

# A2DI: Pré-Traitements et Non Supervisé

John Klein

Université de Lille - CRISTAL UMR CNRS 9189



Où en est-on dans notre problème d'apprentissage supervisé ?

- On sait que si  $n$ , la taille des données est suffisamment grand par rapport à la taille de  $\theta$  alors  $Err_{train}$  ne dévie pas de  $Err_{esp}$ .

Où en est-on dans notre problème d'apprentissage supervisé ?

- On sait que si  $n$ , la taille des données est suffisamment grand par rapport à la taille de  $\theta$  alors  $Err_{train}$  ne dévie pas de  $Err_{esp}$ .
- On sait que des décisions optimales se prennent si on a accès à  $p_{Y|X}$ .

Où en est-on dans notre problème d'apprentissage supervisé ?

- On sait que si  $n$ , la taille des données est suffisamment grand par rapport à la taille de  $\theta$  alors  $Err_{train}$  ne dévie pas de  $Err_{esp}$ .
- On sait que des décisions optimales se prennent si on a accès à  $p_{Y|X}$ .
- On a vu des modèles permettant d'estimer  $p_{Y|X}$  ou  $p_{X,Y}$ .

Où en est-on dans notre problème d'apprentissage supervisé ?

- On sait que si  $n$ , la taille des données est suffisamment grand par rapport à la taille de  $\theta$  alors  $Err_{train}$  ne dévie pas de  $Err_{esp}$ .
- On sait que des décisions optimales se prennent si on a accès à  $p_{Y|X}$ .
- On a vu des modèles permettant d'estimer  $p_{Y|X}$  ou  $p_{X,Y}$ .

La plupart du temps, il est plus efficace d'apprendre ces modèles à partir de vecteurs  $z^{(i)}$  obtenus à partir des exemples  $x^{(i)}$ .

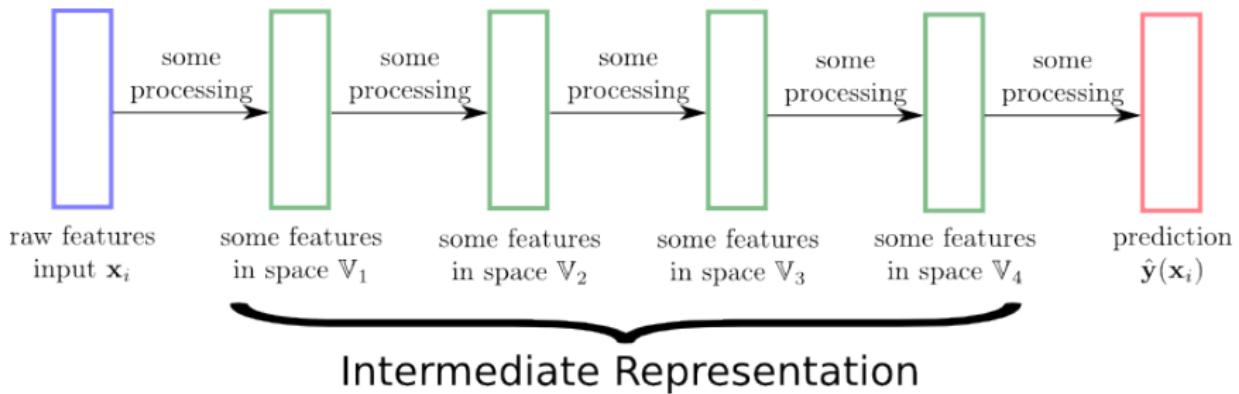
Où en est-on dans notre problème d'apprentissage supervisé ?

- On sait que si  $n$ , la taille des données est suffisamment grand par rapport à la taille de  $\theta$  alors  $Err_{train}$  ne dévie pas de  $Err_{esp}$ .
- On sait que des décisions optimales se prennent si on a accès à  $p_{Y|X}$ .
- On a vu des modèles permettant d'estimer  $p_{Y|X}$  ou  $p_{X,Y}$ .

La plupart du temps, il est plus efficace d'apprendre ces modèles à partir de vecteurs  $z^{(i)}$  obtenus à partir des exemples  $x^{(i)}$ .

Nous allons voir différentes façons de calculer ces vecteurs  $z^{(i)}$  dans ce chapitre.

## Pré-traitements : plusieurs niveaux possibles



# Plan du chapitre

- 1 Pré-traitements basiques
- 2 Réduction de dimension par apprentissage de représentation
- 3 Sélection d'attributs
- 4 Apprentissage non supervisé
- 5 Conclusions
- 6 Annexes

## Pré-traitements : **centrage** des données

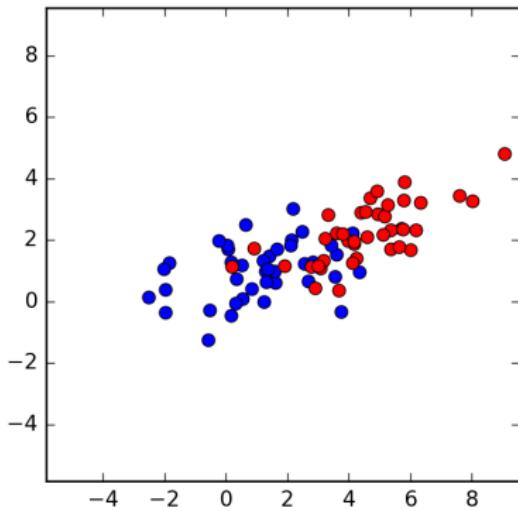
Il s'agit de calculer une représentation intermédiaire selon :

$$\mathbf{z}^{(i)} = \mathbf{x}^{(i)} - \boldsymbol{\mu} \quad \text{avec} \quad \boldsymbol{\mu} = \frac{1}{n} \sum_{i'=1}^n \mathbf{x}^{(i')}.$$
 (1)

## Pré-traitements : **centrage** des données

Il s'agit de calculer une représentation intermédiaire selon :

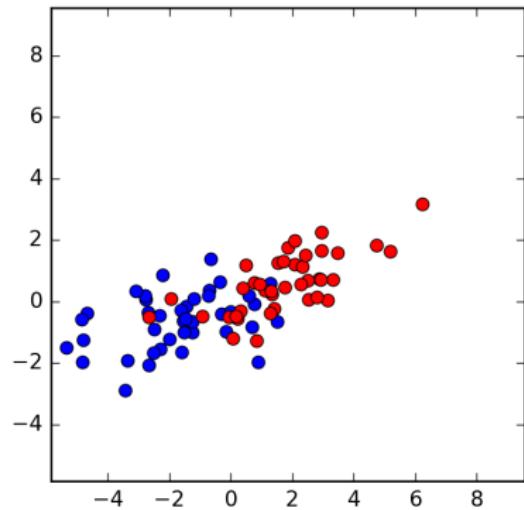
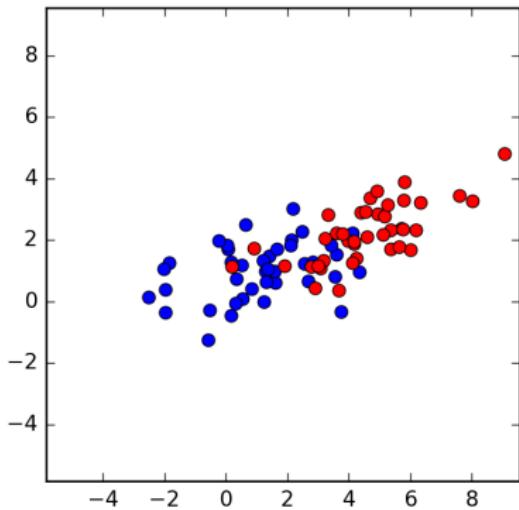
$$\mathbf{z}^{(i)} = \mathbf{x}^{(i)} - \boldsymbol{\mu} \quad \text{avec} \quad \boldsymbol{\mu} = \frac{1}{n} \sum_{i'=1}^n \mathbf{x}^{(i')}.$$
 (1)



## Pré-traitements : **centrage** des données

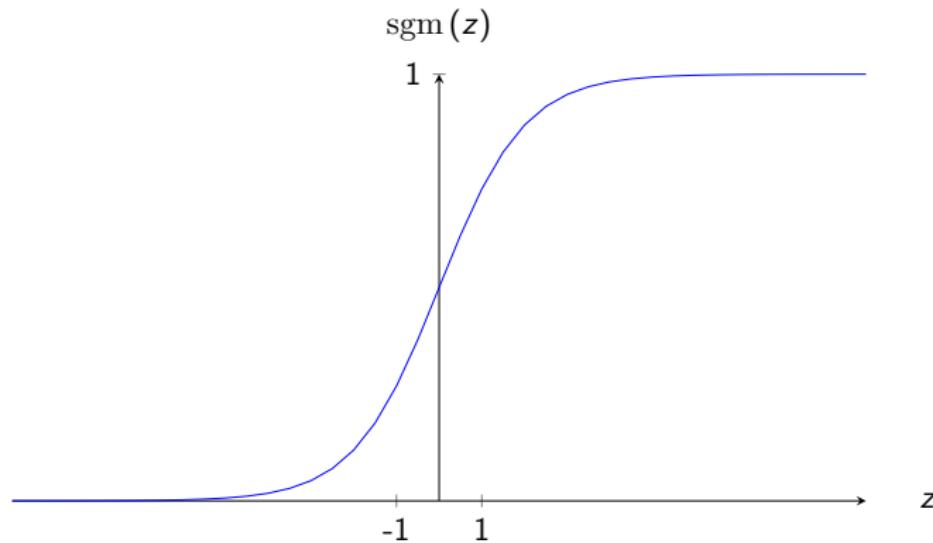
Il s'agit de calculer une représentation intermédiaire selon :

$$\mathbf{z}^{(i)} = \mathbf{x}^{(i)} - \boldsymbol{\mu} \quad \text{avec} \quad \boldsymbol{\mu} = \frac{1}{n} \sum_{i'=1}^n \mathbf{x}^{(i')} \quad (1)$$



## Pré-traitements : **centrage** des données

**Utilité** : ramener les données dans une gamme où les algorithmes ne satureront pas.



## Pré-traitements : centrage + réduction

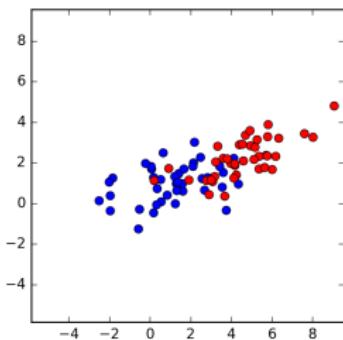
Il s'agit de calculer une représentation intermédiaire selon :

$$z_j^{(i)} = \frac{1}{\hat{\sigma}_j} (x_j^{(i)} - \mu_j) \quad \text{avec} \quad \hat{\sigma}_j = \sqrt{\frac{1}{n} \sum_{i'=1}^n (x_j^{(i')} - \mu_j)^2}. \quad (2)$$

## Pré-traitements : centrage + réduction

Il s'agit de calculer une représentation intermédiaire selon :

$$z_j^{(i)} = \frac{1}{\hat{\sigma}_j} (x_j^{(i)} - \mu_j) \quad \text{avec} \quad \hat{\sigma}_j = \sqrt{\frac{1}{n} \sum_{i'=1}^n (x_j^{(i')} - \mu_j)^2}. \quad (2)$$

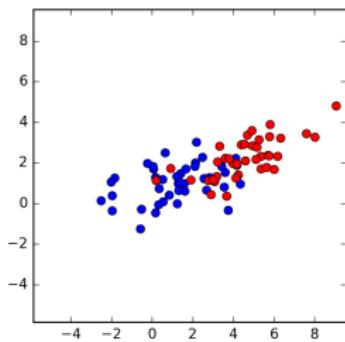


(départ)

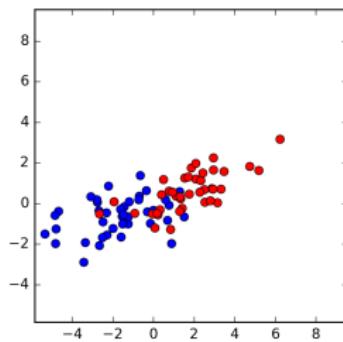
Pré-traitements : **centrage** + **réduction**

Il s'agit de calculer une représentation intermédiaire selon :

$$z_j^{(i)} = \frac{1}{\hat{\sigma}_j} (x_j^{(i)} - \mu_j) \quad \text{avec} \quad \hat{\sigma}_j = \sqrt{\frac{1}{n} \sum_{i'=1}^n (x_j^{(i')} - \mu_j)^2}. \quad (2)$$



(départ)

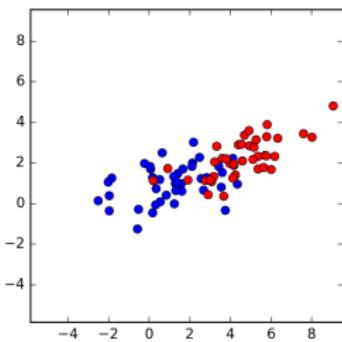


(centré)

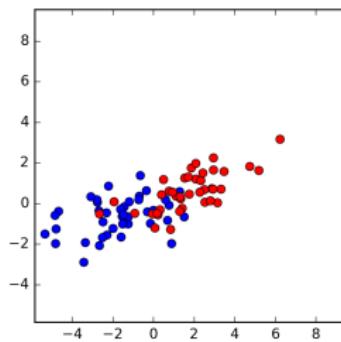
## Pré-traitements : centrage + réduction

Il s'agit de calculer une représentation intermédiaire selon :

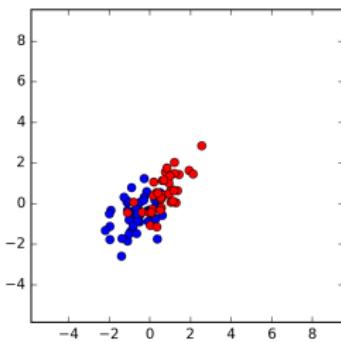
$$z_j^{(i)} = \frac{1}{\hat{\sigma}_j} \left( x_j^{(i)} - \mu_j \right) \quad \text{avec} \quad \hat{\sigma}_j = \sqrt{\frac{1}{n} \sum_{i'=1}^n \left( x_j^{(i')} - \mu_j \right)^2}. \quad (2)$$



(départ)



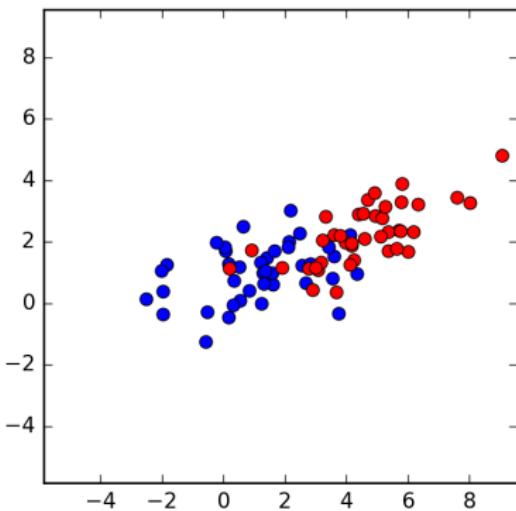
(centré)



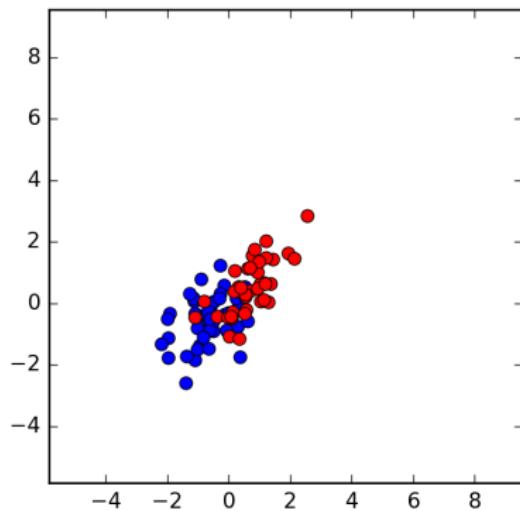
(centré réduit)

## Pré-traitements : centrage + réduction

Utilité : éviter qu'une dimension « prenne toute la place ». (Exemple avec k-PPV)



(départ)



(centré réduit)

## Pré-traitements : centrage + blanchiment (ZCA)

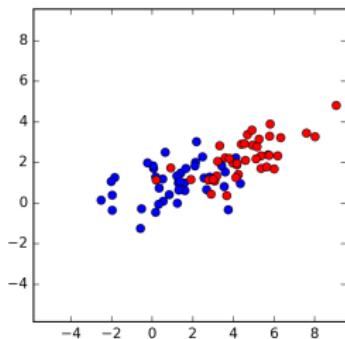
En supposant les  $\mathbf{x}^{(i)}$  centrés, on calcule la représentation intermédiaire selon :

$$\mathbf{z}^{(i)} = \mathbf{W}^T \cdot \mathbf{x}^{(i)} \quad \text{avec} \quad \mathbf{W} = \text{Chol} \left( \left( \mathbf{X} \cdot \mathbf{X}^T \right)^{-1} \right). \quad (3)$$

Pré-traitements : **centrage** + **blanchiment** (ZCA)

En supposant les  $\mathbf{x}^{(i)}$  centrés, on calcule la représentation intermédiaire selon :

$$\mathbf{z}^{(i)} = \mathbf{W}^T \cdot \mathbf{x}^{(i)} \quad \text{avec} \quad \mathbf{W} = \text{Chol} \left( (\mathbf{X} \cdot \mathbf{X}^T)^{-1} \right). \quad (3)$$

