Classification automatique

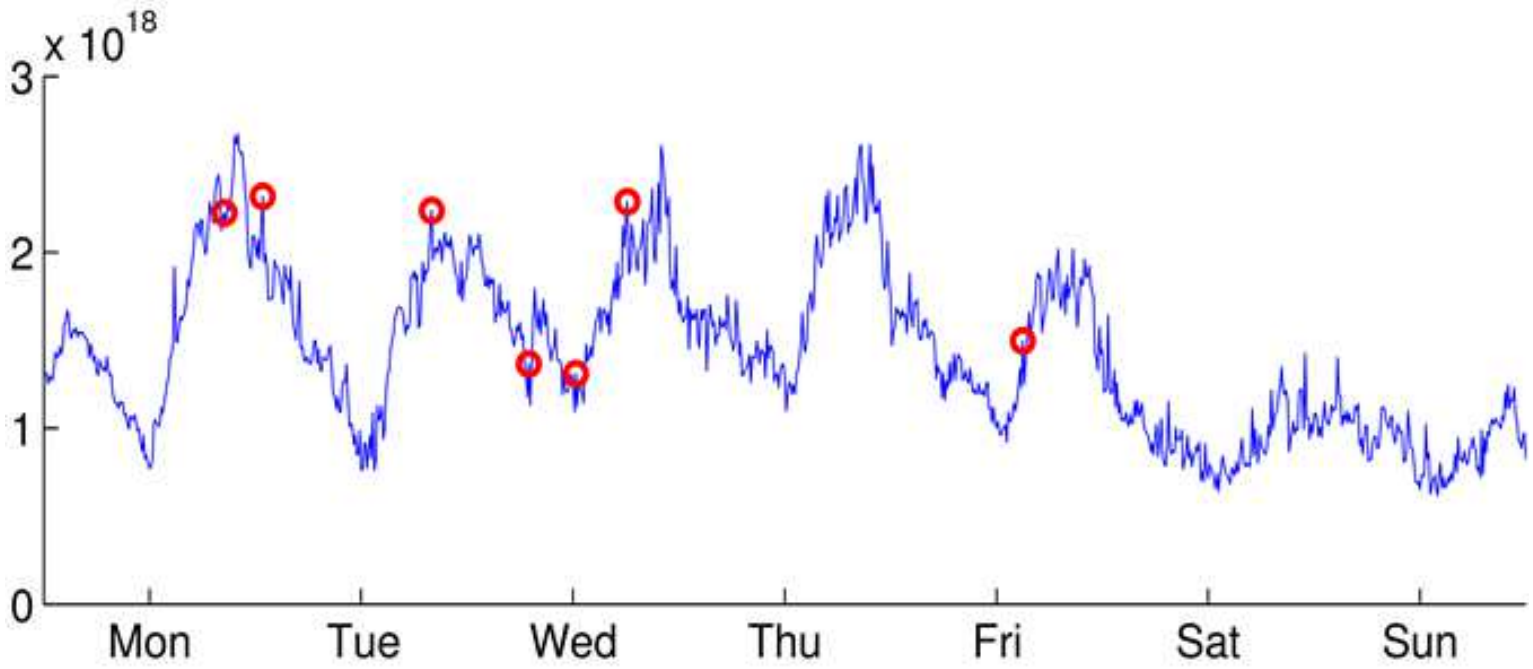
Chapitre 6 : Classification de séries temporelles

Allou Samé
allou.same@ifsttar.fr

Plan

Series temporelles

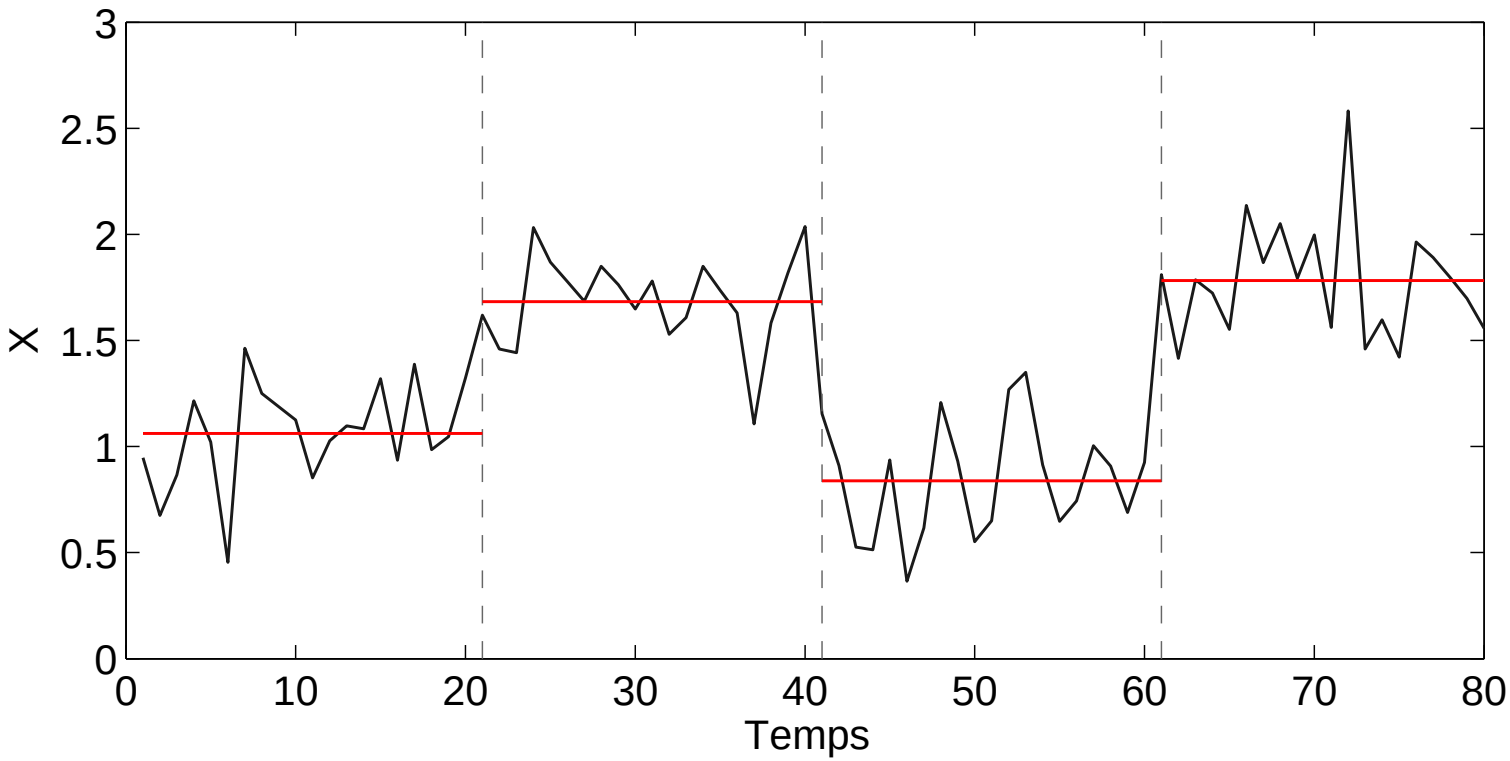
- Suite de valeurs numériques indicées par le temps : $(x_1, \ldots, x_n)$
- **Plusieurs domaines d'application** : économie, finance, météorologie, monitoring à but médical ou non, détection de fraude bancaire, etc.
- **Problèmes** : prédiction, prévision (futur), approximation de fonctions, classification automatique (segmentation d'une série, classification d'un ensemble de séries)



