Module `numpy`. Calcul matriciel

On pourra se reporter au formulaire de l'oral du concours Agro-Veto

```python
import numpy as np
import numpy.linalg as la
```

Construction de matrices et manipulation

■ Exercice 1.

```python
# Exemple de saisie d'une matrice M en donnant la liste de ses lignes
M = np.array([[1,2,3],
              [4,5,6]])
# Obtention de son format
M.shape
```

1. Créez une matrice A de taille $(3,4)$ avec des éléments de votre choix. Vérifiez ses format en utilisant la méthode `shape`.
2. Transformez la matrice A en une matrice de taille $(2,6)$ en utilisant la méthode `reshape`, puis en une matrice $(6,2)$. Quelle contrainte doit respecter le produit $n \times p$ pour effectuer cette opération ?
3. Créez une matrice B de taille $(5,2)$. Transformez-la en une matrice de taille $(2,5)$, puis en une matrice $(10,1)$. Affichez les format à chaque étape.
4. Créez une matrice uniligne (de taille $(1,5)$) et une matrice unicolonne (de taille $(5,1)$). Affichez leurs format et vérifiez que vous obtenez les formats attendus avec `shape`.
5. Créez une matrice C de taille $(6,3)$. Utilisez `reshape` pour la transformer en une matrice $(3,6)$, puis en une matrice $(2,9)$. Vérifiez les format avec `shape`.

M. Patel
2024-2025
Classes de B_2, Rennes

Construction de matrices avec des formules

■ Exercice 2.

1. Complétez, affichez A et corrigez si besoin.

```
# Construction de la matrice de taille 3x4 de coeffs a_ij = i+2j
A = np.zeros((3,4))
for i in range(3):
    for j in range(4):
        A[i,j] =
```

2. Créez une matrice B de taille $(3,3)$ telle que l'élément en position (i,j) soit $b_{i,j} = i + j$. Affichez cette matrice. Affichez cette matrice. Vérifiez les valeurs des éléments.
3. Construisez une matrice C de taille $(4,4)$ telle que l'élément en position (i,j) soit $c_{i,j} = i \times j$. Affichez cette matrice. Vérifiez les valeurs des éléments.
4. Créez une matrice D de taille $(3,4)$ telle que l'élément en position (i,j) soit donné par $d_{i,j} = i - j$. Affichez cette matrice. Vérifiez les valeurs des éléments.
5. En utilisant des boucles `for`, générez une matrice E de taille $(5,5)$ telle que chaque élément soit défini par $e_{i,j} = i^2 + j^2$.
6. Construisez une matrice F de taille $(4,3)$ telle que chaque élément soit défini par $f_{i,j} = (-1)^{i+j} \times (i+j)$. Affichez cette matrice et vérifiez sa construction.
7. Reprendre les exemples précédents en utilisant la commande `np.array` et en construisant la liste des lignes en compréhension.

Extraction de coefficients, lignes et colonnes

■ Exercice 3.

1. Soit A une matrice de taille $(4,5)$ avec des éléments de votre choix. Écrivez le code pour extraire :
   — le coefficient de la première ligne et deuxième colonne ;
   — la première ligne de A ;
   — la dernière colonne de A.
2. Affichez la sous-matrice formée par les lignes 1 à 3 et les colonnes 2 à 4.
3. Créez une matrice B de taille $(5,5)$. Extrayez les éléments diagonaux, puis tous les éléments situés au-dessus de la diagonale principale.
4. Soit C une matrice de taille $(6,4)$. Extrayez les lignes 2 et 4, puis les colonnes 1 et 3.
5. Construisez une matrice D de taille $(7,7)$. Affichez la sous-matrice formée par les lignes et colonnes impaires.

Matrices remarquables

■ Exercice 4.