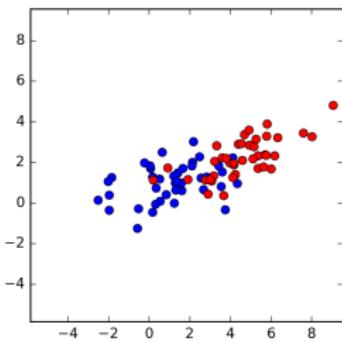


(départ)

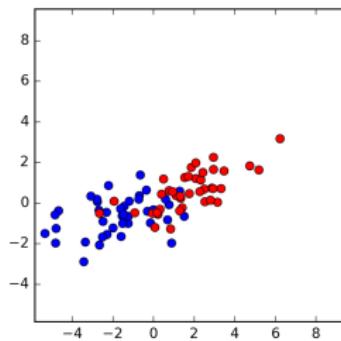
Pré-traitements : **centrage** + **blanchiment** (ZCA)

En supposant les  $\mathbf{x}^{(i)}$  centrés, on calcule la représentation intermédiaire selon :

$$\mathbf{z}^{(i)} = \mathbf{W}^T \cdot \mathbf{x}^{(i)} \quad \text{avec} \quad \mathbf{W} = \text{Chol} \left( (\mathbf{X} \cdot \mathbf{X}^T)^{-1} \right). \quad (3)$$



(départ)

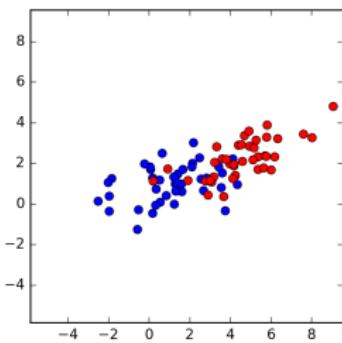


(centré)

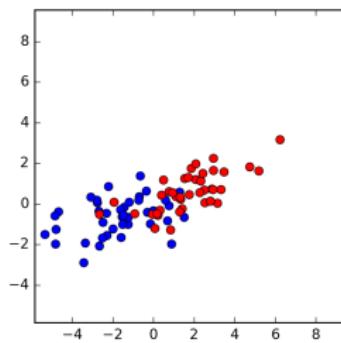
## Pré-traitements : centrage + blanchiment (ZCA)

En supposant les  $\mathbf{x}^{(i)}$  centrés, on calcule la représentation intermédiaire selon :

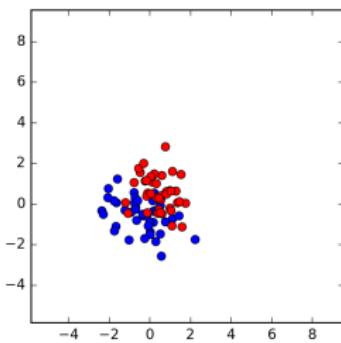
$$\mathbf{z}^{(i)} = \mathbf{W}^T \cdot \mathbf{x}^{(i)} \quad \text{avec} \quad \mathbf{W} = \text{Chol} \left( (\mathbf{X} \cdot \mathbf{X}^T)^{-1} \right). \quad (3)$$



(départ)



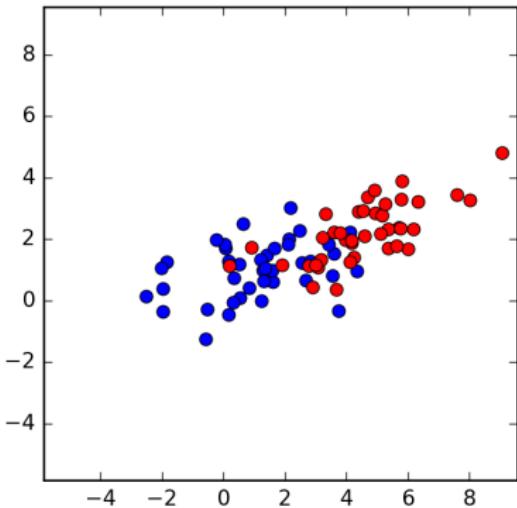
(centré)



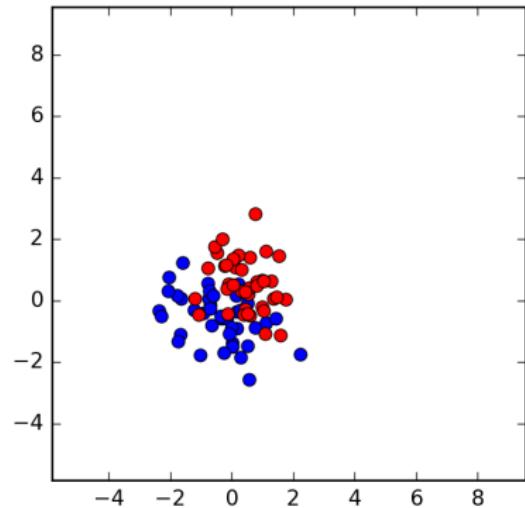
(centré blanchi)

## Pré-traitements : centrage + blanchiment (ZCA)

Utilité : décorrélérer les dimensions.



(départ)



(centré blanchi)

## Pré-traitements : Encodage one-hot

**Utilité** : utiliser des algorithmes acceptant uniquement des *inputs* continues (ex : nnets) sur des données catégoriques.

### Exemple

On s'intéresse à un problème d'approbation de crédit. Les exemples d'apprentissages sont composés des entrées suivantes :

$$\mathbf{x} = [ \text{ salaire } \text{ endettement } \text{ situation familiale } \text{ nbr enfant } ] \quad (4)$$

## Pré-traitements : Encodage one-hot

Principe : rajouter autant d'entrées binaires que de catégories possibles.

### Exemple

On s'intéresse à un problème d'approbation de crédit. Les exemples d'apprentissages sont composés des entrées suivantes :

$$\mathbf{x} = [ \text{ salaire } \text{ endettement } \text{ situation familiale } \text{ nbr enfant } ] \quad (5)$$

Après encodage one-hot, on a :

$$\mathbf{x} = [ \text{ salaire } \text{ endettement } \text{ célib. } \text{ marié } \text{ union libre } \text{ nbr enfant } ] \quad (6)$$

## Pré-traitements : Encodage des données manquantes

### Exemple (continué)

On s'intéresse à un problème d'approbation de crédit. Les exemples d'apprentissages sont composés des entrées suivantes :

$$\mathbf{x} = [ \text{ salaire } \text{ endettement } \text{ situation familiale } \text{ nbr enfant } ] \quad (7)$$

Dans mon dataset, la première entrée de l'exemple  $i$  est **inconnue** :

$$\mathbf{x}^{(i)} = [ \text{ ?? } \text{ 50000 } \text{ c\'elib. } \text{ 0 } ] \quad (8)$$

## Pré-traitements : Encodage des données manquantes

**Solution** : utiliser une variable binaire pour préciser si la donnée est là ou pas.

### Exemple (continué)

Dans mon dataset, la première entrée de l'exemple  $i$  est **inconnue** :

$$\mathbf{x}^{(i)} = [ \text{ ?? } \quad 50000 \quad \text{célib.} \quad 0 ] \quad (9)$$

Elle devient donc :

$$\mathbf{x}^{(i)} = [ \text{ 0 } \quad \text{ 0 } \quad 50000 \quad \text{célib.} \quad 0 ] \quad (10)$$

## Pré-traitements : Encodage des données manquantes

**Solution** : utiliser une variable binaire pour préciser si la donnée est là ou pas.

### Exemple (continué)

Dans mon dataset, la première entrée de l'exemple  $i$  est connue :

$$\mathbf{x}^{(i)} = [ \text{ 2000 } \quad 50000 \quad \text{célib.} \quad 0 ] \quad (11)$$

Elle devient donc :

$$\mathbf{x}^{(i)} = [ \text{ 2000 } \quad 1 \quad 50000 \quad \text{célib.} \quad 0 ] \quad (12)$$

## Pré-traitements : cas particulier des **signaux**

**signal** = fonction du temps, de  
l'espace ou des deux.

**exemple** : parole, musique, image,  
vidéos..

## Pré-traitements : cas particulier des signaux

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

signal = fonction du temps, de l'espace ou des deux.

exemple : parole, musique, image, vidéos..

Filtres d'images



# Plan du chapitre

- 1 Pré-traitements basiques
- 2 Réduction de dimension par apprentissage de représentation
- 3 Sélection d'attributs
- 4 Apprentissage non supervisé
- 5 Conclusions
- 6 Annexes

## Réduction de dimensions : but

- On a parlé dans le 1<sup>er</sup> chapitre du phénomène de la **malédiction de la dimension** !
- Ce phénomène caractérise le fait que le problème d'apprentissage devient beaucoup plus **difficile** quand  $\text{dim}(\mathbb{X})$  croît.
- Cependant, on sait que l'information contenue dans la dimension  $j$  est potentiellement partiellement **répétée** dans une autre dimension  $j'$ .
- On cherche donc un moyen d'éliminer les dimensions redondantes et l'**analyse en composantes principales** (ACP) en est un.

## Réduction de dimensions : but

- On a parlé dans le 1<sup>er</sup> chapitre du phénomène de la **malédiction de la dimension** !
- Ce phénomène caractérise le fait que le problème d'apprentissage devient beaucoup plus **difficile** quand  $\text{dim}(\mathbb{X})$  croît.
- Cependant, on sait que l'information contenue dans la dimension  $j$  est potentiellement partiellement **répétée** dans une autre dimension  $j'$ .
- On cherche donc un moyen d'éliminer les dimensions redondantes et l'**analyse en composantes principales** (ACP) en est un.

## Réduction de dimensions : but

- On a parlé dans le 1<sup>er</sup> chapitre du phénomène de la **malédiction de la dimension** !
- Ce phénomène caractérise le fait que le problème d'apprentissage devient beaucoup plus **difficile** quand  $\text{dim}(\mathbb{X})$  croît.
- Cependant, on sait que l'information contenue dans la dimension  $j$  est potentiellement partiellement **répétée** dans une autre dimension  $j'$ .
- On cherche donc un moyen d'éliminer les dimensions redondantes et l'**analyse en composantes principales** (ACP) en est un.

## Réduction de dimensions : but

- On a parlé dans le 1<sup>er</sup> chapitre du phénomène de la **malédiction de la dimension** !
- Ce phénomène caractérise le fait que le problème d'apprentissage devient beaucoup plus **difficile** quand  $\text{dim}(\mathbb{X})$  croît.
- Cependant, on sait que l'information contenue dans la dimension  $j$  est potentiellement partiellement **répétée** dans une autre dimension  $j'$ .
- On cherche donc un moyen d'éliminer les dimensions redondantes et l'**analyse en composantes principales** (ACP) en est un.

## Apprentissage de représentation : principe

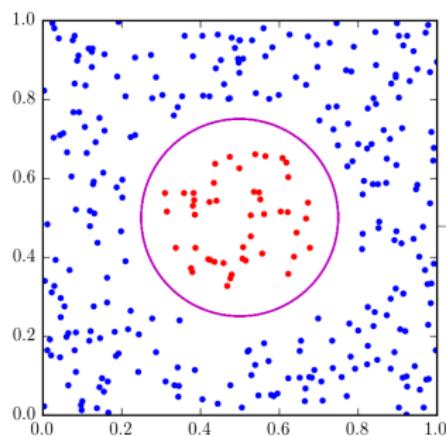
- On parle d'**apprentissage de représentation** quand on transforme les vecteurs d'entrée  $\mathbf{x}$  en un nouveau vecteur  $\mathbf{z}$  et que cette transformation est déduite des données.
- **Espoir** : les  $\mathbf{z}^{(i)}$  sont séparables à l'aide d'algorithmes plus simples

## Apprentissage de représentation : principe

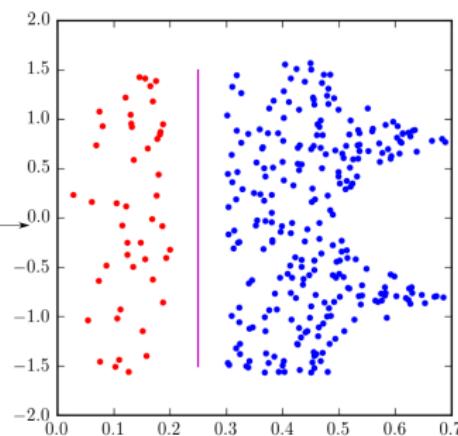
- On parle d'**apprentissage de représentation** quand on transforme les vecteurs d'entrée  $\mathbf{x}$  en un nouveau vecteur  $\mathbf{z}$  et que cette transformation est déduite des données.
- **Espoir** : les  $\mathbf{z}^{(i)}$  sont séparables à l'aide d'algorithmes plus simples

## Apprentissage de représentation : principe

- On parle d'apprentissage de représentation quand on transforme les vecteurs d'entrée  $x$  en un nouveau vecteur  $z$  et que cette transformation est déduite des données.
- Espoir : les  $z^{(i)}$  sont séparables à l'aide d'algorithmes plus simples



Transformation



## Apprentissage de représentation : principe

- Algo + simple  $\rightarrow$  – de paramètres  $\rightarrow$  – de risque d'*overfitting*.
- Si en plus  $\text{dim}(\mathbf{z}) < \text{dim}(\mathbf{x})$  alors en plus je combat la malédiction de la dimension.
- Déjà vu : Auto-encoders et DBNs.

## Apprentissage de représentation : principe

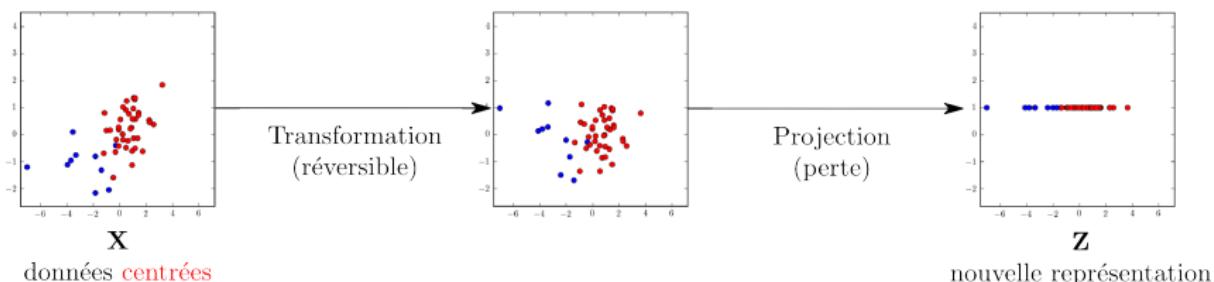
- Algo + simple  $\rightarrow$  – de paramètres  $\rightarrow$  – de risque d'*overfitting*.
- Si en plus  $\text{dim}(\mathbf{z}) < \text{dim}(\mathbf{x})$  alors en plus je combat la malédiction de la dimension.
- Déjà vu : Auto-encoders et DBNs.

## Apprentissage de représentation : principe

- Algo + simple  $\rightarrow$  – de paramètres  $\rightarrow$  – de risque d'*overfitting*.
- Si en plus  $\text{dim}(\mathbf{z}) < \text{dim}(\mathbf{x})$  alors en plus je combat la malédiction de la dimension.
- Déjà vu : Auto-encoders et DBNs.

## Réduction de dimensions : ACP

Approche en 2 temps



Rq : l'ACP fonctionne sur des données **non centrées**, mais les calculs sont légèrement plus simples après centrage.

Intuition géométrique : **projection** orthogonale qui étale les données au max.

## Réduction de dimensions : ACP

Etape n°1 :

- La transformation opérée est similaire à celle du **blanchiment**.
- On procède comme suit :
  - Calcul de la matrice de variance-covariance :  $\Sigma = \mathbf{X} \cdot \mathbf{X}^T$ .
  - On trouve la matrice  $\mathbf{Q}$  tel que  $\Lambda = \mathbf{Q}^T \Sigma \mathbf{Q}$  est diagonale.
- La matrice  $\mathbf{Q}$  est appelée matrice de vecteurs propres.
- Chaque colonne  $\mathbf{q}_j$  de  $\mathbf{Q}$  est appelé **vecteur propre** (*eigenvector*).
- Chaque élément diagonal  $\lambda_j = \Lambda_{jj}$  de  $\Lambda$  est appelée **valeur propre** associée au vecteur propre  $\mathbf{q}_j$ .

## Réduction de dimensions : ACP

Etape n°1 :

- La transformation opérée est similaire à celle du **blanchiment**.
- On procède comme suit :
  - ① Calcul de la matrice de variance-covariance :  $\Sigma = \mathbf{X} \cdot \mathbf{X}^T$ .
  - ② On trouve la matrice  $\mathbf{Q}$  tel que  $\Lambda = \mathbf{Q}^T \Sigma \mathbf{Q}$  est diagonale.
- La matrice  $\mathbf{Q}$  est appelée matrice de vecteurs propres.
- Chaque colonne  $\mathbf{q}_j$  de  $\mathbf{Q}$  est appelé **vecteur propre** (*eigenvector*).
- Chaque élément diagonal  $\lambda_j = \Lambda_{jj}$  de  $\Lambda$  est appelée **valeur propre** associée au vecteur propre  $\mathbf{q}_j$ .

## Réduction de dimensions : ACP

Etape n°1 :

- La transformation opérée est similaire à celle du **blanchiment**.
- On procède comme suit :
  - ① Calcul de la matrice de variance-covariance :  $\Sigma = \mathbf{X} \cdot \mathbf{X}^T$ .
  - ② On trouve la matrice  $\mathbf{Q}$  tel que  $\Lambda = \mathbf{Q}^T \Sigma \mathbf{Q}$  est diagonale.
- La matrice  $\mathbf{Q}$  est appelée matrice de vecteurs propres.
- Chaque colonne  $\mathbf{q}_j$  de  $\mathbf{Q}$  est appelé **vecteur propre** (*eigenvector*).
- Chaque élément diagonal  $\lambda_j = \Lambda_{jj}$  de  $\Lambda$  est appelée **valeur propre** associée au vecteur propre  $\mathbf{q}_j$ .

