(pchisq(x,df=4), add=T, xlim=c(0,14))
lines(ecdf(B[4,]), col="blue")
legend(3, 0.4, legend=c("T4 n=50 sous H0", "Khi2(4)"), col=c("blue", "black"), lty=1, cex=0.7)

par(op)
mtext("ECDF de Tk, et CDF d'un Khi2(k)", side=1, line=-8, font=3, cex=1.5, adj=0)

# Calcul des quantilles des statistiques sous H0.

w = 100000 # Taille échantillon M-C pour les quantilles de chaque statistique.

proba = rep(0, 90) # vecteur des probas sous H0 pour sample
for (i in c(1:90)) {
  proba[i] = log(1 + 1/(i+9) )/log(10)
}

quanti = matrix(0, nrow=6, ncol=3)

v = 1
for (k in c(1:6)) {

  L = matrix(0, nrow=3, ncol=w)

  for (n in c(50, 100, 150)) {

      for (l in c(1:w)) {

        data = rep(0, n)
        for (i in c(1:n)) { # Génération des Xi sous H0 de taille n
          data[i] = sample(x= c(10:99), size=1, prob= proba )
        }
```

```
        if (k <= 4) {
          L[n/50, l] = Test.lisse(k, data, d=2, texte=FALSE)[1] # Tk
        }
        if (k == 5) {
          L[n/50, l] = chisq.benftest(data, digits=2)$statistic # Khi2
        }
        if (k == 6) {
          L[n/50, l] = usq.benftest(data, digits=2, pvalsims=1)$statistic # U^2
        }

      }

    cat("\n", v/0.18, "%")
    v = v+1
  }

  for (i in c(1:3)) {
    quanti[k,i] = quantile(L[i,], 95/100)
  }

}

# Rodriguez
library(BenfordTests)

w = 10000 # Taille échantillons M-C pour le calcul de chaque "puissance".

Q = seq(from=-5, to=5, length=100)
Q[length(Q)+1] = 0
Q[length(Q)+1] = -1

D50 = matrix(0, nrow= 612, ncol= 3)
D100 = matrix(0, nrow= 612, ncol= 3)
D150 = matrix(0, nrow= 612, ncol= 3)

v = 1
for (n in c(50,100,150)) {

  C = matrix(0, nrow= 612, ncol= 3)

  z = 1
  for (gamma in Q) { # On parcout plusieurs gamma pour les différents tests.

    M = rep(0, 6) # Moyenne empirique = "puissance"
    L = matrix(0, nrow=6, ncol=w) # Liste des Y(l)

    proba = rep(0, 90) # vecteur des probas H1 : Rodriguez(gamma) pour sample
    for (i in c(1:90)) {

      if (gamma == -1) {
        proba[i] = log(1 + 1/(i+9))/log(10)
      }
```

```
    if (gamma == 0) {
      proba[i] = (1/810)*( 9+19*log(10)+9*(i+9)*log(i+9) - 9*(i+10)*log(i+10) )
    }

    if ( (gamma != 0) && (gamma != -1) ) {
      proba[i] = (gamma+1)/(90*gamma) - ( (i+10)^(gamma+1) - (i+9)^(gamma+1) )/( gamma*( 100^(gamma+1)
    }

  }

  for (l in c(1:w)) {

    data = rep(0, n)
    for (i in c(1:n)) { # Génération des Xi sous Rodriguez(gamma) de taille n

      data[i] = sample(x= c(10:99), size=1, prob= proba )

    }

    p = rep(0, 6) # Vecteur des statistiques
    for (k in c(1:6)) {

      if (k == 5) {
        p[k] = chisq.benftest(data, digits=2)$statistic # Khi2
      }

      if (k == 6) {
        p[k] = usq.benftest(data, digits=2, pvalsims=1)$statistic # U^2
      }

      if (k <= 4) {
        p[k] = Test.lisse(k, data, d=2, texte=FALSE)[1] # Tk
      }
    }

    for (k in c(1:6)) {

      if (p[k] > quanti[k, n/50] ) {
        M[k] = M[k] + 1
        L[k, l] = 1
      }

