

# Classification de Données Fonctionnelles

Benjamin Auder

CEA - UPMC

6 mai 2009

Thèse depuis 02/2008

Directeur de thèse : Gérard Biau (UPMC)

Encadrant CEA : Bertrand looss (CEA)

# Contexte industriel CEA

Choc thermique pressurisé :

Code thermo-hydraulique coûteux en temps, déterminant les évolutions temporelles de paramètres  $\varphi$  dans l'espace annulaire de la cuve.

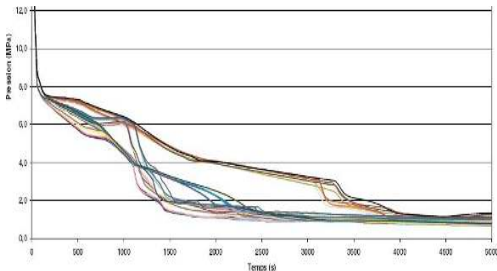


FIG.: Transitoire de pression - Cathare.

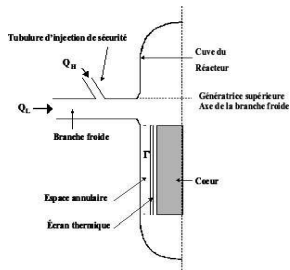


FIG.: Schéma de la zone modélisée

**Objectif :** prédiction de *données fonctionnelles* via un métamodèle.

# Hypothèses

Code de calcul  $B : \mathcal{X} \subset \mathbb{R}^d \longrightarrow \mathcal{Y}$ , avec  $\mathcal{X} = \prod_{i=1}^d [a_i, b_i]$ .

$\mathcal{Y}$  est un espace fonctionnel :  $\mathcal{Y} \subset \mathcal{F}([a, b], \mathbb{R})$ .

## Hypothèse 1

Les sorties du code sont "lisses" :  $\mathcal{Y} = \mathcal{C}^m([a, b], \mathbb{R})$  avec  $m \geq 0$ .

## Hypothèse 2

$B$  est *continu par morceaux* : continu sur  $\mathcal{X}_i$  ouvert connexe,  
 $\mathcal{X} = \text{adh} \left( \bigsqcup_{i=1}^k \mathcal{X}_i \right)$ .

**H1** : l'évolution d'un paramètre physique est au minimum continue.

**H2** : permet de prendre en compte les discontinuités du code.

$\Rightarrow$  Il faut retrouver ces domaines de continuité  $\rightarrow$  *clustering*.

# Illustration

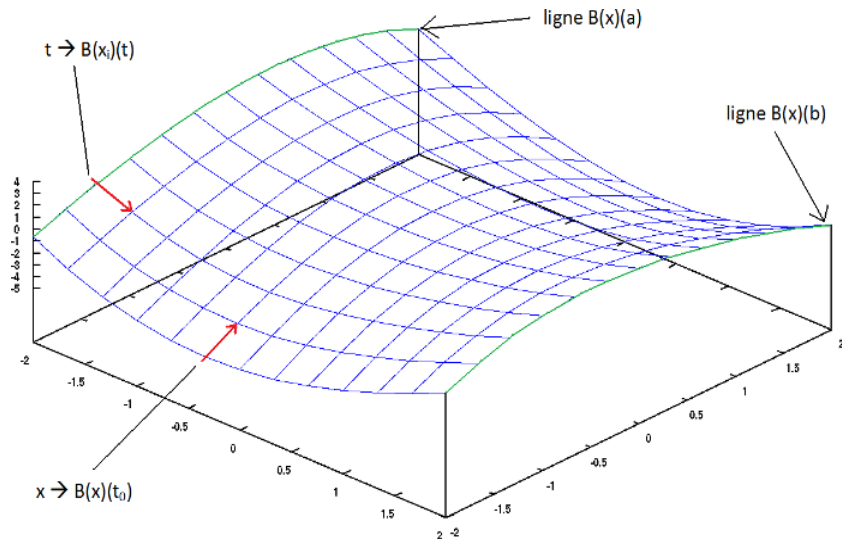


FIG.: Visualisation dans le cas particulier  $d = 1$ .