## Réduction de dimensions : ACP

Etape n°1 :

- La transformation opérée est similaire à celle du **blanchiment**.
- On procède comme suit :
  - ① Calcul de la matrice de variance-covariance :  $\Sigma = \mathbf{X} \cdot \mathbf{X}^T$ .
  - ② On trouve la matrice  $\mathbf{Q}$  tel que  $\Lambda = \mathbf{Q}^T \Sigma \mathbf{Q}$  est diagonale.
- La matrice  $\mathbf{Q}$  est appelée matrice de vecteurs propres.
- Chaque colonne  $\mathbf{q}_j$  de  $\mathbf{Q}$  est appelé **vecteur propre** (*eigenvector*).
- Chaque élément diagonal  $\lambda_j = \Lambda_{jj}$  de  $\Lambda$  est appelée **valeur propre** associée au vecteur propre  $\mathbf{q}_j$ .

## Réduction de dimensions : ACP

Etape n°1 :

- La transformation opérée est similaire à celle du **blanchiment**.
- On procède comme suit :
  - ① Calcul de la matrice de variance-covariance :  $\Sigma = \mathbf{X} \cdot \mathbf{X}^T$ .
  - ② On trouve la matrice  $\mathbf{Q}$  tel que  $\Lambda = \mathbf{Q}^T \Sigma \mathbf{Q}$  est diagonale.
- La matrice  $\mathbf{Q}$  est appelée matrice de vecteurs propres.
- Chaque colonne  $\mathbf{q}_j$  de  $\mathbf{Q}$  est appelé **vecteur propre** (*eigenvector*).
- Chaque élément diagonal  $\lambda_j = \Lambda_{jj}$  de  $\Lambda$  est appelée **valeur propre** associée au vecteur propre  $\mathbf{q}_j$ .

## Réduction de dimensions : ACP

Etape n°1 :

- La transformation opérée est similaire à celle du **blanchiment**.
- On procède comme suit :
  - ① Calcul de la matrice de variance-covariance :  $\Sigma = \mathbf{X} \cdot \mathbf{X}^T$ .
  - ② On trouve la matrice  $\mathbf{Q}$  tel que  $\Lambda = \mathbf{Q}^T \Sigma \mathbf{Q}$  est diagonale.
- La matrice  $\mathbf{Q}$  est appelée matrice de vecteurs propres.
- Chaque colonne  $\mathbf{q}_j$  de  $\mathbf{Q}$  est appelé **vecteur propre (eigenvector)**.
- Chaque élément diagonal  $\lambda_j = \Lambda_{jj}$  de  $\Lambda$  est appelée **valeur propre** associée au vecteur propre  $\mathbf{q}_j$ .

## Réduction de dimensions : ACP

Etape n°1 :

- La transformation opérée est similaire à celle du **blanchiment**.
- On procède comme suit :
  - ① Calcul de la matrice de variance-covariance :  $\Sigma = \mathbf{X} \cdot \mathbf{X}^T$ .
  - ② On trouve la matrice  $\mathbf{Q}$  tel que  $\Lambda = \mathbf{Q}^T \Sigma \mathbf{Q}$  est diagonale.
- La matrice  $\mathbf{Q}$  est appelée matrice de vecteurs propres.
- Chaque colonne  $\mathbf{q}_j$  de  $\mathbf{Q}$  est appelé **vecteur propre** (*eigenvector*).
- Chaque élément diagonal  $\lambda_j = \Lambda_{jj}$  de  $\Lambda$  est appelée **valeur propre** associée au vecteur propre  $\mathbf{q}_j$ .

## Réduction de dimensions : ACP

### Etape n°1 : remarques

- Comme les données sont centrées, la matrice  $\Sigma$  se calcule selon

$$\Sigma_{ij} = \frac{1}{n} \sum_{k=1}^n x_i^{(k)} x_j^{(k)} \text{ (voir Chap. 2).}$$

Elle est donc de taille  $d \times d$ .

- Les vecteurs  $\mathbf{q}_j$  sont orthogonaux ! On a  $\mathbf{q}_j^T \cdot \mathbf{q}_{j'} = 0$  si  $j \neq j'$ .
- Chaque valeur propre  $\lambda_j$  est la variance des données selon la nouvelle dimension  $j$ .

## Réduction de dimensions : ACP

### Etape n°1 : remarques

- Comme les données sont centrées, la matrice  $\Sigma$  se calcule selon

$$\Sigma_{ij} = \frac{1}{n} \sum_{k=1}^n x_i^{(k)} x_j^{(k)} \text{ (voir Chap. 2).}$$

Elle est donc de taille  $d \times d$ .

- Les vecteurs  $\mathbf{q}_j$  sont **orthogonaux** ! On a  $\mathbf{q}_j^T \cdot \mathbf{q}_{j'} = 0$  si  $j \neq j'$ .
- Chaque valeur propre  $\lambda_j$  est la **variance** des données selon la **nouvelle dimension**  $j$ .

## Réduction de dimensions : ACP

### Etape n°1 : remarques

- Comme les données sont centrées, la matrice  $\Sigma$  se calcule selon

$$\Sigma_{ij} = \frac{1}{n} \sum_{k=1}^n x_i^{(k)} x_j^{(k)} \text{ (voir Chap. 2).}$$

Elle est donc de taille  $d \times d$ .

- Les vecteurs  $\mathbf{q}_j$  sont **orthogonaux** ! On a  $\mathbf{q}_j^T \cdot \mathbf{q}_{j'} = 0$  si  $j \neq j'$ .
- Chaque valeur propre  $\lambda_j$  est la **variance** des données selon la **nouvelle dimension**  $j$ .

## Réduction de dimensions : ACP

## Etape n°1 : remarques

- Comme les données sont centrées, la matrice  $\Sigma$  se calcule selon

$$\Sigma_{ij} = \frac{1}{n} \sum_{k=1}^n x_i^{(k)} x_j^{(k)} \text{ (voir Chap. 2).}$$

Elle est donc de taille  $d \times d$ .

- Les vecteurs  $\mathbf{q}_j$  sont **orthogonaux** ! On a  $\mathbf{q}_j^T \cdot \mathbf{q}_{j'} = 0$  si  $j \neq j'$ .
- Chaque valeur propre  $\lambda_j$  est la **variance** des données selon la **nouvelle dimension**  $j$ .  
 → **Hypothèse** : les (nouvelles) dimensions décorrélées à petite variance ne séparent pas bien les classes.

## Réduction de dimensions : ACP

Etape n°2 :

- On élimine toutes les nouvelles dimensions telles que  $\lambda_j < \text{seuil}$ .
- Règle du pouce :

$$\text{seuil} = 0.9 \times \max_j \lambda_j. \quad (13)$$

## Réduction de dimensions : ACP

Etape n°2 :

- On élimine toutes les nouvelles dimensions telles que  $\lambda_j < \text{seuil}$ .

- **Règle du pouce :**

$$\text{seuil} = 0.9 \times \max_j \lambda_j. \quad (13)$$

## Réduction de dimensions : ACP

L'**hypothèse** sur la variance des nouvelles dimensions est elle toujours vraie ?

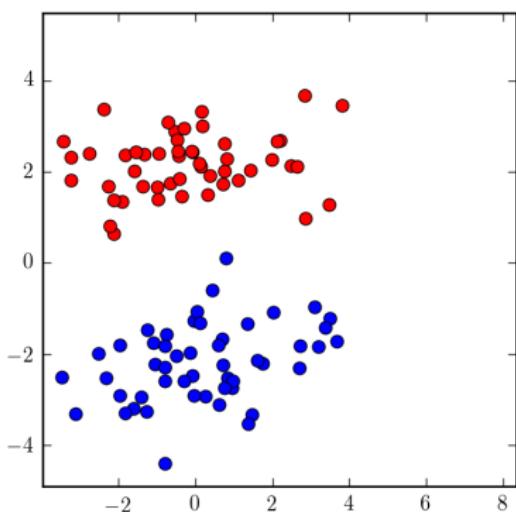
## Réduction de dimensions : ACP

L'**hypothèse** sur la variance des nouvelles dimensions est elle toujours vraie ? **Non !**

## Réduction de dimensions : ACP

L'**hypothèse** sur la variance des nouvelles dimensions est elle toujours vraie ? **Non** !

Exemple des datasets en forme de cigarette

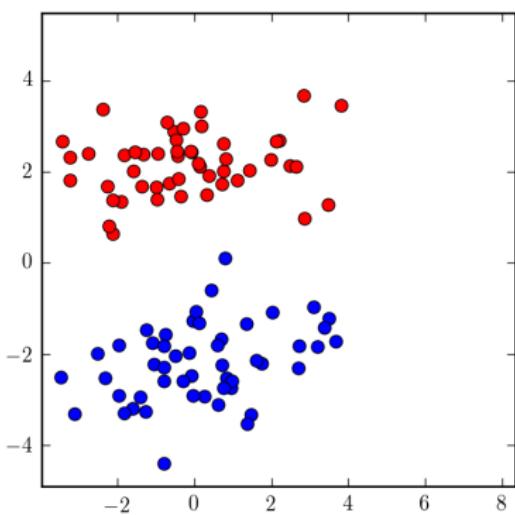


(données centrées)

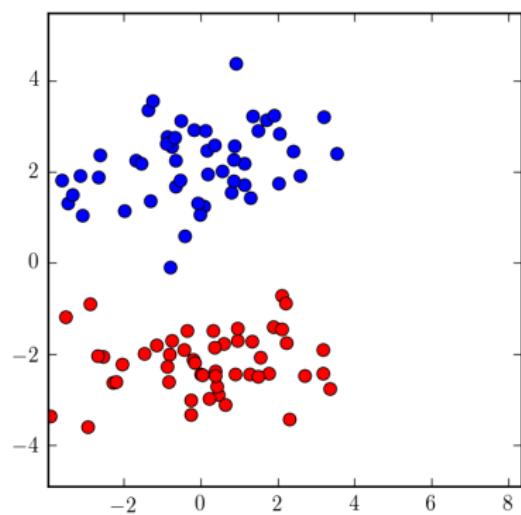
## Réduction de dimensions : ACP

L'**hypothèse** sur la variance des nouvelles dimensions est elle toujours vraie ? **Non** !

Exemple des datasets en forme de cigarette



(données centrées)

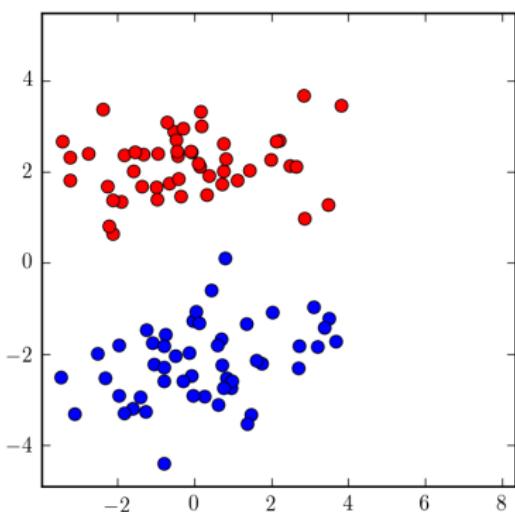


(données transformées)

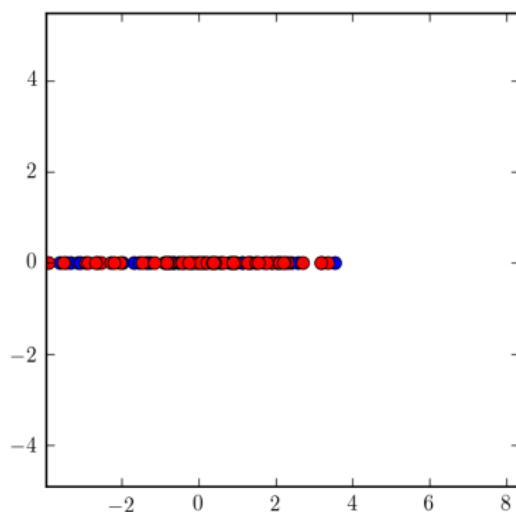
## Réduction de dimensions : ACP

L'**hypothèse** sur la variance des nouvelles dimensions est elle toujours vraie ? **Non** !

Exemple des datasets en forme de cigarette



(données centrées)

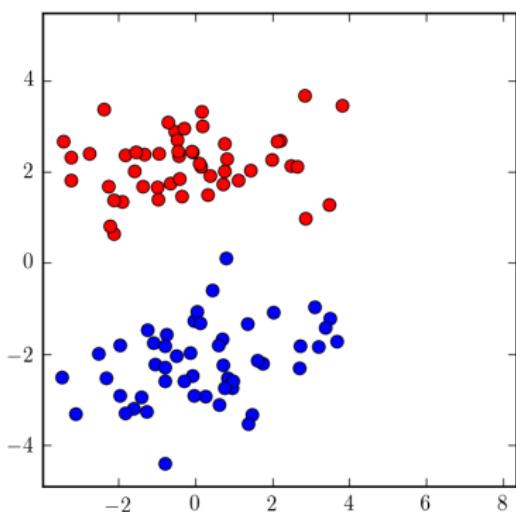


(projétées selon la val.p. max) ↗ ↘ ↙ ↘

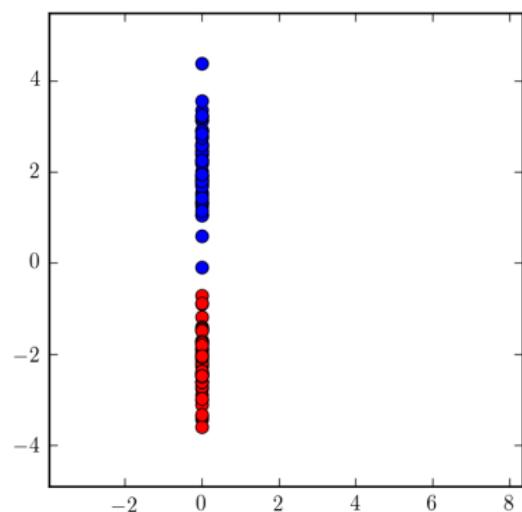
## Réduction de dimensions : ACP

L'**hypothèse** sur la variance des nouvelles dimensions est elle toujours vraie ? **Non !**

Exemple des datasets en forme de cigarette



(données centrées)

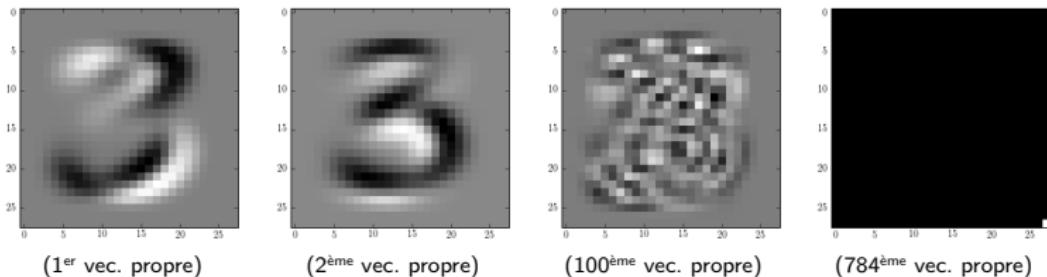


(projétées selon la val.p. min) ↗ ↘ ↙ ↘

## Réduction de dimensions : ACP

Malgré tout, cela fonctionne bien dans de nombreux cas. Voici un exemple sur une partie du dataset mnist qui contient des images manuscrites du chiffres "3".

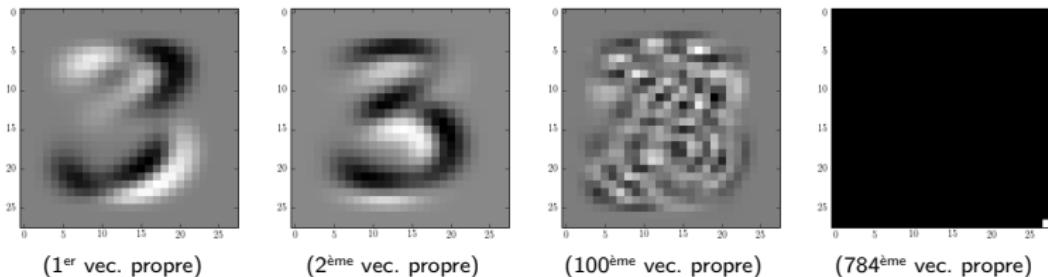
Vecteurs  
Propres  
appris



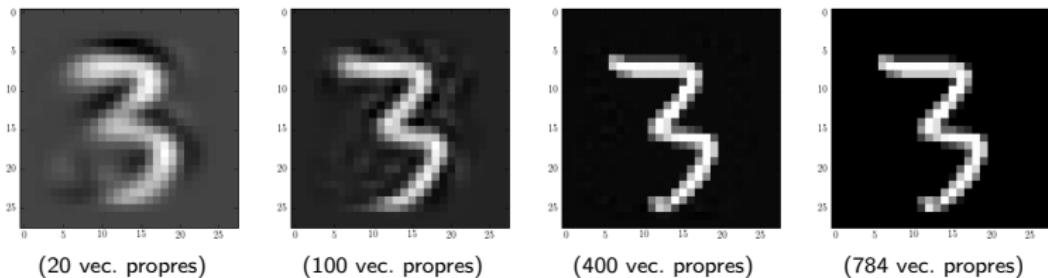
## Réduction de dimensions : ACP

Malgré tout, cela fonctionne bien dans de nombreux cas. Voici un exemple sur une partie du dataset mnist qui contient des images manuscrites du chiffres “3”.

