UNIVERSITÉ DE LIÈGE

Projet – Étude statistique

Éléments de statistiques

Groupe 14

Jean-Baptiste CRISMER (s161640)
François ROZET (s161024)

3ème année de Bachelier Ingénieur civil

Année académique 2018-2019

1 Analyse descriptive

Avant d'utiliser la base de données contenant la population, il faut l'importer. Le script `loadData`[1] a été écrit à cette fin et est dès lors appelé dans tous les autres. Si la matrice `f` (cf. section A.2) est définie lors de son appel, les statistiques qu'elle contient seront alors calculées et stockées dans la structure `stats.dataset`.

1.a Histogrammes

Pour réaliser les histogrammes des taux de natalité et mortalité, on utilise la fonction `histogram` sur leur ensemble de données respectif.

En observant l'histogramme[2] du taux[3] de natalité, on semble observer une décroissance exponentielle. Il se peut, néanmoins, que cette dernière corresponde à la partie décroissante d'une loi Gaussienne. À l'inverse, l'histogramme du taux de mortalité est assimilable à une gaussienne.

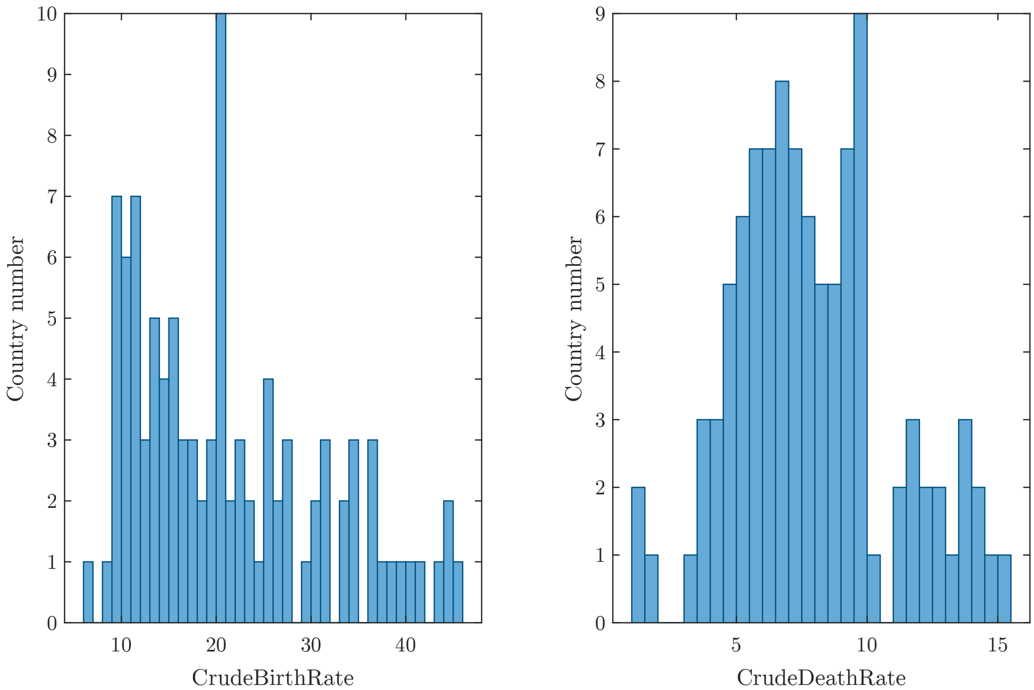

Figure 1 – Histogrammes des taux de natalité et de mortalité.

On peut noter que le taux de natalité, ou plutôt ses valeurs les plus fréquentes, est plus important que le taux de mortalité. Cela se traduit, d'un point de vue démographique,

1. Tous les scripts et fonctions utilisés et demandés sont fournis en annexe.
2. Les figures de ce document étant générées automatiquement à partir de la base de données, leurs axes sont en anglais.
3. Dans ce document les taux, ainsi que certains de leurs estimateurs, sont définis par mille habitants. Par la suite, si aucune unité n'est précisée pour un taux ou un estimateur, c'est qu'il est exprimé en ‰.

par un accroissement général de population. On remarque aussi que la fréquence associée aux valeurs du taux de natalité comprise entre 20 et 21 s'écarte fortement de la tendance générale.

1.b Statistiques

Pour calculer les statistiques moyenne, médiane, mode et écart-type on stocke respectivement les fonctions `mean`, `median`, `mode` et `std`, pré-implémentées dans MATLAB, dans la matrice `f` (cf. section 1).

Se trouvant dans la table 1, ces statistiques calculées pour les taux de natalité et mortalité de toute la population, permettent d'observer que la moyenne et la médiane sont relativement peu distante. Dès lors, les valeurs aberrantes sont soit réparties équitablement autour de la moyenne soit absentes. La seconde hypothèse sera validée plus tard.

À l'inverse, même s'il vaut la valeur la plus représentée, le mode estime très mal les valeurs les plus communes. On remarque aussi que l'écart-type, et dès lors la variance, est relativement important ce qui traduit un grand étalement des données autour de la moyenne.

Taux	Moyenne [‰]	Médiane [‰]	Mode [‰]	Écart-type [‰]
Natalité	21,284	19,900	10,700	9,952
Mortalité	7,866	7,450	5,700	3,008

Table 1 – Statistiques sur les taux de natalité et mortalité de la population.

Ce tableau permet aussi de mettre en évidence que les taux de natalité (11,7) et de mortalité (9,9) de la Belgique sont respectivement inférieur et supérieur à leur moyenne respective. On en déduit que la Belgique possède un accroissement démographique parmi les plus bas de la population.

1.c Taux normaux

Un individu d'une population est défini comme normal au sens de la loi normale s'il appartient à l'intervalle $[\mu - \sigma; \mu + \sigma]$, où μ et σ sont respectivement la moyenne et l'écart-type de la population. Dès lors, un taux de natalité, respectivement mortalité, est normal s'il appartient à [11,282; 31,286], respectivement [4,843; 10,889].

Ainsi, la Belgique est considérée comme normale au sens de la loi normale tant en natalité, qu'en mortalité.

Pour statuer de la normalité des autres pays, on écrit la fonction `isIn` qui détermine si une valeur est contenue dans un intervalle ou non. On compte ainsi que 63% des pays ont un taux de natalité normal et que 68% ont un taux de mortalité normal.

1.d Boîtes à moustaches

Pour réaliser les boîtes à moustaches des taux de natalité et mortalité, on utilise la fonction `boxplot` sur leur ensemble de données respectif.

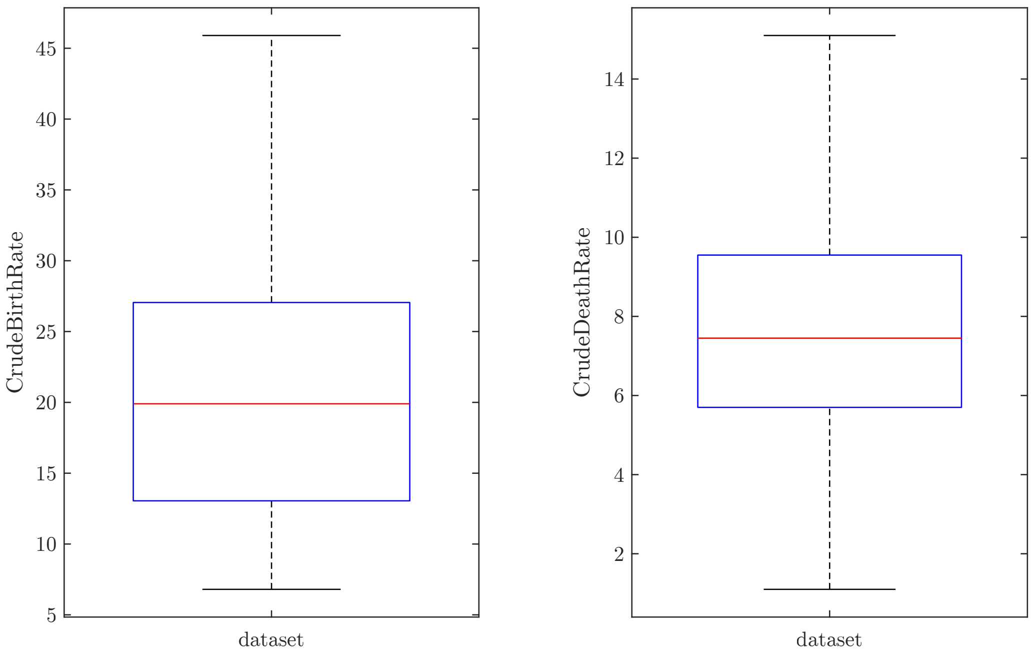

Figure 2 – Boîtes à moustaches des taux de natalité et mortalité.

