Décompositions matricielles

Pascal Romon

Décomposition LU

Qu'est-ce que c'est ?

Décomposition de la matrice $\mathbf{M}$:

$$\mathbf{M} = \mathbf{LU}$$

avec :

- $\mathbf{M}$, $\mathbf{L}$ et $\mathbf{U}$: 3 matrices $n \times n$
- $\mathbf{L}$: matrice unitaire triangulaire inférieure (*Lower*)
- $\mathbf{U}$: matrice triangulaire supérieure (*Upper*)

Décomposition LU

Qu'est-ce que c'est ?

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & l_{32} & 1 & 0 \\ l_{41} & l_{42} & l_{43} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix}$$

On remarquera que non seulement la matrice de gauche est triangulaire inférieure, mais en plus elle n'a que des 1 sur sa diagonale.

Décomposition LU

Principe :

On fait comme un pivot de Gauss, mais

- on effectue les opérations de pivot sous forme de produits de matrices élémentaires
- on mémorise les matrices correspondant aux opérations *inverses*
- on garde le produit de ces matrices

Pourquoi l'inverse ? Parce que si le pivot est $M \mapsto M' = EM$, alors $M = E^{-1}M'$.

Matrices élémentaires

Rappel : une opération sur les lignes d'une matrice (de système linéaire) revient à multiplier à gauche par une matrice élémentaire $\mathbf{E}$.

- permutation des lignes i et j : comme $\mathbf{Id}$ sauf que $e_{ii} = e_{jj} = 0$ et $e_{ij} = e_{ji} = 1$
- $L_j \mathrel{+}= cL_i$: comme $\mathbf{Id}$ avec c en position (j, i) ; comme on suppose $i < j$, $\mathbf{E}_{L_j+=cL_i}$ est *triangulaire inférieure*
- $L_i \mathrel{*}= c$: comme $\mathbf{Id}$ avec c en position (i, i)

Noter que l'opération de soustraction ne génère que avec une diagonale de 1.

$$\mathbf{E}_{L_2 \leftrightarrow L_3} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \quad \mathbf{E}_{L_2 += 7L_1} = \begin{pmatrix} 1 & 0 & 0 \\ 7 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{E}_{L_2 *= -3} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Décomposition LU

Exemple : $\mathbf{M} = \begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix}$

$$\begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix} \begin{array}{c} L_2' = L_2 - 2L_1 \\ L_3' = L_3 + L_1 \\ \longrightarrow \\ \longleftarrow \\ L_2 = L_2' + 2L_1 \\ L_3 = L_3' - L_1 \end{array} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix}$$

Décomposition LU

Exemple :

$$\begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix} \begin{array}{c} L_2' = L_2 - 2L_1 \\ L_3' = L_3 + L_1 \\ \rightarrow \\ \leftarrow \\ L_2 = L_2' + 2L_1 \\ L_3 = L_3' - L_1 \end{array} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix}$$

Notez qu'on a intégré les deux opérations dans une seule matrice.

Décomposition LU

Exemple :

$$\begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \color{red}{\begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix}}$$

$$\begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix} \xrightarrow{L_3' = L_3 - 3L_2} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 0 & -2 \end{bmatrix}$$

Décomposition LU

Exemple :

$$\begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix} \begin{matrix} L_3' = L_3 - 3L_2 \\ \rightarrow \\ \leftarrow \\ L_3 = L_3' {\color{red}+3} L_2 \end{matrix} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 0 & -2 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & {\color{red}3} & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 0 & -2 \end{bmatrix}$$

Décomposition LU

Exemple :

$$\mathbf{M} = \begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix}$$

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

$$= \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 3 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 0 & -2 \end{bmatrix} = \mathbf{LU}$$

