ML 05 Présentation de la décomposition en valeurs singulières (SVD)

1 Contexte

Nous présentons la décomposition en valeurs singulières, une méthode générique de compression d'un tableau rectangulaire de données. Elle possède de nombreuses applications. En particulier, elle permet de calculer en un seul entraînement les régressions ridge pour toutes les valeurs possibles de l'hyperparamètre α. Plus généralement, elle permet souvent d'adopter une perspective géométrique pour mieux comprendre les problèmes et algorithmes de l'apprentissage automatique à partir de données.

2 Recherche d'un sous-espace qui minimise les carrés des écarts à l'espace d'origine

Soit un tableau de données $\mathbf{X}$ composé de N observations en lignes, chacune décrite par P variables en colonnes. x_{ij} est la valeur de la variable j pour l'observation i. Les vecteurs lignes forment N points de l'espace $\mathbb{R}^P$. Les vecteurs colonnes forment P points de l'espace $\mathbb{R}^N$. Nous cherchons à projeter le nuage des points lignes sur un sous-espace $\mathcal{H} \subset \mathbb{R}^P$, tout en minimisant les déformations.

Pour commencer, nous considérons le meilleur sous-espace $\mathcal{H}$ à une dimension, c'est-à-dire une droite définie par son vecteur directeur unitaire $\mathbf{v}$ ("unitaire" signifie ici que sa norme euclidienne est égale à 1, soit $\sqrt{\mathbf{v}^T\mathbf{v}} = 1$, autrement dit, $\mathbf{v}^T\mathbf{v} = 1$). Soit M_i un des N points de $\mathbb{R}^P$. A ce point correspond le vecteur $\mathbf{OM_i}$ aussi noté $\mathbf{x_i}$ car ses coordonnées se lisent sur la i-ème ligne de $\mathbf{X}$. Soit H_i la projection de M_i sur la droite $\mathcal{H}$.

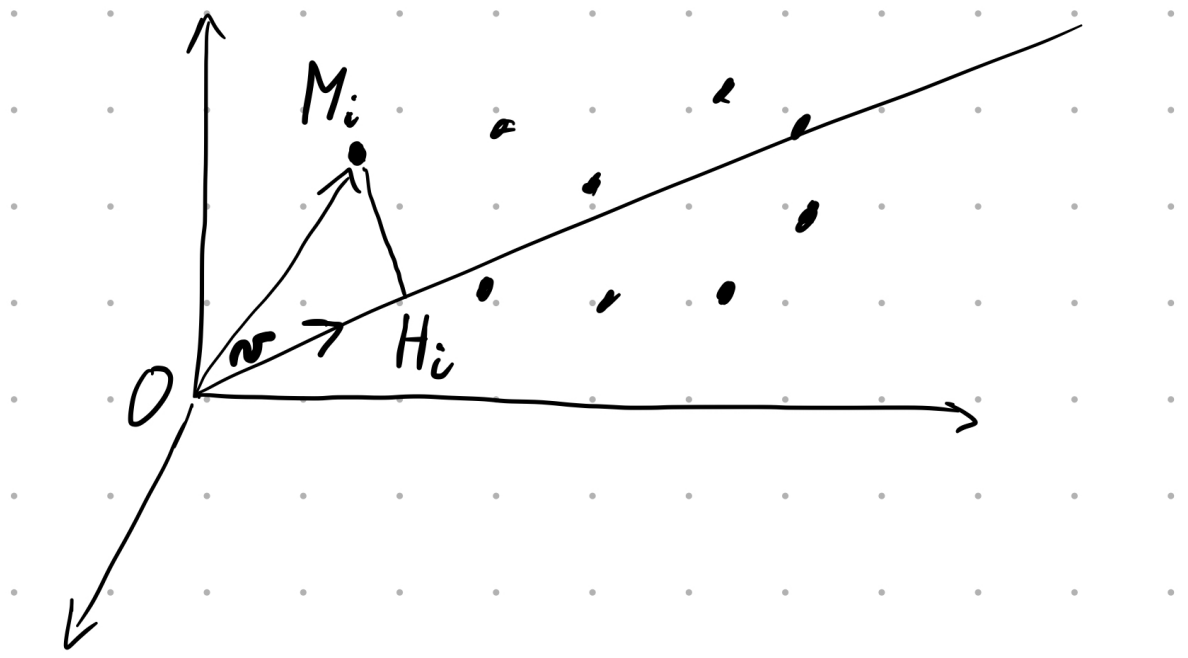


Figure 1: svd1

La longueur OH_i est le produit scalaire de $\mathbf{x_i}$ et de $\mathbf{v}$.

$$OH_i = \mathbf{x_i}^T \mathbf{v} = \sum_{j=1}^{P} x_{ij} v_j$$

Nous obtenons un vecteur des N projections OH_i par le produit de la matrice $\mathbf{X}$ et du vecteur $\mathbf{v}$: $\mathbf{Xv}$. Choisissons pour $\mathcal{H}$ le sous-espace qui minimise la somme des carrés des écarts entre chaque point et sa projection : $\sum_i M_i H_i^2$.