La figure 2 permet de noter l'absence de données aberrantes. S'il y en avait, elles seraient marquées de croix rouges.

Les quartiles retranscrits dans la table 2, sont, quant à eux, calculés à partir de la fonction `quantile`.

Taux	Quartile [‰]		
	1^er^	2^ème^	3^ème^
Natalité	13,05	19,9	27,05
Mortalité	5,7	7,45	9,55

Table 2 – Quartiles des taux de natalité et mortalité.

Ils correspondent effectivement aux valeurs observables dans les boîtes à moustaches. On note aussi que la Belgique se trouve dans les 25% les plus bas en natalité et les plus haut en mortalité.

1.e Polygones des fréquences cumulées

Repris dans la figure 3, les polygones des fréquences cumulées, tracés à l'aide de la fonction `cdfplot`, permettent de statuer à propos de la distribution des taux. Si une loi normale est faiblement discernable pour le taux de mortalité, ce n'est pas le cas pour le taux de natalité.

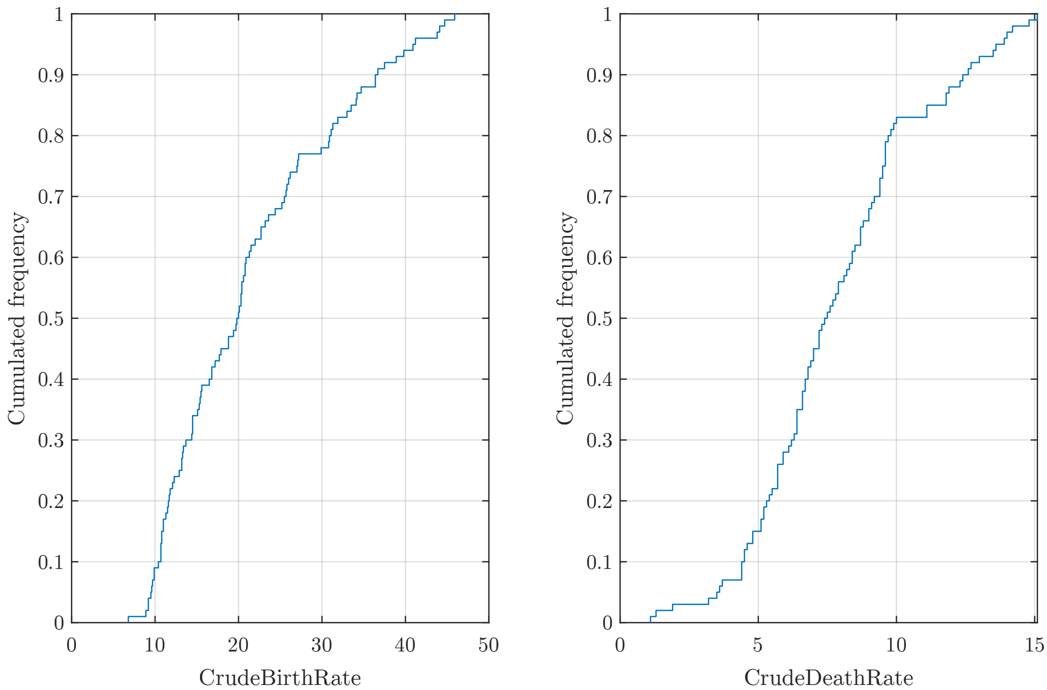

Figure 3 – Polygones des fréquences cumulées des taux de natalité et de mortalité.

Pour calculer la proportion de pays ayant un taux de natalité inférieur ou égal à 20 et supérieur à celui de la Belgique, noté b, on écrit et utilise la fonction `cf` qui détermine la fréquence cumulée (c.-à-d. la proportion de valeurs inférieures ou égales) associée à une valeur x dans une population D_n. Ainsi, il suffit de soustraire la fréquence cumulée de la Belgique à celle de 20 pour trouver la proportion recherchée.

$$\begin{aligned} P(b < x \leq 20) &= P(x \leq 20) - P(x \leq b) \\ &= 0{,}3 \end{aligned}$$

1.f Corrélation

Le coefficient de corrélation entre les taux de natalité et mortalité, calculé à partir de la fonction `corr`, vaut 0,1609. Ce résultat, assez faible, traduit une quasi-absence de corrélation entre les taux de natalité et mortalité.

Cette interprétation est renforcée par la figure 4. En effet, il est difficile d'y observer une quelconque relation entre les deux taux.

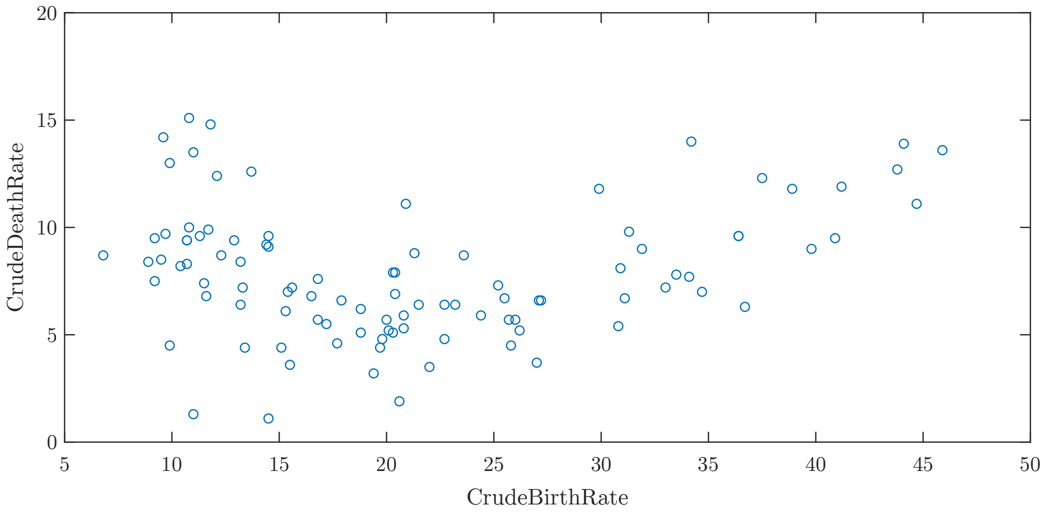

Figure 4 – Nuage de points du taux de mortalité en fonction du celui de natalité.

On peut néanmoins noter la légère tendance des pays ayant un taux de natalité élevé à également posséder un taux de mortalité élevé. Cela pourrait correspondre aux pays du tiers-monde.

2 Génération d'échantillons i.i.d.

Pour tirer les échantillons, le script `pickSamples` a été implémenté. De la même manière que `loadData` (cf. section 1), il calculera, pour chaque échantillon i.i.d. tiré, les statistiques contenues dans la matrice `f` et les stockera dans `stats.sample`.

2.a Un échantillon de vingt pays

i. Statistiques

Les statistiques de chaque échantillon sont obtenues par le script `Q2ai` de la même manière que celles de la population à la section 1.b. Comme le montre la table 3, les statistiques pour plusieurs échantillons restent relativement proches de celles calculées pour la population. Les écarts sont dûs aux pertes d'informations, résultat inévitable d'un échantillonnage.

ii. Boîtes à moustaches

À l'instar de la section 1.d, les boîtes à moustaches sont dessinées par la fonction `boxplot` dans le script `Q2aii`.

Échantillon	Taux	Moyenne [‰]	Médiane [‰]	Écart-type [‰]
1	Natalité	20,795	20,7	7,1825
2		23,015	20,2	10,771
3		20,94	19,7	10,222
1	Mortalité	7,68	8,45	2,7228
2		7,715	7,30	3,8209
3		7,905	8,00	2,9197

Table 3 – Statistiques des taux de natalité et mortalité pour trois échantillons.