Vecteurs  
Propres  
appris



Input  
Reconstruite



## Réduction de dimensions : ACP

L'efficacité visible de l'ACP dans l'exemple précédent vient de la propriété suivante :

### Propriété

Pour un nombre fixe  $K$  de vecteurs orthogonaux  $\mathbf{w}_j$  et de coefficients  $z_j$ , on souhaite reconstruire l'entrée  $\mathbf{x}$  selon

$$\mathbf{x} \approx \hat{\mathbf{x}} = \sum_{j=1}^K z_j \times \mathbf{w}_j. \quad (14)$$

La solution donnée par l'ACP en sélectionnant les vecteurs associées au  $K$  plus grandes valeurs propres et en prenant  $z_j = \mathbf{w}_j^T \cdot \mathbf{x}$  est la meilleure au sens de l'erreur de reconstruction moyenne :

$$\frac{1}{n} \sum_{i=1}^n \|\mathbf{x}^{(i)} - \hat{\mathbf{x}}^{(i)}\|^2 \quad (15)$$

## Décomposition en valeurs singulières (SVD) :

## Décomposition en valeurs singulières (SVD) :

- C'est une méthode très proche de l'ACP mais on part directement de la matrice  $\mathbf{X}$  et non de  $\Sigma$ .

## Décomposition en valeurs singulières (SVD) :

- C'est une méthode très proche de l'ACP mais on part directement de la matrice  $\mathbf{X}$  et non de  $\Sigma$ .
- On obtient la factorisation matricielle suivante :

$$\underbrace{\mathbf{X}^T}_{n \times d} = \underbrace{\mathbf{U}}_{n \times n} \cdot \underbrace{\mathbf{S}}_{n \times d} \cdot \underbrace{\mathbf{V}^T}_{d \times d}. \quad (16)$$

## Décomposition en valeurs singulières (SVD) :

- Toutes les entrées n'appartenant pas à la diagonale principale de  $\mathbf{S}$  sont **nulles**.
- Les éléments diagonaux de  $\mathbf{S}$  sont notés  $s_j = S_{jj}$  et appelées **valeurs singulières**.
- On a la relation

$$s_j = \sqrt{\lambda_j}. \quad (17)$$

## Décomposition en valeurs singulières (SVD) :

- Toutes les entrées n'appartenant pas à la diagonale principale de  $\mathbf{S}$  sont **nulles**.
- Les éléments diagonaux de  $\mathbf{S}$  sont notés  $s_j = S_{jj}$  et appelées **valeurs singulières**.
- On a la relation

$$s_j = \sqrt{\lambda_j}. \quad (17)$$

## Décomposition en valeurs singulières (SVD) :

- Toutes les entrées n'appartenant pas à la diagonale principale de  $\mathbf{S}$  sont **nulles**.
- Les éléments diagonaux de  $\mathbf{S}$  sont notés  $s_j = S_{jj}$  et appelées **valeurs singulières**.
- On a la relation

$$s_j = \sqrt{\lambda_j}. \quad (17)$$

## Décomposition en valeurs singulières (SVD) :

- Les colonnes de la matrice  $\mathbf{U}$  sont appelées **vecteurs singuliers gauches**.
- Les colonnes de la matrice  $\mathbf{V}$  sont appelées **vecteurs singuliers droits**.
- On a les relations

$$\mathbf{U} = \text{eig}\left(\mathbf{X}^T \cdot \mathbf{X}\right), \quad (18)$$

$$\mathbf{V} = \text{eig}\left(\mathbf{X} \cdot \mathbf{X}^T\right) = \mathbf{Q}. \quad (19)$$

## Décomposition en valeurs singulières (SVD) :

- Les colonnes de la matrice  $\mathbf{U}$  sont appelées **vecteurs singuliers gauches**.
- Les colonnes de la matrice  $\mathbf{V}$  sont appelées **vecteurs singuliers droits**.
- On a les relations

$$\mathbf{U} = \text{eig}(\mathbf{X}^T \cdot \mathbf{X}), \quad (18)$$

$$\mathbf{V} = \text{eig}(\mathbf{X} \cdot \mathbf{X}^T) = \mathbf{Q}. \quad (19)$$

## Décomposition en valeurs singulières (SVD) :

- Les colonnes de la matrice  $\mathbf{U}$  sont appelées **vecteurs singuliers gauches**.
- Les colonnes de la matrice  $\mathbf{V}$  sont appelées **vecteurs singuliers droits**.
- On a les relations

$$\mathbf{U} = \text{eig}(\mathbf{X}^T \cdot \mathbf{X}), \quad (18)$$

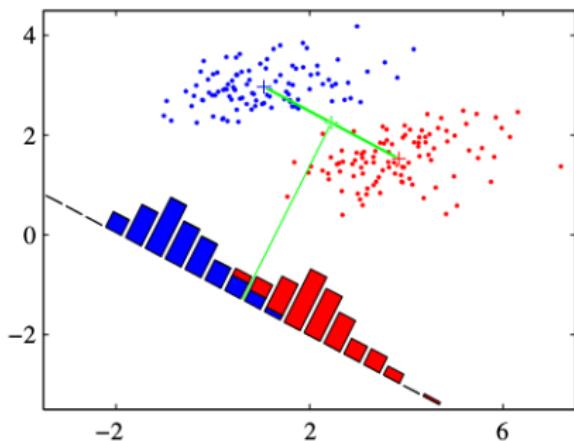
$$\mathbf{V} = \text{eig}(\mathbf{X} \cdot \mathbf{X}^T) = \mathbf{Q}. \quad (19)$$

## Décomposition en valeurs singulières (SVD) :

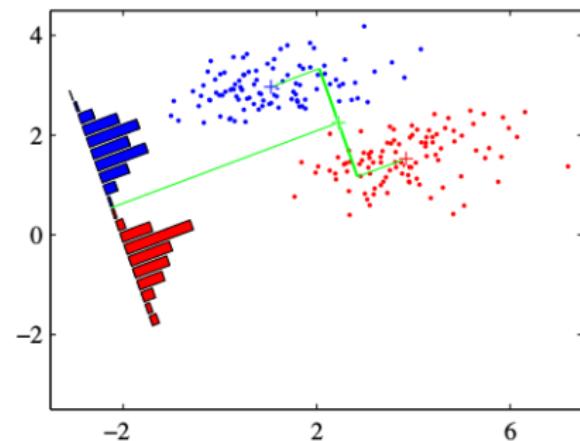
**Conclusion** : Choisir les  $k$  plus gros  $s_j$  (et annuler les autres) aboutit aux **même projets** qu'avec l'**ACP** où nous aurions gardé les  $k$  plus gros  $\lambda_j$

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

- Il s'agit d'une technique **supervisée** où on cherche la droite de vecteur directeur  $\mathbf{v}$  où les données projetées  $\mathbf{z}^{(i)} = \mathbf{v}^T \cdot \mathbf{x}^{(i)}$  seront le **mieux séparées** en fonction de leur classe.



[Bishop 2006]



## Analyse discriminante linéaire de Fisher (Fisher LDA) : Principe

- Soit  $\textcolor{blue}{m}_1$  la moyenne des  $\textcolor{blue}{z}^{(i)}$  dans la classe 1 et  $\textcolor{blue}{m}_2$  celle de la classe 2.
- Pour séparer les classes, j'ai envie d'avoir  $|\textcolor{blue}{m}_2 - \textcolor{blue}{m}_1|$  grand !
- .. mais c'est **insuffisant**, il faut aussi  $\sigma_1$  et  $\sigma_2$  petit.
- Le critère (à maximiser) proposé par Fisher est :

$$\frac{(\textcolor{blue}{m}_2 - \textcolor{blue}{m}_1)^2}{\sigma_1^2 + \sigma_2^2}. \quad (20)$$

## Analyse discriminante linéaire de Fisher (Fisher LDA) : Principe

- Soit  $m_1$  la moyenne des  $z^{(i)}$  dans la classe 1 et  $m_2$  celle de la classe 2.
- Pour séparer les classes, j'ai envie d'avoir  $|m_2 - m_1|$  grand !
- .. mais c'est insuffisant, il faut aussi  $\sigma_1$  et  $\sigma_2$  petit.
- Le critère (à maximiser) proposé par Fisher est :

$$\frac{(m_2 - m_1)^2}{\sigma_1^2 + \sigma_2^2}. \quad (20)$$

## Analyse discriminante linéaire de Fisher (Fisher LDA) : Principe

- Soit  $m_1$  la moyenne des  $z^{(i)}$  dans la classe 1 et  $m_2$  celle de la classe 2.
- Pour séparer les classes, j'ai envie d'avoir  $|m_2 - m_1|$  grand !
- .. mais c'est **insuffisant**, il faut aussi  $\sigma_1$  et  $\sigma_2$  petit.
- Le critère (à maximiser) proposé par Fisher est :

$$\frac{(m_2 - m_1)^2}{\sigma_1^2 + \sigma_2^2}. \quad (20)$$

## Analyse discriminante linéaire de Fisher (Fisher LDA) : Principe

- Soit  $\textcolor{blue}{m}_1$  la moyenne des  $\textcolor{blue}{z}^{(i)}$  dans la classe 1 et  $\textcolor{blue}{m}_2$  celle de la classe 2.
- Pour séparer les classes, j'ai envie d'avoir  $|\textcolor{blue}{m}_2 - \textcolor{blue}{m}_1|$  grand !
- .. mais c'est **insuffisant**, il faut aussi  $\sigma_1$  et  $\sigma_2$  petit.
- Le critère (à maximiser) proposé par Fisher est :

$$\frac{(\textcolor{blue}{m}_2 - \textcolor{blue}{m}_1)^2}{\sigma_1^2 + \sigma_2^2}. \quad (20)$$

## Analyse discriminante linéaire de Fisher (Fisher LDA) : Principe

La **solution** est donnée par :

$$\textcolor{red}{v} \propto (\mathbf{S}_1 + \mathbf{S}_2)^{-1} \cdot (\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_2), \quad (21)$$

avec :

- $\bar{\mathbf{x}}_k$  la moyenne des exemples (avant projection) appartenant à la classe n° $i$ ,
- $\mathbf{S}_i$  la matrice de variance-covariance empirique calculée à partir des exemples (avant projection) appartenant à la classe n° $i$ .

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

- Gros inconvénient : on projette en 1D !!
- Si on voulait classifier derrière, il reste juste à choisir un seuil  $\tau$ . On pourrait *fitter* des Gaussiennes pour chaque distribution conditionnelle des  $Z|Y = c$ .
- Si on a un problème à  $\ell$  classes, alors la méthode est extensible et on projette sur  $\ell - 1$  classes.

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

- Gros inconvénient : on projette en 1D !!
- Si on voulait classifier derrière, il reste juste à choisir un seuil  $\tau$ . On pourrait *fitter* des Gaussiennes pour chaque distribution conditionnelle des  $Z|Y = c$ .
- Si on a un problème à  $\ell$  classes, alors la méthode est extensible et on projette sur  $\ell - 1$  classes.

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

- Gros inconvénient : on projette en 1D !!
- Si on voulait classifier derrière, il reste juste à choisir un seuil  $\tau$ . On pourrait *fitter* des Gaussiennes pour chaque distribution conditionnelle des  $Z|Y = c$ .
- Si on a un problème à  $\ell$  classes, alors la méthode est extensible et on projette sur  $\ell - 1$  classes.

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

### ATTENTION

- Il existe un algorithme d'apprentissage appelé **Analyse discriminante linéaire** (LDA).

- Il s'agit d'un **modèle génératif** où les **class conditional distributions** sont Gaussiennes multivariées :

$$X|Y = c \sim \mathcal{N}(\mu_c, \Sigma) \quad (22)$$

- Dans ce modèle, toutes ces distribution **partagent** la même matrice  $\Sigma$ .
- L'objectif est donc complètement différent de l'**analyse discriminante linéaire de Fisher**.

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

### ATTENTION

- Il existe un algorithme d'apprentissage appelé **Analyse discriminante linéaire** (LDA).
- Il s'agit d'un **modèle génératif** où les **class conditional distributions** sont Gaussiennes multivariées :

$$X|Y = c \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}) \quad (22)$$

- Dans ce modèle, toutes ces distribution **partagent** la même matrice  $\boldsymbol{\Sigma}$ .
- L'objectif est donc complètement différent de l'**analyse discriminante linéaire de Fisher**.

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

### ATTENTION

- Il existe un algorithme d'apprentissage appelé **Analyse discriminante linéaire** (LDA).
- Il s'agit d'un **modèle génératif** où les **class conditional distributions** sont Gaussiennes multivariées :

$$X|Y = c \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}) \quad (22)$$

- Dans ce modèle, toutes ces distribution **partagent** la même matrice  **$\boldsymbol{\Sigma}$** .
- L'objectif est donc complètement différent de l'**analyse discriminante linéaire de Fisher**.

## Analyse discriminante linéaire de Fisher (Fisher LDA) :

### ATTENTION

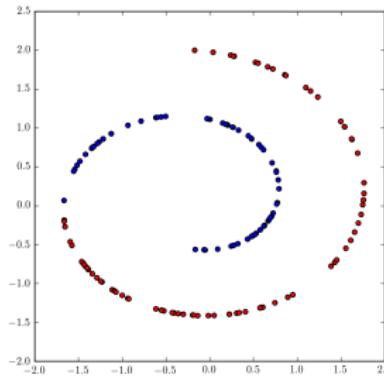
- Il existe un algorithme d'apprentissage appelé **Analyse discriminante linéaire** (LDA).
- Il s'agit d'un **modèle génératif** où les **class conditional distributions** sont Gaussiennes multivariées :

$$X|Y = c \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}) \quad (22)$$

- Dans ce modèle, toutes ces distribution **partagent** la même matrice  **$\boldsymbol{\Sigma}$** .
- L'objectif est donc complètement différent de l'**analyse discriminante linéaire de Fisher**.

## Apprentissage de variétés (*manifold learning*) :

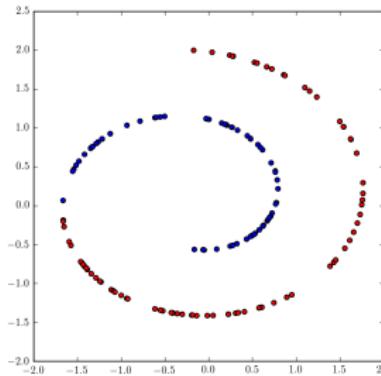
- Il existe des datasets pour lesquels, aucune projection linéaire ne donnera de bons résultats.



- En général, les données vivent dans une zone de l'espace d'attribut qui est beaucoup plus petite que l'espace et dont les contours sont assez lisses.
- Une telle zone est appelée en géométrie variété (*manifold*).

## Apprentissage de variétés (*manifold learning*) :

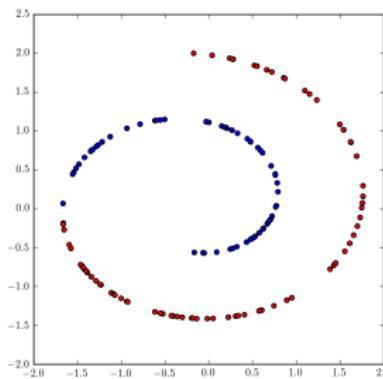
- Il existe des datasets pour lesquels, aucune projection linéaire ne donnera de bons résultats.



- En général, les données vivent dans une zone de l'espace d'attribut qui est beaucoup plus petite que l'espace et dont les contours sont assez lisses.
- Une telle zone est appelée en géométrie variété (*manifold*).

## Apprentissage de variétés (*manifold learning*) :

- Il existe des datasets pour lesquels, aucune projection linéaire ne donnera de bons résultats.



- En général, les données vivent dans une zone de l'espace d'attribut qui est beaucoup plus petite que l'espace et dont les contours sont assez lisses.
- Une telle zone est appelée en géométrie variété (*manifold*).

## Apprentissage de variétés (*manifold learning*) :

Exemples d'algorithmes pour apprendre à « dérouler » la variété.

- Multidimensional Scaling (MDS) : on apprend directement les vecteurs  $\mathbf{z}$  qui vivent dans un espace plus petit et préservent les distances :

$$\sum_{i=1}^n \sum_{i'=1}^n \left( \|\mathbf{x}^{(i)} - \mathbf{x}^{(i')}\| - \|\mathbf{z}^{(i)} - \mathbf{z}^{(i')}\| \right)^2. \quad (23)$$

Cette fonction de coût est minimisée par rapport aux  $\mathbf{z}^{(i)}$ .

# Apprentissage de variétés (*manifold learning*) :

## Exemples d'algorithmes pour apprendre à « dérouler » la variété.

- ISOMAP :

- ➊ construction d'un **graphe** d'adjacence entre les  $x^{(i)}$  proches les un des autres.
- ➋ calcul de distances **géodésiques** entre les  $x^{(i)}$  au sens du chemin de longueur minimal dans le graphe,
- ➌ application de **MDS** avec ces distances.

## Apprentissage de variétés (*manifold learning*) :

Exemples d'algorithmes pour apprendre à « dérouler » la variété.

- ISOMAP :

- ➊ construction d'un **graphe** d'adjacence entre les  $x^{(i)}$  proches les un des autres.
- ➋ calcul de distances **géodésiques** entre les  $x^{(i)}$  au sens du chemin de longueur minimal dans le graphe,
- ➌ application de **MDS** avec ces distances.

## Apprentissage de variétés (*manifold learning*) :

Exemples d'algorithmes pour apprendre à « dérouler » la variété.

