TPinfo : **Extraction et analyse de données**

Objectifs pédagogiques du TP :

- Utilisation de modules et de bibliothèques.
- Lecture d'un fichier de données simples.
- Calculs statistiques sur ces données.
- Représentation graphique.

Le site du gouvernement français `https://www.data.gouv.fr/fr/pages/donnees-coronavirus/` met à disposition en accès libre les données de santé publique concernant l'épidémie de COVID-19.

On y trouve notamment les données hospitalières ou encore le taux d'incidence de l'épidémie.

On travaillera sur le fichier des données quotidiennes de positivité des tests à l'échelle de la France : sp-pe-tb-quot-fra-2021-11-24-19h07.csv.

Remarque : Un fichier du type CSV, pour *comma-separated values*, est un format texte représentant des données tabulaires sous forme de valeurs séparées par des virgules.

- Les colonnes `P_f`, `P_h` et `P` correspondent au nombre de tests positif chez les femmes, les hommes et au total.
- Les colonnes `pop_f`, `pop_h` et `pop` correspondent aux population féminine, masculine et totale.
- La colonne `cl_age90` correspond aux classes d'age par fourchette de 10 ans. La ligne pour laquelle `cl_age90 = 0` correspond à la somme de toutes les classes d'age.

Lecture des données

1. Ouvrir le fichier csv avec un éditeur de texte et lire la première ligne pour en déduire à quelle position se trouve la colonne des classes d'age ainsi que celle des tests positifs.
2. Ouvrir le fichier csv dans python avec la fonction `open` puis utiliser les commandes `read.line()` et/ou `read.lines()` afin de :
   - Lire chaque ligne du fichier et la transformer en une liste contenant les valeurs de chaque ligne à l'aide de la méthode `.split(';')`.
   - Extraire le nombre total de tests positifs chaque jour et les stocker dans une liste.

⚠ Les commandes de lecture créent des chaines de caractères `str` qu'il faut convertir en valeurs entières `int`

Anaylse statistique initiale

3. Écrire une fonction `statistique(liste)` qui prend une liste en entrée et qui renvoie une liste contenant la somme des éléments de la liste, la moyenne $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ et l'écart-type $\sigma_x = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}$.
4. Tracer la représentation graphique du nombre de cas positifs en précisant les noms des axes. Quel soucis observe-t-on ?

Lissage des données

5. Afin d'obtenir des données plus représentatives, on se propose de lisser les données. Repérer les indices correspondant à ces jours et écrire une fonction `lissagedimanche(liste)` qui calcule la moyenne des positifs entre un dimanche et un lundi successifs et qui remplace les valeurs de ces deux jours dans une nouvelle liste.
6. Le résultat est meilleur, mais pas encore très probant. Afin de lisser correctement les résultats, on se propose de faire une moyenne glissante sur 7 jours. Écrire une fonction `lissageglissant(liste)` qui créé une nouvelle liste dont la valeur pour chaque jour (chaque indice) correspond à la moyenne des positifs sur les 3 jours avant et après. On pourra extraire une sous liste avec la commande `liste[i:j]` qui renvoie une liste contenant les éléments `i` à `j` de la liste `liste`.

Compléments

7. À partir des positifs mesurés chaque jour, évaluer grossièrement le taux de reproduction $r = \dfrac{\mathrm{P}_{n+1}}{\mathrm{P}_n}$ du virus d'un jour n au suivant $n+1$.
8. À l'aide la fonction `plt.hist(liste, range = (min, max), bins = int)`, tracer un histogramme de la répartition des jours en fonction du nombre de tests positifs.
9. Proposer une fonction qui permet de tracer un histogramme du nombre de tests positifs en fonction de la classe d'age.