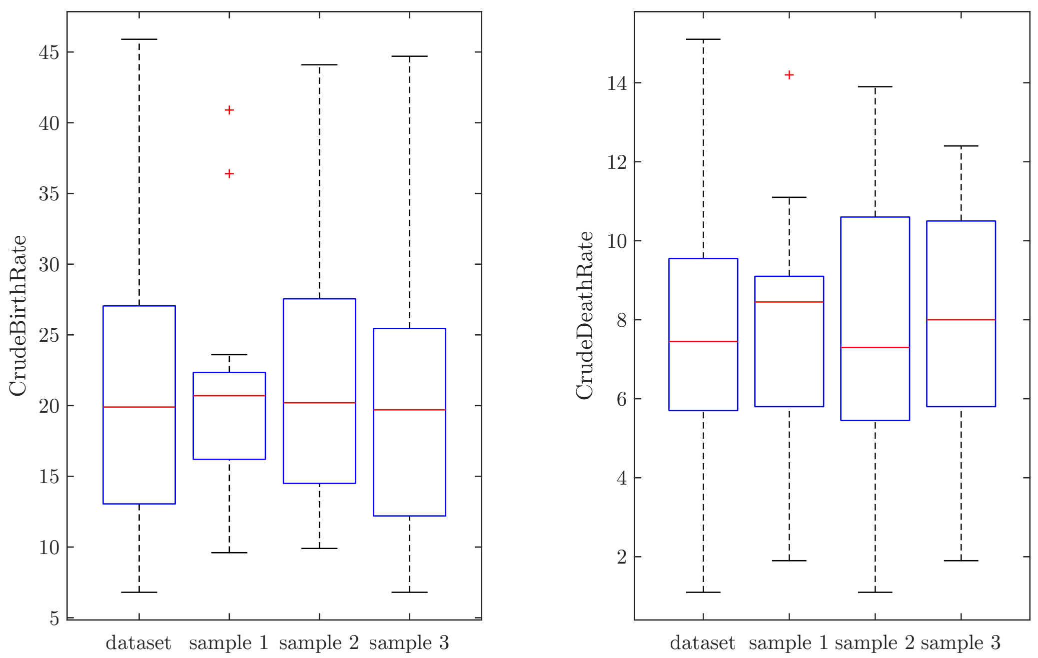

Figure 5 – Boîtes à moustaches des taux de natalité et mortalité pour la population et trois échantillons.

On note, à nouveau, quelques différences entre les échantillons et la population. La plus notable est que, contrairement à la population, les échantillons comportent des valeurs aberrantes. La deuxième, évidente, est que les moustaches des échantillons sont toujours comprises entre celles de la population. La dernière est, qu'en général, la médiane des échantillons reste relativement proche de la médiane de la population contrairement aux quartiles 25 et 75 qui fluctuent sensiblement.

iii. Polygones de fréquences cumulées

De la même manière qu'à la section 1.e, les polygones des fréquences cumulées, pour la population et les échantillons, sont tracés à l'aide de la fonction `cdfplot` par le script

`Q2aiii`.

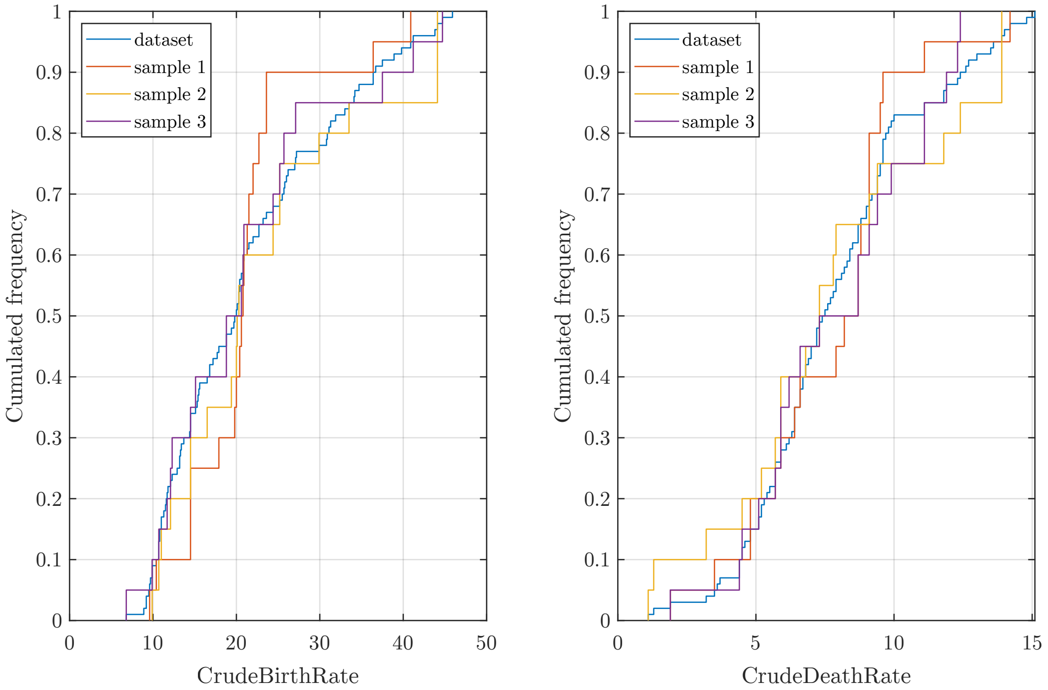

Figure 6 – Polygone des fréquences cumulées des taux de natalité et mortalité pour la population et trois échantillons.

On remarque dans la figure 6 que la distribution des échantillons est moins régulière que celle de la population en ce sens qu'elle comporte de plus grandes « marches ». Elle s'en éloigne aussi de façon significative, ce qui est appuyé par le calcul de la distance de Kolmogorov-Smirnov effectué à l'aide de la fonction `ks2stat`, elle même basée sur la fonction `kstest2`. En effet, les distances données dans la table 4 sont supérieures à 10%, ce qui est loin d'être négligeable.

Échantillon	Taux	Distance de K.-S. [–]
1		0,23
2	Natalité	0,12
3		0,09
1		0,14
2	Mortalité	0,12
3		0,12

Table 4 – Distances de Kolmogorov-Smirnov entre la population et trois échantillons.

2.b Cinq cents échantillons i.i.d. de vingt pays

Ici, contrairement à la section 2.a, les échantillons ne sont plus analysés individuellement. Chacune des statistiques pouvant servir d'estimateur à celles de la population, on s'intéresse à leur distribution respective sur les 500 échantillons.

En tant qu'estimateurs, la moyenne est notée m_χ, la médiane $median_\chi$, l'écart-type s_χ et l'écart-type corrigé s_{n-1}. Leur moyenne est évaluée dans le script `Q2b` et cela à trois reprises pour former la table 5 [4].

Set	Taux	Moyennes de [‰]			
		m_χ	$median_\chi$	s_χ	s_{n-1}
1	Natalité	21,287	19,269	9,605	9,854
2		21,317	19,335	9,619	9,869
3		21,354	19,430	9,649	9,899
1	Mortalité	7,868	7,582	2,861	2,936
2		7,836	7,525	2,884	2,959
3		7,910	7,590	2,891	2,966

Table 5 – Moyennes des estimateurs pour trois sets de 500 échantillons.

Les histogrammes qui suivent ont été tracés par le script `Q2bhist` appelé dans les scripts `Q2bi`, `ii`, `iii` et `iv` et de manière analogue à ceux de la figure 1.a tout en remplaçant les taux de la population par les estimateurs du premier set d'échantillons.

i. Histogrammes de m_χ

Au premier coup d'oeil à la figure 7, on remarque l'allure caractéristique des lois normales pour les deux taux.

Les moyennes de m_χ, reprises dans la table 5 avec les autres estimateurs, sont très, voir extrêmement, proches des moyennes de la population. En effet, les erreurs relatives qu'elles comportent sont toutes inférieures au pourcent.

ii. Histogrammes de $median_\chi$

À nouveau, les distributions, représentées par les histogrammes de la figure 8, décrivent de manière limpide des lois normales. On notera cependant que celle du taux de natalité présente un pic.

Cette fois-ci, les moyennes de $median_\chi$ sont éloignées des taux médians de la population par des erreurs relatives allant jusqu'à dépasser 3% ; un moins bon résultat que m_χ. A

4. À des fins de comparaison avec la population, il peut être utile d'utiliser la table 1.