    }

    }
  M = M/w

  S = rep(0, 6) # Variance empirique
  for (l in c(1:w)) {
    for (k in c(1:6)) {
      S[k] = S[k] + (L[k,l] - M[k])^2
    }
  }
```

```
    S = S/w

    e = rep(0, 6)
    for (k in c(1:6)) {
      e[k] = 1.96*sqrt(S[k]/w) # erreur d'approximation (à 95%)
    }


    for (k in c(1:6)) {
      C[(z + (k-1)*102), ] = c(M[k], e[k], gamma)
    }
    z = z+1

    cat("\n", v/3.06, "%")
    v = v+1

  } # Fin de la boucle sur gamma.


  if (n == 50) {
    D50 = C
  }
  if (n == 100) {
    D100 = C
  }
  if (n == 150) {
    D150 = C
  }

}

# Erreur maximum Monte-Carlo tout test confondu.
print( max(D50[, 2]) ) # n=50
print( max(D100[, 2]) ) # n=100
print( max(D150[, 2]) ) # n=150

op = par( mfrow=c(2,2) )

### n =50 ###

plot(D50[103:202, 3], D50[103:202, 1], col="red", type="line", xlab="n=50", ylab=" ") # T2
abline(a=5/100,b=0, col="red")
lines(D50[1:100, 3], D50[1:100, 1], col="blue") # T1
lines(D50[103:202, 3], D50[103:202, 1], col="red") # T2
lines(D50[205:304, 3], D50[205:304, 1], col="green") # T3
lines(D50[307:406, 3], D50[307:406, 1], col="purple") # T4
lines(D50[409:508, 3], D50[409:508, 1], col="darkred") # Khi2
lines(D50[511:610, 3], D50[511:610, 1], col="black") # U^2

legend(x="bottomright", legend=c("T1", "T2", "T3", "T4", "Khi2", "U^2"), col=c("blue", "red", "green",

### n = 100 ###
```

```r
plot(D100[103:202, 3], D100[103:202, 1], col="red", type="line", xlab="n=100", ylab=" ") # T2
abline(a=5/100,b=0, col="red")
lines(D100[1:100, 3], D100[1:100, 1], col="blue") # T1
lines(D100[103:202, 3], D100[103:202, 1], col="red") # T2
lines(D100[205:304, 3], D100[205:304, 1], col="green") # T3
lines(D100[307:406, 3], D100[307:406, 1], col="purple") # T4
lines(D100[409:508, 3], D100[409:508, 1], col="darkred") # Khi2
lines(D100[511:610, 3], D100[511:610, 1], col="black") # U^2

legend(x="bottomright", legend=c("T1", "T2", "T3", "T4", "Khi2", "U^2"), col=c("blue", "red", "green",

### n = 150 ###

plot(D150[103:202, 3], D150[103:202, 1], col="red", type="line", xlab="n=150", ylab=" ") # T2
abline(a=5/100,b=0, col="red")
lines(D150[1:100, 3], D150[1:100, 1], col="blue") # T1
lines(D150[103:202, 3], D150[103:202, 1], col="red") # T2
lines(D150[205:304, 3], D150[205:304, 1], col="green") # T3
lines(D150[307:406, 3], D150[307:406, 1], col="purple") # T4
lines(D150[409:508, 3], D150[409:508, 1], col="darkred") # Khi2
lines(D150[511:610, 3], D150[511:610, 1], col="black") # U^2

legend(x="bottomright", legend=c("T1", "T2", "T3", "T4", "Khi2", "U^2"), col=c("blue", "red", "green",

par(op)
mtext("  Alternative : Rodriguez", side=1, line=-3, font=3, cex=1.5, adj=1)

# Pietronero
library(BenfordTests)

w = 10000 # Taille échantillons M-C pour le calcul de chaque "puissance".

Q = seq(from=-5, to=5, length=100)
Q[length(Q)+1] = 0
Q[length(Q)+1] = 0

H50 = matrix(0, nrow= 612, ncol= 3)
H100 = matrix(0, nrow= 612, ncol= 3)
H150 = matrix(0, nrow= 612, ncol= 3)

v = 1
for (n in c(50,100,150)) {

  C = matrix(0, nrow= 612, ncol= 3)

  z = 1
  for (gamma in Q) { # On parcout plusieurs gamma pour les différents tests.