Segmentation d'une série temporelle

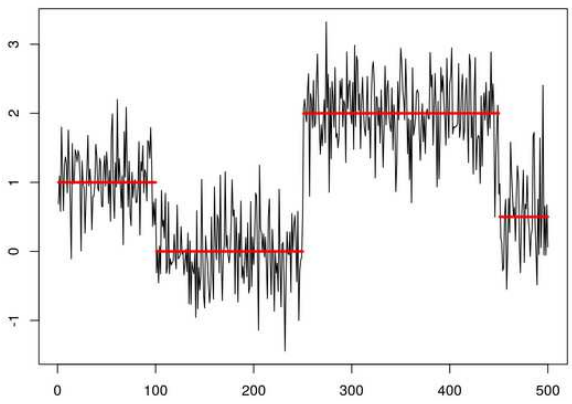
Objectifs

- Résumer/synthétiser une série à l'aide d'un nombre réduit de prototypes
- Trouver les instants de rupture dans la série

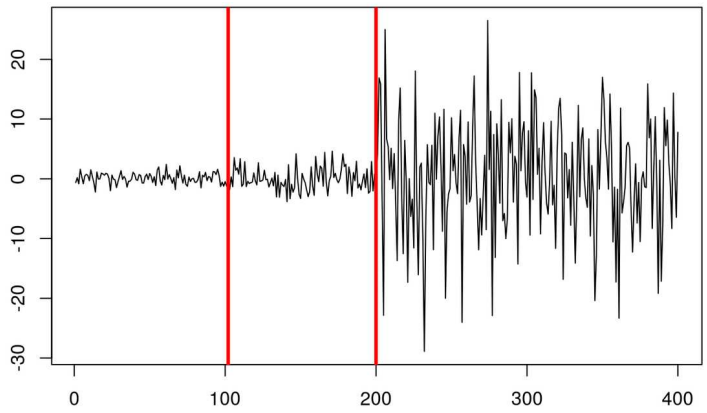


Exemples

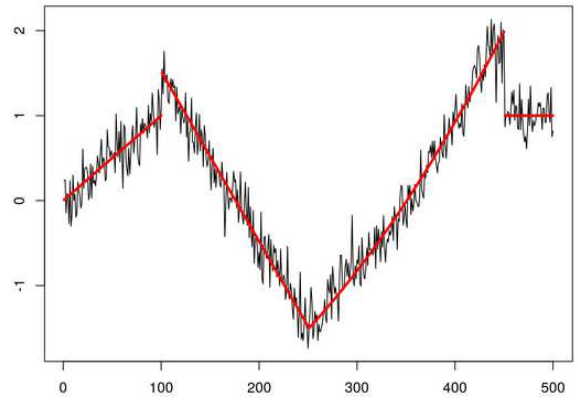
Changement de moyenne



Changement de variance



Changement de pente



Segmentation optimale

Objectifs

Découpage de la série $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ en K segments, qui minimise le critère d'inertie intra-classe ($\longrightarrow$ problème de classification automatique contraint)

$$C(\boldsymbol{\tau}) = \sum_{k=1}^{K} \sum_{t \in I_k} \| \boldsymbol{x}_t - \boldsymbol{\mu}_k \|^2,$$

où les instants de rupture sont notés ici $\boldsymbol{\tau} = (t_1, \ldots, t_{K-1})$ et les segments sont notés $I_1 = [1; t_1 - 1]$, $I_k = [t_k; t_{k+1} - 1]$, $I_K = [t_{K-1}; n]$

La partition obtenue dans ce cas est constituée d'intervalles de temps contigus

Méthode de Fisher et Bellman

Origines

W. D. Fisher (1958) et R. Bellman (1962)

Objectifs

Trouver la segmentation en K segments qui minimise le critère d'inertie intra-classe re-formulé comme suit :

$$\begin{cases} C(\boldsymbol{\tau}) &= D(1, t_1 - 1) + \left(\sum_{k=2}^{K-1} D(t_{k-1}, t_k - 1) \right) + \\ & \quad D(t_{K-1}, n) \quad \text{si } K \geq 3 \\ C(t_1) &= D(1, t_1 - 1) + D(t_1, n) \quad \text{si } K = 2 \end{cases}$$

avec $D(a, b) = \sum_{i=a}^{b} \| \boldsymbol{x}_i - \boldsymbol{\mu}_{ab} \|^2$ et $\boldsymbol{\mu}_{ab} = \frac{\sum_{i=a}^{b} \boldsymbol{x}_i}{b - a + 1}$

Algorithme de programmation dynamique

« toute sous partition d'une partition optimale est optimale »

Etape 1 : calcul de la matrice triangulaire supérieure D

$\forall a = 1, \ldots, n,\ b = a, \ldots, n$

$$D(a,b) = \sum_{i=a}^{b} \| \boldsymbol{x}_i - \boldsymbol{\mu}_{ab} \|^2 \qquad \text{et} \qquad \boldsymbol{\mu}_{ab} = \frac{\sum_{i=a}^{b} \boldsymbol{x}_i}{b-a+1}$$

Etape 2 : calcul récursif des critères optimaux (Forward)

$\forall i = 1, \ldots, n$

$$M_1[i,1] = D[1,i]$$

$\forall k = 2, \ldots, K,\ i = k, \ldots, n$

$$M_1[i,k] = \min \Big\{ M_1[t-1,k-1] + D[t,i] \quad \text{avec} \quad t = k, \ldots, i \Big\}$$

$$M_2[i,k] = \arg\min \Big\{ M_1[t-1,k-1] + D[t,i] \quad \text{avec} \quad t = k, \ldots, i \Big\}$$

Algorithme de programmation dynamique

Etape 3 : calcul des instants de changement optimaux (Backward)

