

Info. TP6: Génération et tracé de données

- <https://learn.heeere.com/2018-infospichi-8a82/>
- http://matplotlib.org/api/pyplot_api.html (doc pyplot)

Rendu : sous forme d'archive

À la fin, vos réponses seront rendues dans une archive zip : cette archive contiendra un fichier `compte-rendu.txt` (avec votre **nom**, **groupe**, et les réponses aux questions précédées d'une apostrophe) et vos **fichiers pythons**.

Pour faire une **archive zip** de votre TP pour le rendu, exécutez, dans le terminal :

```
cd ~/info-spichi/  
zip -r tp6.zip tp6/
```

Important : mise en place

- Suivre les consignes des TP précédents, et travailler dans un dossier `tp6`.

Exercice 1 – Générateur de données aléatoires

Rappel : pour cet exercice, travaillez dans un dossier `ex1` lui même dans le dossier `tp6`.

Rappel TP5: Bluecime veut compléter son système pour fournir des données météorologiques automatiquement aux stations de ski (température, ensoleillement, épaisseur de neige).

Dans 3 jours, une réunion est prévue avec une station pour présenter le rendu des courbes et statistiques dérivées des mesures. Comme il n'y a qu'une semaine de données disponibles, les résultats à montrer sont assez limités. Afin de pouvoir montrer le rendu sur plusieurs semaines tout en ayant des valeurs relativement plausibles, on demande à Régis de créer un générateur de données synthétiques.

Le lendemain, Régis se perd en forêt, pendant que vos collègues partent le rechercher vous êtes chargé de continuer son travail.

Son programme est disponible dans le fichier `generation_donnees.py` téléchargeable sur le site du cours. Cela peut être réalisé directement depuis le terminal avec les commandes :

```
wget https://learn.heeere.com/2018-infospichi-8a82/raw/generation_donnees.py
```

Q1) Régis a la (TRÈS) mauvaise habitude d'écrire ses programmes sans les tester, exécutez le programme et corrigez ses erreurs.

Q2) Pourquoi l'erreur ligne 55 apparaît avant l'erreur ligne 13 ?

Q3) À la ligne 15 on a `donnees[:, 0]`, `donnees[:, 1] = generer_temperatures(nb_jours)`. Est-ce que `donnees[:, :2] = generer_temperatures(nb_jours)` aurait été faux ? Pourquoi ?

Quelqu'un avait suggéré d'ajouter également une mesure de la vitesse du vent. Malheureusement, Régis n'a pas eu le temps de faire la génération de ces données.

On décide de définir la vitesse du vent comme suit:

- le vent n'est pas influencé par le cycle jour/nuit.
- le vent est constant sur des périodes de quelques heures.

Q4) Complétez la fonction `generer_vent` en vous aidant des commentaires à l'intérieur.

Exercice 2 – Tracés des données

Rappel : pour cet exercice, travaillez dans un dossier `ex2` lui même dans le dossier `tp6`.

Q5) Créez un fichier `plot_donnees.py`, dans lequel vous allez générer un mois de données aléatoires.

Pour créer un groupe de graphiques on peut utiliser la méthode `add_subplot` de `plt.Figure`:
`ax = fig.add_subplot(lignes, colonnes, id_subplot):`

```
1 fig = plt.figure()
2 ax = fig.add_subplot(1, 2, 1)
3 ax.plot(tableau1)
4 ax = fig.add_subplot(1, 2, 2)
5 ax.plot(tableau2)
6 plt.show()
```

Ce code donnera 2 graphiques sur un même ligne, le premier contiendra le tracé de `tableau1`, le deuxième celui de `tableau2`.

Q6) Tracez sur deux lignes et deux colonnes, un graphiques contenant les courbes de températures, un graphique contenant la courbe d'ensoleillement, un graphique contenant les courbes d'épaisseur de neige en `mètre`, et un graphique contenant la courbe de la vitesse du vent en `km/h`.

Q7) Comme dans le tp5, n'oubliez pas les légendes. `plt.legend` est disponible pour chaque "subplot" avec `ax.legend`.