# Plan

## 1 Introduction

## 2 Clustering

- Problèmes posés
- Fonction objectif
- Commute Time Clustering
- Tests

## 3 Réduction de la dimension

# Plan

## 1 Introduction

## 2 Clustering

- Problèmes posés
- Fonction objectif
- Commute Time Clustering
- Tests

## 3 Réduction de la dimension

# Objectifs

- Identifier des sous ensembles  $C_i$  de  $\{(x_i, y_i)\} \subset \mathcal{X} \times \mathcal{Y}$ , tels que :
- pour  $i \neq j$ ,  $C_i$  est bien distinct de  $C_j$  ;
  - tous les éléments de  $C_i$  sont similaires.

# Objectifs

→ Identifier des sous ensembles  $C_i$  de  $\{(x_i, y_i)\} \subset \mathcal{X} \times \mathcal{Y}$ , tels que :

- pour  $i \neq j$ ,  $C_i$  est bien distinct de  $C_j$  ;
- tous les éléments de  $C_i$  sont similaires.

Différents points de vue mènent à différentes solutions :

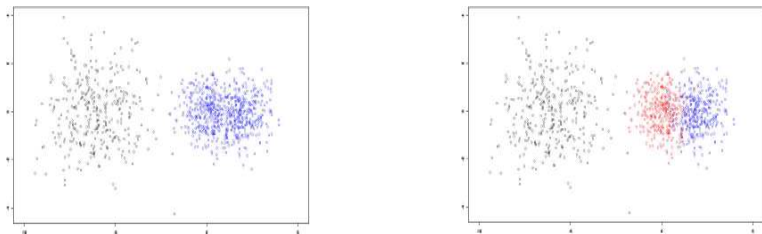


FIG.: Mélange de trois gaussiennes,  $\mu = (-6, 0), (0, 0), (2, 0)$ ,  $\sigma = 2, 0.5, 0.5$ .



# Objectifs

→ Identifier des sous ensembles  $C_i$  de  $\{(x_i, y_i)\} \subset \mathcal{X} \times \mathcal{Y}$ , tels que :

- pour  $i \neq j$ ,  $C_i$  est bien distinct de  $C_j$  ;
- tous les éléments de  $C_i$  sont similaires.

Différents points de vue mènent à différentes solutions :

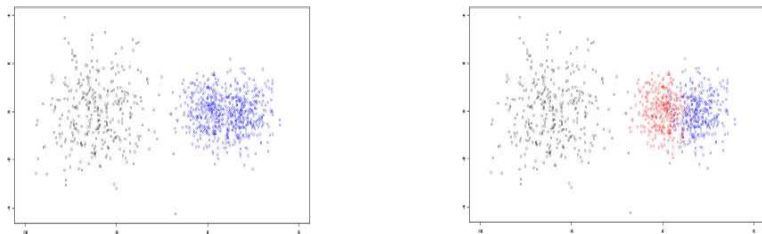


FIG.: Mélange de trois gaussiennes,  $\mu = (-6, 0), (0, 0), (2, 0)$ ,  $\sigma = 2, 0.5, 0.5$ .

⇒ Approche choisie : un algorithme retourne une famille de partitions  $(A_i)_{i=1..m}$  avec  $A_{i+1} \prec A_i$  ;  $A_k$  contient  $k$  groupes.

# Nombre de clusters

Comment choisir le bon partitionnement  $A_k$  ?

Deux options :

- ① critère direct basé sur les distances ;
- ② critère de validation par classification supervisée.

Difficile de donner un critère direct, trop dépendant de la structure des données  $\Rightarrow$  choix 2.

# Nombre de clusters

Comment choisir le bon partitionnement  $A_k$  ?

Deux options :

- ❶ critère direct basé sur les distances ;
- ❷ critère de validation par classification supervisée.

Difficile de donner un critère direct, trop dépendant de la structure des données  $\Rightarrow$  choix 2.

Pour  $A_k$ , erreur de classification estimée par  $n$ -fold cross validation :

$$\text{label} \left( x \in A_k^{(i)} \right) = i.$$

Erreur de classification  $< \delta \Rightarrow$  clustering validé,  $k := k + 1$ .

Sinon "il y a exactement  $k$  clusters".

# Méthode

## Clustering HCT

Clustering hiérarchique : donne une famille de partitions

+

Euclidian Commute Time Distance (ECTD) : gomme l'effet de chaînage du single linkage, en conservant ses avantages.

## Clustering HCT

Clustering hiérarchique : donne une famille de partitions

+

Euclidian Commute Time Distance (ECTD) : gomme l'effet de chaînage du single linkage, en conservant ses avantages.