1. Utilisez `np.zeros((n, p))` et `np.ones((n, p))` pour créer respectivement une matrice nulle et une matrice remplie de 1 et de taille $(3,3)$.
2. Créez la matrice identité de taille $(4,4)$ avec `np.eye`.
3. Créez une matrice E de taille $(5,5)$ ayant des 1 sur le contour et des 0 à l'intérieur, en utilisant des combinaisons de `zeros` et `ones`.
4. Construisez une matrice diagonale de taille $(4,4)$ avec des valeurs de 2 sur la diagonale en utilisant `np.eye`.
5. Créez une matrice F de taille $(6,6)$ avec des 1 sur la diagonale principale et -1 sur la première diagonale au-dessus et en dessous de la diagonale principale.

Produit matriciel

■ Exercice 5.

1. **a)** Si A B sont deux matrices de formats compatibles, et $C = AB$, rappeler l'expression du coefficient général $c_{i,j}$ de la matrice C.

   b) Implémentez une fonction `mult(A, B)` pour le produit de deux matrices A et B formats compatibles, sans utiliser les fonctions de produit de `numpy`.
2. Testez votre fonction avec deux matrices de format $(2,3)$ et $(3,2)$, puis vérifiez avec `np.dot`.
3. Multipliez la matrice A par elle-même pour calculer A^2, puis A^3. Comparez les résultats avec `np.dot`.
4. Créez deux matrices aléatoires de votre choix de format $(3,3)$. Calculez leur produit avec `mult` et `np.dot`.
5. Écrivez une fonction qui calcule le produit de trois matrices $(A \times B \times C)$ et vérifiez sa compatibilité pour des matrices de format $(2,2)$, $(2,3)$, $(3,2)$.

Puissances de matrices

■ Exercice 6.

1. **a)** Implémentez une fonction `puissance(A, n)` qui renvoie la puissance n-ième d'une matrice carrée A, en multipliant successivement la matrice par elle-même.

   b) Implémentez une version récursive de votre fonction.
2. Comparez les résultats avec `la.matrix_power`.
3. Testez votre fonction sur une matrice identité de taille $(3,3)$ et vérifiez que $A^n = A$ pour $n > 1$.
4. Calculez la puissance 5 d'une matrice diagonale $(3,3)$ avec des valeurs de 2, 3, et 4 sur la diagonale.
5. Pour une matrice $B = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$, calculez B^4 et comparez avec les résultats de `matrix_power`.

Utilisation du module `numpy.linalg`

■ Exercice 7.

1. Calculez le rang des matrices suivantes :

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

2. Calculez l'inverse de $C = \begin{pmatrix} 1 & 2 & 3 \\ 0 & -1 & 2 \\ 4 & 0 & -1 \end{pmatrix}$ avec `la.inv` et vérifiez que $C \times C^{-1} = I_3$.

3. Soit $E = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 4 \\ 5 & 6 & 0 \end{pmatrix}$. Calculez le rang, les valeurs propres, et essayez de déterminer si E est diagonalisable.

4. Calculez le rang d'une matrice nulle de taille $(4, 4)$ et d'une matrice de 1 de taille $(4, 4)$.

M. Patel
2024-2025
Classes de B_2, Rennes

PYTHON
AGRO-VETO
2023

Listes

`[]`	Créer une liste vide	`L.pop(k)`	Renvoie le $k^{\text{ème}}$ élément de la liste `L` et l'enlève de `L`
`[a]*n`	Créer une liste avec n fois l'élément `a`	`L.remove(a)`	Enlève une fois la valeur `a` de la liste `L`
`L.append(a)`	Ajoute l'élément `a` à la fin de la liste `L`	`max(L)`	Renvoie le plus grand élément de la liste `L`
`L1 + L2`	Concatène les deux listes `L1` et `L2`	`min(L)`	Renvoie le plus petit élément de la liste `L`
`len(L)`	Renvoie le nombre d'éléments de la liste `L`	`sum(L)`	Renvoie la somme de tous les éléments de la liste `L`