$k = K - 1$

$r = n$

Tant que $k \geq 1$

$t_k = M_2(r, k+1)$

$r = t_k - 1$

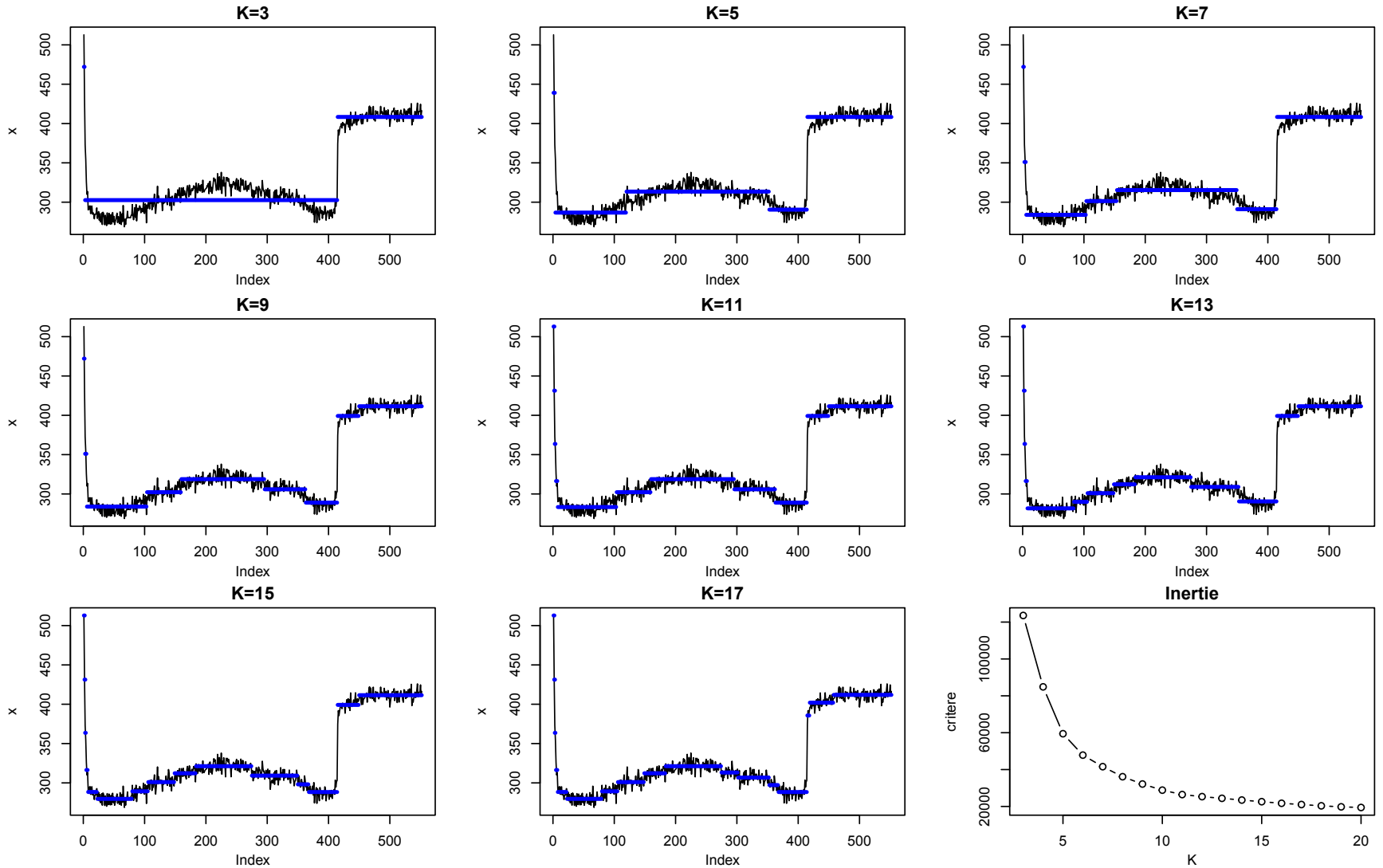
$k = k - 1$

Choix de K

- Méthode du coude appliquée au critère d'inertie intra-classe
- Critère intra-classe pénalisé

Exemples

Exemple de segmentation (données aiguillage)



Autres méthodes de segmentation

- Chaînes de markov Cachées ou Hidden Markov Models (HMM)
- Modèles de mélange (avec des contraintes sur les poids du mélange)
- Méthodes de segmentation on line

Segmentation de séries temporelles sous R

```
library(changepoint)
set.seed(10)
x=c(rnorm(100,0,1),rnorm(100,1,1),
                    rnorm(100,0,1),rnorm(100,0.2,1))
ts.plot(x,xlab="Index")

# choix automatique du nombre de segments
res=cpt.mean(x,method="PELT")
cpts(res)
plot(res,type="l",cpt.col="blue",xlab="Index",cpt.width=4)

# choix manuel du nombre de segment (fixé ici à 4)
res=cpt.mean(x,penalty="Manual",pen.value=0,Q=4,
                                        method="SegNeigh")
cpts(res)
plot(res,type="l",cpt.col="blue",xlab="Index",cpt.width=4)
```

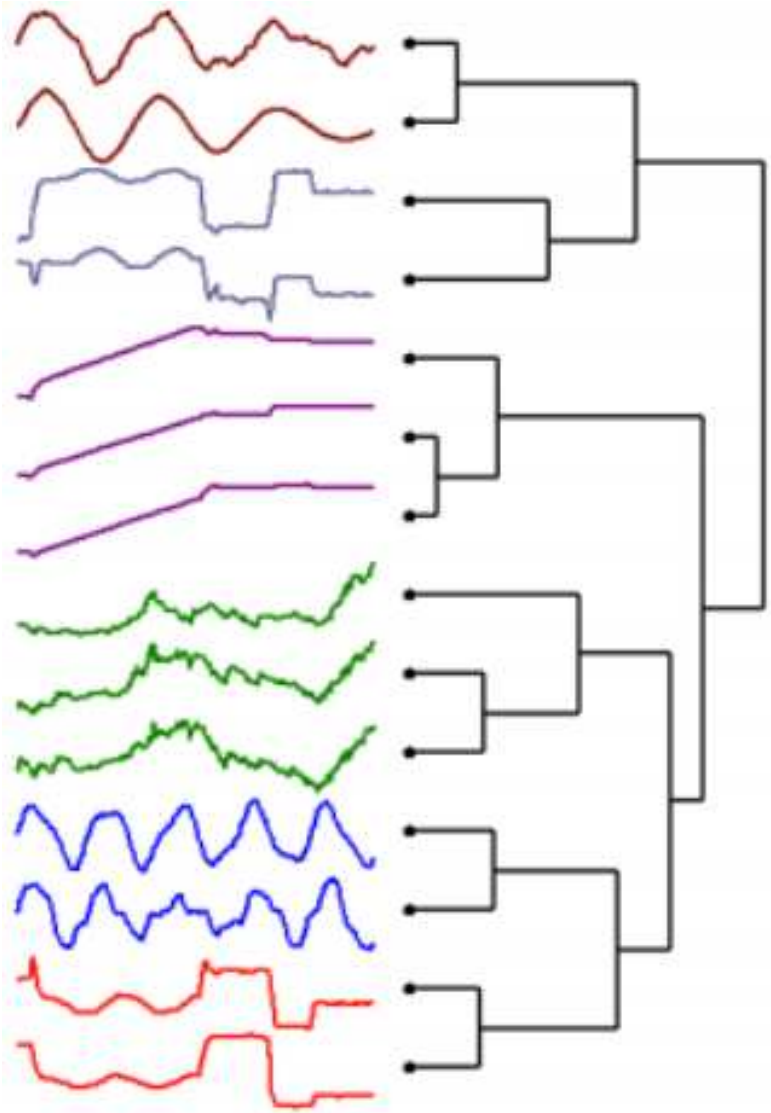
Classification de séries temporelles

Objectifs

Regrouper un ensemble de séries temporelles en classes homogènes

Notations

$(\mathbf{x}_1, \ldots, \mathbf{x}_n)$: ensemble de n séries temporelles, avec $\mathbf{x}_i = (x_1, \ldots, x_N)$



Différentes familles de méthodes

Famille 1 : méthodes directes

- Chaque série temporelle est traitée comme une observation multivariée
- La classification s'effectue par des méthodes classiques (ex. CAH, k-means, PAM, Fuzzy k-means) et en utilisant des distances telles que
  - la distance euclidienne
  - la distance basée sur la corrélation
  - la distance Dynamic Time Warping (DTW) (voir ci-après)

Différentes familles de méthodes

Famille 2 : méthodes basées sur l'extraction de caractéristiques

- Etape 1 : extraction préalable de caractéristiques pour chaque série
  - segmentation et discrétisation : ex. méthode SAX
  - extraction de coefficients de Fourier ou d'ondelettes
  - extraction de coefficients à partir de modèles de type AR
  - analyse en composantes principales fonctionnelles (FPCA)
- Etape 2 : classification des vecteurs de caractéristiques à l'aide de méthodes classiques