Chronologie :

- 1 algorithme HCT sur sorties fonctionnelles  $B(x_i)$  ;
- 2 validation croisée sur les sorties pour déterminer  $k_0$  ;
- 3 algorithme HCT sur les entrées  $B^{-1}(C_i)$  pour chaque cluster ;
- 4  $k_0$  validations croisées sur les entrées pour déterminer  $k \geq k_0$ .

# Plan

## 1 Introduction

## 2 Clustering

- Problèmes posés
- **Fonction objectif**
- Commute Time Clustering
- Tests

## 3 Réduction de la dimension

## Point de départ

Graphe  $G = (S, A)$ ,  $S$  sommets représentant les éléments à classer. Une arête  $(s_1, s_2) \in A$  a pour poids  $w_{(s_1, s_2)}$ , similarité entre  $s_1$  et  $s_2$ .

Graphe  $k$ -NN ( $G_k$ ) : deux points trop éloignés ont pour similarité 0.

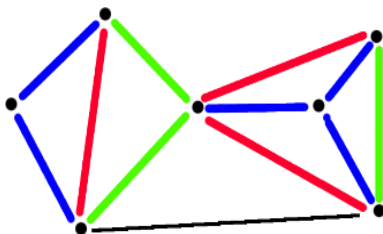


FIG.: 1-NN bleu, 2-NN vert, 3-NN rouge, lien avec  $k$  trop grand en noir.

## Point de départ

Grphe  $G = (S, A)$ ,  $S$  sommets représentant les éléments à classer. Une arête  $(s_1, s_2) \in A$  a pour poids  $w_{(s_1, s_2)}$ , similarité entre  $s_1$  et  $s_2$ .

Grphe  $k$ -NN ( $G_k$ ) : deux points trop éloignés ont pour similarité 0.

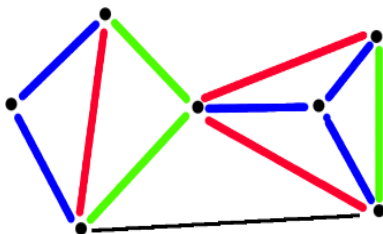


FIG.: 1-NN bleu, 2-NN vert, 3-NN rouge, lien avec  $k$  trop grand en noir.

- $k$  trop petit : on sous-estime les similarités ;
- $k$  trop grand : on sur-estime les similarités.

Heuristique :  $k =$  plus petit entier tel que

$$\text{parties connexes}(G_k) = \text{parties connexes}(G_{k-1}).$$



# Coupe minimale

## Coupe minimale normalisée

À  $k$  fixé, minimiser  $\text{NCut} = \sum_{i=1}^k \frac{\text{cut}(A_i, \bar{A}_i)}{\text{Vol}(A_i)}$ , avec  $\text{cut}(A, \bar{A}) = \sum_{i \in A, j \in \bar{A}} w_{ij}$ .

NCut est lié aux probabilité de transitions :

$$\mathbb{P}(\bar{A}|A) = \frac{\text{cut}(A, \bar{A})}{\text{Vol}(A)}.$$

Problème équivalent : minimiser  $\sum_{i=1}^k \mathbb{P}(\bar{A}_i|A_i)$  (Shi & Meilă, 2000).

NP-complet  $\Rightarrow$  relaxation.

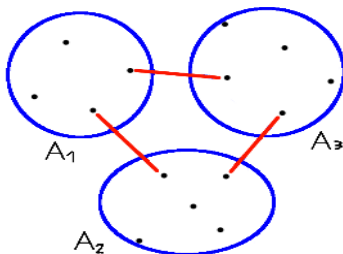


FIG.: Illustration :

# Plan

## 1 Introduction

## 2 Clustering

- Problèmes posés
- Fonction objectif
- Commute Time Clustering
- Tests

## 3 Réduction de la dimension

# Clustering spectral

Laplacien du graphe :  $L = D - W$ ,

$W$  matrice de similarité contenant les poids des arêtes,

$D$  matrice diagonale avec  $D_{i,i} = \sum_{j \in V(i)} w_{ij}$ .

# Clustering spectral

Laplacien du graphe :  $L = D - W$ ,

$W$  matrice de similarité contenant les poids des arêtes,

$D$  matrice diagonale avec  $D_{i,i} = \sum_{j \in V(i)} w_{ij}$ .