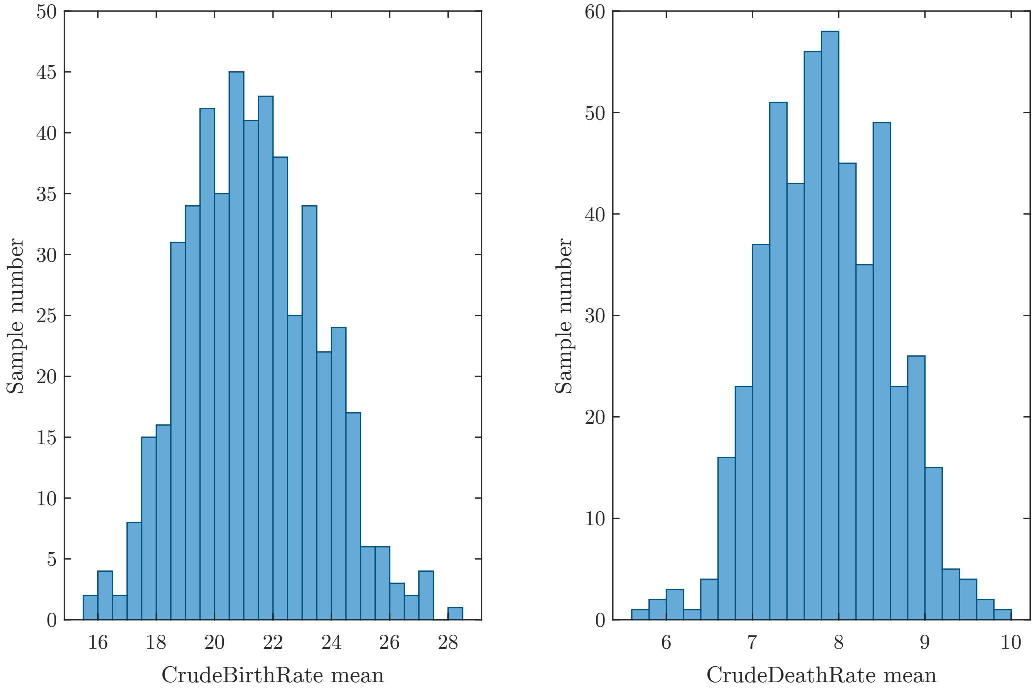

Figure 7 – Histogrammes de m_χ pour 500 échantillons.

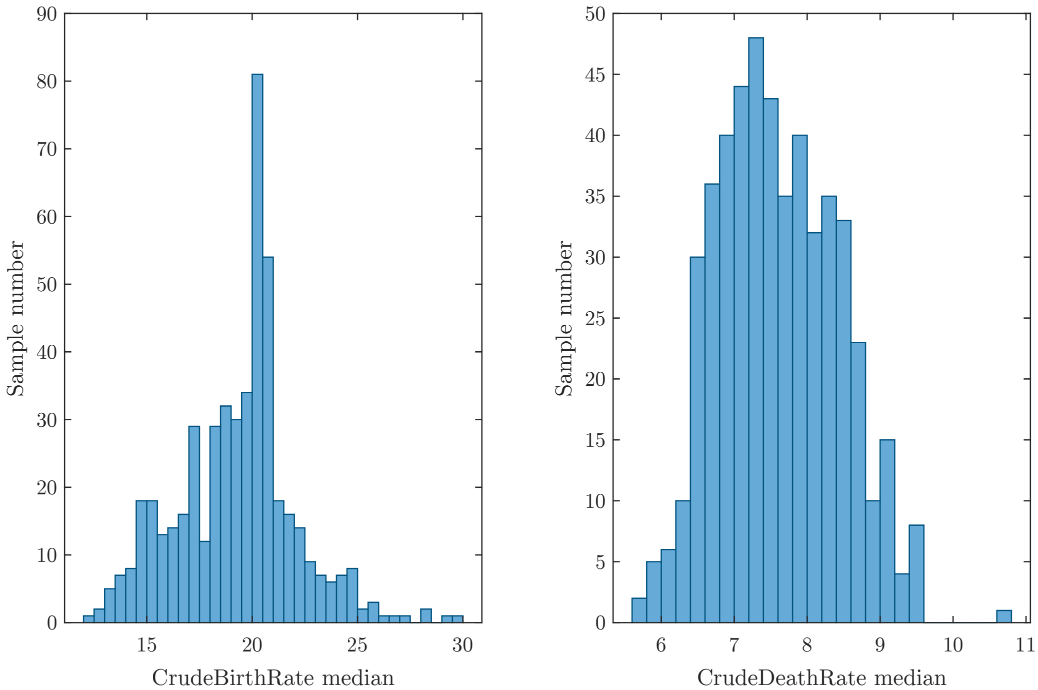

Figure 8 – Histogrammes de $median_\chi$ pour 500 échantillons.

fortiori, ses moyennes sont plus éloignées encore du taux moyen, qu'elles sous-estiment largement. Néanmoins, ce résultat est parfaitement compréhensible, puisque la médiane de la population sous-estime elle même le taux moyen. La médiane constitue donc un moins bon estimateur du taux moyen que la moyenne.

Théoriquement, il est souvent dit de la médiane qu'elle est un meilleur estimateur que la moyenne. En effet, contrairement à la moyenne, la médiane est généralement peu touchée, et donc peu biaisée, par la présence de valeurs aberrantes. Cependant, dans notre cas, notre population ne contient pas de données aberrantes (cf. section 1.d) et, dès lors, la médiane perd son avantage sur la moyenne.

iii. Histogrammes de s_χ

À première vue, on semble reconnaître dans la figure 9 des lois normales. Cependant, nous savons que la distribution de la variance, c.-à-d. le carré de s_χ suit, à un coefficient multiplicatif près, une loi Chi-carré à $n-1$ degrés de liberté (χ^2_{n-1}). Dès lors, s_χ ne peut pas suivre une loi normale.

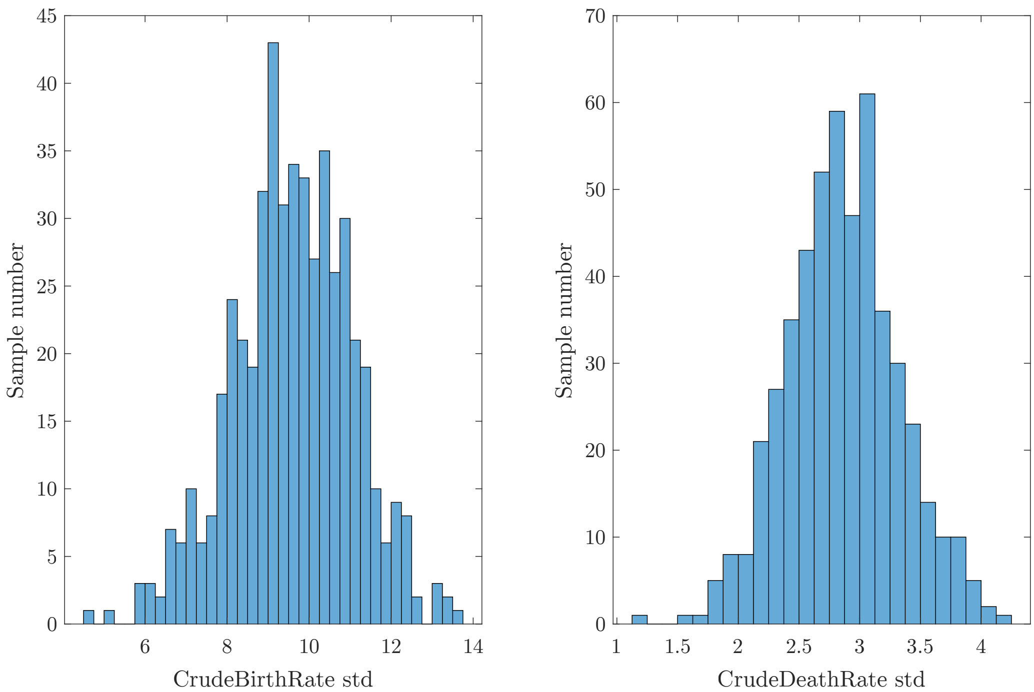

Figure 9 – Histogrammes de s_χ pour 500 échantillons.

Concernant la moyenne de l'estimateur s_χ, on remarque qu'elle sous-estime systématiquement et de manière significative l'écart-type de la population. Cette sous-estimation a déjà été discutée dans le cadre du cours et est le résultat de l'utilisation de la moyenne de l'échantillon à la place de celui de la population dans le calcul de l'écart-type. Pour résoudre ce problème, il suffit d'introduire la correction de Bessel, c.-à-d. l'écart-type corrigé [5], noté s_{n-1}.