- ISOMAP :

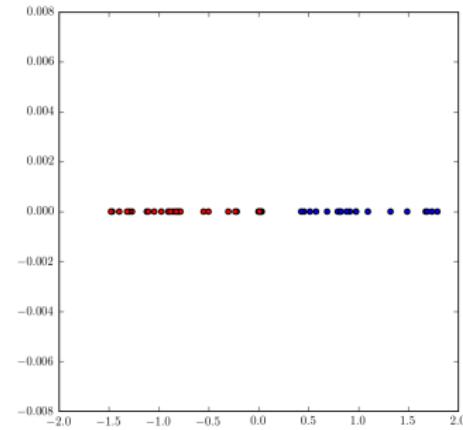
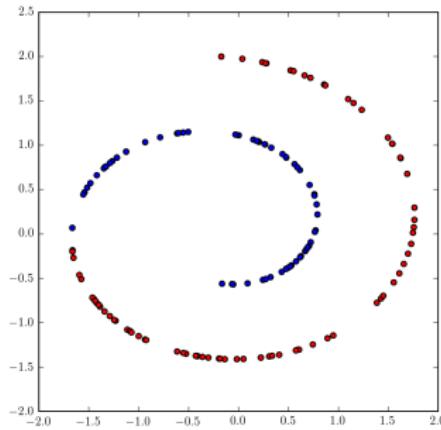
- ➊ construction d'un **graphe** d'adjacence entre les  $\mathbf{x}^{(i)}$  proches les un des autres.
- ➋ calcul de distances **géodésiques** entre les  $\mathbf{x}^{(i)}$  au sens du chemin de longueur minimal dans le graphe,
- ➌ application de **MDS** avec ces distances.

## Apprentissage de variétés (*manifold learning*) :

Exemples d'algorithmes pour apprendre à « dérouler » la variété.

- **ISOMAP :**

- ➊ construction d'un **graphe** d'adjacence entre les  $\mathbf{x}^{(i)}$  proches les un des autres.
- ➋ calcul de distances **géodésiques** entre les  $\mathbf{x}^{(i)}$  au sens du chemin de longueur minimal dans le graphe,
- ➌ application de **MDS** avec ces distances.



Apprentissage de variétés (*manifold learning*) :

Autres Exemples :

- Local Linear Embedding (LLE) : on apprend pour une approximation linéaire de chaque  $\mathbf{x}^{(i)}$  à partir de ses voisins. On cherche les  $\mathbf{z}^{(i)}$  qui vivent dans un espace plus petit et **préservent** ces approximations.
- ACP à noyaux : On va calculer les val. propres et vec. propres de la matrice de covariance

$$\frac{1}{n} \sum_{i=1}^n \Phi(\mathbf{x}^{(i)}) \Phi(\mathbf{x}^{(i)})^T,$$

où  $\Phi$  est une transformation engendrant un noyau<sup>1</sup>

$$\kappa(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \Phi(\mathbf{x}^{(i)})^T \Phi(\mathbf{x}^{(j)}).$$

---

1. fonction avec quelques propriétés spécifiques.

Apprentissage de variétés (*manifold learning*) :

Autres Exemples :

- Local Linear Embedding (LLE) : on apprend pour une approximation linéaire de chaque  $\mathbf{x}^{(i)}$  à partir de ses voisins. On cherche les  $\mathbf{z}^{(i)}$  qui vivent dans un espace plus petit et **préservent** ces approximations.
- ACP à noyaux : On va calculer les val. propres et vec. propres de la matrice de covariance

$$\frac{1}{n} \sum_{i=1}^n \Phi(\mathbf{x}^{(i)}) \Phi(\mathbf{x}^{(i)})^T,$$

où  $\Phi$  est une transformation engendrant un noyau<sup>1</sup>

$$\kappa(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \Phi(\mathbf{x}^{(i)})^T \Phi(\mathbf{x}^{(j)}).$$

---

1. fonction avec quelques propriétés spécifiques.

# Apprentissage de variétés (*manifold learning*) :

Autres Exemples :

- **Self Organizing Maps (SOM)** : Il s'agit de *fitter* une grille 2D (en général) aux données → surtout utile pour de la **visualisation** de données

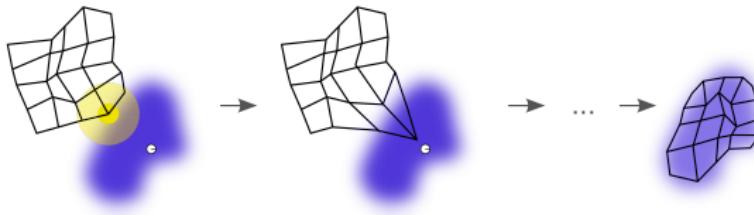
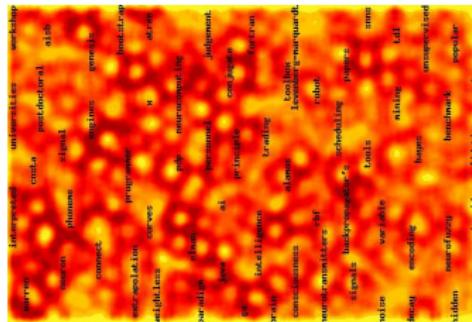


image from the SOM wikipedia page.



**FIGURE 14.19.** Heatmap representation of the SOM model fit to a corpus of 12,088 strings grouped according to their semantic content (courtesy WEBSOM homepage). The lighter areas indicate higher-density areas. Populated nodes are automatically labeled according to typical content.

fig. tirée de *Elements of Statistical Learning*, Hastie and Tibshirani 2008.

# Plan du chapitre

- 1 Pré-traitements basiques
- 2 Réduction de dimension par apprentissage de représentation
- 3 Sélection d'attributs
- 4 Apprentissage non supervisé
- 5 Conclusions
- 6 Annexes

## Sélection d'attributs :

- On a toujours la même envie de réduire la dimension, mais on veut le faire **directement** sur les vecteurs  $\mathbf{x}^{(i)}$

## Sélection d'attributs :

- On a toujours la même envie de réduire la dimension, mais on veut le faire **directement** sur les vecteurs  $\mathbf{x}^{(i)}$

$$\mathbf{x}^{(i)} = [0.2 \quad -2.0 \quad 10.2 \quad 2.9 \quad 1.2 \quad 0.5]^T$$

## Sélection d'attributs :

- On a toujours la même envie de réduire la dimension, mais on veut le faire **directement** sur les vecteurs  $\mathbf{x}^{(i)}$

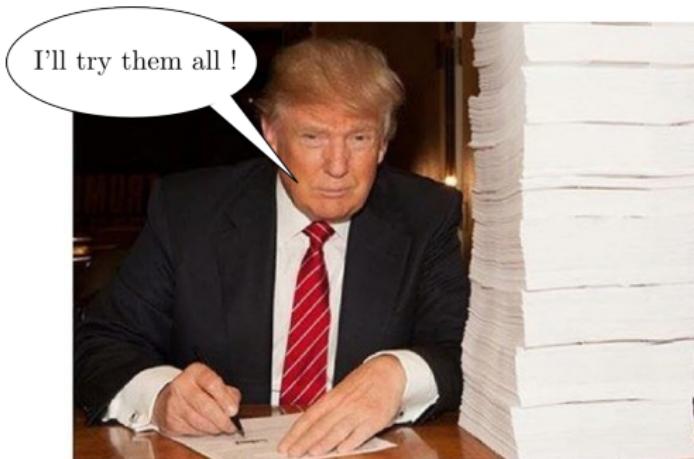
$$\mathbf{x}^{(i)} = [0.2 \quad -2.0 \quad 10.2 \quad 2.9 \quad 1.2 \quad 0.5]^T$$

## Sélection d'attributs :

- 1<sup>ère</sup> idée : sélection **exhaustive** en utilisant  $Err_{test}$

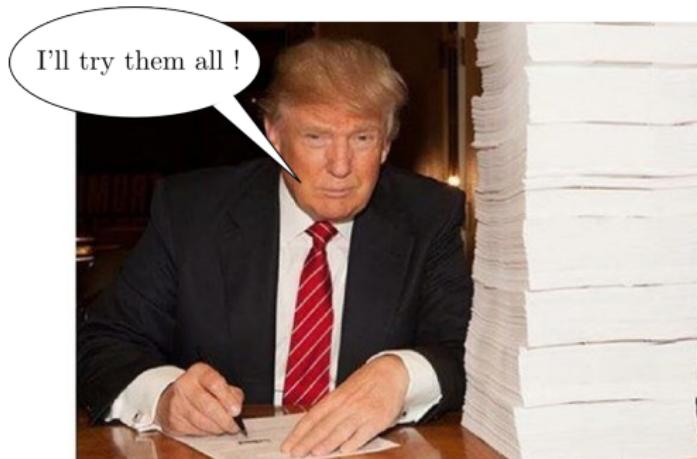
## Sélection d'attributs :

- 1<sup>ère</sup> idée : sélection **exhaustive** en utilisant  $Err_{test}$



## Sélection d'attributs :

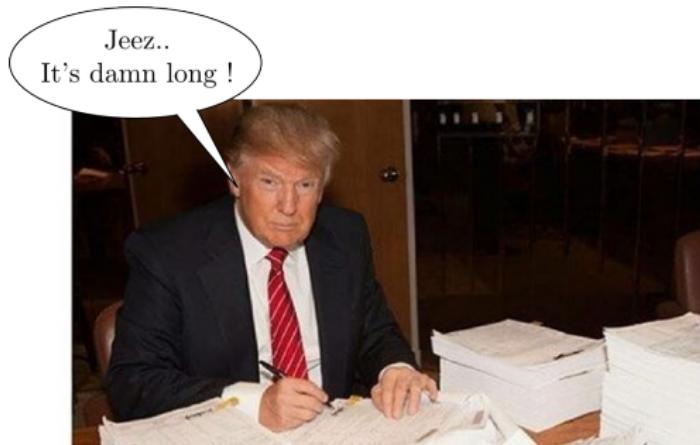
- 1<sup>ère</sup> idée : sélection **exhaustive** en utilisant  $Err_{test}$



.. mais il y  $2^d$  sélections possibles !

## Sélection d'attributs :

- 1<sup>ère</sup> idée : sélection **exhaustive** en utilisant  $Err_{test}$



.. mais il y  $2^d$  sélections possibles !

## Sélection d'attributs :

- 2<sup>ème</sup> idée : on va les prendre **un par un**, toujours en utilisant  $Err_{test}$ .
- On commence avec **aucun** attribut.
- On entraîne l'algo dimension par dimension ( $d$  entraînements) et on garde la meilleure.
- On itère ensuite en entraînant avec celle gardée plus une des restantes ( $d - 1$  entraînements)

## Sélection d'attributs :

- 2<sup>ème</sup> idée : on va les prendre **un par un**, toujours en utilisant  $Err_{test}$ .
- On commence avec **aucun** attribut.
- On entraîne l'algo dimension par dimension ( $d$  entraînements) et on garde la meilleure.
- On itère ensuite en entraînant avec celle gardée plus une des restantes ( $d - 1$  entraînements)

## Sélection d'attributs :

- 2<sup>ème</sup> idée : on va les prendre **un par un**, toujours en utilisant  $Err_{test}$ .
- On commence avec **aucun** attribut.
- On entraîne l'algo dimension par dimension ( $d$  entraînements) et on garde la meilleure.
- On itère ensuite en entraînant avec celle gardée plus une des restantes ( $d - 1$  entraînements)

## Sélection d'attributs :

- 2<sup>ème</sup> idée : on va les prendre **un par un**, toujours en utilisant  $Err_{test}$ .
- On commence avec **aucun** attribut.
- On entraîne l'algo dimension par dimension ( $d$  entraînements) et on garde la meilleure.
- On itère ensuite en entraînant avec celle gardée plus une des restantes ( $d - 1$  entraînements)

## Sélection d'attributs :

- etc.. On arrête quand on atteint un nombre convenu à l'avance de dimensions ou que  $Err_{test}$  augmente.
- On parle de Forward Feature Selection.
- L'approche par éliminations successives (Backward) est aussi possible.
- Dans le pire des cas, on fait  $\frac{d(d+1)}{2}$  apprentissages.

## Sélection d'attributs :

- etc.. On arrête quand on atteint un nombre convenu à l'avance de dimensions ou que  $Err_{test}$  augmente.
- On parle de **Forward Feature Selection**.
- L'approche par éliminations successives (**Backward**) est aussi possible.
- Dans le pire des cas, on fait  $\frac{d(d+1)}{2}$  apprentissages.

## Sélection d'attributs :

- etc.. On arrête quand on atteint un nombre convenu à l'avance de dimensions ou que  $Err_{test}$  augmente.
- On parle de **Forward Feature Selection**.
- L'approche par éliminations successives (**Backward**) est aussi possible.
- Dans le pire des cas, on fait  $\frac{d(d+1)}{2}$  apprentissages.

## Sélection d'attributs :

- etc.. On arrête quand on atteint un nombre convenu à l'avance de dimensions ou que  $Err_{test}$  augmente.
- On parle de **Forward Feature Selection**.
- L'approche par éliminations successives (**Backward**) est aussi possible.
- Dans le pire des cas, on fait  $\frac{d(d+1)}{2}$  apprentissages.

Sélection d'attributs : parcimonie (*sparsity*)

- Vous rappelez-vous de la régression linéaire dans le cas Bayésien ?

$$p_{\theta|\mathcal{D}}(\mathbf{w}, b, \sigma) \propto \underbrace{\prod_{i=1}^n \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y^{(i)} - (\mathbf{w}^T \cdot \mathbf{x}^{(i)} + b))^2}{2\sigma^2}}}_{\text{likelihood}} \times \underbrace{\frac{1}{\sqrt{2\pi}\sigma_0} e^{-\frac{[\mathbf{w} \ b] \cdot \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}}{2\sigma_0^2}}}_{\text{prior sur } \mathbf{w} \text{ et } b},$$

$$J(\boldsymbol{\theta}) = \text{NLL}(\mathbf{w}, b, \sigma) + \log(\sqrt{2\pi}\sigma_0) + \frac{[\mathbf{w} \ b] \cdot \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}}{2\sigma_0^2}.$$

## Sélection d'attributs : parcimonie (*sparsity*)

- $\sigma_0$  est en réalité un **hyperparamètre** qu'on ne peut pas apprendre..
- Sortons le du vecteur de paramètres et posons plus simplement :

$$\boldsymbol{\theta} = \begin{bmatrix} w \\ b \end{bmatrix}. \quad (24)$$

- Au final, on peut se contenter de minimiser :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \times \|\boldsymbol{\theta}\|_2^2.$$

où  $\delta > 0$  est un **hyperparamètre** qui remplace  $\sigma_0$ .

- Ce terme de régularisation est appelé **pénalité  $L_2$**  et l'approche est connue sous le nom de **Ridge regression**.

## Sélection d'attributs : parcimonie (*sparsity*)

- $\sigma_0$  est en réalité un **hyperparamètre** qu'on ne peut pas apprendre..
- Sortons le du vecteur de paramètres et posons plus simplement :

$$\boldsymbol{\theta} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}. \quad (24)$$

- Au final, on peut se contenter de minimiser :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \times \|\boldsymbol{\theta}\|_2^2.$$

où  $\delta > 0$  est un **hyperparamètre** qui remplace  $\sigma_0$ .

- Ce terme de régularisation est appelé **pénalité  $L_2$**  et l'approche est connue sous le nom de **Ridge regression**.

Sélection d'attributs : parcimonie (*sparsity*)

- $\sigma_0$  est en réalité un **hyperparamètre** qu'on ne peut pas apprendre..
- Sortons le du vecteur de paramètres et posons plus simplement :

$$\boldsymbol{\theta} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}. \quad (24)$$

- Au final, on peut se contenter de minimiser :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \times \|\boldsymbol{\theta}\|_2^2.$$

où  $\delta > 0$  est un **hyperparamètre** qui remplace  $\sigma_0$ .

- Ce terme de régularisation est appelé **pénalité  $L_2$**  et l'approche est connue sous le nom de **Ridge regression**.

Sélection d'attributs : parcimonie (*sparsity*)

- $\sigma_0$  est en réalité un **hyperparamètre** qu'on ne peut pas apprendre..
- Sortons le du vecteur de paramètres et posons plus simplement :

$$\boldsymbol{\theta} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}. \quad (24)$$

- Au final, on peut se contenter de minimiser :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \times \|\boldsymbol{\theta}\|_2^2.$$

où  $\delta > 0$  est un **hyperparamètre** qui remplace  $\sigma_0$ .

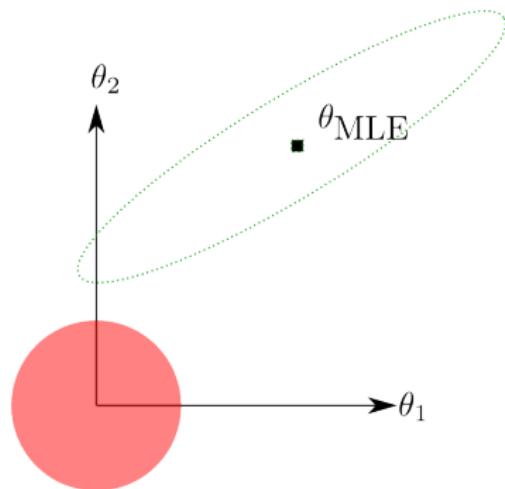
- Ce terme de régularisation est appelé **pénalité  $L_2$**  et l'approche est connue sous le nom de **Ridge regression**.