Différentes familles de méthodes

Famille 3 : modèles probabilistes adaptés aux séries temporelles

- Modèles (chaque classe est régie par une distribution de probabilité)
  - Mélanges de régressions polynomiales
  - Mélanges de modèles ARMA
  - Mélanges de processus markoviens
- Algorithmes
  - Estimation des paramètres des modèles à l'aide d'algorithmes adaptés (ex. algorithme Espérance-Maximisation (EM))

Classification basée sur des distances classiques

Distance euclidienne

$$d_{EUC}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2}$$

Distance basée sur la corrélation

$$d_{COR}(\mathbf{x}, \mathbf{y}) = \sqrt{2(1 - |COR(\mathbf{x}, \mathbf{y})|)}$$

où $COR(\mathbf{x}, \mathbf{y})$ est le coefficient de corrélation entre les séries $\mathbf{x}$ et $\mathbf{y}$

Classification

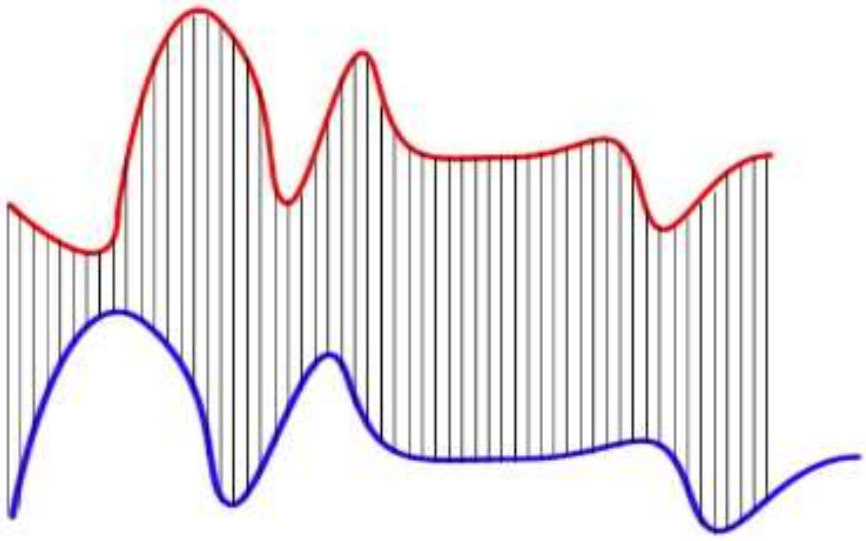
Regroupement automatique des séries temporelles en utilisant des techniques classiques (ex. CAH, PAM, k-means) à partir des distances (euclidienne ou basée sur la corrélation)

Classification basée sur la distance DTW

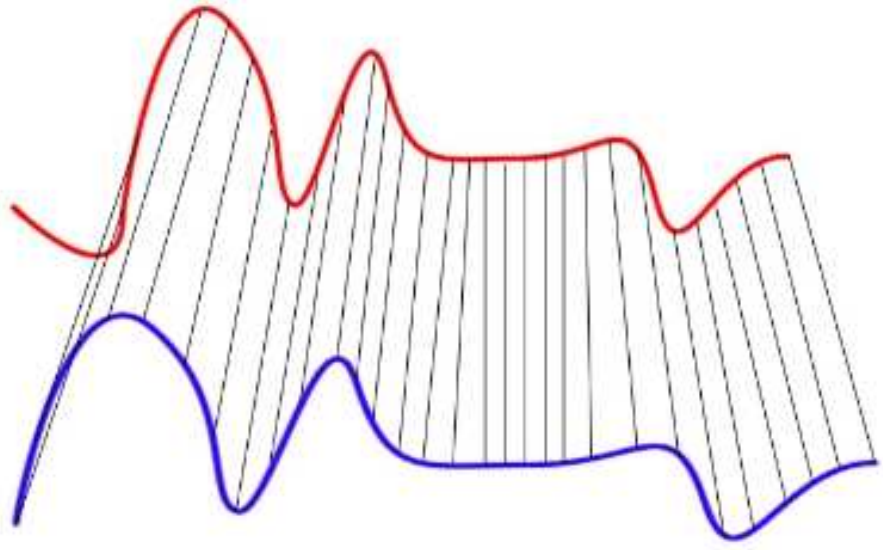
Distance DTW

- Mesure l'appariement (alignement) optimal entre deux séries
- Tient compte de la forme des séries

distance euclidienne



distance DTW



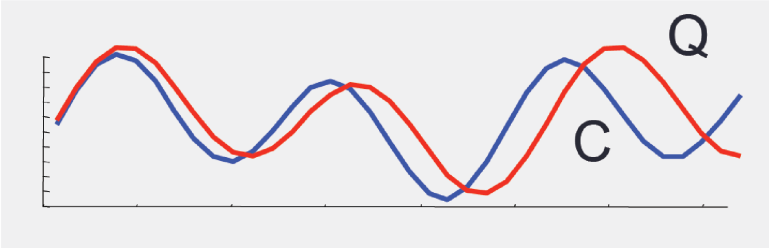
Classification

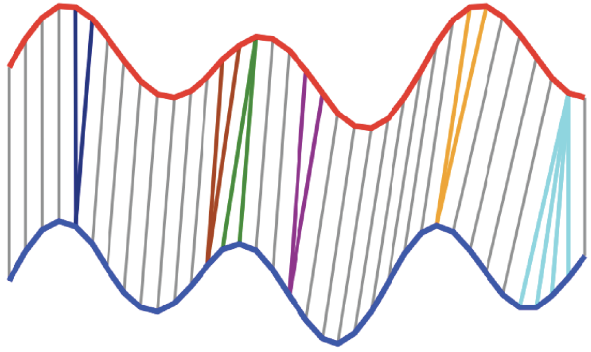
Application des techniques classiques (ex. CAH, PAM) à partir du tableau de distances DTW

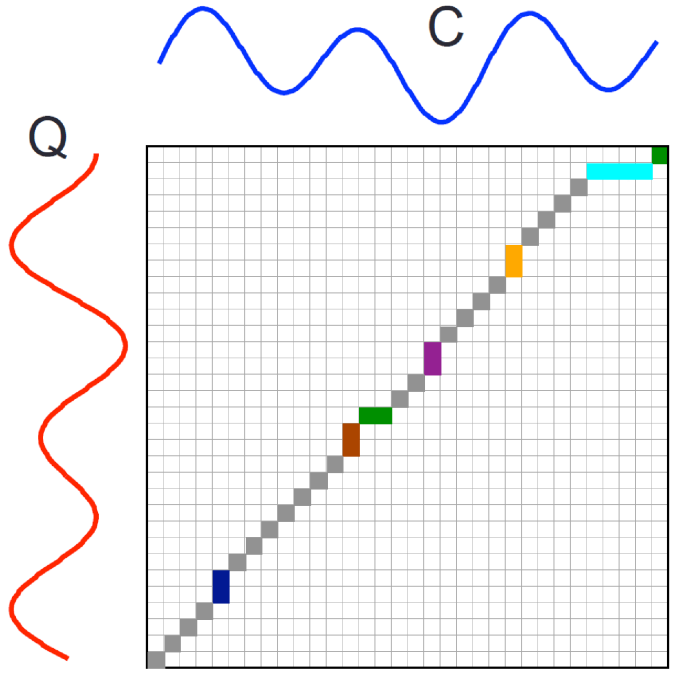
Classification basée sur la distance DTW