Q8) Utilisez `ax.set_title(titre)` pour définir un titre explicite aux différents graphiques.

On peut modifier les valeurs affichées sur les axes de la façon suivante:

```
1 tableau = generation_donnees.generer_donnees(30)
2 fig = plt.figure()
3 ax = fig.add_subplot(1, 1, 1)
4 ax.plot(tableau[:, 0])
5
6 ax.set_xticks([0, 15*24, 30*24])
7 ax.set_xticklabels(["a", "b", "c"])
8 plt.show()
```

`ax.set_xticks` définit les indices des valeurs à afficher sur l'axe des abscisses, dans l'exemple, seront affichées 3 valeurs, une au début, une au milieu, et une à la fin de l'axe des abscisses. `ax.set_xticklabels` définit les valeurs à afficher sur l'axe des abscisses, dans l'exemple, seront affichées "a" au début, "b" au milieu, et "c" à la fin de l'axe des abscisses.

Note: utilisez `ax.set_yticks` et `ax.set_yticklabels` pour changer l'axe des ordonnées.

Q9) Sur chaque graphique, affichez sur l'axe des abscisses le début de chaque jour. Les jours seront numérotés de 1 à 30.

Les méthodes de la question précédente ne changent que l'affichage de l'axe. pour définir les limites réelles de des axes on utilise `ax.set_ylim(bottom, top)`

```
1 tableau = generation_donnees.generer_donnees(30)
2 fig = plt.figure()
3 ax = fig.add_subplot(1, 1, 1)
4 ax.plot(tableau[:, 0])
5
```

```
6 ax.set_ylim(0, 5)
7 plt.show()
```

Dans cet exemple l'axe des ordonnées commencera à 0 et finira à 5, les valeurs inférieures à 0 et celle supérieures à 5 ne seront donc pas affichées.

Note: utilisez `ax.set_xlim` pour changer l'axe des ordonnées.

Q10) Vous connaissez les valeurs maximales et minimales de chaque mesures, utilisez `ax.set_ylim` de sorte que les 4 graphiques aient chacun un axe des ordonnées constant (sans perdre des données).

Pour créer des diagrammes en “camembert” on utilise la méthode `ax.pie`:

```
1 tableau = generation_donnees.generer_donnees(30)
2 fig = plt.figure()
3 ax = fig.add_subplot(1, 1, 1)
4 ax.pie([50, 15, 25, 10], labels=["50%", "15%", "25%", "10%"])
5 plt.plot
```

Le premier tableau contient les proportion de chaque part, labels sert à légènder les parts.

Q11) Dans un nouveau graphique, utilisez `ax.pie` pour afficher les pourcentages d'heures où la moyenne des épaisseurs de neige est supérieure à 170cm incluse, entre 169cm et 140cm incluses, et inférieure à 139cm incluse.

Aide: Vous pouvez utiliser `np.bincount(tableau)` qui permet de compter le nombre d'occurrences de chaque valeur d'un tableau d'entiers. `np.bincount([5,2,5])` par exemple renvoie `[0,0,1,0,0,2]`, aucun 0, aucun 1, un 2, aucun 3, aucun 4, deux 5 (s'arrête à la valeur maximale).

Q12) Créez un groupe de graphiques de 1 ligne et 3 colonnes.

Q13) En utilisant la fonction `ax.bar(x, y)`, dans le 1er graphique, tracez les températures minimales au soleil mesurées à 0h, 1h , ..., 23h, dans le 2ème, les températures moyennes, et dans le 3ème, les températures maximales.

Documentation `ax.bar`: `ax.bar(left, height, width)`

- `left` : tableau de valeur, coordonnées x du côté gauche des barres
- `height` : tableau de valeur, hauteur des barres
- `width` : valeur, largeur des barres

Q14) Grâce au détail des arguments de `ax.bar`, refaites la question précédente sur un seul graphique, de sorte qu'à chaque heure soient affichées 3 barres: la première pour la température minimale, la deuxième pour la température moyenne, et la troisième pour la valeur maximale