Réécriture de NCut (von Luxburg, 2006) :

$$h \in \mathbb{R}^{N \times k}, h_{ij} = \begin{cases} \frac{1}{\sqrt{\text{Vol}(A_j)}} & \text{si } i \in A_j, \\ 0 & \text{sinon.} \end{cases}$$

Après calculs :  ${}^t H H = I$ ,  ${}^t h_i D h_i = 1$ , et  ${}^t h_i L h_i = 2 \frac{\text{cut}(A_i, \bar{A}_i)}{\text{Vol}(A_i)}$ .

## Clustering spectral

Laplacien du graphe :  $L = D - W$ ,

$W$  matrice de similarité contenant les poids des arêtes,

$D$  matrice diagonale avec  $D_{i,i} = \sum_{j \in V(i)} w_{ij}$ .

Réécriture de NCut (von Luxburg, 2006) :

$$h \in \mathbb{R}^{N \times k}, h_{ij} = \begin{cases} \frac{1}{\sqrt{\text{Vol}(A_j)}} & \text{si } i \in A_j, \\ 0 & \text{sinon.} \end{cases}$$

Après calculs :  ${}^t H H = I$ ,  ${}^t h_i D h_i = 1$ , et  ${}^t h_i L h_i = 2 \frac{\text{cut}(A_i, \bar{A}_i)}{\text{Vol}(A_i)}$ .

### Problème équivalent à NCut

$$\min_{A_1, \dots, A_k} \text{Tr}({}^t H L H), \text{ sous } {}^t H D H = I,$$

Colonnes de  $H$  = vecteurs indicateurs de la composition des clusters.

Relaxation : on résout pour  $H$  quelconque dans  $\mathbb{R}^{N \times k}$ ,

puis  $k$ -means sur les lignes  $\Rightarrow$  solution approchée.

# Euclidian Commute Time Distance

Marche aléatoire  $M$  : distribution initiale  $\pi$  sur les sommets,  $s_t$  le sommet courant à l'instant  $t$ .

$$\mathbb{P}(s_{t+1}|s_t) = \frac{W(s_t, s_{t+1})}{\sum_{a \in V(s_t)} W(s_t, a)}$$

→ Si  $M$  prend peu de temps en moyenne pour aller de  $s_1$  à  $s_2$  puis revenir, alors  $s_1$  et  $s_2$  sont dans deux clusters identiques. Mesure de dissimilarité :

$$d^2(s_1, s_2) = \mathbb{E}[t, s_t = s_2 | s_0 = s_1] + \mathbb{E}[t, s_t = s_1 | s_0 = s_2].$$

# Euclidian Commute Time Distance

Marche aléatoire  $M$  : distribution initiale  $\pi$  sur les sommets,  $s_t$  le sommet courant à l'instant  $t$ .

$$\mathbb{P}(s_{t+1}|s_t) = \frac{w(s_t, s_{t+1})}{\sum_{a \in V(s_t)} w(s_t, a)}$$

→ Si  $M$  prend peu de temps en moyenne pour aller de  $s_1$  à  $s_2$  puis revenir, alors  $s_1$  et  $s_2$  sont dans deux clusters identiques. Mesure de dissimilarité :

$$d^2(s_1, s_2) = \mathbb{E}[t, s_t = s_2 | s_0 = s_1] + \mathbb{E}[t, s_t = s_1 | s_0 = s_2].$$

$d$  est une distance euclidienne dans un espace de représentation à  $N$  dimensions :

$$d(i, j)^2 = \text{Vol}(G)^{-1} (e_i - e_j)^T L^+ (e_i - e_j) \quad (\text{Saerens et al., 2004});$$

$L^+$  = pseudo-inverse du laplacien du graphe ;  $(e_i)$  base canonique de  $\mathbb{R}^N$ .

# Algorithme utilisé : HCT

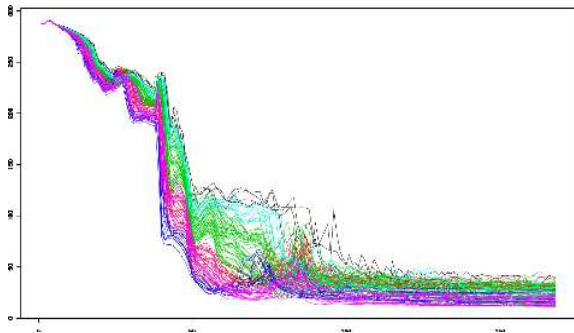
- 1 Choix d'une mesure de similarité sur les données ;
- 2 Construction du graphe  $k$ -NN avec  $k$  plus petit entier tel que parties connexes( $G_k$ ) = parties connexes( $G_{k-1}$ ) ;
- 3 Calcul de la matrice  $M$  des commute distances, avec l'inverse du laplacien  $L^+$  ou itérativement ;
- 4 Clustering hiérarchique sur les sommets du graphe  $k$ -NN, utilisant  $M$ .



