Mémo Python

1 Structures

1.1 Boucles et définition de fonction

Test if:

```
if test :
   commandes
elif test :
   commandes
else :
   commandes
```

Boucle while:

```
while test :
   commandes
```

Boucle for:

```
for i in liste :
   commandes
```

La liste sera éventuellement donnée par `range(a,b)` *i.e.* la liste des *entiers* $\{a, a+1, ..., b-1\}$.

Fonction:

```
def fonction(arg_1, ..., arg_n) :
   commandes
   return resultat
```

2 Numpy

Rappel : pour utiliser le module Numpy, il faut d'abord l'importer par la commande

```
import numpy as np
```

Une fonction de la librairie Numpy sera appelée par la commande `np.fonction`

Pour l'algèbre linéaire numérique, Numpy contient un sous-module particulier `linalg`. Une fonction du sous-module `linalg` sera appelée par la commande `np.linalg.fonction`

2.1 Matrices

Une matrice $A = (a_{ij})_{1\leq i\leq n, 1\leq j\leq p}$ est définie par

```
A = np.array([[a_11, a_12,...,a_1p], ..., [a_n1, a_n2,...,a_np]])
```

Si A et B sont de mêmes tailles, `A + B`, `A - B` renvoient la somme $A+B$, la différence $A-B$.

Fonction Numpy	Résultat
`np.zeros((n,p))`	Renvoie la matrice nulle de taille (n,p)
`np.ones((n,p))`	Renvoie la matrice de taille (n,p) où toutes les entrées sont égales à 1
`np.eye(n)`	Renvoie la matrice identité de taille (n,n)
`A@B`	Produit matriciel AB
`np.dot(A,B)`	Produit matriciel AB
`np.linalg.solve(A,b)`	Retourne la solution x du système $Ax = b$
`np.tranpose(A)`	Transposée de A
`np.linalg.eigvals(A)`	Valeurs propres de A
`np.linalg.eig(A)`	Vecteurs propres de A
`np.linalg.det(A)`	Déterminant de A
`np.linalg.inv(A)`	Inverse de A

Warning

Pour deux tableaux Numpy `A,B`, la commande `A*B` effectue le produit *terme à terme* et retourne le tableau de coordonnées $(a_{ij}b_{ij})$.

3 Graphiques

Pour tracer des graphiques, il faut utiliser le sous-module `matplotlib.pyplot` que l'on importe par :

```
import matplotlib.pyplot as plt
```

Pour tracer la courbe des points $(x_k, y_k)_k$, le script sera

```
plt.plot(X, Y)
plt.show()
```

où `X` contient les abscisses $(x_k)_k$ et `Y` contient les ordonnées $(y_k)_k$.
Il existe des options pour tracer en échelle log ou log-log une courbe

Fonction	Résultat
`plt.semilogx(X,Y)`	Trace la courbe passant par les points $(x_k, y_k)_k$
`plt.ylabel('Y')`	Nomme l'axe des ordonnées `Y`
`plt.legend(loc='best')`	Affiche une légende

Pour tracer 2 courbes sur un même graphique, on tapera:

```
plt.plot(X1, Y1)
plt.plot(X2, Y2)
plt.show()
```

Il existe différentes options à `plt.plot` pour personnaliser les courbes : `plt.plot(X1, Y1, 'r^')` affiche des triangles rouges. On peut remplacer `r` par une autre lettre pour d'autres couleurs (`g`, `b`, `y`,...) et `^` par un autre symbole pour d'autres formes (`o`, `s`, `+`,...).

Commandes supplémentaires à ajouter avant `plt.show()` :

Fonction	Résultat
`plt.xlabel('X')`	Nomme l'axe des abscisses `X`
`plt.ylabel('Y')`	Nomme l'axe des ordonnées `Y`
`plt.legend(loc='best')`	Affiche une légende

Pour la légende, il faut spécifier un nom pour chaque courbe par l'option `plt.plot(X1, Y1, 'r^', label='nom')`.