Distance DTW

- Mesure l'appariement (alignement) optimal entre deux séries
- Tient compte de la forme des séries







Classification

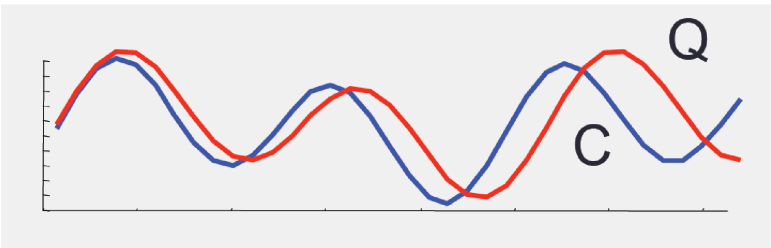
Application des techniques classiques (ex. CAH, PAM) à partir du tableau de distances DTW

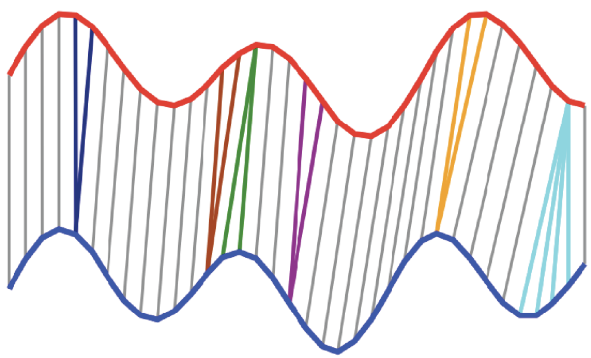
Classification basée sur la distance DTW

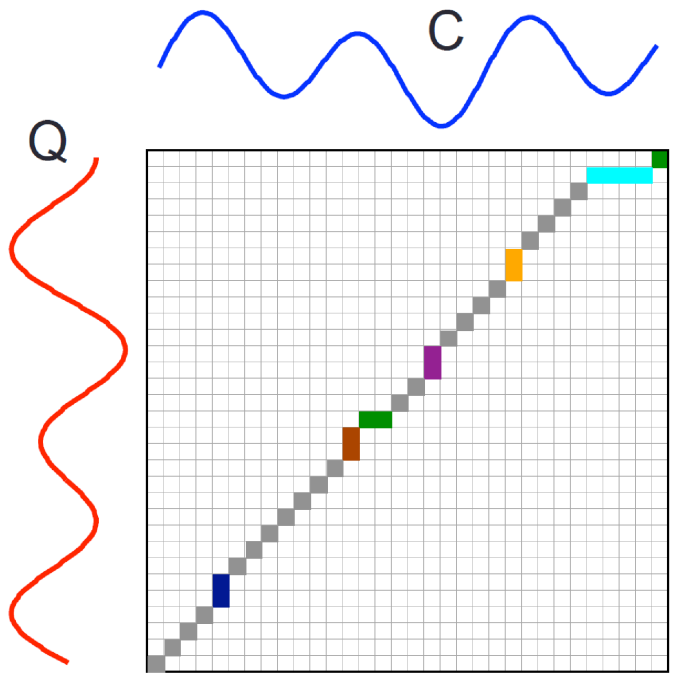
Distance DTW

$$d_{DTW}(\mathbf{x}, \mathbf{y}) = \min_{r \in M} \sum_{i=1,\ldots,m} |x_{a_i} - y_{b_i}|^2$$

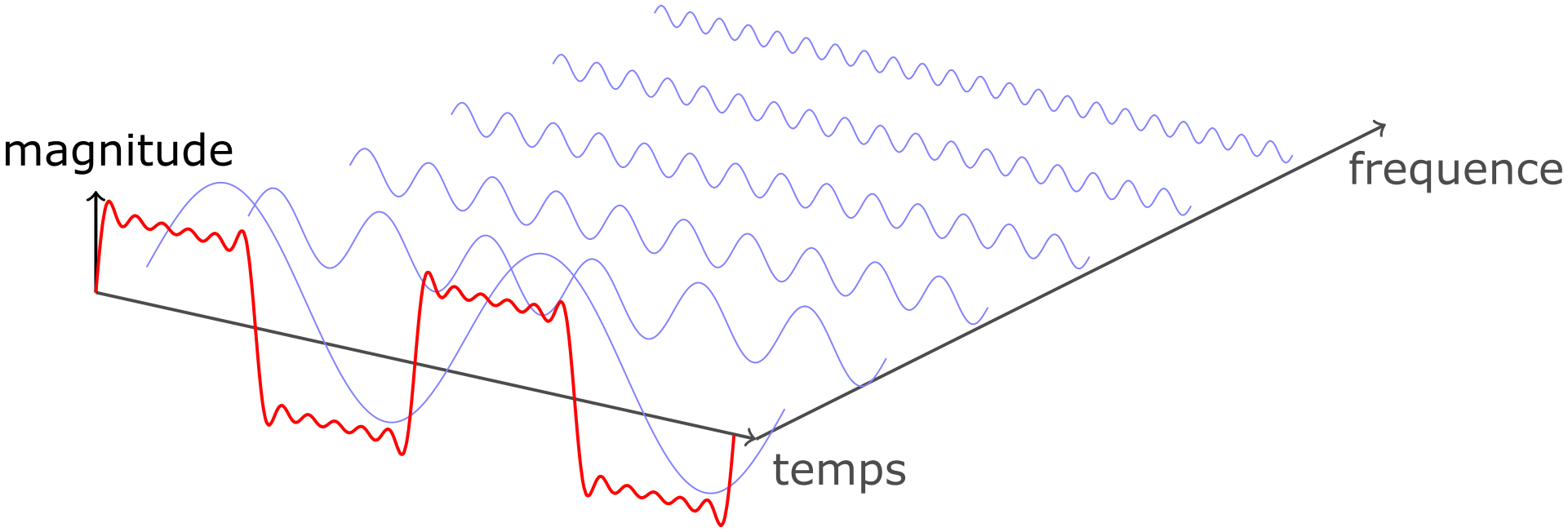
où M est l'ensemble des paires $r = ((a_1, b_1), \ldots, (a_m, b_m))$ vérifiant $a_i, b_i \in \{1, \ldots, N\}$ tel que $a_1 = b_1 = 1$, $a_m = b_m = N$, et $a_{i+1} = (a_i \text{ ou } a_i + 1)$, $b_{i+1} = (b_i \text{ ou } b_i + 1)$







Classification basée sur la décomposition en série de Fourier



Décomposition en séries de Fourier

$$\mathbf{x} \approx \sum_{j=1}^{M} \alpha_{m1}^{(x)} \cos\left(\frac{2\pi j}{N} t\right) + \alpha_{m2}^{(x)} \sin\left(\frac{2\pi j}{N} t\right)$$

Généralement, on choisit $M \leq [\frac{N}{2}] + 1$

Classification basée sur la décomposition en série de Fourier

Méthode de classification

Application des techniques classiques (ex. CAH, PAM, k-means) en utilisant la distance euclidienne