# Algorithme utilisé : HCT

- 1 Choix d'une mesure de similarité sur les données ;
- 2 Construction du graphe  $k$ -NN avec  $k$  plus petit entier tel que parties connexes( $G_k$ ) = parties connexes( $G_{k-1}$ ) ;
- 3 Calcul de la matrice  $M$  des commute distances, avec l'inverse du laplacien  $L^+$  ou itérativement ;
- 4 Clustering hiérarchique sur les sommets du graphe  $k$ -NN, utilisant  $M$ .

Six clusters sur  
100 sorties de code  
Cathare :



# Plan

## 1 Introduction

## 2 Clustering

- Problèmes posés
- Fonction objectif
- Commute Time Clustering
- Tests

## 3 Réduction de la dimension

# Comparaisons I

Méthode	Taux d'erreur
K-Means	6%
Hierarchique + Ward	6.2%
Clustering Spectral	5.9%
HCT	5.8%

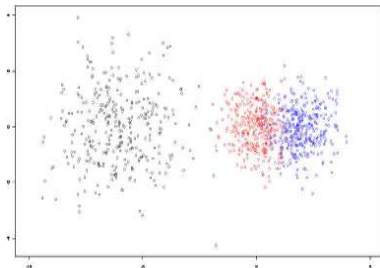


FIG.: Mélange de trois gaussiennes.

→ Résultats comparables.

## Comparaisons II

Méthode	Taux d'erreur
K-Means	41%
Hierarchique + Ward	45%
Clustering Spectral	0%
HCT	0%

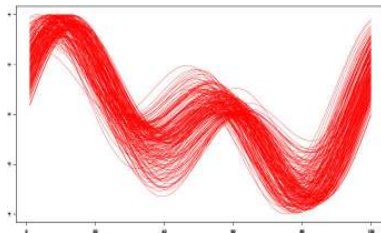
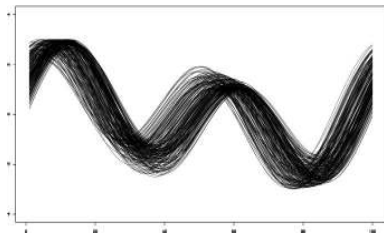


FIG.:  $f_{a,b,c,d}$  sur  $[0, 2\pi]$ , groupe 1 à gauche, groupe 2 à droite.

Fonction  $f_{a,b,c,d} : x \mapsto a \cos(x) + b \sin(x) + c \cos(2x) + d \sin(2x)$ .

$(a, b)$  uniforme sur  $S(0, 1)$  pour tous les points ( $N=400$ ),

$(c, d) : \mathcal{U}(S(0, 1))$  (noir, 1 à  $N/2$ ), puis  $\mathcal{U}(S(0, 2))$  (rouge,  $N/2$  à  $N$ ).

# Bilan

## Avantages

- + pas de recherche de vecteurs propres ;
- + performances comparables à celles du clustering spectral.

## Inconvénients

- assez sensible aux outliers (améliorable) ;
- choix du paramètre  $k$  assez arbitraire.

# Bilan

## Avantages

- + pas de recherche de vecteurs propres ;
- + performances comparables à celles du clustering spectral.

## Inconvénients

- assez sensible aux outliers (améliorable) ;
- choix du paramètre  $k$  assez arbitraire.

Distance  $L^2 \rightarrow$  mesure de similarité  $\rightarrow$  commute distance, algorithmique.  
Seule la mesure de similarité est utile. Exemples :

- extension de la corrélation sur les rangs au cadre fonctionnel (Heckman & Zamar, 2000) ;
- similarités du type  $e^{-\gamma \|f^{(k)} - g^{(k)}\|^\alpha}$  ;
- $e^{-\gamma D(f,g)}$  avec  $D$  divergence de Bregman symétrique ..etc.