Sélection d'attributs : parcimonie (*sparsity*)

- Ridge regression

Le problème peut en fait être ré-écrit comme :

$$\arg \min_{\theta \text{ t.q. } \|\theta\|_2 \leq B} \text{NLL}(\theta) \quad (25)$$



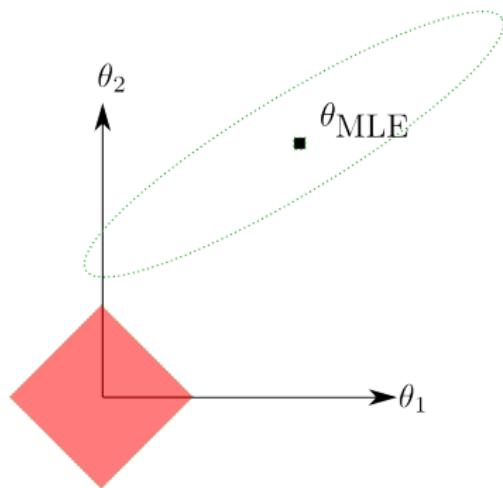
Sélection d'attributs : parcimonie (*sparsity*)

## • Lasso

Changeons légèrement  
le problème :

$$\arg \min_{\theta} \text{NLL}(\theta) \quad (26)$$

t.q.  $\|\theta\|_1 \leq B$

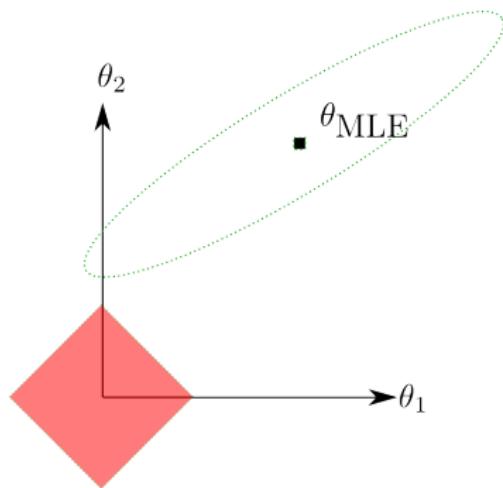


Sélection d'attributs : parcimonie (*sparsity*)

## • Lasso

Changeons légèrement  
le problème :

$$\arg \min_{\theta \text{ t.q. } \|\theta\|_1 \leq B} \text{NLL}(\theta) \quad (26)$$



→ Certaines dimensions sont annulées !

Sélection d'attributs : parcimonie (*sparsity*)

- Lasso peut aussi se ré-exprimer comme le problème de minimisation de la fonction de coût  $J$  :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \|\boldsymbol{\theta}\|_1, \quad (27)$$

(28)

Sélection d'attributs : parcimonie (*sparsity*)

- Lasso peut aussi se ré-exprimer comme le problème de minimisation de la fonction de coût  $J$  :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \|\boldsymbol{\theta}\|_1, \quad (27)$$

$$= (\mathbf{y} - \mathbf{X}^T \cdot \boldsymbol{\theta})^T \cdot (\mathbf{y} - \mathbf{X}^T \cdot \boldsymbol{\theta}) + \delta \|\boldsymbol{\theta}\|_1, \quad (28)$$

(29)

## Sélection d'attributs : parcimonie (*sparsity*)

- Lasso peut aussi se ré-exprimer comme le problème de minimisation de la fonction de coût  $J$  :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \|\boldsymbol{\theta}\|_1, \quad (27)$$

$$= (\mathbf{y} - \mathbf{X}^T \cdot \boldsymbol{\theta})^T \cdot (\mathbf{y} - \mathbf{X}^T \cdot \boldsymbol{\theta}) + \delta \|\boldsymbol{\theta}\|_1, \quad (28)$$

$$= \sum_{i=1}^n (y^{(i)} - \boldsymbol{\theta}^T \cdot \mathbf{x}^{(i)})^2 + \delta \sum_{j=1}^d |\theta_j|, \quad (29)$$

(30)

## Sélection d'attributs : parcimonie (*sparsity*)

- Lasso peut aussi se ré-exprimer comme le problème de minimisation de la fonction de coût  $J$  :

$$J(\boldsymbol{\theta}) = \text{NLL}(\boldsymbol{\theta}) + \delta \|\boldsymbol{\theta}\|_1, \quad (27)$$

$$= (\mathbf{y} - \mathbf{X}^T \cdot \boldsymbol{\theta})^T \cdot (\mathbf{y} - \mathbf{X}^T \cdot \boldsymbol{\theta}) + \delta \|\boldsymbol{\theta}\|_1, \quad (28)$$

$$= \sum_{i=1}^n (y^{(i)} - \boldsymbol{\theta}^T \cdot \mathbf{x}^{(i)})^2 + \delta \sum_{j=1}^d |\theta_j|, \quad (29)$$

$$= \sum_{i=1}^n \left( y^{(i)} - \theta_j x_j^{(i)} - \boldsymbol{\theta}_{-j}^T \cdot \mathbf{x}_{-j}^{(i)} \right)^2 + \delta \sum_{j=1}^d |\theta_j| \quad (30)$$

Sélection d'attributs : parcimonie (*sparsity*)

- Lasso : on a une fonction de coût  $J$  convexe, on va donc dériver :

$$\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) = \frac{\partial}{\partial \theta_j} \sum_{i=1}^n \left( y^{(i)} - \theta_j x_j^{(i)} - \boldsymbol{\theta}_{-j}^\top \cdot \mathbf{x}_{-j}^{(i)} \right)^2 + \delta \frac{\partial}{\partial \theta_j} \sum_{j=1}^d |\theta_j|$$

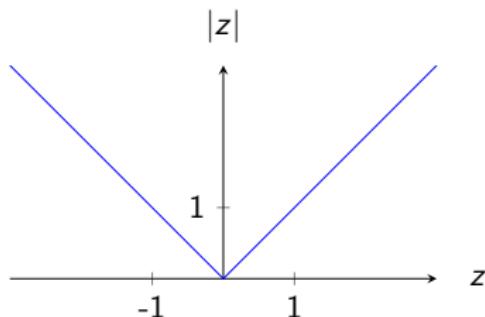
## Sélection d'attributs : parcimonie (*sparsity*)

- **Problème** : la valeur absolue n'est **pas dérivable** en zéro.



Sélection d'attributs : parcimonie (*sparsity*)

- **Problème** : la valeur absolue n'est pas dérivable en zéro.



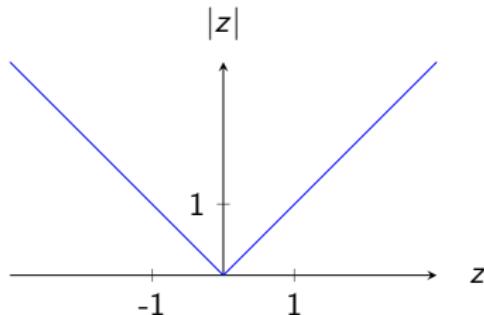
$$\frac{d}{dz} |z| = \begin{cases} -1 & \text{si } z < 0 \\ ?? & \text{si } z = 0 \\ 1 & \text{si } z > 0 \end{cases}$$

## Sélection d'attributs : parcimonie (*sparsity*)

- Nouveau concept : le sous-différentiel.

Sélection d'attributs : parcimonie (*sparsity*)

- Nouveau concept : le sous-différentiel.



$$\partial|z| = \begin{cases} -1 & \text{si } z < 0 \\ [-1; 1] & \text{si } z = 0 \\ 1 & \text{si } z > 0 \end{cases}$$

## Sélection d'attributs : parcimonie (*sparsity*)

- Calculons le sous-différentiel de  $J$  par rapport à  $\theta_j$  :

$$\begin{aligned}\partial_{\theta_j} J &= A_j \theta_j - B_j + \delta \times \partial |\theta_j|, \\ &= \begin{cases} A_j \theta_j - B_j - \delta & \text{si } \theta_j < 0 \\ [-B_j - \delta; -B_j + \delta] & \text{si } \theta_j = 0 \\ A_j \theta_j - B_j + \delta & \text{si } \theta_j > 0 \end{cases} \quad (31)\end{aligned}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Calculons le sous-différentiel de  $J$  par rapport à  $\theta_j$  :

$$\begin{aligned}\partial_{\theta_j} J &= A_j \theta_j - B_j + \delta \times \partial |\theta_j|, \\ &= \begin{cases} A_j \theta_j - B_j - \delta & \text{si } \theta_j < 0 \\ [-B_j - \delta; -B_j + \delta] & \text{si } \theta_j = 0 \\ A_j \theta_j - B_j + \delta & \text{si } \theta_j > 0 \end{cases} \quad (31)\end{aligned}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Calculons le sous-différentiel de  $J$  par rapport à  $\theta_j$  :

$$\begin{aligned}\partial_{\theta_j} J &= A_j \theta_j - B_j + \delta \times \partial |\theta_j|, \\ &= \begin{cases} A_j \theta_j - B_j - \delta & \text{si } \theta_j < 0 \\ [-B_j - \delta; -B_j + \delta] & \text{si } \theta_j = 0 \\ A_j \theta_j - B_j + \delta & \text{si } \theta_j > 0 \end{cases} \quad (31)\end{aligned}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Calculons le sous-différentiel de  $J$  par rapport à  $\theta_j$  : et maintenant ?
- Comme d'habitude, notre estimé de  $\theta_j$  sera la valeur  $\hat{\theta}_j$  telle que

$$\partial_{\hat{\theta}_j} J = 0,$$

(32)

Sélection d'attributs : parcimonie (*sparsity*)

- Calculons le sous-différentiel de  $J$  par rapport à  $\theta_j$  : et maintenant ?
- Comme d'habitude, notre estimé de  $\theta_j$  sera la valeur  $\hat{\theta}_j$  telle que

$$\begin{aligned} \partial_{\hat{\theta}_j} J &= 0, \\ \Leftrightarrow \begin{cases} A_j \hat{\theta}_j - B_j - \delta & \text{si } \theta_j < 0 \\ [-B_j - \delta; -B_j + \delta] & \text{si } \theta_j = 0 \\ A_j \hat{\theta}_j - B_j + \delta & \text{si } \theta_j > 0 \end{cases} &= 0 \end{aligned} \quad (32)$$

Sélection d'attributs : parcimonie (*sparsity*)

- Calculons le sous-différentiel de  $J$  par rapport à  $\theta_j$  : et maintenant ?
- Comme d'habitude, notre estimé de  $\theta_j$  sera la valeur  $\hat{\theta}_j$  telle que

$$\begin{aligned} \partial_{\hat{\theta}_j} J &= 0, \\ \Leftrightarrow \begin{cases} A_j \hat{\theta}_j - B_j - \delta & \text{si } \theta_j < 0 \\ [-B_j - \delta; -B_j + \delta] & \text{si } \theta_j = 0 \\ A_j \hat{\theta}_j - B_j + \delta & \text{si } \theta_j > 0 \end{cases} &= 0 \quad (32) \end{aligned}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Commençons par le 1<sup>er</sup> cas : si  $\hat{\theta}_j < 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j - \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j + \delta}{A_j}. \end{aligned} \tag{33}$$

- Quand choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j < 0 &\Leftrightarrow \frac{B_j + \delta}{A_j} < 0, \\ &\Leftrightarrow B_j + \delta < 0, \\ &\Leftrightarrow B_j < -\delta. \end{aligned} \tag{34}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Commençons par le 1<sup>er</sup> cas : si  $\hat{\theta}_j < 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j - \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j + \delta}{A_j}. \end{aligned} \tag{33}$$

- Quand choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j < 0 &\Leftrightarrow \frac{B_j + \delta}{A_j} < 0, \\ &\Leftrightarrow B_j + \delta < 0, \\ &\Leftrightarrow B_j < -\delta. \end{aligned} \tag{34}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Commençons par le 1<sup>er</sup> cas : si  $\hat{\theta}_j < 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j - \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j + \delta}{A_j}. \end{aligned} \tag{33}$$

- **Quand** choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j < 0 &\Leftrightarrow \frac{B_j + \delta}{A_j} < 0, \\ &\Leftrightarrow B_j + \delta < 0, \\ &\Leftrightarrow B_j < -\delta. \end{aligned} \tag{34}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Commençons par le 1<sup>er</sup> cas : si  $\hat{\theta}_j < 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j - \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j + \delta}{A_j}. \end{aligned} \tag{33}$$

- **Quand** choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j < 0 &\Leftrightarrow \frac{B_j + \delta}{A_j} < 0, \\ &\Leftrightarrow B_j + \delta < 0, \\ &\Leftrightarrow B_j < -\delta. \end{aligned} \tag{34}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Le 3<sup>ème</sup> cas est similaire : si  $\hat{\theta}_j > 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j + \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j - \delta}{A_j}. \end{aligned} \tag{35}$$

- Quand choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j > 0 &\Leftrightarrow \frac{B_j - \delta}{A_j} > 0, \\ &\Leftrightarrow B_j - \delta > 0, \\ &\Leftrightarrow B_j > \delta. \end{aligned} \tag{36}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Le 3<sup>ème</sup> cas est similaire : si  $\hat{\theta}_j > 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j + \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j - \delta}{A_j}. \end{aligned} \tag{35}$$

- Quand choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j > 0 &\Leftrightarrow \frac{B_j - \delta}{A_j} > 0, \\ &\Leftrightarrow B_j - \delta > 0, \\ &\Leftrightarrow B_j > \delta. \end{aligned} \tag{36}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Le 3<sup>ème</sup> cas est similaire : si  $\hat{\theta}_j > 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j + \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j - \delta}{A_j}. \end{aligned} \tag{35}$$

- Quand choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j > 0 &\Leftrightarrow \frac{B_j - \delta}{A_j} > 0, \\ &\Leftrightarrow B_j - \delta > 0, \\ &\Leftrightarrow B_j > \delta. \end{aligned} \tag{36}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Le 3<sup>ème</sup> cas est similaire : si  $\hat{\theta}_j > 0$  :

$$\begin{aligned} A_j \hat{\theta}_j - B_j + \delta &= 0, \\ \Leftrightarrow \hat{\theta}_j &= \frac{B_j - \delta}{A_j}. \end{aligned} \tag{35}$$

- Quand choisit-on cet estimé ?

$$\begin{aligned} \hat{\theta}_j > 0 &\Leftrightarrow \frac{B_j - \delta}{A_j} > 0, \\ &\Leftrightarrow B_j - \delta > 0, \\ &\Leftrightarrow B_j > \delta. \end{aligned} \tag{36}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Et pour le 2<sup>ème</sup> cas alors ?? quand  $\hat{\theta}_j = 0$  ?
- Et bien quand  $\hat{\theta}_j = 0$  ... on choisit  $\hat{\theta}_j = 0 !!!$
- Quand choisit-on cet estimé ?

$$\begin{aligned} -B_j - \delta &\leq \partial J(\hat{\theta}_j) = 0 \leq -B_j + \delta, \\ \Leftrightarrow -\delta &\leq B_j \leq \delta. \end{aligned} \tag{37}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Et pour le 2<sup>ème</sup> cas alors ?? quand  $\hat{\theta}_j = 0$  ?
- Et bien quand  $\hat{\theta}_j = 0$  ... on choisit  $\hat{\theta}_j = 0 !!!$
- Quand choisit-on cet estimé ?

$$\begin{aligned} -B_j - \delta &\leq \partial J(\hat{\theta}_j) = 0 \leq -B_j + \delta, \\ \Leftrightarrow -\delta &\leq B_j \leq \delta. \end{aligned} \tag{37}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Et pour le 2<sup>ème</sup> cas alors ?? quand  $\hat{\theta}_j = 0$  ?
- Et bien quand  $\hat{\theta}_j = 0$  ... on choisit  $\hat{\theta}_j = 0 !!!$
- **Quand** choisit-on cet estimé ?

$$\begin{aligned} -B_j - \delta &\leq \partial J(\hat{\theta}_j) = 0 \leq -B_j + \delta, \\ \Leftrightarrow -\delta &\leq B_j \leq \delta. \end{aligned} \tag{37}$$

Sélection d'attributs : parcimonie (*sparsity*)

- Bilan :

$$\hat{\theta}_j = \begin{cases} \frac{B_j + \delta}{A_j} & \text{si } B_j < -\delta \\ 0 & \text{si } B_j \in [-\delta; \delta] \\ \frac{B_j - \delta}{A_j} & \text{si } B_j > \delta \end{cases} \quad (38)$$

- Attention, cela ne donne l'estimé que pour une seule dimension
- Il faut itérer pour les autres.

Sélection d'attributs : parcimonie (*sparsity*)

- Bilan :

$$\hat{\theta}_j = \begin{cases} \frac{B_j + \delta}{A_j} & \text{si } B_j < -\delta \\ 0 & \text{si } B_j \in [-\delta; \delta] \\ \frac{B_j - \delta}{A_j} & \text{si } B_j > \delta \end{cases} \quad (38)$$

- Attention, cela ne donne l'estimé que pour une seule dimension
- Il faut itérer pour les autres.

Sélection d'attributs : parcimonie (*sparsity*)

- Bilan :

$$\hat{\theta}_j = \begin{cases} \frac{B_j + \delta}{A_j} & \text{si } B_j < -\delta \\ 0 & \text{si } B_j \in [-\delta; \delta] \\ \frac{B_j - \delta}{A_j} & \text{si } B_j > \delta \end{cases} \quad (38)$$

- Attention, cela ne donne l'estimé que pour une seule dimension
- Il faut **itérer** pour les autres.

## Lasso

Initialiser  $\hat{\theta}$  et  $\delta$ .

Calculer  $A_j$ ,  $\forall j : A_j \leftarrow 2 \sum_{i=1}^n (x_j^{(i)})^2$ .

**while** pas convergé **do**

**for**  $j$  de 1 à  $d$  **do**

    Calculer  $B_j \leftarrow 2 \sum_{i=1}^n (y^{(i)} - \hat{\theta}_{-j}^T \cdot x_{-j}^{(i)}) x_j^{(i)}$ .

**if**  $B_j < -\delta$  **then**

      Mettre à jour le paramètre :  $\hat{\theta}_j \leftarrow \frac{B_j + \delta}{A_j}$ .

**else if**  $B_j > \delta$  **then**

      Mettre à jour le paramètre :  $\hat{\theta}_j \leftarrow \frac{B_j - \delta}{A_j}$ .