Le théorème de Pythagore donne $\sum_i M_i H_i^2 = \sum_i OM_i^2 - \sum_i OH_i^2$.

Puisque $\sum_i OM_i^2$ est indépendant de $\mathbf{v}$, nous devons maximiser

$$\sum_i OH_i^2 = (\mathbf{Xv})^T (\mathbf{Xv}) = \mathbf{v}^T \mathbf{X}^T \mathbf{Xv} \tag{1}$$

... sous la contrainte $\mathbf{v}^T \mathbf{v} = 1$.

Nous notons $\mathbf{v_1}$ ce vecteur directeur du meilleur sous-espace de dimension 1 pour le nuage des N observations de $\mathcal{R}^P$. Le meilleur sous-espace de dimension 2 contient $\mathbf{v_1}$ et il faut le compléter avec un second vecteur unitaire $\mathbf{v_2}$, orthogonal à $\mathbf{v_1}$ et qui maximise $\mathbf{v_2}^T \mathbf{X}^T \mathbf{X} \mathbf{v_2}$. Etc.

Nous montrerons plus loin que $\mathbf{v_1}$ est le vecteur propre de $\mathbf{X}^T\mathbf{X}$ associé à la plus grande valeur propre λ_1, cette dernière étant justement le maximum de la forme quadratique de l'équation (1). De même, $\mathbf{v_2}$ est le vecteur propre de $\mathbf{X}^T\mathbf{X}$ associé à la seconde plus grande valeur propre λ_2. Etc.

Nous venons de voir que les vecteurs orthogonaux $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_k}$ forment une base d'un sous-espace k-dimensionnel sur lequel peuvent être projetés les N vecteurs lignes de $\mathbf{X}$ pour minimiser les carrés des écarts à l'espace d'origine qui était de dimension (au plus) $max(N, P)$.

Nous construirons de même des vecteurs orthogonaux $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_k}$ qui forment une base d'un sous-espace k-dimensionnel sur lequel peuvent être projetés les P vecteurs colonnes de $\mathbf{X}$ pour minimiser les carrés des écarts à l'espace d'origine qui était de dimension (au plus) $max(N, P)$.

Nous trouverons de façon similaire que $\mathbf{u_1}$ est le vecteur propre de $\mathbf{X}\mathbf{X}^T$ associé à la plus grande valeur propre μ_1. De même, $\mathbf{u_2}$ est le vecteur propre de $\mathbf{X}\mathbf{X}^T$ associé à la seconde plus grande valeur propre μ_2. Etc.

Les vecteurs $\mathbf{u_1}, \mathbf{u_2}, \ldots$ et $\mathbf{v_1}, \mathbf{v_2}, \ldots$ sont les axes principaux de la matrice des données $\mathbf{X}$.

Nous pouvons écrire :

$$\begin{cases} \mathbf{X}^T \mathbf{X} \mathbf{v}_\alpha = \lambda_\alpha \mathbf{v}_\alpha \\ \mathbf{X}\mathbf{X}^T \mathbf{u}_\alpha = \mu_\alpha \mathbf{u}_\alpha \end{cases} \tag{2}$$

En prémultipliant la première des deux équations du système (2) par $\mathbf{X}$, nous obtenons :

$$\left(\mathbf{X}\mathbf{X}^T\right) \mathbf{X}\mathbf{v}_\alpha = \lambda_\alpha \left(\mathbf{X}\mathbf{v}_\alpha\right)$$

Cette dernière équation nous indique qu'au vecteur propre $\mathbf{v}_\alpha$ de $\mathbf{X}^T\mathbf{X}$ correspond un vecteur propre $(\mathbf{X}\mathbf{v}_\alpha)$ de $\mathbf{X}\mathbf{X}^T$ de même valeur propre λ_α. Comme μ_1 est la plus grande valeur propre de $\mathbf{X}\mathbf{X}^T$, nous avons montré que $\lambda_1 \leq \mu_1$

De même, en prémultipliant la seconde des deux équations du système (2) par $\mathbf{X}^T$, nous montrerions que $\mu_1 \leq \lambda_1$. Donc, $\lambda_1 = \mu_1$ et en général, $\lambda_\alpha = \mu_\alpha$.