# Plan

## 1 Introduction

## 2 Clustering

- Problèmes posés
- Fonction objectif
- Commute Time Clustering
- Tests

## 3 Réduction de la dimension

# Vue globale

Étapes :

- 1 clustering en  $k$  groupes  $C_i$  ;
- 2 classification  $C$  des entrées ;
- 3  $P_i$  : réduction de la dimension sur le cluster  $C_i$  ;
- 4 régression  $R_i$  sur les coordonnées réduites pour chaque  $C_i$  ;

Puis prédiction d'une nouvelle sortie en classant l'entrée  $x_0$  dans  $C(x_0)$ , et application de  $R_{C(x_0)}$  avant de restituer la fonction :

$$\text{Prédiction}(x_0) = P_{C(x_0)}^{-1}(R_{C(x_0)}(x_0)).$$

Objectif :  $P_i|C_i$ .



# Courbes principales

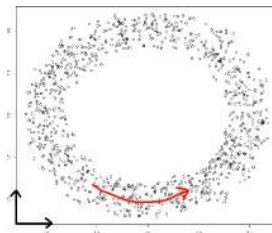


FIG.: Distribution autour d'un cercle.

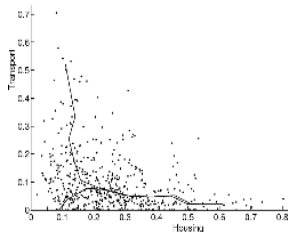


FIG.: Deux "1<sup>eres</sup>" courbes principales.

# Courbes principales

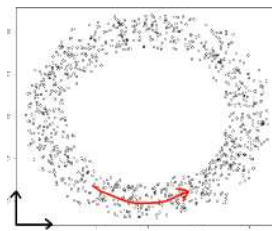


FIG.: Distribution autour d'un cercle.

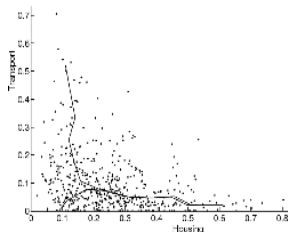


FIG.: Deux "1<sup>eres</sup>" courbes principales.

Courbe "optimale" passerait par tous les points  $\Rightarrow$  régularisation.

*Self-consistency* (Hastie, 1984) :

Une courbe paramétrée  $s$  est principale si  $\forall t \in [\alpha, \beta], s(t) = \mathbb{E}[X | t_X = t]$ .

*Longeur bornée* (Kégl et al., 2000) :

$s^*$  minimise l'erreur quadratique moyenne sous la contrainte  $l(s) \leq L$ .

*Courbure bornée* (Sandilya & Kulkarni, 2002) :

$s^*$  minimise l'erreur quadratique moyenne sous la contrainte  $\kappa(s) \leq K$ .

# Adaptation au cadre fonctionnel

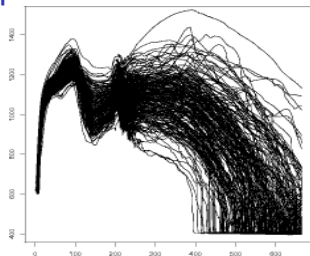


FIG.: 200 sorties du code de calcul Cathare, 665 points de discrétisation.

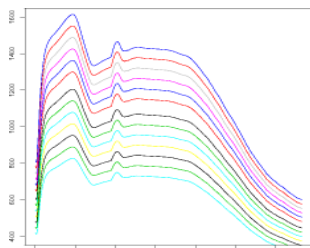


FIG.: Segment  $t \mapsto tv$ ,  $v$  1<sup>ère</sup> CP : 20% de la variance.

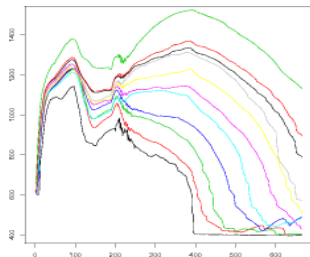


FIG.: 1<sup>ère</sup> CP : 58% de la variance.

Ensuite, il faudrait déterminer une 2<sup>nde</sup> (..etc) CP en chaque "point" de la courbe ...

# Conclusion

Réglages des paramètres du clustering ?

- Taille d'un voisinage dans le graphe.
- Mesure de similarité.

Optimiser efficacité de la représentation vs. qualité de la prédiction sur coordonnées réduites ?

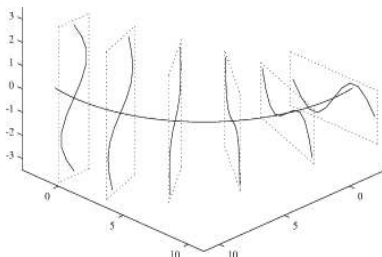


FIG.: Exemple de surface principale en 2D.

Autres méthodes à explorer : isomap, LLE, laplacian eigenmaps ...