    M = rep(0, 6) # Moyenne empirique = "puissance"
    L = matrix(0, nrow=6, ncol=w) # Liste des Y(l)

    proba = rep(0, 90) # vecteur des probas H1 : Pietronero(gamma) pour sample
    for (i in c(1:90)) {
```

```
  if (gamma == 0) {
    proba[i] = log(1 + 1/(i+9))/log(10)
  }

  if (gamma != 0) {
    proba[i] = ( (i+9)^(-gamma) - (i+10)^(-gamma) )/( 10^(-gamma) - 100^(-gamma) )
  }

}

for (l in c(1:w)) {

  data = rep(0, n)
  for (i in c(1:n)) { # Génération des Xi sous Pietronero(gamma) de taille n

    data[i] = sample(x= c(10:99), size=1, prob= proba )

  }

  p = rep(0, 6) # Vecteur des statistiques
  for (k in c(1:6)) {

    if (k == 5) {
      p[k] = chisq.benftest(data, digits=2)$statistic # Khi2
    }

    if (k == 6) {
      p[k] = usq.benftest(data, digits=2, pvalsims=1)$statistic # U^2
    }

    if (k <= 4) {
      p[k] = Test.lisse(k, data, d=2, texte=FALSE)[1] # Tk
    }
  }

  for (k in c(1:6)) {

    if (p[k] > quanti[k, n/50] ) {
      M[k] = M[k] + 1
      L[k, l] = 1
    }

  }

  }
M = M/w

S = rep(0, 6) # Variance empirique
for (l in c(1:w)) {
  for (k in c(1:6)) {
    S[k] = S[k] + (L[k,l] - M[k])^2
  }
}
```

```
    S = S/w

    e = rep(0, 6)
    for (k in c(1:6)) {
      e[k] = 1.96*sqrt(S[k]/w) # erreur d'approximation (à 95%)
    }


    for (k in c(1:6)) {
      C[(z + (k-1)*102), ] = c(M[k], e[k], gamma)
    }
    z = z+1

    cat("\n", v/3.06, "%")
    v = v+1

  } # Fin de la boucle sur gamma.


  if (n == 50) {
    H50 = C
  }
  if (n == 100) {
    H100 = C
  }
  if (n == 150) {
    H150 = C
  }

}

# Erreur maximum Monte-Carlo tout test confondu.
print( max(H50[, 2]) ) # n=50
print( max(H100[, 2]) ) # n=100
print( max(H150[, 2]) ) # n=150

op = par( mfrow=c(2,2) )

### n =50 ###

plot(H50[409:508, 3], H50[409:508, 1], col="darkred", type="line", xlab="n=50", ylab=" ") # Khi2
abline(a=5/100,b=0, col="red")
abline(a=0,b=0, col="black")
lines(H50[1:100, 3], H50[1:100, 1], col="blue") # T1
lines(H50[103:202, 3], H50[103:202, 1], col="red") # T2
lines(H50[205:304, 3], H50[205:304, 1], col="green") # T3
lines(H50[307:406, 3], H50[307:406, 1], col="purple") # T4
lines(H50[409:508, 3], H50[409:508, 1], col="darkred") # Khi2
lines(H50[511:610, 3], H50[511:610, 1], col="black") # U^2

legend(x="bottomright", legend=c("T1", "T2", "T3", "T4", "Khi2", "U^2"), col=c("blue", "red", "green",

### n = 100 ###
```

```r
plot(H100[409:508, 3], H100[409:508, 1], col="darkred", type="line", xlab="n=100", ylab=" ") # Khi2
abline(a=5/100,b=0, col="red")
abline(a=0,b=0, col="black")
lines(H100[1:100, 3], H100[1:100, 1], col="blue") # T1
lines(H100[103:202, 3], H100[103:202, 1], col="red") # T2
lines(H100[205:304, 3], H100[205:304, 1], col="green") # T3
lines(H100[307:406, 3], H100[307:406, 1], col="purple") # T4
lines(H100[409:508, 3], H100[409:508, 1], col="darkred") # Khi2
lines(H100[511:610, 3], H100[511:610, 1], col="black") # U^2

legend(x="bottomright", legend=c("T1", "T2", "T3", "T4", "Khi2", "U^2"), col=c("blue", "red", "green",