Présentes dans la table 5, les moyennes de ce nouvel estimateur sont effectivement beaucoup plus proches de l'écart-type de la population tout en lui restant inférieures. On peut,

5. Son calcul numérique se fait aussi par le biais de la fonction `std`.

en effet, prouver à l'aide de l'inégalité de Jensen que s_{n-1} sous-estime l'écart-type de la population.

iv. Histogrammes de la distance de Kolmogorov-Smirnov

Le calcul des distances de K.-S. est complété par le script `Q2biv` d'une manière similaire à celle du script `Q2aiii`.

Bien qu'ils soient à nouveau assimilables à des lois normales, les histogrammes présentés dans la figure 10 semblent cette fois-ci asymétriques par rapport aux valeurs les plus représentées.

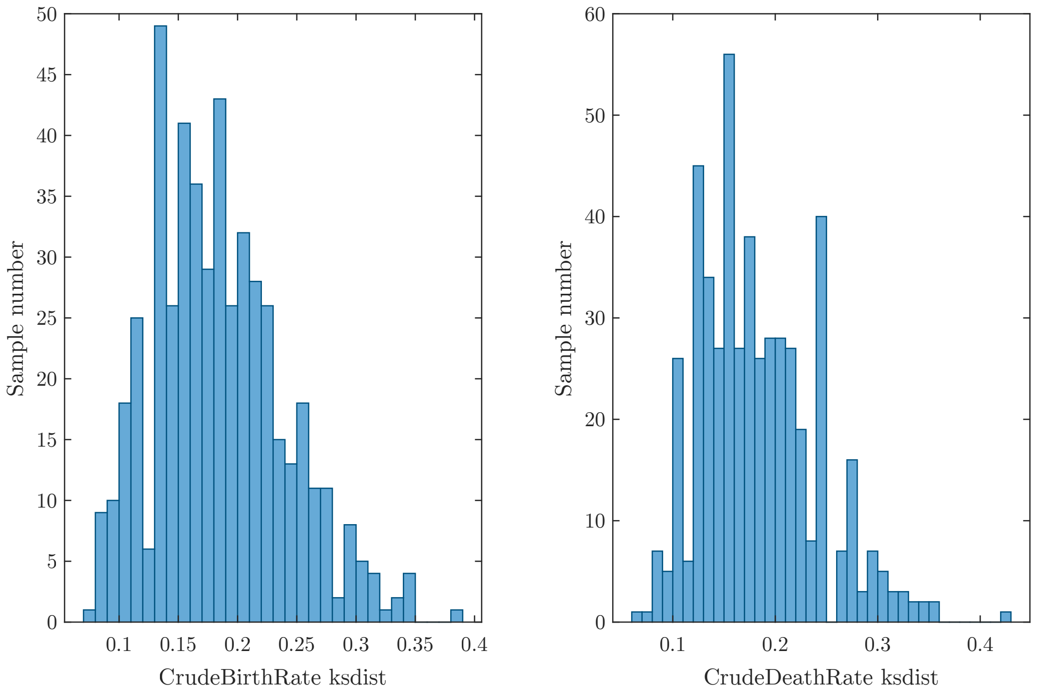

Figure 10 – Histogrammes des distances de Kolmogorov Smirnov entre les polygones des fréquences cumulées des taux pour la population et les échantillons.

3 Estimation

À partir de maintenant, n'est plus considéré que le taux de natalité.

On souhaite, dans les sections 3.a, 3.b et 3.c, étudier l'impact de la taille des échantillons sur la qualité des estimations de μ, le taux moyen, fournies par la moyenne et la médiane. Ainsi, de façon similaire à la section 2.b, le biais (par rapport à μ) et la variance des estimateurs m_χ et $median_\chi$ sont déterminés par les scripts `Q3a2b` et `Q3c` pour des sets de 100 échantillons de respectivement 20 et 50 pays.

Taille	Set	Biais de [‰]		Variance de [‰^{2}]	
		m_χ	$median_\chi$	m_χ	$median_\chi$
20	1	$3{,}24 \times 10^{-2}$	$-1{,}555$	5,366	8,402
	2	$-4{,}68 \times 10^{-2}$	$-2{,}151$	4,336	7,768
	3	$7{,}02 \times 10^{-2}$	$-2{,}0995$	4,944	8,966
50	4	$-2{,}07 \times 10^{-2}$	$-1{,}9745$	1,5196	2,1314
	5	$2{,}75 \times 10^{-2}$	$-1{,}9395$	2,6351	4,0747
	6	$-4{,}02 \times 10^{-2}$	$-2{,}034$	2,3859	3,9806

Table 6 – Biais et variances des estimateurs m_χ et $median_\chi$ pour des échantillons du taux de natalité de 20 et 50 pays.

3.a Moyenne

Comme la section i. le laissait présager, on remarque à l'aide de la table 6 que le biais de l'estimateur m_χ est très faible par rapport à la moyenne de la population. Il est intéressant de noter que le biais peut être positif comme négatif, ce qui traduit une distribution non-biaisée autour du taux moyen.

La variance semble quant à elle orbiter près de la valeur 5. Il est cependant difficile d'interpréter ce nombre car il représente une distance « au carré ». Pour interpréter, on utilise donc sa racine, l'écart-type, orbitant autour de 2,2. Cette valeur traduit l'étalement de l'estimateur m_χ et est relativement faible.

3.b Médiane

Il a été dit précédemment que l'estimateur médiane sous-estimait le taux moyen. Ce résultat apparaît nettement dans la table 6, le biais étant systématiquement négatif. On notera que ce biais reste proche du biais réel offert par la médiane de la population.

On apprend aussi, grâce à la variance, que la médiane possède une distribution plus étalée que la moyenne.

3.c Échantillons de cinquante pays

En augmentant la taille des échantillons, deux phénomènes notables se produisent. Le premier est que les biais fluctuent moins et le second que les variances ont diminué. Ce dernier est une conséquence évidente de l'augmentation de la taille des échantillons.

Le premier, quant à lui, décrit la stabilisation, avec la taille des échantillons, de la moyenne des estimateurs. Puisque le biais de l'estimateur m_χ tend vers 0, on peut dire que sa moyenne tend bien vers le taux moyen. De manière analogue, on attend de la moyenne de l'estimateur $median_\chi$ qu'elle tende vers le taux médian. Pour confirmer cette hypothèse, il faut vérifier que le biais de $median_\chi$ tende bien vers le biais réel de la médiane de la population. Cependant, bien qu'ils ne s'en soient pas éloignés, les biais pour des échantillons de 50 pays n'en sont pas réellement plus proches que ceux de 20 pays. Néanmoins, avec une taille d'échantillon beaucoup plus grande (i.e. 1000), cette convergence devient très notable.

Ainsi, il est naturel pour l'estimateur $median_\chi$ de sous-estimer le taux moyen puisqu'il estime sans-biais le taux médian, lui étant inférieur.

3.d Intervalles de confiance

Pour construire les intervalles de confiance associés aux échantillons, il a été fait le choix arbitraire que l'écart-type de la population était inconnu. Dès lors, dans l'hypothèse d'une distribution normale de la variable parente, les bornes des intervalles dépendent des estimateurs m_χ et s_{n-1} :

$$m_\chi - x_{1-\frac{\alpha}{2}} \frac{s_{n-1}}{\sqrt{n}} \leq \mu \leq m_\chi + x_{1-\frac{\alpha}{2}} \frac{s_{n-1}}{\sqrt{n}}$$

Où μ est le taux moyen et $x_{1-\frac{\alpha}{2}}$ est la valeur associée à la proportion $1 - \frac{\alpha}{2}$ par la loi de construction utilisée. En l'occurrence, il est demandé d'utiliser deux lois :

i. Une loi de student [6] : $x_{1-\frac{\alpha}{2}} = t_{1-\frac{\alpha}{2}} = 2{,}093$.
ii. Une loi de Gauss : $x_{1-\frac{\alpha}{2}} = u_{1-\frac{\alpha}{2}} = 1{,}96$.

Dans le script `Q3d`, $x_{1-\frac{\alpha}{2}}$ est déterminée à l'aide de la fonction `icdf` qui prend en paramètres une loi parmi celles stockées dans la matrice `g`, et la proportion $1 - \frac{\alpha}{2}$. Ensuite, la fonction `hasIn` vérifie pour chaque intervalle construit s'il contient le taux moyen ou non.