Méthode

Algorithm 1: Décomposition LU

input: une matrice $\mathbf{M}_{n\times n}$ à décomposer

$\mathbf{L} = \mathbf{Id}$, $\mathbf{U} = \mathbf{M}$

foreach *pivot* i *de* 1 *à* $n-1$ **do**

- $\widehat{\mathbf{L}} = \mathbf{Id}$
- **foreach** *ligne* j *de* $i+1$ *à* n **do**
  - $\widehat{\mathbf{L}}_{j,i} = -\frac{\mathbf{U}_{j,i}}{\mathbf{U}_{i,i}}$
- $\mathbf{U} = \widehat{\mathbf{L}}\mathbf{U}$
- $\mathbf{L} = \mathbf{L}\widehat{\mathbf{L}}^{-1}$ ($\widehat{\mathbf{L}}^{-1}$ facile à calculer)

return $\mathbf{L}$ et $\mathbf{U}$ ($\mathbf{U}$ n'est triangulaire supérieur qu'à la fin)

À chaque étape on garde $\mathbf{LU} = \mathbf{M}$.

Illustration de l'algorithme

$$\mathbf{M} = \begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix}$$

❶ $\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $\mathbf{U} = \begin{bmatrix} 2 & 1 & -1 \\ 4 & 6 & 1 \\ -2 & 11 & 8 \end{bmatrix}$

❷ $\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$ $\mathbf{U} = \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 12 & 7 \end{bmatrix}$

❸ $\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 3 & 1 \end{bmatrix}$ $\mathbf{U} = \begin{bmatrix} 2 & 1 & -1 \\ 0 & 4 & 3 \\ 0 & 0 & -2 \end{bmatrix}$

Décomposition LU

Propriétés :

- $\det(\mathbf{M}) = \det(\mathbf{L}) \times \det(\mathbf{U}) = \det(\mathbf{U})$
- la décomposition LU n'existe pas toujours, même si $\mathbf{M}$ est régulière, par ex. $\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$ (inversement, elle peut exister avec $\mathbf{M}$ singulière)
- méthode de *pivot partiel* : existence si $\mathbf{M}$ est régulière et qu'on s'autorise à permuter les lignes (ou les colonnes) de $\mathbf{M}$ (décomposition appelée LUP ou PLU)
- méthode de *pivot total* : on augmente la stabilité numérique en prenant le plus grand pivot (en valeur absolue) par permutation ligne *et* colonnes ; cela permet de traiter les matrices singulières
- si $\mathbf{M}$ est régulière, la décomposition $\mathbf{M} = \mathbf{LU}$ est unique, quand $\mathbf{L}$ n'a que des 1 sur sa diagonale

Application

Résolution de système linéaire :

$$\mathbf{Mx} = \mathbf{b}$$

- $\mathbf{LUx} = \mathbf{b}$
- on résout : $\mathbf{Ly} = \mathbf{b}$ (élimination)
- on résout : $\mathbf{Ux} = \mathbf{y}$ (élimination)

Application

Inverser une matrice :

$$\mathbf{M}\mathbf{M}^{-1} = \mathbf{Id}$$

- décomposition : $\mathbf{M} = \mathbf{L}\mathbf{U}$
- on résout n systèmes : $\mathbf{L}\mathbf{U}(\mathbf{M}^{-1})_j = \mathbf{e}_j$ ($\mathbf{A}_j$: j-ème colonne de $\mathbf{A}$)

$\rightarrow$ inverse utilisée par MATLAB par défaut.

Application

Réduction d'erreur :

Soit le système : $\mathbf{Ax} = \mathbf{b}$
pour lequel on trouve une solution $\mathbf{x}_0$ telle que :

$$\mathbf{Ax}_0 = \mathbf{b}_0 \simeq \mathbf{b} \qquad \text{(erreurs numériques)}$$

On a

$$\mathbf{A}(\mathbf{x}_0 + \delta\mathbf{x}) = \mathbf{b} \quad \Leftrightarrow \quad \mathbf{Ax}_0 + \mathbf{A}\delta\mathbf{x} = \mathbf{b}$$

soit

$$\mathbf{A}\delta\mathbf{x} = \mathbf{b} - \mathbf{Ax}_0$$

On résout le système et on obtient : $\mathbf{x} = \mathbf{x}_0 + \delta\mathbf{x}$
La décomposition de $\mathbf{A}$ est utilisée pour résoudre les 2 systèmes.