### n = 150 ###

plot(H150[409:508, 3], H150[409:508, 1], col="darkred", type="line", xlab="n=150", ylab=" ") # Khi2
abline(a=5/100,b=0, col="red")
abline(a=0,b=0, col="black")
lines(H150[1:100, 3], H150[1:100, 1], col="blue") # T1
lines(H150[103:202, 3], H150[103:202, 1], col="red") # T2
lines(H150[205:304, 3], H150[205:304, 1], col="green") # T3
lines(H150[307:406, 3], H150[307:406, 1], col="purple") # T4
lines(H150[409:508, 3], H150[409:508, 1], col="darkred") # Khi2
lines(H150[511:610, 3], H150[511:610, 1], col="black") # U^2

legend(x="bottomright", legend=c("T1", "T2", "T3", "T4", "Khi2", "U^2"), col=c("blue", "red", "green",

par(op)
mtext("  Alternative : Pietronero", side=1, line=-3, font=3, cex=1.5, adj=1)

library(BenfordTests)

test.maison = function (data) { # Pour n=100 !

  t1 = Test.lisse(2, data, d=2, texte=FALSE)[1] # T2
  t2 = usq.benftest(data, digits=2, pvalsims=1)$statistic # U^2

  a = 0
  if (t1 > quanti_bis[1]) {
    a = 1
  }
  if (t2 > quanti_bis[2]) {
    a = 1
  }

  return(a)

}

# Calcul des quantilles des statistiques sous H0 (3.2%).

n = 100

w = 50000 # Taille échantillon M-C pour les quantilles de chaque statistique.
```

```
proba = rep(0, 90) # vecteur des probas sous H0 pour sample
for (i in c(1:90)) {
  proba[i] = log(1 + 1/(i+9) )/log(10)
}

quanti_bis = rep(0,2)

v = 1
for (k in c(1:2)) {

  L = rep(0,w)

  for (l in c(1:w)) {

    data = rep(0, n)
    for (i in c(1:n)) { # Génération des Xi sous H0 de taille n
      data[i] = sample(x= c(10:99), size=1, prob= proba )
    }

    if (k == 1) {
      L[l] = Test.lisse(2, data, d=2, texte=FALSE)[1] # Tk
    }

    if (k == 2) {
      L[l] = usq.benftest(data, digits=2, pvalsims=1)$statistic # U^2
    }

  }

  quanti_bis[k] = quantile(L, 96.8/100)

}

library(BenfordTests)

n = 100

### Pietronero ###

w = 10000 # Taille échantillons M-C pour le calcul de chaque "puissance".
Q = seq(from=-5, to=5, length=100)
Q[length(Q)+1] = 0
Q[length(Q)+1] = 0

EP = matrix(0, nrow= 102, ncol= 3)

  z = 1
  for (gamma in Q) { # On parcout plusieurs gamma pour les différents tests.

    M = 0 # Moyenne empirique = "puissance"
    L = rep(0,w) # Liste des Y(l)

    proba = rep(0, 90) # vecteur des probas H1 : Pietronero(gamma) pour sample
```

```
    for (i in c(1:90)) {

      if (gamma == 0) {
        proba[i] = log(1 + 1/(i+9))/log(10)
      }

      if (gamma != 0) {
        proba[i] = ( (i+9)^(-gamma) - (i+10)^(-gamma) )/( 10^(-gamma) - 100^(-gamma) )
      }

    }

    for (l in c(1:w)) {

      data = rep(0, n)
      for (i in c(1:n)) { # Génération des Xi sous Pietronero(gamma) de taille n

        data[i] = sample(x= c(10:99), size=1, prob= proba )

      }

      p = test.maison(data)

      if (p == 1) {
        M = M + 1
        L[l] = 1
      }

      }
    M = M/w

    S = 0 # Variance empirique
    for (l in c(1:w)) {
      S = S + (L[l] - M)^2
    }
    S = S/w

    e = 1.96*sqrt(S/w) # erreur d'approximation (à 95%)

    EP[z, ] = c(M, e, gamma)
    z = z+1

    cat("\n", (z-1)/1.02, "%")

  } # Fin de la boucle sur gamma.

library(BenfordTests)

n = 100

### Rodriguez ###

w = 10000 # Taille échantillons M-C pour le calcul de chaque "puissance".
```

```
Q = seq(from=-5, to=5, length=100)
Q[length(Q)+1] = 0
Q[length(Q)+1] = -1

ER = matrix(0, nrow= 102, ncol= 3)

  z = 1
  for (gamma in Q) { # On parcout plusieurs gamma pour les différents tests.

