UNIVERSITÉ JEAN MONNET SAINT-ÉTIENNE

Info. TP5: Bactéries et premiers plots

(au moment du 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 tp5.zip tp5/
```

Important : mise en place

- Suivre les consignes des TP précédents, et travailler dans un dossier tp5.

Exercice 1 – Simulation de Bactéries

Rappel : pour cet exercice, travaillez dans un dossier `ex1` lui même dans le dossier `tp5`.

But de l'exercice : en partant d'un programme réalisant une simulation (totalement aléatoire), vous allez simuler l'évolution d'une colonie de bactéries. Une touche permet d'injecter des bactéries dans notre simulation.

Les questions vous font ajouter, étape par étape, différents éléments de la simulation. Il faut lancer votre programme à chaque étape pour valider que vous avez bien fait cette étape.

Q1) Ouvrez et comprenez le programme contenu dans `bacteries.py`. Aide 0 : il faut bien lire le programme. Aide 1 : lancez le programme et essayer les différentes touches pour comprendre comme il fonctionne. Aide 2 : dans le programme vous verrez qu'une fonction peut renvoyer plusieurs valeurs en renvoyant un n-uplet (triplet ici), et que l'on peut aussi/alors « déconstruire » un n-uplet pour récupérer les valeurs.

Q2) Faites que le substrat soit constant au fil du temps (au lieu d'être tiré aléatoirement à chaque pas de temps), c'est à dire : dans la fonction de calcul de l'état suivant (`etat_suivant`), la valeur renvoyé pour le substrat doit être la même que celle reçue.

Q3) On va faire que les bactéries se diffusent (se déplacent dans le voisinage). À chaque pas de simulation (c'est à dire dans la fonction `etat_suivant`), faites qu'une variable `flou` soit remplie en appliquant la fonction `diffuser` au tableau courant des bactéries. Pour l'instant, faites que l'état suivant des bactéries soit exactement la valeur de `flou`. Comme pour toutes les questions, pensez à tester.

Q4) La fonction `diffuser` n'est pas complète : elle ne diffuse que verticalement. Comprenez et modifiez la fonction `diffuser` pour qu'elle diffuse aussi selon l'axe horizontal. Aide : il vous faut extraire des blocs qui font tout sauf une ligne (la première ou la dernière). Pensez à diviser par 4 pour obtenir une moyenne, puisque vous ajoutez maintenant 4 blocs.

Q5) En utilisant la touche 2 pour bien voir, observez que la fonction `diffuser` fait « partir » l'ensemble des bactéries d'une case vers les cases voisines. Il est donc incorrect dans notre simulation de remplacer l'état des bactéries par la variable `flou`. Nous allons donc faire que seulement 20 pourcents des bactéries se déplacent à chaque fois. Définissez donc une variable `diff = 0.2` et faites qu'en chaque position, la nouvelle valeur de bactérie soit calculée avec la formule $diff \times flou + (1 - diff) \times bact$, où $bact$ est la quantité de bactérie à l'état précédent.

Q6) On va nourrir nos bactéries et faire qu'elles se multiplient. Changez l'initialisation du substrat pour qu'il y aie de la nourriture de partout (c'est à dire des 1 partout).

Q7) Dans l'évolution du système, faites une nouvelle variable `miam` qui contiendra, pour chaque position, le produit entre la quantité de bactérie (après diffusion) à cette position et la quantité de substrat à cette position, le tout multiplié par 0.1 (un facteur qui représente la voracité des bactéries). Ajouter la valeur calculée au nombre de bactéries (les bactéries ont mangé et ont créé de nouvelles bactéries.

Q8) Pour l'instant les bactéries se multiplient trop vite. Il y a deux raisons : elles ne meurent jamais et le substrat est infiniment disponible. Pour les sous-questions A et B ci-dessous, on va soustraire des valeurs, il faut donc utiliser `np.clip(....)` pour s'assurer que le résultat n'est jamais négatif (par exemple le forcer à être en 0 et 10). A) Faites que le substrat soit diminué de la valeur de `miam` à chaque pas de simulation. B) Faites que le nombre de bactéries (après reproduction) soit diminué d'une constante valant 0.0025.

Q9) On va créer un substrat avec une forme particulière. À l'initialisation de la simulation, « coupez le substrat en deux zones» en mettant à 0 les lignes 10 à 14, puis « recréez un lien entre les zones » en mettant à 1 les colonnes 10 à 12 (tout inclus).

Q10) On va faire que la « voracité » des bactéries soit non pas une constante (0.1) mais dépende de la quantité de « catalyseur » à la position donnée. Faites que la variable d'état catalyseur soit remplie avec la valeur 0.1 partout et utilisez ce tableau au lieu du 0.1 dans le calcul de `miam`. On s'attend à ce que le comportement du programme reste le même.

Q11) On va vérifier qu'une variation de quantité de catalyseur change la dynamique des bactéries. Initialisez à 0.3 la zone supérieure (le lignes jusqu'à la 10).