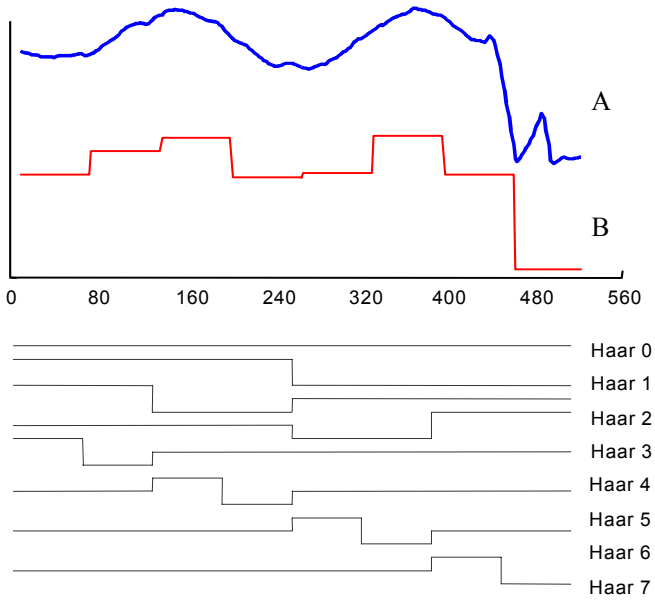
$$d_{DFT}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{m=1}^{M} (\alpha_m^{(x)} - \alpha_m^{(y)})^2}$$

$\alpha_m^{(x)}$ et $\alpha_m^{(y)}$: coefficients issus de la décomposition de $\mathbf{x}$ et $\mathbf{y}$

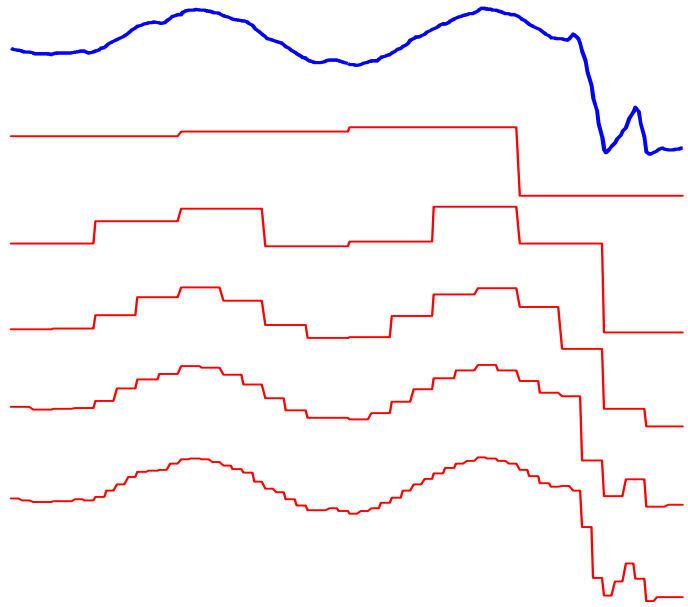
$$\begin{aligned} \mathbf{x}_t &\approx \sum_{j=1}^{M} \alpha_{m1}^{(x)} \cos\left(\frac{2\pi j}{N} t\right) + \alpha_{m2}^{(x)} \sin\left(\frac{2\pi j}{N} t\right) \\ \mathbf{y}_t &\approx \sum_{j=1}^{M} \alpha_{m1}^{(y)} \cos\left(\frac{2\pi j}{N} t\right) + \alpha_{m2}^{(y)} \sin\left(\frac{2\pi j}{N} t\right) \end{aligned}$$

Classification basée sur les ondelettes

Décomposition d'une série selon une base d'ondelettes



Différents niveaux d'approximation



Classification de séries

Application des techniques classiques (ex. CAH, PAM, k-means)

$$d_{DWT}(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{m=1}^{M}(\alpha_m^{(x)} - \alpha_m^{(y)})^2}$$

$\alpha_m^{(x)}$ et $\alpha_m^{(y)}$: coefficients issus de la décomposition de $\mathbf{x}$ et $\mathbf{y}$

Classification basée sur le processus ARIMA

$$d_{AR}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{m=1}^{M} (\gamma_m^{(x)} - \gamma_m^{(y)})^2}$$

$\gamma_m^{(x)}$ et $\gamma_m^{(y)}$: coefficients issus de la modélisation ARIMA de $\mathbf{x}$ et $\mathbf{y}$

Il est possible d'utiliser d'autres distances telles que la distance de Mahalanobis sur les coefficients des modèles ARIMA

Classification basée sur l'ACP fonctionnelle

Méthode

1. Transformation de chaque série $\mathbf{x}_i$ en une courbe ou fonction $f_i(t)$: lissage polynomial, par splines ou par séries de Fourier
2. Discrétisation des fonctions : création, pour chaque série $\mathbf{x}_i$, du vecteur

$$\widetilde{\mathbf{x}}_i = \big(f_i(\tau_1), \ldots, f_i(\tau_m)\big)$$

   où les $\tau_j \in \{1, \ldots, N\}$ sont des instants communs à toutes les séries
3. Réduction de la dimension : application d'une ACP sur les données $(\widetilde{\mathbf{x}}_1, \ldots, \widetilde{\mathbf{x}}_n)$
4. Classification des scores fournis par l'ACP à l'aide des méthodes de classification déjà étudiées dans les chapitres précédents

Classification basée sur l'approximation SAX (Symbolic Aggregate approXimation)

Objectifs

- Transformer les séries à classifier en séquences symboliques de longueur plus petite que les séries initiales
- Opérer la classification sur les séries symboliques

Notations

- Série temporelle : $\mathbf{x} = (x_1, \ldots, x_n)$
- Approximation constante par morceaux de la série

$$\bar{\mathbf{x}} = (\bar{x}_1, \ldots, \bar{x}_m)$$

avec m le nombre de segments choisi généralement $<< n$

- Représentation symbolique de la série

$$\hat{\mathbf{x}} = (\hat{x}_1, \ldots, \hat{x}_m) \qquad \text{ex. } \hat{\mathbf{x}} = (aabcaabbcdcca)$$

avec $\hat{x}_j \in \mathcal{A}$, où $\mathcal{A}$ est l'alphabet choisi. Exemple $\mathcal{A} = \{a, b, c, d\}$

