Notes de calcul scientifique en Python

Alban Gossard

Les imports suivants sont utilisés:

```
import numpy as np
import matplotlib.pyplot as plt
```

1 Structures de données pour le calcul scientifique

1.1 Vecteur - Array

Un array de taille 5 rempli de 0:

```
a=np.zeros(5)
```

Un array de taille 5 rempli de 2:

```
a=np.ones(5)*2
```

1.2 Matrice

Note: Eviter autant que possible d'utiliser des listes d'arrays. Toujours réfléchir à la taille de vos éléments et déclarer la taille dès l'initialisation.
Une matrice de taille 5×6 remplie de 0:

```
a=np.zeros((5,6))
```

2 Calcul vectoriel

2.1 Additionner des vecteurs

En Python on évite autant que possible de remplir les éléments d'un vecteur un par un, on procède par bloc quand cela est possible.

```
# A EVITER
n=42
# generation de vecteurs de nombre aleatoires
a=np.random.randn(n)
b=np.random.randn(n)
c=np.random.randn(n)
for i in range(n):
    c[i]=a[i]+b[i]
```

```
# PLUTOT FAIRE
n=42
a=np.random.randn(n)
b=np.random.randn(n)
c=a+b
```

2.2 Générer un ensemble de valeurs

Des entiers de 0 à $n-1$:

```
a=np.arange(42)
```

9 nombres uniformément espacés de 2 à 10:

```
a=np.linspace(2,10,9)
```

9 nombres: 10^{-8}, 10^{-7} jusqu'à 10^{0}:

```
a=np.logspace(-8,0,9)
```

ou alors:

```
k=np.arange(-8,1)
a=10**k
```

$n \geq 1$ nombres discrétisant l'intervalle $[a, b]$ avec $a < b$ tels que: $x_j = a + (b-a)\frac{j}{n} \qquad 1 \leq j \leq n$

```
a=-8; b=4
n=42
k=np.arange(n+1)
x=a+(b-a)*k/n
# ou: x=np.linspace(a,b,n)
```

2.3 Produit scalaire de vecteurs

Calculer le produit scalaire entre deux vecteurs:

```
a=np.random.randn(4)
b=np.random.randn(4)
print(np.dot(a,b))
```

2.4 Produit matrice-vecteur

```
A=np.random.randn(4,6)
b=np.random.randn(6)
print(np.dot(A,b))
```

2.5 Affectation

Affecter les éléments qui appartiennent aux 2 premières colonnes et 3 premières lignes d'un array en 2D:

```
a=np.zeros((5,6))
a[:2,:3]=1
```

3 Affichage avec matplotlib

Afficher en échelle log les axes x et y:

```
a=np.linspace(0,1e2,100)
b=np.exp(-a)
plt.loglog(a,b)
plt.show()
```

Afficher en échelle log l'axe y seulement:

```
a=np.linspace(0,1e2,100)
b=np.exp(-a)
plt.semilogy(a,b)
plt.show()
```

Afficher un nuage de points:

```
a=np.random.uniform(-1,1,size=(10,2))
plt.scatter(a[:,0],a[:,1])
plt.show()
```

4 Résolution d'un système linéaire

Résoudre $Ax = b$ avec A une matrice réelle symétrique définie positive. Pour définir une matrice A qui soit réelle symétrique définie positive nous partons d'une matrice aléatoire B: $A = \frac{B+B^T}{2} + 10^{-3}\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}$

```
B=np.random.randn(3,3)
A=(B+B.T)/2+1e-3*np.eye(3) # pour que la matrice soit symetrique definie positive
b=np.random.randn(3)
x=np.linalg.solve(A,b)
print("error=",np.linalg.norm(A.dot(x)-b))
```