Q12) On va essayer de sur-vitaminer nos bactéries. Initialisez à 1 la colonne 11 du catalyseur.

Q13) On va maintenant charger les données initiales à partir d'un fichier. Nous pourrions charger les 3 tableaux à partir de fichiers mais nous allons nous contenter des valeurs de catalyseur. Commencez par faire que le substrat ne soit plus découpé en deux zones (mettre des 1 partout). Utilisez la fonction numpy `genfromtxt` pour charger le fichier `test1.csv` (valeurs séparées par des virgules) qui servira de valeur initiale pour `catalyseur`.

Exercice 2 – Courbes simples.

Rappel : pour cet exercice, travaillez dans un dossier `ex2` lui même dans le dossier `tp5`.

'Q15) Exécutez `simple_plot.py`, ouvrez le fichier et commentez la ligne `plt.figure...` puis ré-exécutez le. Est-ce qu'il fonctionne encore ? Pourquoi ?

'Q16) Dans `simple_plot.py`, transformez `rand_tab` en tableau à 2 dimensions contenant 15 lignes, tracez sa courbe. Dans le graphe produit par matplotlib, à quoi correspondent les dimensions du tableau passé en paramètre ?

'Q17) Dans `simple_plot.py`, créez un tableau `tab` à une dimension contenant des valeurs entières allant de 100 à 115 exclu, tracez la courbe `plot(tab, rand_tab)`. Qu'est-ce qui a changé ? Pourquoi ?

'Q18) Inversez `tab` et `rand_tab` dans un nouveau tracé. Qu'est-ce qui change ? Pourquoi ?

Q19) Exécutez `op_plot.py`. Ajoutez une légende cohérente et qui ne doit pas chevaucher les courbes. Aide : cherchez la documentation de `plt.legend` pour comprendre le paramètre `loc`, ou essayez de changer la valeur de `loc` pour voir ce qu'il se passe.

Q20) Tracez les courbes `y=x` et `y=x²` dans l'intervalle [-25;25]. Faites attention à avoir les bonnes valeurs affichées sur les axes et que les courbes soient suffisamment "lissées" (et n'oubliez pas la légende qui ne doit pas chevaucher les courbes).

Q21) Tracez les courbes sinusoid et cosinus dans l'intervalle [-25;25] (avec la même attention que pour la courbe précédente).

Bonus: trouvez un moyen de mettre une légende sans chevaucher les courbes. La documentation peut (ou pas) être utile : https://matplotlib.org/api/pyplot_api.html#matplotlib.pyplot.legend

Exercice 3 – Application télésiège #niveau 1#.

Rappel : pour cet exercice, travaillez dans un dossier `ex3` lui même dans le dossier `tp5`.

Bluecime est une entreprise grenobloise qui propose un système pour s'assurer de la sécurité des passagers d'un télésiège. Ce système est composé d'une caméra filmant l'embarquement d'un télésiège et d'un ordinateur branché à la caméra qui effectue les traitements nécessaires afin de détecter si un problème survient à l'embarquement.

Afin d'améliorer ses ventes, l'entreprise veut essayer de fournir plus de services afin d'augmenter la valeur du système avec un minimum d'ingénierie. Une idée serait de proposer des outils statistiques supplémentaires et automatiques aux gérants de station.

Une première expérimentation est lancée sur un télésiège afin d'essayer de fournir des statistiques météorologique: Une sonde placée sous la caméra capte la température à l'ombre, une autre sur la caméra capte la température au soleil. Une troisième sonde mesure l'ensoleillement. Trois jauges sont placées à différents endroits du champ de vision de la caméra afin de pouvoir mesurer l'épaisseur de la neige.

On a expérimenté ce nouveau système pendant une semaine. Chacune des données a été mesurée toutes les heures et vous sont fournies dans un fichier csv `telesiege_mesures.csv` On vous demande ensuite de produire des courbes de statistique, n'oubliez pas de fournir des graphiques LISIBLES et légendées correctement (de façon cohérente et sans chevauchement).

Ouvrez le fichier pour prendre connaissance du format.

Q22) Créez un fichier `appli_plot.py`, à l'aide de la fonction `genfromtxt` de `numpy`, chargez le fichier csv dans un tableau numpy et tracez les données (on rappelle, le plus lisiblement possible).

Q23) Tracez, sur un même graphique, la plus grande épaisseur de neige, la plus petite, et la moyenne des 3 jauges mesurées au cours de la semaine.

Q24) Tracez la moyenne de l'épaisseur de la neige mesurée au cours de la semaine dans les 3 jauges, en mètre.

Q25) Tracez, sur un même graphique, les moyennes quotidiennes des températures à l'ombre et celles au soleil.

Q26) Tracez, sur un même graphique, l'évolution de la luminosité observée chacun des jours de l'expérience.

Le `Wh/m²` est une unité de mesure de l'ensoleillement très utilisée dans l'industrie, `1Wh = 3600J`.

Q27) Tracez, sur un même graphique, les mesures d'ensoleillement effectuées les différentes **journées** de l'expérience `en kWh/m²`.