Set	Loi de	
	student	Gauss
1	93	92
2	96	96
3	97	95

Table 7 – Nombre d'intervalles contenant μ pour trois sets de 100 échantillons de 20 pays en fonction de la loi utilisée.

On remarque, en premier lieu, que le nombre d'intervalles contenant μ est toujours plus grand pour la loi de student que pour celle de Gauss. Cela est dû au fait que, pour les mêmes m_χ et s_{n-1}, l'intervalle construit avec la loi de student est plus grand que celui construit avec celle de Gauss, car $u_{1-\frac{\alpha}{2}} \leq t_{1-\frac{\alpha}{2}}$.

En deuxième lieu, on observe que la proportion d'intervalles rejetés est proche de α, ce qui est logique. Cependant, les valeurs obtenues sont parfois supérieures à ce qui est attendu

6. Le degré de liberté qu'il faut utiliser est $n - 1$, c.-à-d. la taille des échantillons réduit d'une unité.

(95 %). Cela est dû au hasard de l'échantillonnage. En effet, en augmentant le nombre d'échantillons à 10 000 (cf. table 8), on note que le nombre d'intervalles contenant μ y est systématiquement inférieur.

Set	Loi de	
	student	Gauss
1	9413	9266
2	9420	9259
3	9419	9282

Table 8 – Nombre d'intervalles contenant μ pour trois sets de 10 000 échantillons de 20 pays en fonction de la loi utilisée.

Ce résultat est plus probant et traduit le fait que la distribution de la loi parente, en tout cas dans notre base de donnée, ne suit pas exactement une loi normale, bien qu'elle y raisonnablement assimilable.

4 Tests d'hypothèse

Tirer 100 fois un échantillon de 40 pays par organisme, c.-à-d. l'état belge et les quatre instituts, comme il est demandé dans l'énoncé, revient à tirer 500 échantillons et les répartir ensuite aléatoirement entre eux. C'est ce qui est implémenté dans le script `Q4`.

Pour déterminer x et p, les proportions de pays ayant un taux de natalité plus faible que la Belgique respectivement dans la population et dans les échantillons, on utilise la fonction `cf` [7]. Ce faisant, pour la population, on trouve $x = 0{,}2$.

Pour vérifier l'hypothèse H_0, il faut satisfaire à un test d'hypothèse unilatéral à droite pour une proportion. Cela signifie que les proportions p doivent satisfaire l'inégalité

$$\begin{aligned} p &\leq x + u_{1-\frac{\alpha}{2}} \sqrt{\frac{x(1-x)}{n}} \\ &\leq 0{,}3240 \end{aligned}$$

Cependant, ce test n'est valide que si on peut assimiler la distribution de la variable p à une loi normale de moyenne x et d'écart-type $\sqrt{x(1-x)}$, ce qui n'est valable que lorsque $\min(nx;\, n(1-x)) \geq 5$. Ce dernier critère étant validé, le test l'est aussi.

Ainsi, il suffit de comparer chaque p avec la borne calculée pour déterminer le nombre d'échantillons qui vérifie l'hypothèse nulle. Les résultats de cette comparaison sont fournis dans la table 9.

7. Il est important de remarquer que, sans un paramètre additionnel, `cf` détermine la proportion des valeurs « plus faible ou égale ».

Set	Belgique	Institut				OMS
		1	2	3	4	
1	4	6	8	2	0	15
2	6	5	7	5	3	20
3	5	6	6	2	2	18

Table 9 – Nombre de rejets de H_0 pour trois sets de 5 fois 100 échantillons de 40 pays.

4.a Comparaison à α

Le nombre de fois où l'état belge a rejeté l'hypothèse nulle orbite autour de 5. Puisque 100 tests ont été effectués, on obtient bien une valeur proche de α, c.-à-d. 5%. Cependant, on remarque pour les autres instituts que cette valeur fluctue. Pour s'assurer de la proportion proche de α d'hypothèse rejetée, on effectue les tests pour des sets de 5 fois 10 000.

Set	Belgique	Institut				OMS
		1	2	3	4	
1	442	438	429	478	429	1676
2	433	390	467	414	439	1601
3	439	422	440	426	424	1611

Table 10 – Nombre de rejets de H_0 pour trois sets de 5 fois 10 000 échantillons de 40 pays.

Les résultats de cette seconde expérience, présentés dans la table 10, confirment l'impression initiale. On note néanmoins que la proportion est toujours inférieure à α ; différence probablement due à l'hypothèse d'une loi normale.

4.b Considération de l'OMS

Sur les trois sets de 100 échantillons, l'OMS considére, en moyenne, 17,7 fois que la Belgique a un taux de natalité faible. Ce nombre de rejets est beaucoup plus élevé que celui fourni par l'état belge. Néanmoins, cela est compréhensible. Il suffit d'un seul rejetant l'hypothèse pour que l'OMS le fasse à son tour. Ainsi, en supposant que, comme la Belgique, chaque institut rejette H_0 avec une probabilité α, l'OMS le fera avec une probabilité

$$p_4 = 1 - C_4^0(1-\alpha)^4 = 0{,}1855 \simeq 4\alpha$$

En observant les proportions de rejets de l'OMS dans les deux tableaux précédents, on trouve effectivement des valeurs qui en sont proches.

4.c Équilibrage

Pour réduire la différence entre la proportion de rejets de l'état belge et celle l'OMS, il faut réduire l'avantage qu'ont, par leur nombre, les instituts. Plusieurs méthodes permettent d'obtenir ce résultat :

- Puisque le nombre d'instituts, `ni` dans le code, est leur principal avantage, diminuer ce dernier est une solution. Notamment, on peut déterminer la proportion de rejets de l'OMS pour 3 et 2 instituts, respectivement.

$$p_3 = 1 - C_3^0(1-\alpha)^3 = 0{,}1426 \simeq 3\alpha$$
$$p_2 = 1 - C_2^0(1-\alpha)^2 = 0{,}0975 \simeq 2\alpha$$

- Augmenter la tolérance, `tol` dans le code, de l'OMS est aussi une solution valable. En effet, s'il faut maintenant au moins deux instituts rejetant l'hypothèse pour que l'OMS le fasse, on a comme proportion

$$p'_4 = 1 - C_4^0(1-\alpha)^4 - C_4^1\alpha(1-\alpha)^3 = 0{,}0140 < \alpha$$

  Néanmoins cette proportion est plus faible encore que celle de l'état belge, qui en devient avantagé.

- Diminuer le seuil de signification des instituts ou augmenter celui de Belgique sont aussi des solutions valables.

A Ligne de conduite

A.1 Cohérence

Dans un soucis de cohérence, chaque script répondant à une question est partitionné de la même manière. Premièrement, dans la section `Parameters` se trouvent les paramètres liés à la question, comme la taille ou le nombre des échantillons. Ensuite, la section `Calls` appelle d'autres scripts dont l'utilisation est récurrente. Cela permet de rendre les codes plus concis. Vient après la section `Compute` qui est le corps principal du script. C'est là que sont effectués les calculs.

Finalement, les sections `Plot` et `Display` servent à afficher respectivement les graphiques et les résultats numériques.

A.2 Modularité

Pour rendre le code plus modulaire, l'appel de fonctions calculant une statistique sur un ensemble de données se fait via une matrice (`f` ou `g`). Dans cette matrice chaque ligne représente une fonction qui doit être appelée et contient, dans l'ordre, un nom arbitraire associé, le nom d'appel de la fonction et ses paramètres optionnels. Il suffit dès lors d'itérer sur les lignes de cette matrice pour calculer chaque statistique.