    M = 0 # Moyenne empirique = "puissance"
    L = rep(0,w) # Liste des Y(l)

    proba = rep(0, 90) # vecteur des probas H1 : Rodriguez(gamma) pour sample
    for (i in c(1:90)) {

      if (gamma == -1) {
        proba[i] = log(1 + 1/(i+9))/log(10)
      }

      if (gamma == 0) {
        proba[i] = (1/810)*( 9+19*log(10)+9*(i+9)*log(i+9) - 9*(i+10)*log(i+10) )
      }

      if ( (gamma != 0) && (gamma != -1) ) {
        proba[i] = (gamma+1)/(90*gamma) - ( (i+10)^(gamma+1) - (i+9)^(gamma+1) )/( gamma*( 100^(gamma+1
      }

    }

    for (l in c(1:w)) {

      data = rep(0, n)
      for (i in c(1:n)) { # Génération des Xi sous Rodriguez(gamma) de taille n

        data[i] = sample(x= c(10:99), size=1, prob= proba )

      }

      p = test.maison(data)

      if (p == 1) {
        M = M + 1
        L[l] = 1
      }

      }
    M = M/w

    S = 0 # Variance empirique
    for (l in c(1:w)) {
      S = S + (L[l] - M)^2
    }
    S = S/w
```

```
    e = 1.96*sqrt(S/w) # erreur d'approximation (à 95%)

    ER[z, ] = c(M, e, gamma)
    z = z+1

    cat("\n", (z-1)/1.02, "%")

  } # Fin de la boucle sur gamma.

### Graphiques ###
par( mfrow=c(1,2) )


plot(D100[103:202, 3], D100[103:202, 1], col="red", type="line", xlab="n=100, Rodriguez", ylab=" ") # T
abline(a=5/100,b=0, col="red")
abline(a=2.5/100,b=0, col="black")
lines(D100[103:202, 3], D100[103:202, 1], col="red") # T2
lines(ER[1:100, 3], ER[1:100, 1], col="blue") # maison

legend(x="bottomright", legend=c("T2", "Maison"), col=c("red", "blue"), lty=1, cex=0.6)


plot(H100[103:202, 3], H100[103:202, 1], col="red", type="line", xlab="n=100, Pietronero", ylab=" ") #
abline(a=5/100,b=0, col="red")
abline(a=2.5/100,b=0, col="black")
lines(H100[103:202, 3], H100[103:202, 1], col="red") # T2
lines(EP[1:100, 3], EP[1:100, 1], col="blue") # maison

legend(x="bottomright", legend=c("T2", "Maison"), col=c("red", "blue"), lty=1, cex=0.6)

library(BenfordTests)
n = 100

w = 10000

proba = rep(0, 90) # vecteur des probas sous H0 pour sample
for (i in c(1:90)) {
  proba[i] = log(1 + 1/(i+9) )/log(10)
}


v = 1

M2 = 0 # T2 et U^2
M3 = 0 # T2 et Khi2
M = 0 # U^2 et Khi2

for (l in c(1:w)) {

  data = rep(0, n)
  for (i in c(1:n)) { # Génération des Xi sous H0 de taille n
    data[i] = sample(x= c(10:99), size=1, prob= proba )
  }
```

```
  p1 = Test.lisse(2, data, d=2, texte=FALSE)[2] # T2

  p2 = usq.benftest(data, digits=2)$p.value # U^2

  p3 = chisq.benftest(data, digits=2, pvalmethod = "simulate")$p.value # Khi2


  if ( (p1 <= 2.5/100) && (p2 <= 2.5/100) ) {
    M2 = M2+1
  }
  if ( (p1 <= 2.5/100) && (p3 <= 2.5/100) )  {
    M3 = M3+1
  }
  if ( (p3 <= 2.5/100) && (p2 <= 2.5/100) )  {
    M = M+1
  }