Application

Calcul du déterminant :

$$\underbrace{\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix}}_{\mathbf{M}} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{21} & 1 & 0 & 0 \\ l_{31} & l_{32} & 1 & 0 \\ l_{41} & l_{42} & l_{43} & 1 \end{bmatrix}}_{\mathbf{L}} \underbrace{\begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & u_{22} & u_{23} & u_{24} \\ 0 & 0 & u_{33} & u_{34} \\ 0 & 0 & 0 & u_{44} \end{bmatrix}}_{\mathbf{U}}$$

$$\det(\mathbf{M}) = \det(\mathbf{L})\det(\mathbf{U}) = \det(\mathbf{U}) = \prod_i \mathbf{U}_{i,i}$$

car $\det(\mathbf{L}) = 1$.

Si $\mathbf{M} = \mathbf{PLU}$ où $\mathbf{P}$ est une matrice de permutation, alors $\det(\mathbf{M}) = \det(\mathbf{P})\det(\mathbf{U}) = (-1)^k \det(\mathbf{U})$ où k est le nombre de transpositions (échange de lignes ou colonnes).

Application

Calcul approché des valeurs propres par algorithme LU.

Décomposition QR

Qu'est-ce que c'est ?

Décomposition de la matrice carrée $\mathbf{M}$: $\mathbf{M} = \mathbf{QR}$

avec :

- $\mathbf{M}$, $\mathbf{Q}$ et $\mathbf{R}$: 3 matrices $n \times n$
- $\mathbf{Q}$: matrice orthogonale $(\mathbf{Q}^\top \mathbf{Q} = \mathbf{Id})$
- $\mathbf{R}$: matrice triangulaire supérieure

Remarque : $\mathbf{Q}^\top \mathbf{Q} = \mathbf{Id}$ est équivalent à dire que les colonnes de $\mathbf{Q}$ forment une base orthonormée.

Décomposition QR

Qu'est-ce que c'est ?

$$\begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ q_{41} & q_{42} & q_{43} & q_{44} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ 0 & r_{22} & r_{23} & r_{24} \\ 0 & 0 & r_{33} & r_{34} \\ 0 & 0 & 0 & r_{44} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{m}_{\bullet 1} & \mathbf{m}_{\bullet 2} & \cdots & \mathbf{m}_{\bullet 4} \end{bmatrix} = \begin{bmatrix} \mathbf{q}_{\bullet 1} & \mathbf{q}_{\bullet 2} & \cdots & \mathbf{q}_{\bullet 4} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ 0 & r_{22} & r_{23} & r_{24} \\ 0 & 0 & r_{33} & r_{34} \\ 0 & 0 & 0 & r_{44} \end{bmatrix}$$

c-à-d
$$\begin{cases} \mathbf{m}_{\bullet 1} & = r_{11}\mathbf{q}_{\bullet 1} \\ \mathbf{m}_{\bullet 2} & = r_{12}\mathbf{q}_{\bullet 1} + r_{22}\mathbf{q}_{\bullet 2} \\ \mathbf{m}_{\bullet 3} & = r_{13}\mathbf{q}_{\bullet 1} + r_{23}\mathbf{q}_{\bullet 2} + r_{33}\mathbf{q}_{\bullet 3} \\ \mathbf{m}_{\bullet 4} & = r_{14}\mathbf{q}_{\bullet 1} + r_{24}\mathbf{q}_{\bullet 2} + r_{34}\mathbf{q}_{\bullet 3} + r_{44}\mathbf{q}_{\bullet 4} \end{cases}$$

Autrement dit,

$$\begin{cases} \mathsf{Vect}(\mathbf{q}_{\bullet 1}) = \mathsf{Vect}(\mathbf{m}_{\bullet 1}) \\ \mathsf{Vect}(\mathbf{q}_{\bullet 1}, \mathbf{q}_{\bullet 2}) = \mathsf{Vect}(\mathbf{m}_{\bullet 1}, \mathbf{m}_{\bullet 2}) \end{cases}$$