Calculons la norme euclidienne, aussi appelée norme L2, de $\mathbf{X}\mathbf{v}_\alpha$.

$$
\begin{aligned}
&\text{Norme euclidienne de } \mathbf{X}\mathbf{v}_\alpha \\
&= \{ \text{ Par définition de la norme euclidienne.}\} \\
&\sqrt{(\mathbf{X}\mathbf{v}_\alpha)^T (\mathbf{X}\mathbf{v}_\alpha)} \\
&= \{\text{Propriété de la transposition}\} \\
&\sqrt{\mathbf{v}_\alpha{}^T \mathbf{X}^T \mathbf{X}\mathbf{v}_\alpha} \\
&= \{\text{Voir équation (2)}\} \\
&\sqrt{\lambda_\alpha \mathbf{v}_\alpha{}^T \mathbf{v}_\alpha} \\
&= \{\text{Par construction, } \mathbf{v}_\alpha \text{est de norme 1.}\} \\
&\sqrt{\lambda_\alpha}
\end{aligned}
$$

Nous avons montré plus haut que $(\mathbf{X}\mathbf{v}_\alpha)$ est un vecteur propre de $\mathbf{X}\mathbf{X}^T$ de valeur propre associée λ_α. Or, $\mathbf{u}_\alpha$ est le vecteur propre unitaire (i.e., de norme euclidienne égale à 1) de $(\mathbf{X}\mathbf{v}_\alpha)$ associé à la valeur propre λ_α. C'est pourquoi, nous pouvons écrire :

$$
\begin{cases}
\mathbf{u}_\alpha = \frac{1}{\sqrt{\lambda_\alpha}} \mathbf{X}\mathbf{v}_\alpha \\
\mathbf{v}_\alpha = \frac{1}{\sqrt{\lambda_\alpha}} \mathbf{X}^T \mathbf{u}_\alpha
\end{cases}
$$

A partir de l'égalité $\mathbf{X}\mathbf{v}_\alpha = \mathbf{u}_\alpha \sqrt{\lambda_\alpha}$, nous post-multiplions par $\mathbf{v}_\alpha{}^T$ et nous sommons sur l'ensemble des P axes principaux (en supposant, sans perte de généralité, que $N > P$) :

$$
\mathbf{X} \left(\sum_{\alpha=1}^{P} \mathbf{v}_\alpha \mathbf{v}_\alpha{}^T \right) = \sum_{\alpha=1}^{P} \sqrt{\lambda_\alpha} \mathbf{u}_\alpha \mathbf{v}_\alpha{}^T
$$

Soit $\mathbf{V}$ la matrice formée des P vecteurs $\mathbf{v}_\alpha$ en colonnes. Comme les vecteurs $\mathbf{v}_\alpha$ sont orthogonaux deux à deux et de norme 1, $\mathbf{V}\mathbf{V}^T$ est la matrice identité $\mathbf{I_P}$. Donc, $\sum_{\alpha=1}^{P} \mathbf{v}_\alpha \mathbf{v}_\alpha{}^T = \mathbf{V}\mathbf{V}^T = \mathbf{I_P}$. Nous obtenons finalement :

$$
\mathbf{X} = \sum_{\alpha=1}^{P} \sqrt{\lambda_\alpha} \mathbf{u}_\alpha \mathbf{v}_\alpha{}^T \tag{3}
$$

Nous avons montré que la matrice $\mathbf{X}$ peut s'écrire comme une somme de matrices de rang 1 qui sont les produits des vecteurs $\mathbf{u}$ (appelés *vecteurs singuliers à gauche*) et $\mathbf{v}$ (appelés *vecteurs singuliers à droite*) pondérés par les *valeurs singulières* $\sqrt{\lambda_\alpha}$. L'équation (3) est celle de la *décomposition en valeurs singulières* de $\mathbf{X}$.

Nous pouvons écrire ce résultat sous forme matricielle en introduisant les matrices $\mathbf{U}$, $\mathbf{V}$ et $\mathbf{D}$. Sans perte de généralité, nous exprimons les résultats pour $N > P$. La matrice $\mathbf{U}$, de dimension $N \times P$, contient en colonnes les vecteurs singuliers à gauche $\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_P}$. La matrice $\mathbf{V}$, de dimension $P \times P$ contient en colonnes les vecteurs singuliers à droite $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_P}$. La matrice diagonale $\mathbf{D}$ contient sur sa diagonale les valeurs singulières $\sqrt{\lambda_1}, \sqrt{\lambda_2}, \ldots, \sqrt{\lambda_P}$. La décomposition en valeurs singulière de toute matrice $\mathbf{X}$ s'écrit alors :

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

Notons que le rang de la matrice $\mathbf{X}$, c'est-à-dire le nombre de colonnes indépendantes (ou le nombre de lignes indépendantes, c'est équivalent), est égal au nombre de valeurs singulières non nulles.