}
M2 = M2/w
M3 = M3/w
M = M/w

# Calcul des quantilles des statistiques sous H0 (2.6%).

n = 100

w = 50000 # Taille échantillon M-C pour les quantilles de chaque statistique.

proba = rep(0, 90) # vecteur des probas sous H0 pour sample
for (i in c(1:90)) {
  proba[i] = log(1 + 1/(i+9) )/log(10)
}

quanti_bis2 = rep(0,2)

L = matrix(0, nrow=2, ncol=w)

for (l in c(1:w)) {

  data = rep(0, n)
  for (i in c(1:n)) { # Génération des Xi sous H0 de taille n
    data[i] = sample(x= c(10:99), size=1, prob= proba )
  }

  L[1,l] = Test.lisse(2, data, d=2, texte=FALSE)[1] # T2

  L[2,l] = chisq.benftest(data, digits=2)$statistic # Khi2

}

quanti_bis2[1] = quantile(L[1,], 97.4/100) # test à 2.6%
quanti_bis2[2] = quantile(L[2,], 97.4/100) # test à 2.6%
```

```r
test.maison2 = function (data) { # Pour n=100 !

  t1 = Test.lisse(2, data, d=2, texte=FALSE)[1] # T2
  t2 = chisq.benftest(data, digits=2)$statistic # Khi2

  a = 0
  if (t1 > quanti_bis2[1]) {
    a = 1
  }
  if (t2 > quanti_bis2[2]) {
    a = 1
  }

  return(a)

}

library(BenfordTests)

n = 100

### Rodriguez ###

w = 10000 # Taille échantillons M-C pour le calcul de chaque "puissance".
Q = seq(from=-5, to=5, length=100)
Q[length(Q)+1] = 0
Q[length(Q)+1] = -1

ER2 = matrix(0, nrow= 102, ncol= 3)

  z = 1
  for (gamma in Q) { # On parcout plusieurs gamma pour les différents tests.

    M = 0 # Moyenne empirique = "puissance"
    L = rep(0,w) # Liste des Y(l)

    proba = rep(0, 90) # vecteur des probas H1 : Rodriguez(gamma) pour sample
    for (i in c(1:90)) {

      if (gamma == -1) {
        proba[i] = log(1 + 1/(i+9))/log(10)
      }

      if (gamma == 0) {
        proba[i] = (1/810)*( 9+19*log(10)+9*(i+9)*log(i+9) - 9*(i+10)*log(i+10) )
      }

      if ( (gamma != 0) && (gamma != -1) ) {
        proba[i] = (gamma+1)/(90*gamma) - ( (i+10)^(gamma+1) - (i+9)^(gamma+1) )/( gamma*( 100^(gamma+1)
      }

    }
```

```
    for (l in c(1:w)) {

      data = rep(0, n)
      for (i in c(1:n)) { # Génération des Xi sous Rodriguez(gamma) de taille n

        data[i] = sample(x= c(10:99), size=1, prob= proba )

      }

      p = test.maison2(data)

      if (p == 1) {
        M = M + 1
        L[l] = 1
      }

      }
    M = M/w

    S = 0 # Variance empirique
    for (l in c(1:w)) {
      S = S + (L[l] - M)^2
    }
    S = S/w

    e = 1.96*sqrt(S/w) # erreur d'approximation (à 95%)

    ER2[z, ] = c(M, e, gamma)
    z = z+1

    cat("\n", (z-1)/1.02, "%")

  } # Fin de la boucle sur gamma.




n = 100

### Pietronero ###

w = 10000 # Taille échantillons M-C pour le calcul de chaque "puissance".
Q = seq(from=-5, to=5, length=100)
Q[length(Q)+1] = 0
Q[length(Q)+1] = 0

EP2 = matrix(0, nrow= 102, ncol= 3)

  z = 1
  for (gamma in Q) { # On parcout plusieurs gamma pour les différents tests.
```