B Scripts

```
addpath('resources/csv/');
addpath('scripts/');
addpath('functions/');
```

Listing 1 – `startup.m`

```
% Setup
dataset = readtable('db_stat14.csv', 'ReadRowNames', true);
index = (dataset.Properties.VariableNames)';

% Compute
if exist('f', 'var') == 1 % if #f is set
    for i = 1:size(index, 1) % for each population
        stats.dataset.(index{i}) = table;
        for j = 1:size(f, 1) % for each function
            stats.dataset.(index{i}).(f{j, 1}) = feval(f{j, 2},
                dataset.(index{i}), f{j, 3}{:});
        end
    end
end
```

Listing 2 – `loadData.m`

```
% Setup
sample = cell(m, 1);

for i = 1:m % pick samples
    sample{i} = dataset(randi([1 size(dataset, 1)], n, 1), :);
end

% Compute
for i = 1:size(index, 1) % for each population
    stats.sample.(index{i}) = table;
    for j = 1:size(f, 1) % for each function
        temp = zeros(m, 1);
        for k = 1:m % for each sample
            temp(k) = feval(f{j, 2}, sample{k}.(index{i}), f{j, 3}{:});
        end
        stats.sample.(index{i}).(f{j, 1}) = temp;
    end
end
```

Listing 3 – pickSamples.m

```
%% Parameters

space = [1, 0.5];

%% Calls

loadData;

%% Plot

for i = 1:size(index, 1)
    % Setup nice edges
    temp = min(dataset.(index{i}));
    xmin = temp - mod(temp, space(i));
    temp = max(dataset.(index{i}));
    xmax = temp - mod(temp, space(i)) + space(i);
    edges = xmin:space(i):xmax;

    % Plot
    subplot(1, 2, i);
    histogram(dataset.(index{i}), edges);
    xlabel(index{i});
    ylabel(strcat('Country number'));
end

%% Clear workspace

clearvars -except dataset index;
```

Listing 4 – Q1a.m

```
%% Parameters
```

```

f = {
    'mean', 'mean', {};
    'median', 'median', {};
    'mode', 'mode', {};
    'std', 'std', {1}
};

%% Calls

loadData;

%% Display

% Setup
tab = table;
for i = 1:size(index, 1)
    tab(end + 1, :) = stats.dataset.(index{i});
end
tab.Properties.RowNames = index;

% Display
disp(tab);

%% Clear workspace

clearvars -except dataset index stats tab;
```

Listing 5 – `Q1b.m`

```
%% Parameters

f = {
    'mean', 'mean', {};
    'std', 'std', {1}
};
country = 'Belgium';

%% Calls

loadData;

%% Compute

% Setup
isNormed = table;
iCountry = find(strcmp(dataset.Properties.RowNames, country)); %
    search country

% Compute interval
for i = 1:size(index, 1)
    temp = [-1, 1] * stats.dataset.(index{i}).std;
    stats.dataset.(index{i}).interval = temp +
        stats.dataset.(index{i}).mean;
end
```

```

% Compute proportion
for i = 1:size(index, 1)
    isNormed.(index{i}) = isIn(dataset.(index{i}),
        stats.dataset.(index{i}).interval);
    stats.dataset.(index{i}).proportion = sum(isNormed.(index{i}), 1)
        / size(dataset, 1);
end
isNormed.Properties.RowNames = dataset.Properties.RowNames;

%% Display

% Setup
tab = table;
for i = 1:size(index, 1)
    tab(end + 1, :) = stats.dataset.(index{i});
end
tab.Properties.RowNames = index;

% Display
disp(tab);
disp(isNormed(iCountry, :));

%% Clear workspace

clearvars -except dataset index stats isNormed iCountry tab;
```

Listing 6 – `Q1c.m`

```
%% Parameters

f = {
    'quantile25', 'quantile', {0.25};
    'median', 'median', {};
    'quantile75', 'quantile', {0.75}
};

%% Calls

loadData;

%% Plot

for i = 1:size(index, 1)
    subplot(1, 2, i);
    boxplot(dataset.(index{i}), 'Labels', 'dataset', 'Widths', 0.8);
    ylabel(index{i});
end

%% Display

% Setup
tab = table;
for i = 1:size(index, 1)
    tab(end + 1, :) = stats.dataset.(index{i});
```

```
end
tab.Properties.RowNames = index;

% Display
disp(tab);

%% Clear workspace

clearvars -except dataset index stats;
```

Listing 7 – `Q1d.m`

```
%% Parameters

country = 'Belgium';
supp = 20;

%% Calls

loadData;

%% Compute

% Setup
tab = table;
iCountry = find(strcmp(dataset.Properties.RowNames, country)); % search country

% Compute
temp = cf(dataset.(index{1}), supp);
tab.proportion = temp - cf(dataset.(index{1}), dataset{iCountry, index{1}});

%% Display

disp(tab);

%% Plot

for i = 1:size(index, 1)
    subplot(1, 2, i);
    cdfplot(dataset.(index{i}));
    ylabel('Cumulated frequency');
    xlabel(index{i});
end

%% Clear workspace

clearvars -except dataset index tab;
```

Listing 8 – `Q1e.m`

```
%% Calls

```

```
loadData;

%% Compute

% Setup
tab = table;

% Compute
tab.correlation = corr(dataset.(index{1}), dataset.(index{2}));

%% Display

disp(tab);

%% Plot

scatter(dataset.(index{1}), dataset.(index{2}));
xlabel((index{1}));
ylabel((index{2}));
daspect([1 1 1]);

%% Clear workspace

clearvars -except dataset index tab;
```

Listing 9 – `Q1f.m`

```
%% Parameters

n = 20;
m = 3;
f = {
    'mean', 'mean', {};
    'median', 'median', {};
    'std', 'std', {1}
};

%% Calls

loadData;
pickSamples;

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(stats.sample.(index{i}));
end

%% Clear workspace

clearvars -except dataset index stats;
```

Listing 10 – `Q2ai.m`

```
%% Parameters

n = 20;
m = 3;
f = {
    'mean', 'mean', {};
    'median', 'median', {};
    'std', 'std', {1}
};

%% Calls

loadData;
pickSamples;

%% Plot

for i = 1:size(index, 1)
    % Setup
    a = [dataset.(index{i})];
    b = zeros(size(dataset, 1), 1);

    names = fieldnames(stats);
    labels = {names{1}};
    for j = 1:m
        a = [a; sample{j}.(index{i});];
        b = [b; j * ones(n, 1) ];
        labels{end + 1, 1} = [names{2}, ' ', num2str(j)];
    end

    % Plot
    subplot(1, 2, i);
    boxplot(a, b, 'Labels', labels, 'Widths', 0.8);
    ylabel(index{i});
end

%% Clear workspace

clearvars -except dataset index stats;
```

Listing 11 – Q2aii.m

```
%% Parameters

n = 20;
m = 3;
f = {};
g = {'ksdist', 'ks2stat', {}};

%% Calls

loadData;
pickSamples;

```

```
%% Compute

for i = 1:size(index, 1)
    g{3} = {dataset.(index{i})};
    temp = zeros(m, 1);
    for j = 1:m
        temp(j) = feval(g{2}, sample{j}.(index{i}), g{3}{:});
    end
    stats.sample.(index{i}).(g{1}) = temp;
end

%% Plot

for i = 1:size(index, 1)
    % Setup
    names = fieldnames(stats);
    legends = {names{1}};
    for j = 1:m
        legends{end + 1, 1} = [names{2}, ' ', num2str(j)];
    end

    % Plot
    subplot(1, 2, i);
    cdfplot(dataset.(index{i}));
    hold on
        for j = 1:m
            cdfplot(sample{j}.(index{i}));
        end
    hold off
    legend(legends);
    ylabel('Cumulated frequency');
    xlabel(index{i});
end

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(stats.sample.(index{i}));
end

%% Clear workspace

clearvars -except dataset index stats;
```

Listing 12 – `Q2aiii.m`

```
%% Parameters

n = 20;
m = 500;
f = {
    'mean', 'mean', {};
    'median', 'median', {};
    'std', 'std', {1};
```

```
    'std_corr', 'std', {}
};
g = {'mean', 'mean', {}};

%% Calls

loadData;
pickSamples;

%% Compute

for i = 1:size(index, 1)
    tab.(index{i}) = table;
    for j = 1:size(f, 1) % for each function in #f
        temp = table;
        for k = 1:size(g, 1) % for each function in #g
            temp.(g{k, 1}) = feval(g{k, 2},
                stats.sample.(index{i}).(f{j, 1}), g{k, 3}{:});
        end
        tab.(index{i})(end + 1, :) = temp;
    end
    tab.(index{i}).Properties.RowNames = f(:, 1);
end
```

Listing 13 – Q2b.m

```
for i = 1:size(index, 1)
    % Setup nice edges
    temp = min(stats.sample.(index{i}).(f{k, 1}));
    xmin = temp - mod(temp, space(i));
    temp = max(stats.sample.(index{i}).(f{k, 1}));
    xmax = temp - mod(temp, space(i)) + space(i);
    edges = xmin:space(i):xmax;

    % Plot
    subplot(1, 2, i);
    histogram(stats.sample.(index{i}).(f{k, 1}), edges);
    xlabel(strcat(index{i}, {' '}, f{k, 1}));
    ylabel('Sample number');
end
```