Décomposition QR

Souvent en optimisation on veut projeter un vecteur $\mathbf{w}$ sur l'espace perpendiculaire aux vecteurs $\mathbf{v}_1, \dots, \mathbf{v}_k$ (directions interdites, ou contraintes). Il est tentant d'écrire

$$P(\mathbf{w}) = \mathbf{w} - \sum_{i=1}^{k} \frac{\mathbf{w}^\top \mathbf{v}_i}{\mathbf{v}_i^\top \mathbf{v}_i} \mathbf{v}_i$$

mais ce n'est vrai que si les $\mathbf{v}_i$ sont orthogonaux ($\mathbf{v}_j^\top \mathbf{v}_i = \delta_{ij} \mathbf{v}_i^\top \mathbf{v}_i$).

$$\begin{aligned} \mathbf{v}_j^\top P(\mathbf{w}) &= \mathbf{v}_j^\top \mathbf{w} - \sum_{i=1}^{k} \frac{\mathbf{w}^\top \mathbf{v}_i}{\mathbf{v}_i^\top \mathbf{v}_i} \mathbf{v}_j^\top \mathbf{v}_i = \mathbf{v}_j^\top \mathbf{w} - \sum_{i=1}^{k} \frac{\mathbf{w}^\top \mathbf{v}_i}{\mathbf{v}_i^\top \mathbf{v}_i} \delta_{ij} \mathbf{v}_i^\top \mathbf{v}_i \\ &= \mathbf{v}_j^\top \mathbf{w} - \mathbf{w}^\top \mathbf{v}_j = 0 \end{aligned}$$

Pour pouvoir le faire (surtout de façon répétée), il faut remplacer $\mathrm{Vect}(\mathbf{v}_1, \dots, \mathbf{v}_k)$ par $\mathrm{Vect}(\mathbf{u}_1, \dots, \mathbf{u}_k)$ où les $\mathbf{u}_i$ sont orthogonaux (voire orthonormés).

Algorithme de Householder

On peut contruire $\mathbf{Q}$ et $\mathbf{R}$ à la main (méthode dite de Gram–Schmidt) de type pivot, mais il existe un algorithme bien plus rapide. Une matrice de Householder $\mathbf{Q} = \mathbf{Id} - 2\mathbf{v}\mathbf{v}^\top$ est une réflexion par rapport au plan perpendiculaire à $\mathbf{v}$ (supposé de norme 1). Si on pose $\mathbf{u} = \mathbf{x} \pm \|\mathbf{x}\|\mathbf{e}_1$ et $\mathbf{v} = \mathbf{u}/\|\mathbf{u}\|$, la réflexion échangera $\mathbf{x}/\|\mathbf{x}\|$ et $\pm\mathbf{e}_1$. Le choix du signe se fait en évitant d'avoir un vecteur proche de zéro (au cas où $\mathbf{x}$ est proche de $\pm\mathbf{e}_1$).
En choisissant comme $\mathbf{x}$ la première colonne de $\mathbf{M}$, on a

$$\mathbf{QM} = \left[\begin{array}{c|ccc} \pm 1 & \star & \cdots & \star \\ \hline 0 & & & \\ \vdots & & \mathbf{M}' & \\ 0 & & & \end{array}\right]$$

On applique la même procédure récursivement à $\mathbf{M}'$.

Méthode de Householder

Algorithm 2: Décomposition QR

input: une matrice $\mathbf{M}_{n\times n}$ à décomposer

$\mathbf{Q} = \mathbf{Id}$, $\mathbf{R} = \mathbf{M}$

foreach *block-colonne* $\mathbf{x} = \mathbf{R}_{*,i}$ **do**

- $\alpha = \|\mathbf{x}\|_2$
- **if** $\mathbf{x}_i < 0$ **then** $\alpha = -\alpha$
- $\mathbf{u} = \mathbf{x} - \alpha\mathbf{e}_i$
- $\mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|_2}$
- $\widehat{\mathbf{Q}} = \mathbf{Id}_n - 2\mathbf{v}\mathbf{v}^\top$
- $\mathbf{R} = \widehat{\mathbf{Q}}\mathbf{R}$
- $\mathbf{Q} = \mathbf{Q}\widehat{\mathbf{Q}}$ (on devrait avoir $\widehat{\mathbf{Q}}^{-1}$ mais $\widehat{\mathbf{Q}}^{-1} = \widehat{\mathbf{Q}}$)

return $\mathbf{Q}$ et $\mathbf{R}$

À tout moment on a $\mathbf{QR} = \mathbf{M}$

Décomposition QR

- la décomposition s'étend aux matrices rectangulaires $n \times m$ tant que $n \geq m$
- on peut aussi ajouter des permutations pour augmenter la stabilité numérique, et obtenir le rang de matrices singulières