```
    M = 0 # Moyenne empirique = "puissance"
    L = rep(0,w) # Liste des Y(l)

    proba = rep(0, 90) # vecteur des probas H1 : Pietronero(gamma) pour sample
    for (i in c(1:90)) {

      if (gamma == 0) {
        proba[i] = log(1 + 1/(i+9))/log(10)
      }

      if (gamma != 0) {
        proba[i] = ( (i+9)^(-gamma) - (i+10)^(-gamma) )/( 10^(-gamma) - 100^(-gamma) )
      }

    }

    for (l in c(1:w)) {

      data = rep(0, n)
      for (i in c(1:n)) { # Génération des Xi sous Pietronero(gamma) de taille n

        data[i] = sample(x= c(10:99), size=1, prob= proba )

      }

      p = test.maison2(data)

      if (p == 1) {
        M = M + 1
        L[l] = 1
      }

      }
    M = M/w

    S = 0 # Variance empirique
    for (l in c(1:w)) {
      S = S + (L[l] - M)^2
    }
    S = S/w

    e = 1.96*sqrt(S/w) # erreur d'approximation (à 95%)

    EP2[z, ] = c(M, e, gamma)
    z = z+1

    cat("\n", (z-1)/1.02, "%")

  } # Fin de la boucle sur gamma.

### Graphiques ###
par( mfrow=c(1,2) )
```

```
plot(D100[103:202, 3], D100[103:202, 1], col="red", type="line", xlab="n=100, Rodriguez", ylab=" ") # T
abline(a=5/100,b=0, col="red")
#abline(a=2.5/100,b=0, col="black")
lines(D100[103:202, 3], D100[103:202, 1], col="red") # T2
lines(ER2[1:100, 3], ER2[1:100, 1], col="blue") # maison2

legend(x="bottomright", legend=c("T2", "Maison2"), col=c("red", "blue"), lty=1, cex=0.6)


plot(H100[103:202, 3], H100[103:202, 1], col="red", type="line", xlab="n=100, Pietronero", ylab=" ") #
abline(a=5/100,b=0, col="red")
#abline(a=2.5/100,b=0, col="black")
lines(H100[103:202, 3], H100[103:202, 1], col="red") # T2
lines(EP2[1:100, 3], EP2[1:100, 1], col="blue") # maison2

legend(x="bottomright", legend=c("T2", "Maison2"), col=c("red", "blue"), lty=1, cex=0.6)

#-----------------------------------------------------------------------------------------------
```

CSSEGI. 2020. *Données CSSE*. https://www.openscience.fr/IMG/pdf/iste_mas20v3n1_1.pdf.

DELAHAYE, JEAN-PAUL. n.d. *UNE EXPLICATION POUR LA LOI DE BENFORD*. https://www.cristal.univ-lille.fr/profil/jdelahay/pls:2018:299.pdf.

DELAHAYE, JEAN-PAUL, and Nicolas GAUVRIT. n.d. *POURQUOI LA LOI DE BENFORD n'EST PAS MYSTèRIEUSE*. https://journals.openedition.org/msh/10363?file=1.

Desgraupes, B. n.d. *Méthodes Statistiques : Tests d'adequation*. https://bdesgraupes.pagesperso-orange.fr/UPX/L2/MethStats_seance_11_doc.pdf.

Gilles DUCHARME and Credo VOVOR-DASSU. 2020. *benfordSmoothTest*. https://github.com/CSSEGISandData/COVID-19.

Gilles R.DUCHARME and Credo VOVOR-DASSU. n.d. *Test d'adéquation Loi de Benford*. https://www.openscience.fr/IMG/pdf/iste_mas20v3n1_1.pdf.

Kolmogorov-Smirnov. n.d. *Fonction de Repartition Empirique*. https://mistis.inrialpes.fr/software/SMEL/cours/ts/node7.html.

Leonid I. GALTCHOUK. n.d. *STATISTIQUES TESTS d'HYPOTHèSES*. https://www.universalis.fr/encyclopedie/tests-d-hypotheses-statistiques/8-tests-d-ajustement/#.

R Core Team. 2022. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Test de Kolmogorov-Smirnov. n.d. https://mistis.inrialpes.fr/software/SMEL/cours/ts/node7.html.

Test Du Khi-Deux. n.d. https://fr.wikipedia.org/wiki/Test_du_%CF%87%C2%B2.