SAX : Symbolic Aggregate approXimation

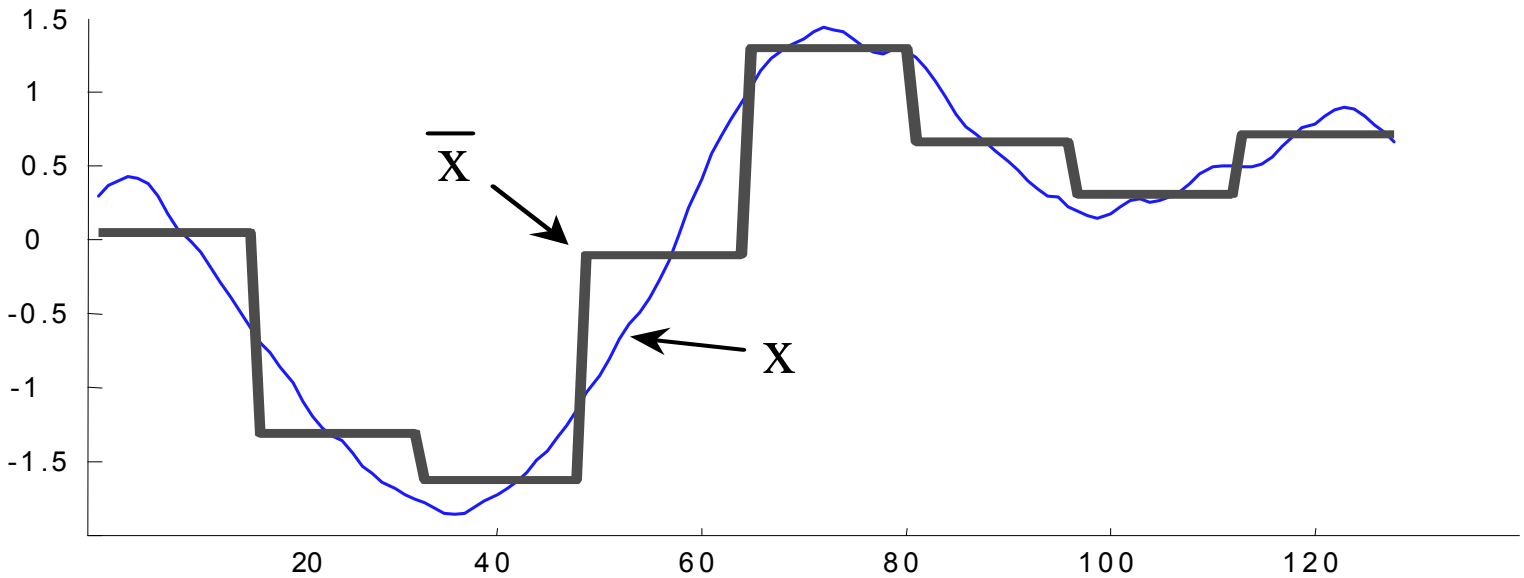
Principales étapes

1. Transformation de la série $\mathbf{x}$ en une série $\bar{\mathbf{x}}$ constante par morceaux (PAA)
2. Transformation de la série $\bar{\mathbf{x}}$ en une séquence symbolique $\hat{\mathbf{x}}$
3. Utilisation de la représentation symbolique pour classifier les séries

SAX (Etape 1) : Piecewise Aggregate Approximation (PAA)

1. Subdivision de l'intervalle de temps $\{1, \ldots, n\}$ en m intervalles de même longueur égale à n/m
2. Calcul de la moyenne de chaque intervalle

$$\bar{x}_j = \frac{m}{n} \sum_{i=\frac{n}{m}(i-1)+1}^{\frac{n}{m}i} x_i$$



SAX (Etape 2) : agrégation des segments

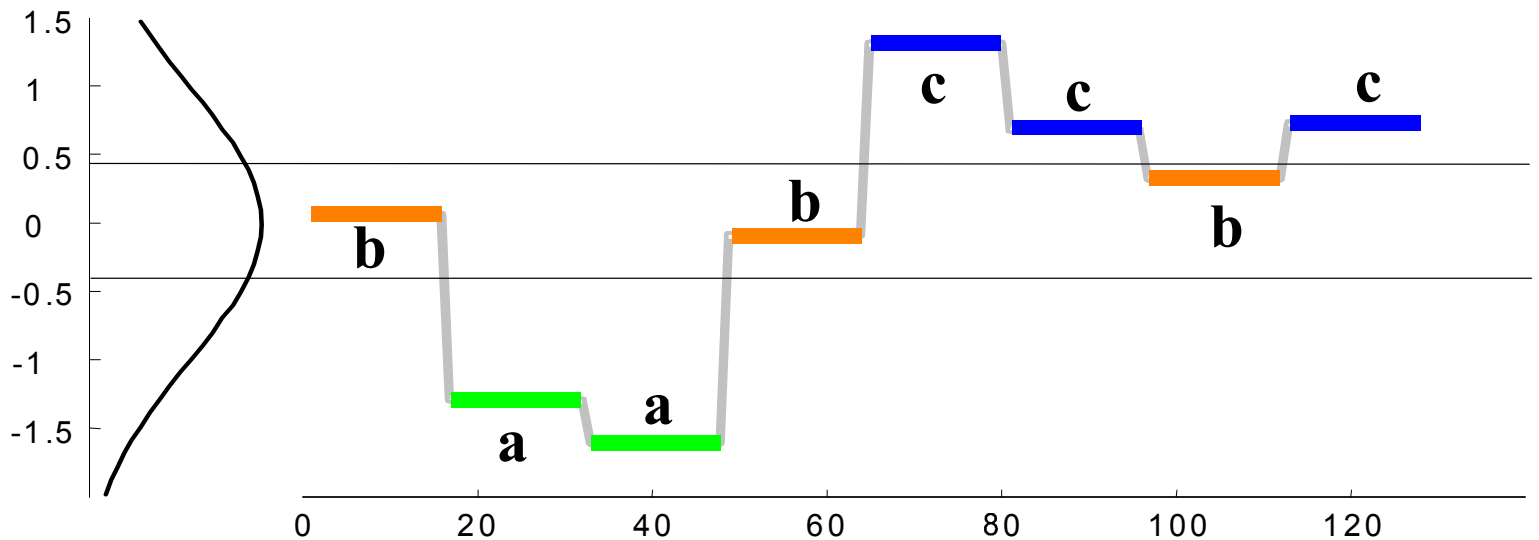
1. Découpage de l'axe des ordonnées en $p > 2$ segments :

$$[-\infty; \beta_1[\ ;\ [\beta_1; \beta_2[\ ;\ \ldots\ ;\ [\beta_{p-2}; \beta_{p-1}[\ ;\ [\beta_{p-1}; \infty[$$

avec $\beta_u = F^{-1}_{\mathcal{N}(0,1)}(u/p)$: fractile d'ordre $\frac{u}{p}$ de la loi $\mathcal{N}(0,1)$

2. Agrégation des segments :

$$\begin{array}{lrll} \text{si} & \bar{x}_j \in]-\infty; \beta_1[& \text{alors} & \hat{x}_j = a \\ \text{si} & \bar{x}_j \in [\beta_1; \beta_2[& \text{alors} & \hat{x}_j = b \\ \text{si} & \bar{x}_j \in [\beta_2; +\infty[& \text{alors} & \hat{x}_j = c \end{array}$$

