



# Représentations vectorielles I. Intro

18 mars 2019

—Lexicologie, terminologie, dictionnairique—

## Ressources recommandées :

- <https://www.fast.ai/2017/07/17/num-lin-alg/> (opérations matricielles)
- Jurafsky and Martin chap. 6 <https://web.stanford.edu/~jurafsky/slp3/6.pdf>

## Commandes pour utiliser des matrices avec numpy

Un vecteur est une matrice de dimensions  $1 \times n$ , où une liste numpy : `a = np.array([1,2,5])`  
Le  $i$ -ième élément du vecteur : `a[i]`  
Accéder aux éléments avec indice  $\geq i$  du vecteur : `a[i :]`  
Accéder aux éléments avec indice  $< i$  du vecteur : `a[:i]`  
Une matrice (dimensions  $n \times m$ ) est une liste de listes : `b = np.array([[3,2,9],[1,5,1],[0,3,4]])`  
Accéder à la ligne  $i$  de la matrice : `b[i]` ou `b[i, :]`  
Accéder à la colonne  $j$  de la matrice : `b[:, j]`  
L'élément  $i, j$  de la matrice est celle de l' $i$ -ième ligne,  $j$ -ième colonne : `b[i, j]`  
Créer une matrice de dimensions  $5 \times 6$  avec des valeurs zéro partout : `c = np.zeros([5,6])`  
Transposée d'une matrice  $A$  :  $A_{ij}$  devient  $B_{ji}$ , soit on échange les lignes et les colonnes.



La **longueur** du vecteur (aussi appelé sa **norme l2**), où  $n$  est le nombre de ses dimensions :

$$|x|_2 = \sqrt{\sum_{i=1}^n x_i^2} \quad (1)$$

(cf. théorème de Pythagore pour un vecteur de deux dimensions).

La **norme l1** du vecteur est la somme de toutes les valeurs :

$$|x|_1 = \sum_{i=1}^n x_i \quad (2)$$

Il s'agit dans les deux cas des normes **du vecteur** ; pour obtenir le vecteur normalisé, il faut diviser le vecteur par la norme.

**Multiplication de deux vecteurs :**

avec le même nombre  $n$  de dimensions (*produit scalaire, dot product, inner product*) :

$$x \times y = x_1y_1 + x_2y_2 + \dots + x_ny_n = \sum_{i=1}^n x_iy_i \quad (3)$$

Par exemple :

```
a = np.array([1, 0, 3])
```

```
b = np.array([0, 9, 9])
```

$$a \times b = 1 \times 0 + 0 \times 9 + 3 \times 9 = 27 \quad (4)$$

Multiplication vecteur-vecteur, vecteur-matrice, matrice-matrice dans numpy :

$$c = np.dot(a, b) \quad (5)$$

ou

$$c = np.matmul(a, b) \quad (6)$$

**Multiplication matrice-vecteur :**

la multiplication d'une matrice avec un vecteur donne un vecteur, dont le  $i$ -ième élément est le produit scalaire de la ligne  $i$  de la matrice et du vecteur :



$$\begin{bmatrix} 1 & 3 \\ 4 & 0 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 \times 1 + 3 \times 5 = 16 \\ 4 \times 1 + 0 \times 5 = 4 \\ 2 \times 1 + 1 \times 5 = 7 \end{bmatrix}$$

**I. Multiplication de matrice avec un vecteur de colonne.**

La matrice a les dimensions  $m \times n$ , et le vecteur  $n \times 1$  (ou  $1 \times n$ ), donc le nombre des colonnes de la matrice = la dimensionnalité du vecteur.

$$\begin{bmatrix} 1 & 3 \\ 4 & 0 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 5 \end{bmatrix} = \begin{bmatrix} 16 \\ 4 \\ 7 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 3 \\ 4 & 0 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 5 \end{bmatrix} = \begin{bmatrix} 16 & 4 & 7 \end{bmatrix}$$

### Multiplication matrice-matrice :

produit des matrices A de taille  $m \times n$  et matrice B de taille  $n \times p$  (nombre des colonnes dans A = nb des lignes dans B) :

$$C_{ij} = \sum_{k=1}^n A_{ik} B_{kj} \quad (7)$$

donc on multiplie les lignes de A avec les colonnes de B. Intuitivement : on prend les colonnes de la matrice B un par un et on fait le même calcul que plus haut sous "multiplication de matrice avec un vecteur".

$C_{ij}$  contient donc le produit des vecteurs de *l'i-ème ligne de A et de la j-ème colonne de B*.

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 \\ 5 & 6 & 9 & 9 \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 3 & 3 \\ 9 & 2 \\ 5 & 1 \end{bmatrix} = \begin{bmatrix} 8 & 4 \\ 32 & 18 \\ 144 & 50 \end{bmatrix}$$

### Exercices I.

1. A la main : calculez le résultat.

— vecteur - vecteur :



$$\begin{bmatrix} 4 & 3 & 8 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0 & 2 & 8 & 2 & 1 & 8 \end{bmatrix} =$$

— matrice - vecteur :

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 \\ 5 & 6 & 9 & 9 \end{bmatrix} \times \begin{bmatrix} 3 \\ 7 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} \phantom{0} \\ \phantom{0} \\ \phantom{0} \end{bmatrix}$$

2. Créez une matrice A  $15 \times 10$  peuplée avec les nombres entiers naturels impairs en ordre croissant de gauche à droite et de haut en bas. (Tip : vous pouvez utiliser `np.arange`, puis `np.reshape` pour modifier les dimensions une fois que vous avez la suite des valeurs).