Décomposition QR

$$\mathbf{M} = \mathbf{Q}\mathbf{R}$$

Applications :

- résoudre le système linéaire $\mathbf{Mx} = \mathbf{b}$, $\mathbf{Rx} = \mathbf{Q}^\top\mathbf{b}$.
- inverse de matrice : comme pour la décomposition LU.
- calcul du déterminant : $\det(\mathbf{M}) = \pm\det(\mathbf{R}) = \pm\prod_i \mathbf{R}_{i,i}$, car $\det(\mathbf{Q}) = \pm 1$

Cholesky

Introduction :
Permet de décomposer une matrice symétrique définie positive $\mathbf{A}$ en $\mathbf{A} = \mathbf{L}\mathbf{L}^\top$ où $\mathbf{L}$ est une matrice triangulaire inférieure.

Définie positive :
$\mathbf{A}$ est définie positive si : $\forall \mathbf{x} \neq \mathbf{0} \quad \mathbf{x}^\top \mathbf{A} \mathbf{x} > 0$

Deux fois plus rapide que LU, mais ne marche que pour les matrices symétriques définies positives.

Calcul

Calcul explicite :

$$L_{ii} = \sqrt{(A_{ii} - \sum_{k=0}^{i-1} L_{ik}^2)} \qquad \text{diagonale}$$

$$L_{ij} = \frac{1}{L_{jj}} \left(A_{ij} - \sum_{k=0}^{j-1} L_{ik} L_{jk} \right) \qquad \text{pour } i > j$$

Pour calculer L_{ij} on n'utilise que des éléments situés *à gauche* (sauf L_{jj}), on peut donc calculer ces éléments en allant de gauche à droite, colonne par colonne, ou ligne par ligne.

Diagonalisation

Introduction :

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$$

avec :

- $\mathbf{A}$: matrice $n \times n$.
- $\mathbf{P}$: matrice $n \times n$.
- $\mathbf{D}$: matrice $n \times n$ diagonale.

Diagonalisation

$$\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$$

Propriétés :

- les éléments de $\mathbf{D}$ sont les valeurs propres de $\mathbf{A}$ (éventuellement répétées).
- rang de $\mathbf{A}$: nombre de valeurs propres $\neq 0$.
- les colonnes de $\mathbf{P}$ sont les vecteurs propres de $\mathbf{A}$ associés aux valeur propres correspondance de $\mathbf{M}$.

Diagonalisation

Application :

- inverse (peu utilisée)
- Analyse en Composantes Principales (*PCA* en anglais)

Décomposition en Valeurs Singulières

Introduction : (SVD = *Singular Value Decomposition*)

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^\top \quad (= \mathbf{U}\mathbf{D}\mathbf{V}^{-1})$$

avec :

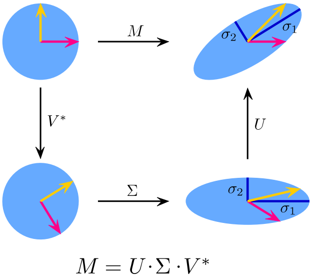
- $\mathbf{A}$: matrice $m \times n$, $m \geq n$.
- $\mathbf{U}$: matrice $m \times m$, orthogonale.
- $\mathbf{D}$: matrice diagonale rectangulaire $m \times n$, diagonale si $m = n$.
- $\mathbf{V}$ (donc $\mathbf{V}^\top$) : matrice $n \times n$ orthogonale.

Plus général que la diagonalisation : matrices rectangulaires aussi, et $\mathbf{V}$ différent de $\mathbf{U}$.