Par ailleurs, nous obtenons une reconstitution approchée de rang k de $\mathbf{X}$ en ne conservant dans l'équation (3) que les k plus grandes valeurs singulières $\sqrt{\lambda_1}, \sqrt{\lambda_2}, \ldots, \sqrt{\lambda_k}$ et en annulant toutes les autres.

2.1 Illustrations du SVD avec la compression d'une image

Considérons l'image d'un hortensia en fleurs.

Nous convertissons cette image, originellement au format JPEG, en un format en niveaux de gris, simple à manipuler, appelé PGM (*Portable Grey Map*). Dans ce format non compressé, l'image est représentée par une matrice dont chaque valeur, comprise entre 0 et 1, représente le niveau de gris d'un pixel. Nous opérons cette transformation grâce au programme *imagemagick* avec la commande `convert hortensia.jpg hortensia.pgm`.

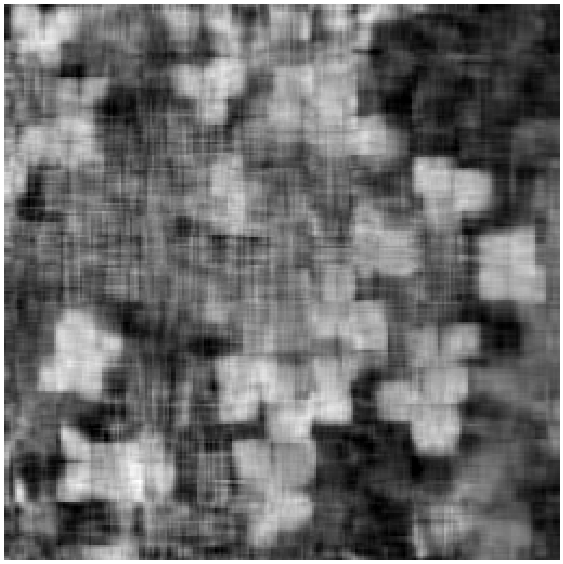
Ensuite, nous appliquons la décomposition en valeurs singulières à la matrice de l'image en niveaux de gris. Nous utilisons le résultat du SVD pour reconstituer une version compressée de l'image en ne conservant qu'un certain nombre des plus grandes valeurs singulières.

```
X <- read_grey_img("images/hortensia.pgm")

par(mfrow=c(1,2), oma=c(1,1,0,0)+0.1, mar=c(0,0,1,1)+0.1);
print_grey_img(X, main="image originale d=256", asp=1);
print_grey_img(compress_SVD(X,16), main="d=16", asp=1);
```

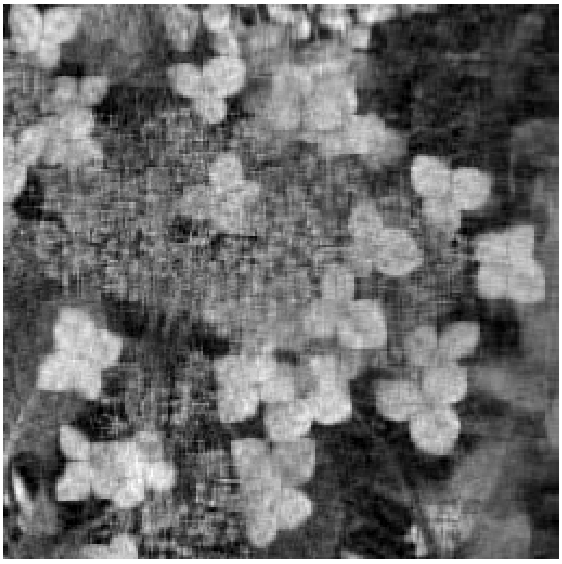
image originale d=256

d=16



```
print_grey_img(compress_SVD(X,32), main="d=32", asp=1);
print_grey_img(compress_SVD(X,64), main="d=64", asp=1);
```

d=32



d=64

```
print_grey_img(compress_SVD(X,128), main="d=128", asp=1);
print_grey_img(compress_SVD(X,256), main="d=256", asp=1);
```

d=128

d=256