**else**

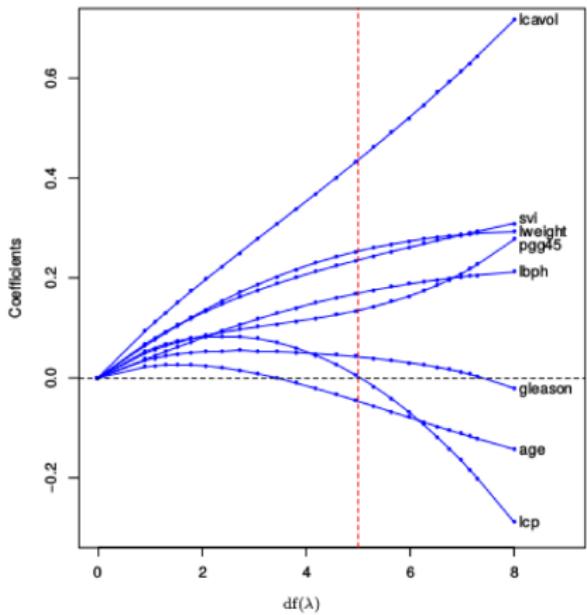
      Mettre à jour le paramètre :  $\hat{\theta}_j \leftarrow 0$ .

**end if**

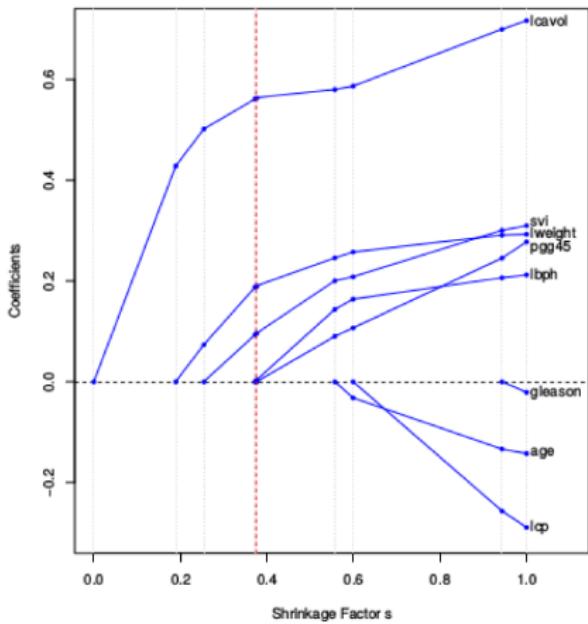
**end for**

**end while**

Ridge



Lasso



## Sélection d'attributs : parcimonie (*sparsity*)

- Les modèles Ridge et Lasso sont les solutions du problème de minimisation de la NLL à laquelle on a ajouté une pénalité  $L_2$  et  $L_1$  respectivement.
- La norme  $L_p$  s'écrit :

$$\|\mathbf{x}\|_p = \left( \sum_{j=1}^d |x_j|^p \right)^{1/p}, \quad \forall \mathbf{x} \in \mathbb{R}^d. \quad (39)$$

- Du coup .. si  $p = 1$  est plus *sparse* que  $p = 2$ .. est ce que choisir  $p$  encore plus petit ne serait pas mieux ?
- Oui, mais on perd la *convexité* de la fonction  $J$  (minima locaux).

## Sélection d'attributs : parcimonie (*sparsity*)

- Les modèles Ridge et Lasso sont les solutions du problème de minimisation de la NLL à laquelle on a ajouté une pénalité  $L_2$  et  $L_1$  respectivement.
- La norme  $L_p$  s'écrit :

$$\|\mathbf{x}\|_p = \left( \sum_{j=1}^d |x_j|^p \right)^{1/p}, \quad \forall \mathbf{x} \in \mathbb{R}^d. \quad (39)$$

- Du coup .. si  $p = 1$  est plus *sparse* que  $p = 2$ .. est ce que choisir  $p$  encore plus petit ne serait pas mieux ?
- Oui, mais on perd la *convexité* de la fonction  $J$  (minima locaux).

## Sélection d'attributs : parcimonie (*sparsity*)

- Les modèles Ridge et Lasso sont les solutions du problème de minimisation de la NLL à laquelle on a ajouté une pénalité  $L_2$  et  $L_1$  respectivement.
- La norme  $L_p$  s'écrit :

$$\|\mathbf{x}\|_p = \left( \sum_{j=1}^d |x_j|^p \right)^{1/p}, \quad \forall \mathbf{x} \in \mathbb{R}^d. \quad (39)$$

- Du coup .. si  $p = 1$  est plus *sparse* que  $p = 2$ .. est ce que choisir  $p$  encore plus petit ne serait pas mieux ?
- Oui, mais on perd la *convexité* de la fonction  $J$  (minima locaux).

Sélection d'attributs : parcimonie (*sparsity*)

- Les modèles Ridge et Lasso sont les solutions du problème de minimisation de la NLL à laquelle on a ajouté une pénalité  $L_2$  et  $L_1$  respectivement.
- La norme  $L_p$  s'écrit :

$$\|\mathbf{x}\|_p = \left( \sum_{j=1}^d |x_j|^p \right)^{1/p}, \quad \forall \mathbf{x} \in \mathbb{R}^d. \quad (39)$$

- Du coup .. si  $p = 1$  est plus *sparse* que  $p = 2$ .. est ce que choisir  $p$  encore plus petit ne serait pas mieux ?
- Oui, mais on perd la *convexité* de la fonction  $J$  (minima locaux).

Sélection d'attributs : parcimonie (*sparsity*)

- Pénalité  $L_0$  : définition

- La norme  $L_p$  s'écrit :

$$\|\mathbf{x}\|_0 = \sum_{j=1}^d \mathbb{I}_0(x_j), \quad \forall \mathbf{x} \in \mathbb{R}^d. \quad (40)$$

Sélection d'attributs : parcimonie (*sparsity*)

- Pénalité  $L_0$  : définition

- La norme  $L_p$  s'écrit :

$$\|\mathbf{x}\|_0 = \sum_{j=1}^d 1 - \mathbb{I}_0(x_j), \quad \forall \mathbf{x} \in \mathbb{R}^d. \quad (40)$$

# Plan du chapitre

- 1 Pré-traitements basiques
- 2 Réduction de dimension par apprentissage de représentation
- 3 Sélection d'attributs
- 4 Apprentissage non supervisé
- 5 Conclusions
- 6 Annexes

## Non supervisé : K-means

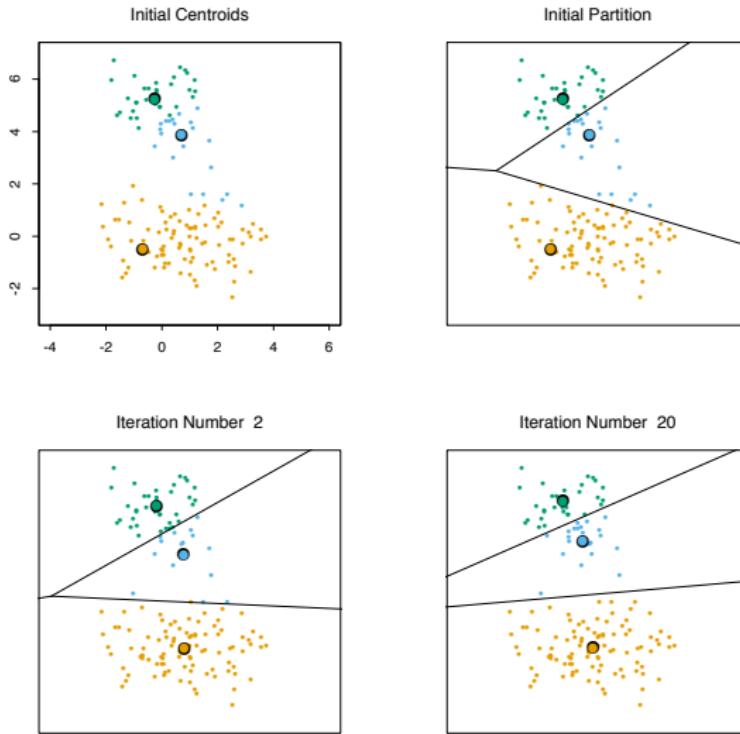
**Objectif** : identifier l'existence de groupes/agrégats/clusters au sein de données non étiquetées (*classes  $c_k$  inconnues*)

### Principe :

- Choisir le nombre de clusters  $K$ ,
- À partir de  $K$  centres initiaux  $\mu_k$  (points dans l'*input space*) :
  - pour chaque  $1 \leq k \leq K$ , identifier l'ensemble des points  $c_k$  plus proche de  $\mu_k$  que de tout autre centre,
  - remplacer  $\mu_k$  par le centre (moyenne) de  $c_k$
- Itérer jusqu'à convergence (démontrée mathématiquement).

# Non supervisé : K-means

**Illustration** fig. tirée de *Elements of Statistical Learning*, Hastie and Tibshirani 2008.



## Non supervisé : K-means

### Principe

- **Points initiaux** :  $K$  entrées choisies au hasard...  
⇒ lancer plusieurs fois l'algorithme et conserver le meilleur résultat,
- **Meilleur résultat** : minimise  $\sum_{k=1}^K \sum_{C(i)=k} \|\mathbf{x}^{(i)} - \boldsymbol{\mu}_k\|^2$
- **Limitations** :
  - choix de  $K$  ? non imbrication lorsque  $K$  décroît...
  - importance relative des clusters, hiérarchie ?
- **Version soft générative** : algorithme EM (Expectation-Minimization)  
⇒ cours Estimation, Hastie p. 272, Bishop p. 430) pour un modèle de mélange de  $K$  distributions.

## Non supervisé : K-means

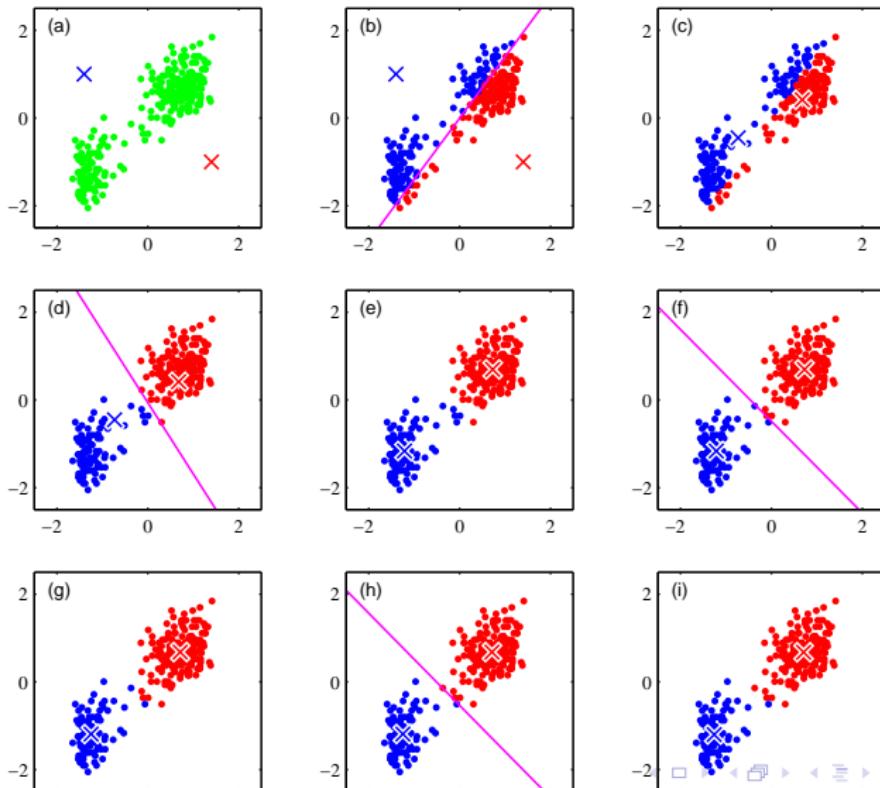
**Exemple** : Geyser Old Faithful

**Données :**

- 272 mesures, éruptions du geyser Old Faithful à Yellowstone
  - $x_1$  : durée des éruptions en minutes,
  - $x_2$  : temps d'attente avant la prochaine éruption
- identification de 2 types d'éruptions

## Non supervisé : K-means

## Exemple : Geyser Old Faithful

fig. tirée de *Pattern Recognition and Machine Learning*, Bishop 2006.

## Non supervisé : K-means

**Exemple** : Génétique et tumeurs cancéreuses

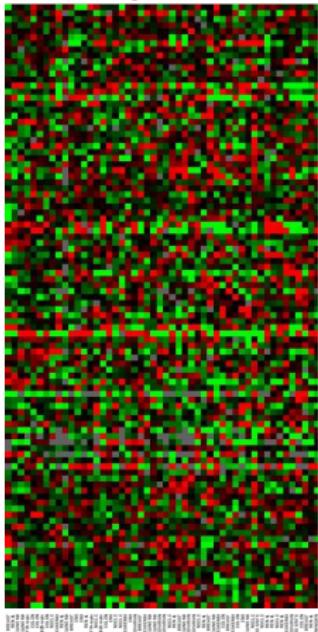


fig. tirée de *Elements of Statistical Learning*, Hastie and Tibshirani 2008.

### Données :

- Matrice  $6830 \times 64$  nombres réels mesurant :
  - l'expression d'un gène particulier = lignes
  - chez un sujet donné = colonnes
- 64 vecteurs d'entrée  $\mathbf{x}^{(i)}$  de  $d = 6830$  valeurs.
- étiquettes/cibles :  $y^{(i)}$  = breast (cancer du sein), melanoma (peau)...

⇒ utilisées a posteriori pour vérifier la pertinence du clustering.

## Non supervisé : K-means

**Exemple** : Génétique et tumeurs cancéreuses

Résultats pour K=3 :

Cluster	Breast	CNS	Colon	K562	Leukemia	MCF7
1	3	5	0	0	0	0
2	2	0	0	2	6	2
3	2	0	7	0	0	0
Cluster	Melanoma	NSCLC	Ovarian	Prostate	Renal	Unknown
1	1	7	6	2	9	1
2	7	2	0	0	0	0
3	0	0	0	0	0	0

## Non supervisé : K-means

**Exemple** : Quantification vectorielle d'images

1 pixel =  $\mathbf{x}^{(i)} = (x_R, x_V, x_B)$   $\Rightarrow$  clustering 3d

utilisation de  $K$  couleurs seulement

$K = 2$



$K = 3$



$K = 10$



Original image



## Non supervisé : K-medoids

**Objectif** : identifier l'existence de clusters au sein de données non étiquetées, à partir des *distances* entre les points seulement

### Principe :

- Choisir le nombre de clusters  $K$ ,
- Pour une affectation des points  $\mathbf{x}^{(i)}$  aux clusters  $c_k \Leftrightarrow \boldsymbol{\mu}_k$  :
  - trouver l'observation minimisant la distance à tous les autres points au sein de chaque cluster :

$$1 \leq k \leq K, \quad i_k^* = \arg \min_{i \in c_k} \sum_{q \in c_k} d(\mathbf{x}^{(i)}, \mathbf{x}^{(q)})$$

- les centres deviennent :  $\boldsymbol{\mu}_k = \mathbf{x}^{(i_k^*)}$
- affecter chaque entrée  $\mathbf{x}^{(i)}$  à la classe du centre le plus proche
- Itérer jusqu'à convergence (démontrée mathématiquement).

## Non supervisé : K-medoids

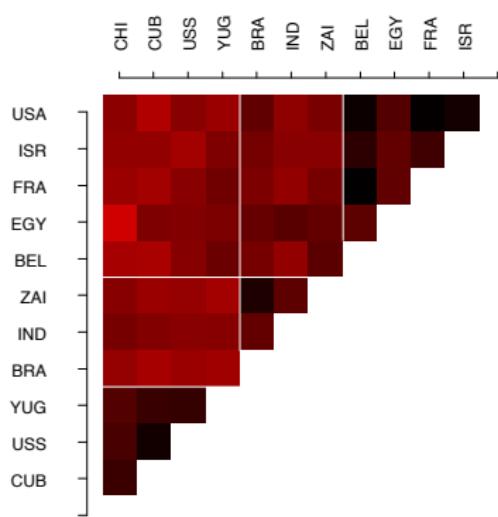
**Exemple** : en 1990, des étudiants de Sciences-Po ont noté sur 10 les "distances" entre 2 pays parmi 12 pays (Belgique, Brésil, Chili...)

	BEL	BRA	CHI	CUB	EGY	FRA	IND	ISR	USA	USS	YUG
BRA	5.58										
CHI	7.00	6.50									
CUB	7.08	7.00	3.83								
EGY	4.83	5.08	8.17	5.83							
FRA	2.17	5.75	6.67	6.92	4.92						
IND	6.42	5.00	5.58	6.00	4.67	6.42					
ISR	3.42	5.50	6.42	6.42	5.00	3.92	6.17				
USA	2.50	4.92	6.25	7.33	4.50	2.25	6.33	2.75			
USS	6.08	6.67	4.25	2.67	6.00	6.17	6.17	6.92	6.17		
YUG	5.25	6.83	4.50	3.75	5.75	5.42	6.08	5.83	6.67	3.67	
ZAI	4.75	3.00	6.08	6.67	5.00	5.58	4.83	6.17	5.67	6.50	6.92

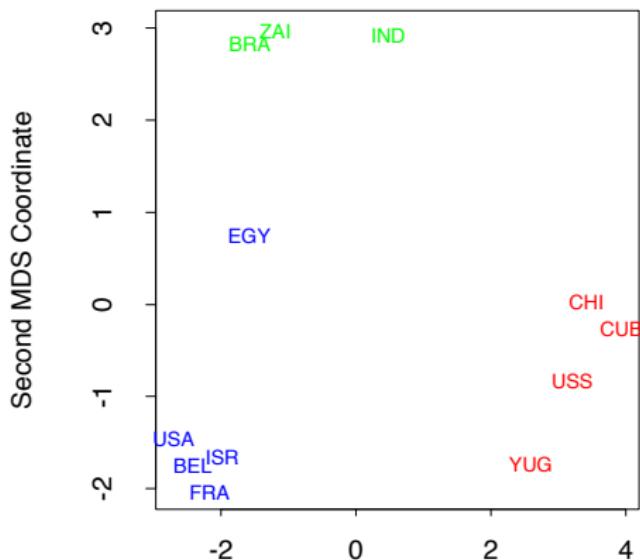
## Non supervisé : K-medoids

**Exemple** : en 1990, des étudiants de Sciences-Po ont noté sur 10 les "distances" entre 2 pays parmi 12 pays (Belgique, Brésil, Chili...)