SVD

Interprétation géométrique : $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$

La transformation $\mathbf{A}$ est une rotation $\mathbf{V}^\top$, suivie de dilatations suivants les axes des coordonnées, suivie d'une nouvelle rotation.



By Georg-Johann - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=11342212

SVD

Propriétés :

- les éléments de la diagonale de $\mathbf{D}$ sont les valeurs singulières de $\mathbf{A}$.
- rang de $\mathbf{A}$: nombre de valeurs singulières $\neq 0$.
- en annulant les plus petites valeurs singulières, on obtient des matrices de rang inférieur à $\mathbf{A}$ les plus proches de $\mathbf{A}$.

$\sigma > 0$ est une *valeur singulière* s'il existe deux vecteurs non nuls $\mathbf{u}, \mathbf{v}$ (pas forcément de même dimension !) tels que $\mathbf{M}\mathbf{u} = \sigma\mathbf{v}$ et $\mathbf{M}^\top\mathbf{v} = \sigma\mathbf{u}$.

SVD

Calcul de $\mathbf{A}^{-1}$:

$$\begin{aligned}
\mathbf{A} &= \mathbf{U}\mathbf{D}\mathbf{V}^\top \\
\\
\mathbf{A}^{-1} &= (\mathbf{U}\mathbf{D}\mathbf{V}^\top)^{-1} \\
&= \mathbf{V}\mathbf{D}^{-1}\mathbf{U}^{-1} \\
&= \mathbf{V}\mathbf{D}^{-1}\mathbf{U}^\top
\end{aligned}$$

- $(\mathbf{V}^\top)^{-1} = \mathbf{V}$ car $\mathbf{V}$ est orthogonale.
- $\mathbf{U}^{-1} = \mathbf{U}^\top$ car $\mathbf{U}$ est orthogonale.
- Ici, $\mathbf{D}$ est diagonale donc facile à inverser.

Moindres carrés

Matrice $\mathbf{A}$ non carrée:

On note $\mathbf{A}^+ = (\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{A}$ la matrice pseudo-inverse de $\mathbf{A}$. Grâce à la SVD, $\mathbf{A}^+$ se calcule par:

$$\begin{aligned}\mathbf{A}^+ &= (\mathbf{UDV}^\top\mathbf{VD}^\top\mathbf{U}^\top)^{-1}\mathbf{UDV}^\top \\ &= (\mathbf{UDD}^\top\mathbf{U}^\top)^{-1}\mathbf{UDV}^\top = \mathbf{U}(\mathbf{DD}^\top)^{-1}\mathbf{U}^\top\mathbf{UDV}^\top \\ &= \mathbf{U}(\mathbf{DD}^\top)^{-1}\mathbf{DV}^\top = \mathbf{VD}^+\mathbf{U}^\top\end{aligned}$$

où $\mathbf{D}^+$ la matrice obtenue en inversant les éléments non nuls de $\mathbf{D}$.
Remarques:

- $(\mathbf{A}^+)^+ = \mathbf{A}$
- Si $m = n$ alors $\mathbf{A}^+ = \mathbf{A}^{-1}$

Null space

Propriété :

La SVD permet aussi de résoudre des systèmes d'équations du genre :

$$\mathbf{Ax} = \mathbf{0}$$

- Pivot de Gauss $\rightarrow$ solution triviale $\mathbf{x} = \mathbf{0}$
- SVD $\rightarrow$ vecteur $\mathbf{x} \neq \mathbf{0}$ tel que :
    - $\|\mathbf{x}\|_2 = 1$.
    - $\|\mathbf{Ax}\|_2$ soit minimal.
    - solution : il s'agit du vecteur propre (colonne de $\mathbf{V}$) associé à la plus petite valeur singulière de $\mathbf{D}$.

Décompositions

Bilan :

décomposition	conditions matrice	rapidité	fiabilité précision	rang
LU pivot partiel	matrice régulière	$++$	$+$	•
LU pivot total	•	$-$	$+++$	oui
QR Householder	•	$++$	$+$	•
QR pivot partiel	•	$+$	$++$	oui
QR pivot total	•	$-$	$+++$	oui
Cholesky	symétrique déf. > 0	$++++$	$+$	•
SVD	•	$-$	$++++$	oui