Une matrice peut être représentée par une liste, un array ou une matrix.
Les types array (ou ndarray) et matrix sont définis dans numpy.

<u>Différences entre liste et array</u>

On peut modifier la taille d'une liste (pop, del, append, insert), mais pas d'un array (il faut en faire une copie dans un array de taille différente).

Les listes peuvent contenir des éléments de types différents, pas les array.

En contrepartie, les opérations sur les arrays sont souvent plus rapides, ils prennent moins de place en mémoire que les listes, et permettent d'effectuer des calculs que ne permettent pas les listes.

<u>Type matrix</u>

Une matrix est un array à deux dimensions pour lequel on a redéfini les opérations matricielles. Par exemple, A*B effectue :

- une multiplication terme à terme si A et B sont du type array,
- une multiplication matricielle si A et B sont du type matrix.

<u>Remarque</u> : np.dot(A,B) renvoie le produit scalaire de A et B si A et B sont deux arrays de dimension 1 et le produit matriciel de A et B si A et B sont deux arrays de dimension 2.
np.mat(A)*np.mat(B) aura le même effet.

<u>Quelques exemples utiles</u>

- np.zeros((4,3)) renvoie un array à 4 lignes et 3 colonnes rempli de 0 (plus précisément un array de *n* array contenant *p* zéros)
- np.diag([1,2,3]) renvoie une matrice diagonale du type array avec 1, 2, 3 sur la diagonale.

On peut préciser le type : np.zeros((3,3),int). Par défaut, il s'agit de float.

- np.array([[1,2],[3,4]]) renvoie
```
array([[ 1, 2],
       [ 3, 4]])
```
- np.matrix([[1,2],[3,4]]) renvoie une matrice similaire.
- np.matrix([1,2,3,4]).reshape((2,2)) a le même effet.

M désigne un array ou une matrix à $n \geq 2$ lignes et $p \geq 4$ colonnes.

- M[1][3] renvoie l'élément ligne 2 colonne 4 <u>si M est du type liste ou array</u>, M[1,3] fait la même chose <u>si M est du type array ou matrix</u>.
- M[1] renvoie la ligne 2, M[:,3] renvoie la colonne 4.
- M[:,:2] renvoie les deux premières colonnes.
- M.T : transposée de M (ne fonctionne pas avec les <u>array</u> à une ligne, il faut utiliser M.reshape((p,1)) dans ce cas).
- np.linalg.matrix_rank(M) : rang de M
- np.linalg.det(M) : déterminant de M
- np.poly(M) : polynôme caractéristique
- np.linalg.eigvals(M) : valeurs propres de M
- np.linalg.eig(M) : tuple ([valeurs propres],[vecteurs propres de M]), les vecteurs propres sont à lire verticalement (matrice de passage).
- np.linalg.inv(M) : inverse de M, même effet que M**(-1) si M est du type matrix
- np.linalg.solve(A,B) : résout AX = B, où A et B sont deux matrices
- np.set_printoptions(precision = 4,suppress = True) agit sur l'affichage des résultats précédents (4 décimales, et pas d'écriture scientifique pour les petites valeurs)

Activité 1

Soit $A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$.

1. Calculer A^{-1}.
2. Vérifier que A est diagonalisable, déterminer ses valeurs propres et la matrice de passage P vers la nouvelle base.
3. Calculer $P^{-1}AP$. Vérifier que l'on trouve bien le résultat attendu.

4. Déterminer le polynôme caractéristique de A, retrouver ses racines avec la fonction roots.
5. Résoudre le système AX = ${}^t(i\ 0\ 1)$.
On montre que $A^n = \frac{4^n-1}{3}A + \frac{4-4^n}{3}I_3$.
6. Ecrire un programme testant cette formule pour une centaine de valeurs de n et renvoyant un booléen indiquant si la formule est toujours vérifiée.
La formule semble-t-elle valable pour n négatif ?

Activité 2 – Oral Centrale Mathématiques 2

Pour les simulations informatiques sous Python, on importera les bibliothèques scientifiques à l'aide des instructions suivantes :

```
import numpy as np
import numpy.linalg as alg
```

Soit $(a_k)_{k\in[[1,n]]} \in \mathbb{R}^n$. On considère la matrice A($a_1,\ldots, a_n$) suivante :

$$A = \begin{pmatrix} 0 & a_1 & a_2 & \ldots & a_{n-1} & a_n \\ a_1 & 0 & a_2 & \ldots & a_{n-1} & a_n \\ a_1 & a_2 & 0 & \ldots & a_{n-1} & a_n \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_1 & a_2 & a_3 & \ldots & a_n & 0 \end{pmatrix}$$

1. On suppose dans cette question que $n = 4$.
a. Écrire une fonction Python de paramètre (a, b, c, d) qui calcule la matrice $A(a, b, c, d)$.
b. Donner les valeurs propres des matrices $A(1, 2, 3, 4)$, $A(4, 2, 3, 1)$ et $A(–3, –1, 1, 2)$.
En déduire une conjecture sur les valeurs propres de la matrice $A(a, b, c, d)$.
c. Donner les vecteurs propres des matrices $A(1, 2, 3, 4)$, $A(4, 2, 3, 1)$ et $A(–3, –1, 1, 2)$.
On pourra faire des rapports des coordonnées d'un vecteur pour l'identifier.
Que peut-on dire de la matrice dans ces trois cas ?

Les questions mathématiques suivantes ne sont pas l'objet de notre TD d'informatique.

2. *Montrer les conjectures précédentes dans le cas général.*
3. On suppose que les réels $(a_i)_{i\in[[1,n]]}$ *sont strictement positifs et que* $\sum_{i=1}^n a_i = 1$. *Que peut-on dire de la suite* $(A(a_1, \ldots a_n)^m)_{m\in\mathbb{N}}$ *?*

Activité 3 – Calcul intégral et valeurs propres

Rappel : les fonctions permettant le calcul approché d'intégrale sont quad, trapz, simps, et dblquad, tplquad pour les intégrales multiples, de la bibliothèque scipy.integrate.

On considère la matrice $A(t) = \begin{pmatrix} 1 & 2 & 3 \\ 2 & t & 4 \\ 3 & 4 & 5 \end{pmatrix}$, et la fonction f définie sur $\mathbb{R}$ par $f(t) = \mathrm{Max}(\mathrm{Sp}(A(t)))$.
1. Quel semble être le comportement de f en $-\infty$? en $+\infty$?
2. Calculer $\int_{-10}^{10} f(t)dt$, à l'aide d'un programme, puis en utilisant la bibliothèque scipy.integrate.
3. Représenter la fonction f sur [–10, 10], [0,100] …