Listing 14 – Q2bhist.m

```
%% Parameters

space = [0.5, 0.20];

%% Calls

Q2b;

%% Plot

k = 1;
```

```
Q2bhist;

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(tab.(index{i})(k, :));
end

%% Clear workspace

clearvars -except dataset index stats tab;
```

Listing 15 – `Q2bi.m`

```
%% Parameters

space = [0.5, 0.2];

%% Calls

Q2b;

%% Plot

k = 2;
Q2bhist;

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(tab.(index{i})(k, :));
end

%% Clear workspace

clearvars -except dataset index stats tab;
```

Listing 16 – `Q2bii.m`

```
%% Parameters

space = [0.25, 0.125];

%% Calls

Q2b;

%% Plot

k = 3;
Q2bhist;

```

```
%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(tab.(index{i})(k, :));
end

%% Clear workspace

clearvars -except dataset index stats tab;
```

Listing 17 – Q2biii.m

```
%% Parameters

n = 20;
m = 100;
f = {
    'mean', 'mean', {};
    'median', 'median', {}
};
g = {
    'gap', 'gap', {};
    'var', 'var', {1}
};

%% Calls

loadData;
pickSamples;

%% Comute

for i = 1:size(index, 1)
    tab.(index{i}) = table;
    g{1, 3} = {stats.dataset.(index{i}).mean}; % change @gap
        parameter according to the population studied
    for j = 1:size(g, 1) % for each function in #g
        temp = zeros(size(f, 1), 1);
        for k = 1:size(f, 1) % for each function in #f
            temp(k) = feval(g{j, 2}, stats.sample.(index{i}).(f{k,
                1}), g{j, 3}{:});
        end
        tab.(index{i}).(g{j, 1}) = temp;
    end
    tab.(index{i}).Properties.RowNames = f(:, 1);
end

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(tab.(index{i}));
end

```

```
%% Clear workspace

clearvars -except dataset index stats tab;
```

Listing 18 – `Q3a2b.m`

```
%% Parameters

n = 50;
m = 100;
f = {
    'mean', 'mean', {};
    'median', 'median', {}
};
g = {
    'gap', 'gap', {};
    'var', 'var', {1}
};

%% Calls

loadData;
pickSamples;

%% Comute

for i = 1:size(index, 1)
    tab.(index{i}) = table;
    g{1, 3} = {stats.dataset.(index{i}).mean}; % change @gap
        parameter according to the population studied
    for j = 1:size(g, 1) % for each function in #g
        temp = zeros(size(f, 1), 1);
        for k = 1:size(f, 1) % for each function in #f
            temp(k) = feval(g{j, 2}, stats.sample.(index{i}).(f{k,
                1}), g{j, 3}{:});
        end
        tab.(index{i}).(g{j, 1}) = temp;
    end
    tab.(index{i}).Properties.RowNames = f(:, 1);
end

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(tab.(index{i}));
end

%% Clear workspace

clearvars -except dataset index stats tab;
```

Listing 19 – `Q3c.m`

```
%% Parameters

n = 20;
m = 100;
f = {
    'mean', 'mean', {};
    'std', 'std', {1};
    'std_corr', 'std', {0}
};
g = {
    'student', 'T', {n - 1};
    'normal', 'Normal', {0, 1}
};
p = 0.95;

%% Calls

loadData;
pickSamples;

%% Compute

% Setup
alpha = 1 - p;

for i = 1:size(index, 1)
    tab.(index{i}) = table;

    % Compute intervals
    temp = zeros(size(g, 1), 1);
    for j = 1:size(g, 1) % for each distribution law in #g
        intervals = [-1, 1] * icdf(g{j, 2}, 1 - alpha / 2, g{j, 3}{:});
        intervals = intervals .* stats.sample.(index{i}).std_corr * n^(-1/2);
        intervals = intervals + stats.sample.(index{i}).mean;

        % Count
        temp(j) = sum(hasIn(stats.dataset.(index{i}).mean, intervals));
    end

    tab.(index{i}).number = temp;
    tab.(index{i}).Properties.RowNames = g(:, 1);
end

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(tab.(index{i}));
end

%% Clear workspace

clearvars -except dataset index stats tab;
```

Listing 20 – `Q3d.m`

```
%% Parameters

n = 40;
ni = 4;
l = 100;
m = (ni + 1) * l;
f = {
    'mean', 'mean', {};
    'std', 'std', {1}
};
g = {'x', 'cf', {[], 1}};
country = 'Belgium';
p = 0.95;
tol = 0;

%% Calls

loadData;
pickSamples;

%% Compute

% Setup
k = size(f, 1) + 1;
f(k, :) = g;

iCountry = find(strcmp(dataset.Properties.RowNames, country)); % search #country index

alpha = 1 - p;

% Compute x
for i = 1:size(index, 1)
    f{k, 3}{1} = dataset{iCountry, index{i}};
    stats.dataset.(index{i}).(f{k, 1}) = feval(f{k, 2}, dataset.(index{i}), f{k, 3}{:});
    temp = zeros(m, 1);
    for j = 1:m
        temp(j) = feval(f{k, 2}, sample{j}.(index{i}), f{k, 3}{:});
    end
    stats.sample.(index{i}).(f{k, 1}) = temp;
end

% Compute H0
for i = 1:size(index, 1)
    temp = (stats.dataset.(index{i}).(f{k, 1}) * (1 - stats.dataset.(index{i}).(f{k, 1})) / n)^(1/2);
    temp = stats.dataset.(index{i}).(f{k, 1}) + temp * icdf('Normal', 1 - alpha / 2, 0, 1);
    temp = temp >= stats.sample.(index{i}).(f{k, 1}); % for each of the #m samples
```

```

    % Arrange #temp into #ni institutes
    H0.(index{i}) = table;
    H0.(index{i}).(country) = temp(1:l);
    for j = 1:ni
        H0.(index{i}).(['Institute_' num2str(j)]) = temp(1 + j * l:(1 + j) * l);
    end
    H0.(index{i}).('OMS') = sum(H0.(index{i}){:, 2:end}, 2) + tol >= ni;

    % Count
    number.(index{i}) = array2table(1 - sum(H0.(index{i}){:,:}, 1));
    number.(index{i}).Properties.VariableNames = H0.(index{i}).Properties.VariableNames;
end

%% Display

for i = 1:size(index, 1)
    disp([index{i} ' :']);
    disp(number.(index{i}));
end

%% Clear workspace

clearvars -except dataset index stats H0 number;
```

Listing 21 – Q4.m

C Fonctions

Lors de l'écriture des fonctions, nous avons fait le choix de ne pas les rendre robustes, c.-à-d. que nous supposons leurs entrées valides et ne les vérifions pas.

```
function y = cf(Dn, x, varargin)
%% cf

% Compute #y, the cumulative frequency of #x in the population #Dn.

%% Defaults

defaults = {0};
idx = ~cellfun('isempty',varargin);
defaults(idx) = varargin(idx);

%% Code
    y = zeros(size(x));
    for i = 1:numel(x)
        if defaults{1} == 0
            temp = sum(Dn(:) <= x(i), 1);
        else
            temp = sum(Dn(:) < x(i), 1);
        end
        y(i) = temp / numel(Dn);
    end
end
```

Listing 22 – `cf.m`

```
function y = gap(x, mu)
%% gap

% Compute #y, the gap between the mean of #x's values and #mu.

%% Code
    y = mean(x) - mu;
end
```

Listing 23 – `gap.m`

```
function y = hasIn(x, in)
%% hasIn

% Return in #y, for each interval contained in #in, if #x is included
    or not in the interval.

%% Code
    y = (x(1) >= in(:, 1)) .* (x(1) <= in(:, end));
end
```

Listing 24 – `hasIn.m`

```
function y = isIn(x, in)
%% isIn

% Return in #y, for each value contained in #x, if the value is
    included or not in the interval #in.

%% Code
    y = (x >= in(1)) .* (x <= in(end));
end
```

Listing 25 – `isIn.m`

```
function ks2stat = ks2stat(x1, x2)
%% ks2stat

% Compute #ks2stat, the Kolmogorov-Smirnov distance between #x1 and
    #x2 samples.

%% Code
    [~, ~, ks2stat] = kstest2(x1, x2);
end
```

Listing 26 – `ks2stat.m`