Numpy

`import numpy as np`

`np.array()`	Transforme une liste en matrice `numpy`
`np.linspace(a,b,n)`	Crée une matrice ligne de n valeurs uniformément réparties entre a et b (inclus)
`np.zeros([n,m])`	Crée la matrice nulle de taille $n \times m$
`np.eye(n)`	Crée la matrice identité de taille n
`np.diag(L)`	Crée la matrice diagonale dont les termes diagonaux sont les éléments de la liste `L`
`np.transpose(M)`	Renvoie la transposée de M
`np.dot(M,P)`	Renvoie le produit matriciel MP
`np.sum(M)`	Renvoie la somme de tous les éléments de M
`np.prod(M)`	Renvoie le produit de tous les éléments de M
`np.max(M)`	Renvoie le plus grand élément de M
`np.min(M)`	Renvoie le plus petit élément de M
`np.shape(M)`	Renvoie dans un couple le format de la matrice M
`np.size(M)`	Renvoie le nombre d'éléments de M

Numpy.linalg

`import numpy.linalg as la`

`la.inv(M)`	Renvoie l'inverse de la matrice M si elle est inversible
`la.eigvals(M)`	Renvoie la liste des valeurs propres de M
`la.eig(M)`	Renvoie un couple `L,P` où `L` est la liste des valeurs propres de M et `P` la matrice de passage associée
`la.matrix_rank(M)`	Renvoie le rang de M

Random

`import random as rd`

`rd.random()`	Simule une réalisation d'une variable $X \hookrightarrow \mathcal{U}([0,1])$
`rd.randint(a,b)`	Simule une réalisation d'une variable $X \hookrightarrow \mathcal{U}([\![a,b]\!])$
`rd.gauss(0,1)`	Simule une réalisation d'une variable $X \hookrightarrow \mathcal{N}(0,1)$
`rd.choice(L)`	Choisit aléatoirement un élément de la liste `L`

Math

`import math as m`

`m.atan(x)`	Renvoie $\arctan(x)$	`m.sqrt(x)`	Renvoie $\sqrt{x}$ si $x \geq 0$
`m.floor(x)`	Renvoie $\lfloor x \rfloor$	`m.log(x)`	Renvoie $\ln(x)$ si $x > 0$
`m.factorial(n)`	Renvoie $n!$ si $n \in \mathbb{N}$	`m.exp(x)`	Renvoie e^x

Logique

`a == b`	Teste l'égalité « $a = b$ »
`a != b`	Teste « $a \neq b$ »
`a < b`	Teste « $a < b$ »
`a <= b`	Teste « $a \leq b$ »
`a > b`	Teste « $a > b$ »
`a => b`	Teste « $a \geq b$ »
`not A`	Renvoie la négation de A
`A and B`	Renvoie « A et B »
`A or B`	Renvoie « A ou B »
`True`	Constante booléenne « Vrai »
`False`	Constante booléenne « Faux »

Matplotlib.pyplot

`import matplotlib.pyplot as plt`

`plt.plot(X,Y,'+-r')`	Génère la courbe des points définis par les listes `X` et `Y` (abscisses et ordonnées) avec les options : • symbole : `'.'` point, `'o'` rond, `'h'` hexagone, `'+'` plus, `'x'` croix, `'*'` étoile, … • ligne : `'-'` trait plein, `'--'` pointillé, `'-.'` alterné, … • couleur : `'b'` **bleu** , `'r'` **rouge** , `'g'` **vert** , `'c'` **cyan** , `'m'` **magenta** , `'k'` **noir** , …
`plt.bar(X,Y)`	Génère l'histogramme des points définis par les listes `X` et `Y` (abscisses et ordonnées)
`plt.axis('equal')`	Rend le repère orthonormé
`plt.xlim(xmin,xmax)`	Fixe les bornes de l'axe des abscisses
`plt.ylim(ymin,ymax)`	Fixe les bornes de l'axe des ordonnées
`plt.show()`	Affiche le graphique

Cette liste est non exhaustive. Les candidats sont libres d'utiliser les commandes de leur choix.