- Sélectionnez colonnes avec indice 2-5, dans une matrice B.
- Sélectionnez les lignes 0-4, dans une matrice C.
- Créez un vecteur z de dimension 10, avec une valeur de 1 à l'indice 4, zéro ailleurs.
- Calculez le produit scalaire de la première ligne de A avec z.
- Calculez le produit de la matrice C avec z.

3. A la main : calculez le résultat.

$$\begin{bmatrix} 3 & 5 & 2 \\ 9 & 3 & 8 \\ 1 & 6 & 3 \end{bmatrix} \times \begin{bmatrix} 6 \\ 7 \\ 8 \end{bmatrix} =$$
$$\begin{bmatrix} 4 & 8 & 1 & 2 \\ 7 & 3 & 0 & 7 \\ 5 & 1 & 5 & 5 \end{bmatrix} \times \begin{bmatrix} 5 & 5 \\ 9 & 1 \\ 2 & 2 \\ 0 & 1 \end{bmatrix} =$$

4. Créez (automatiquement) une matrice A  $10 \times 10$  peuplée avec les nombres de 0 à 99, de gauche à droite et de haut en bas.
- Sélectionnez les parties correspondant aux nombres de 60 à 99, dans une matrice B.
  - Sélectionnez les multiples de 10 dans A, dans un vecteur y.
  - Sélectionnez les lignes de 3 à 6 incluses, et les colonnes de 0 à 5 incluses, dans une matrice C.
  - Créez un vecteur z avec les multiples de 3 entre 21 et 48 inclus.
  - Calculez le produit scalaire des multiples de 10 (A) et des multiples de 3.
  - Pour chaque ligne 60..69, 70..79 jusqu'à 99, calculez le produit scalaire du vecteur correspondant avec le vecteur z.
5. <https://github.com/fastai/numerical-linear-algebra/blob/master/README.md>
- Le premier exercice sous "Matrix and Tensor Products" (AIDS)
  - Le deuxième exercice sous "Matrix and Tensor Products" (shopping)

### Exercices TAL. Vecteurs de mots

1. Nous allons créer des représentations vectorielles des mots à partir de co-occurrences dans un corpus. On va créer une matrice M dans lequel chaque ligne correspond à un mot  $w$ , et chaque colonne correspond à un mot de contexte  $c$ . La liste des mots  $w$  et  $c$ , identiques, se trouve dans le fichier words.lst.

$M_{i,j}$  est donné par le nombre de co-occurrence du mot  $w_i$  avec le mot  $c_j$  dans une fenêtre de 5 mots. La co-occurrence d'un mot  $w$  avec lui-même n'est pas définie, seulement les mots du contexte sont pris en compte.

Complétez le programme vector1.py selon les instructions pour créer M.

2. Utiliser les indices  $i, j$  des mots  $w_i$  et  $c_j$  dans M, et la liste *word*, pour faire le lien entre la matrice M et les mots du vocabulaire. Ecrivez une fonction qui demande à l'utilisateur de rentrer deux mots, et renvoie le nombre de co-occurrences des deux mots. Évitez l'exception si le mot n'est pas dans la matrice.

3. La norme l1 du vecteur est la somme de toutes les valeurs :

$$|x|_1 = \sum_{i=1}^n x_i \quad (8)$$

où  $n$  est le nombre des dimensions. Diviser les valeurs du vecteur de co-occurrences brutes par la norme l1 donne les valeurs de 'fréquence relative', qui peuvent être interprétées comme une distribution de probabilité, parce qu'elles somment à 1.

La norme l2 du vecteur est sa longueur :

$$|x|_2 = \sqrt{\sum_{i=1}^n x_i^2} \quad (9)$$

Il s'agit dans les deux cas des normes **du vecteur** ; pour obtenir le vecteur normalisé, il faut diviser les valeurs de chaque dimension par la norme.

Normalisez la matrice **par colonnes**, selon la norme l2. Utilisez `np.linalg.norm`.

4. Les mots sont représentés comme des vecteurs. Nous cherchons les mots les plus similaires. Nous utiliserons la similarité cosinus entre deux vecteurs. Elle est comprise entre -1 (vecteurs opposés) et 1 (similaires).

$$\cos(x, y) = \frac{\sum_{i=1}^N x_i y_i}{\sqrt{\sum_{i=1}^N (x_i)^2} \sqrt{\sum_{i=1}^N (y_i)^2}} \quad (10)$$

Ecrivez une fonction qui prend deux mots en entrée, et renvoie la similarité cosinus entre leurs vecteurs. Testez-la avec des mots sur la liste `topfreq`.

Utilisez `np.linalg.norm` pour la norme des deux vecteurs.