Clustering en  $K = 3$  groupes + représentation simplifiée :



Reordered Dissimilarity Matrix



First MDS Coordinate

## Non supervisé : Spectral Clustering

- Soit  $\mathbf{W} = [W_{ij}]$  une matrice de **similarité** entre exemples :

$$W_{ij} = \exp\left(\frac{-\|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|^2}{a}\right) \text{ (par exemple)}$$

- On construit le **graphe**  $G$  tel que
  - chaque exemple  $\mathbf{x}^{(i)}$  est un noeud,
  - chaque arrête relie deux noeuds si  $W_{ij} >$  seuil,
  - les arrêtes existantes sont pondérées par  $W_{ij}$
- Clustering  $\Leftrightarrow$  Partitionnement du graphe.
- Bon partitionnement : couper les arrêtes de poids faibles.

## Non supervisé : Spectral Clustering

- Soit  $\mathbf{L}$  le Laplacien du graphe  $G$  :

$$\mathbf{L} = \begin{pmatrix} \sum_i W_{0i} & 0 & \dots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \sum_i W_{ni} \end{pmatrix} - \mathbf{W}$$

- Soient  $(\mathbf{z}^{(j)})_{j=1}^m$  les  $m$  vecteurs propres de  $\mathbf{L}$  associées aux plus petites valeurs propres et soit la matrice de taille  $n \times m$

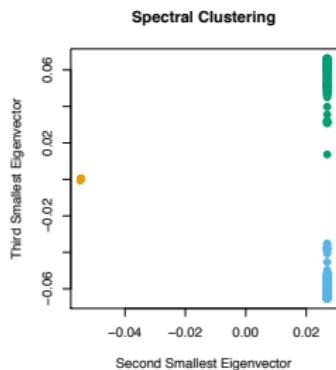
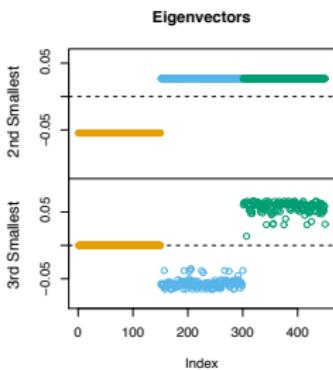
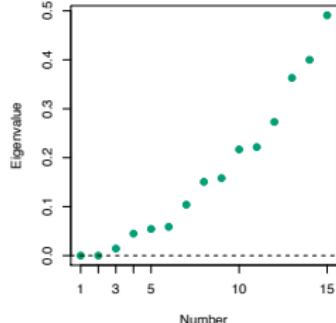
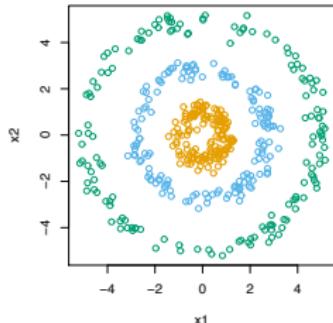
$$\mathbf{Z} = \begin{pmatrix} \left[ \mathbf{z}^{(1)} \right] & \dots & \left[ \mathbf{z}^{(m)} \right] \end{pmatrix} \quad (41)$$

## Non supervisé : Spectral Clustering

- Nouvelle représentation :  $\tilde{\mathbf{z}}^{(i)} = \text{ligne}_i(\mathbf{Z}) = Z_{i,:}$  pour l'exemple  $\mathbf{x}^{(i)}$ .
- On peut ensuite, par exemple, utiliser K-means sur les  $\tilde{\mathbf{z}}^{(i)}$ .

# Non supervisé : Spectral Clustering

**Exemple** : fig. tirée de *Elements of Statistical Learning*, Hastie and Tibshirani 2008.



## Messages importants du chapitre :

- Les **pré-traitements** ont un impact fort sur les performances globables.
- Ils permettent de tirer **meilleur parti** des algorithmes vus dans les autres chapitres et de **contourner certains problèmes** (données manquantes, mixtes, etc.).
- L'**ACP** est une méthode **non-supervisée** qui permet de **réduire** drastiquement la **dimension** et **décorrèle** les composantes.
- .. mais elle peut **effacer une dimension discriminante**.
- Les modèles qui utilisent une **pénalité sparse** encodent naturellement une **sélection d'attributs** qui est apprise.
- Les méthodes **non supervisées** reposent essentiellement sur le regroupement des exemples à l'aide de calculs de similarités bien sentis.

## Messages importants du chapitre :

- Les **pré-traitements** ont un impact fort sur les performances globables.
- Ils permettent de tirer **meilleur parti** des algorithmes vus dans les autres chapitres et de **contourner certains problèmes** (données manquantes, mixtes, etc.).
- L'**ACP** est une méthode **non-supervisée** qui permet de **réduire** drastiquement la **dimension** et **décorrèle** les composantes.
- .. mais elle peut **effacer une dimension discriminante**.
- Les modèles qui utilisent une **pénalité sparse** encodent naturellement une **sélection d'attributs** qui est apprise.
- Les méthodes **non supervisées** reposent essentiellement sur le regroupement des exemples à l'aide de calculs de similarités bien sentis.

## Messages importants du chapitre :

- Les pré-traitements ont un impact fort sur les performances globables.
- Ils permettent de tirer meilleur parti des algorithmes vus dans les autres chapitres et de contourner certains problèmes (données manquantes, mixtes, etc.).
- L'ACP est une méthode non-supervisée qui permet de réduire drastiquement la dimension et décorrèle les composantes.
  - .. mais elle peut effacer une dimension discriminante.
- Les modèles qui utilisent une pénalité sparse encodent naturellement une sélection d'attributs qui est apprise.
- Les méthodes non supervisées reposent essentiellement sur le regroupement des exemples à l'aide de calculs de similarités bien sentis.

## Messages importants du chapitre :

- Les pré-traitements ont un impact fort sur les performances globales.
- Ils permettent de tirer meilleur parti des algorithmes vus dans les autres chapitres et de contourner certains problèmes (données manquantes, mixtes, etc.).
- L'ACP est une méthode non-supervisée qui permet de réduire drastiquement la dimension et décorrèle les composantes.
- .. mais elle peut effacer une dimension discriminante.
- Les modèles qui utilisent une pénalité sparse encodent naturellement une sélection d'attributs qui est apprise.
- Les méthodes non supervisées reposent essentiellement sur le regroupement des exemples à l'aide de calculs de similarités bien sentis.

## Messages importants du chapitre :

- Les pré-traitements ont un impact fort sur les performances globales.
- Ils permettent de tirer meilleur parti des algorithmes vus dans les autres chapitres et de contourner certains problèmes (données manquantes, mixtes, etc.).
- L'ACP est une méthode non-supervisée qui permet de réduire drastiquement la dimension et décorrèle les composantes.
- .. mais elle peut effacer une dimension discriminante.
- Les modèles qui utilisent une pénalité sparse encodent naturellement une sélection d'attributs qui est apprise.
- Les méthodes non supervisées reposent essentiellement sur le regroupement des exemples à l'aide de calculs de similarités bien sentis.

## Messages importants du chapitre :

- Les pré-traitements ont un impact fort sur les performances globales.
- Ils permettent de tirer meilleur parti des algorithmes vus dans les autres chapitres et de contourner certains problèmes (données manquantes, mixtes, etc.).
- L'ACP est une méthode non-supervisée qui permet de réduire drastiquement la dimension et décorrèle les composantes.
- .. mais elle peut effacer une dimension discriminante.
- Les modèles qui utilisent une pénalité sparse encodent naturellement une sélection d'attributs qui est apprise.
- Les méthodes non supervisées reposent essentiellement sur le regroupement des exemples à l'aide de calculs de similarités bien sentis.

ACP : Pourquoi la val. propre  $\lambda_j$  est-elle la variance des données selon la “nouvelle dimension  $j$ ” dont le vecteur directeur est le vecteur propre  $\mathbf{q}_j$  ?

ACP : Pourquoi la val. propre  $\lambda_j$  est-elle la variance des données selon la "nouvelle dimension  $j$ " dont le vecteur directeur est le vecteur propre  $\mathbf{q}_j$  ?

En fait, c'est comme ça qu'on a construit la transformation de l'ACP :  
On cherche  $\mathbf{q}_1$  tel que

$$\text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \max_{\mathbf{q} \text{ t.q. } \|\mathbf{q}\|_2=1} \text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] \quad (42)$$

ACP : Pourquoi la val. propre  $\lambda_j$  est-elle la variance des données selon la "nouvelle dimension  $j$ " dont le vecteur directeur est le vecteur propre  $\mathbf{q}_j$  ?

En fait, c'est comme ça qu'on a construit la transformation de l'ACP :  
On cherche  $\mathbf{q}_1$  tel que

$$\text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \max_{\mathbf{q} \text{ t.q. } \|\mathbf{q}\|_2=1} \text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] \quad (42)$$

On comprend cette équation comme un problème de maximisation où je cherche une droite dont le vecteur directeur  $\mathbf{q}_1$  est de norme unitaire et les projetés de mes exemples sur cette droite ont une variance maximale.

ACP : Pourquoi la val. propre  $\lambda_j$  est-elle la variance des données selon la "nouvelle dimension  $j$ " dont le vecteur directeur est le vecteur propre  $\mathbf{q}_j$  ?

En fait, c'est comme ça qu'on a construit la transformation de l'ACP :  
On cherche  $\mathbf{q}_1$  tel que

$$\text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \max_{\mathbf{q} \text{ t.q. } \|\mathbf{q}\|_2=1} \text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] \quad (42)$$

On comprend cette équation comme un problème de maximisation où je cherche une droite dont le vecteur directeur  $\mathbf{q}_1$  est de norme unitaire et les projetés de mes exemples sur cette droite ont une variance maximale.

ACP : Pourquoi la val. propre  $\lambda_j$  est-elle la variance des données selon la "nouvelle dimension  $j$ " dont le vecteur directeur est le vecteur propre  $\mathbf{q}_j$  ?

En fait, c'est comme ça qu'on a construit la transformation de l'ACP :  
On cherche  $\mathbf{q}_1$  tel que

$$\text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \max_{\mathbf{q} \text{ t.q. } \|\mathbf{q}\|_2=1} \text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] \quad (42)$$

On comprend cette équation comme un problème de maximisation où je cherche une droite dont le vecteur directeur  $\mathbf{q}_1$  est de norme unitaire et les projetés de mes exemples sur cette droite ont une variance maximale.

Le projeté de l'exemple  $\mathbf{x}^{(i)}$  sur cette droite est  $\mathbf{q}_1^T \cdot \mathbf{x}^{(i)}$ .

## ACP

Partons de la définition du terme de variance :

$$\text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \frac{1}{n} \sum_{i=1}^n \left( \mathbf{q}^T \cdot \mathbf{x}^{(i)} - \mathbb{E} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] \right)^2 \quad (43)$$

## ACP

Partons de la définition du terme de variance :

$$\text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \frac{1}{n} \sum_{i=1}^n \left( \mathbf{q}^T \cdot \mathbf{x}^{(i)} - \mathbb{E} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] \right)^2 \quad (43)$$

Le terme d'espérance est la moyenne des projections :

$$\mathbb{E} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \frac{1}{n} \sum_{i=1}^n \mathbf{q}_1^T \cdot \mathbf{x}^{(i)}, \quad (44)$$

$$= \mathbf{q}_1^T \cdot \frac{1}{n} \sum_{i=1}^n \mathbf{x}^{(i)}, \quad (45)$$

$$= \mathbf{q}_1^T \cdot \boldsymbol{\mu}_x, \quad (46)$$

où  $\boldsymbol{\mu}_x$  est la moyenne des exemples d'apprentissage.

## ACP

Notre terme de variance devient :

$$\begin{aligned}
 \text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] &= \frac{1}{n} \sum_{i=1}^n \left( \mathbf{q}^T \cdot \mathbf{x}^{(i)} - \mathbf{q}^T \cdot \boldsymbol{\mu}_x \right)^2, \\
 &= \frac{1}{n} \sum_{i=1}^n \mathbf{q}^T \cdot \left( \mathbf{x}^{(i)} - \boldsymbol{\mu}_x \right) \cdot \left( \mathbf{x}^{(i)} - \boldsymbol{\mu}_x \right)^T \cdot \mathbf{q}, \\
 &= \mathbf{q}^T \cdot \left( \frac{1}{n} \sum_{i=1}^n \left( \mathbf{x}^{(i)} - \boldsymbol{\mu}_x \right) \cdot \left( \mathbf{x}^{(i)} - \boldsymbol{\mu}_x \right)^T \right) \cdot \mathbf{q}, \\
 &= \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q},
 \end{aligned} \tag{47}$$

et  $\boldsymbol{\Sigma}$  est la matrice de variance-covariance de nos exemples d'apprentissage.

## ACP

Avec l'astuce du **Langrangien**, j'introduis un paramètre  $\lambda$  et mon problème de maximisation devient :

$$\text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \max_{\mathbf{q}} \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \|\mathbf{q}\|_2), \quad (48)$$

$$= \max_{\mathbf{q}} \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \mathbf{q}^T \cdot \mathbf{q}). \quad (49)$$

## ACP

Avec l'astuce du **Langrangien**, j'introduis un paramètre  $\lambda$  et mon problème de maximisation devient :

$$\text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \max_{\mathbf{q}} \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \|\mathbf{q}\|_2), \quad (48)$$

$$= \max_{\mathbf{q}} \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \mathbf{q}^T \cdot \mathbf{q}). \quad (49)$$

Cette expression est une **forme quadratique convexe**. Elle est donc maximale en un unique point  $\mathbf{q}$  où ses dérivées sont nulles. On résout donc :

$$\frac{d}{d\mathbf{q}} \left\{ \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \mathbf{q}^T \cdot \mathbf{q}) \right\} = 0, \quad (50)$$

$$\Leftrightarrow 2\boldsymbol{\Sigma} \cdot \mathbf{q} - 2\lambda\mathbf{q} = 0, \quad (51)$$

$$\Leftrightarrow \boldsymbol{\Sigma} \cdot \mathbf{q} = \lambda\mathbf{q}. \quad (52)$$

## ACP

Avec l'astuce du **Langrangien**, j'introduis un paramètre  $\lambda$  et mon problème de maximisation devient :

$$\text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \max_{\mathbf{q}} \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \|\mathbf{q}\|_2), \quad (48)$$

$$= \max_{\mathbf{q}} \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \mathbf{q}^T \cdot \mathbf{q}). \quad (49)$$

Cette expression est une **forme quadratique convexe**. Elle est donc maximale en un unique point  $\mathbf{q}$  où ses dérivées sont nulles. On résout donc :

$$\frac{d}{d\mathbf{q}} \left\{ \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} + \lambda (1 - \mathbf{q}^T \cdot \mathbf{q}) \right\} = 0, \quad (50)$$

$$\Leftrightarrow 2\boldsymbol{\Sigma} \cdot \mathbf{q} - 2\lambda\mathbf{q} = 0, \quad (51)$$

$$\Leftrightarrow \boldsymbol{\Sigma} \cdot \mathbf{q} = \lambda\mathbf{q}. \quad (52)$$

La relation ci-dessus est la définition de ce qu'est être un vecteur propre  $\mathbf{q}$  associé à la valeur propre  $\lambda$ .

## ACP

Montrons que la valeur propre  $\lambda$  est la variance des données projetées sur  $\mathbf{q}$  :

$$\boldsymbol{\Sigma} \cdot \mathbf{q} = \lambda \mathbf{q}, \quad (53)$$

$$\Leftrightarrow \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} = \lambda \mathbf{q}^T \cdot \mathbf{q}, \quad (54)$$

$$\Leftrightarrow \text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \lambda. \quad (55)$$

## ACP

Montrons que la valeur propre  $\lambda$  est la variance des données projetées sur  $\mathbf{q}$  :

$$\boldsymbol{\Sigma} \cdot \mathbf{q} = \lambda \mathbf{q}, \quad (53)$$

$$\Leftrightarrow \mathbf{q}^T \cdot \boldsymbol{\Sigma} \cdot \mathbf{q} = \lambda \mathbf{q}^T \cdot \mathbf{q}, \quad (54)$$

$$\Leftrightarrow \text{Var} \left[ \left\{ \mathbf{q}_1^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] = \lambda. \quad (55)$$

Pour finir, il suffit d'**itérer** en cherchant le prochain vecteur  $\mathbf{q}_2$  qui maximisera encore la variance après projection mais qui sera **orthogonal** (donc décorrélé) à  $\mathbf{q}_1$  :

$$\mathbf{q}_2 = \arg \max_{\mathbf{q} \text{ t.q. } \|\mathbf{q}\|_2=1 \text{ et } \mathbf{q} \cdot \mathbf{q}_1=0} \text{Var} \left[ \left\{ \mathbf{q}^T \cdot \mathbf{x}^{(i)} \right\}_{i=1}^n \right] \quad (56)$$