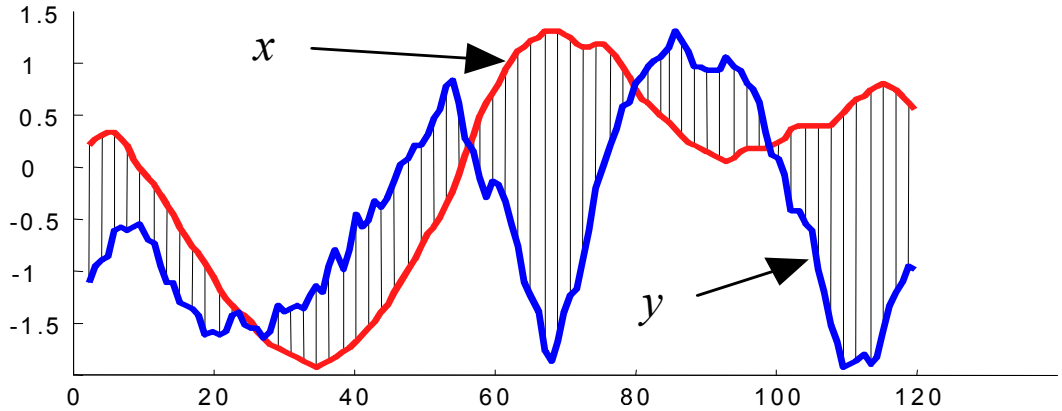


SAX (Etape 2) : agrégation des segments

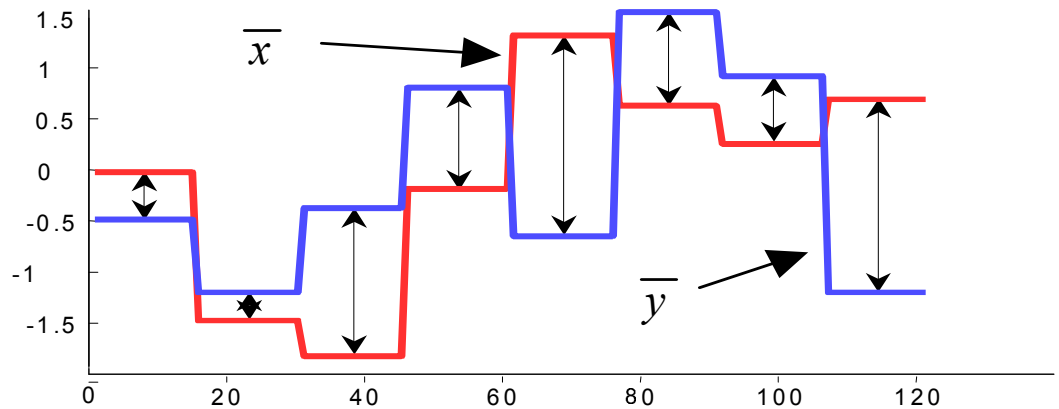
	Valeurs de β							
p	3	4	5	6	7	8	9	10
β_1	-0.43	-0.67	-0.84	-0.97	-1.07	-1.15	-1.22	-1.28
β_2	0.43	0	-0.25	-0.43	-0.57	-0.67	-0.76	-0.84
β_3		0.67	0.25	0	-0.18	-0.32	-0.43	-0.52
β_4			0.84	0.43	0.18	0	-0.14	-0.25
β_5				0.97	0.57	0.32	0.14	0
β_6					1.07	0.67	0.43	0.25
β_7						1.15	0.76	0.52
β_8							1.22	0.84
β_9								1.28

SAX (Etape 3) : distance entre séries temporelles

$$D(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$



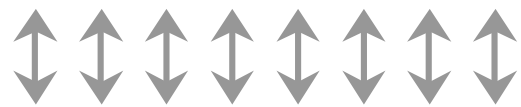
$$D(\bar{\mathbf{x}}, \bar{\mathbf{y}}) = \sqrt{\frac{n}{m}} \sqrt{\sum_{j=1}^{m} (\bar{x}_j - \bar{y}_j)^2}$$



SAX (Etape 3) : distance entre séries temporelles

$$D(\hat{\mathbf{x}}, \hat{\mathbf{y}}) = \sqrt{\frac{n}{m}} \sqrt{\sum_{j=1}^{m} \widetilde{d}(\hat{x}_j, \hat{y}_j)^2}$$

$$\hat{x} = \mathbf{b\ a\ a\ b\ c\ c\ b\ c}$$



$$\hat{y} = \mathbf{b\ a\ b\ c\ a\ c\ c\ a}$$

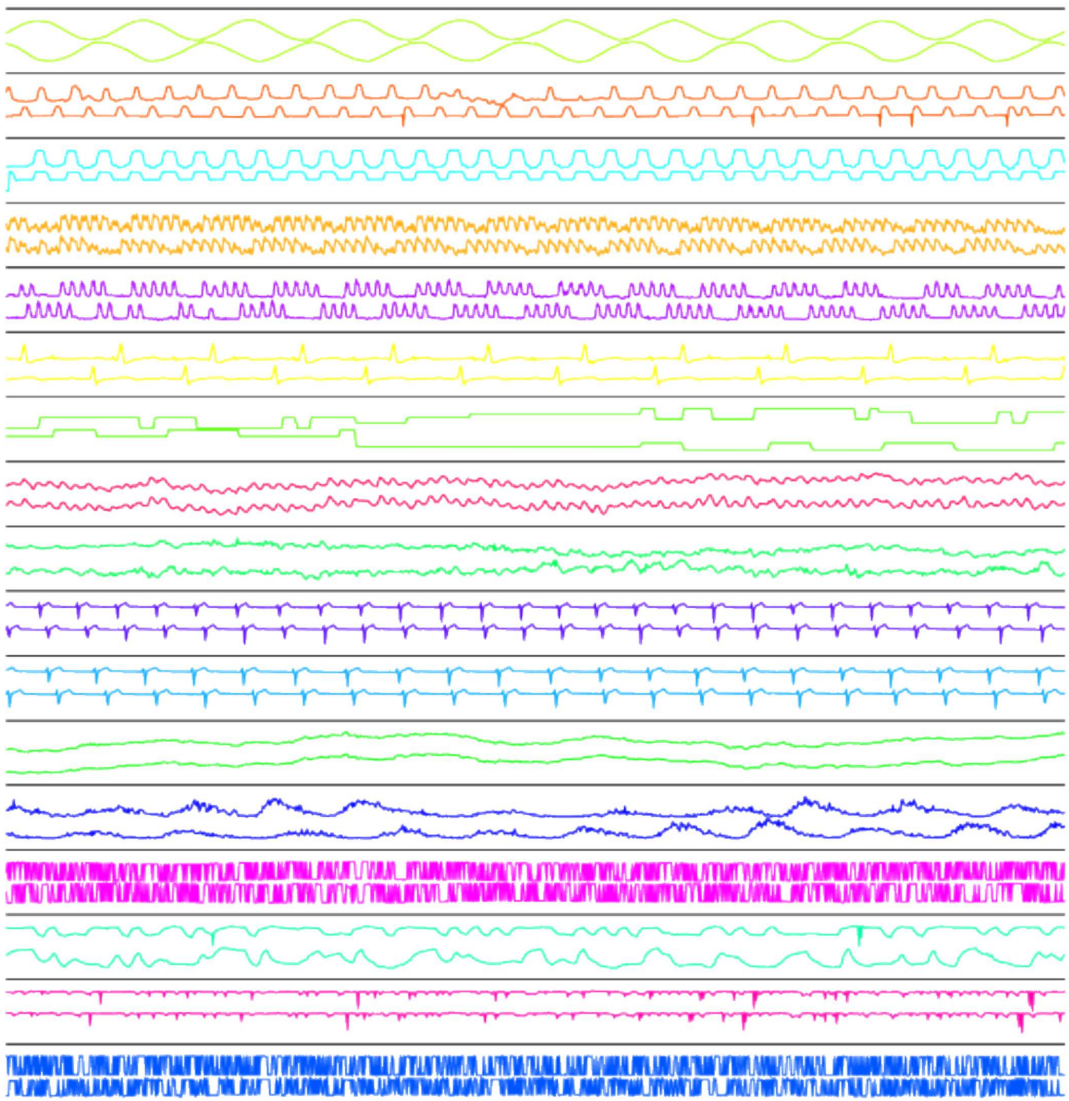
avec

$$\widetilde{d}(c_r, c_s) = \begin{cases} 0 & \text{si } |r-s| \leq 1 \\ \beta_{\max(r,s)-1} - \beta_{\min(r,s)} & \text{sinon} \end{cases}$$

avec $r, s = 1, \ldots, p$ et $c_r, c_s \in \mathcal{A}$

Exemple d'application

Appariement de séries temporelles issues de différents domaines (Keogh et al. 2004)



Classification de séries temporelles avec R

```
library(TSdist)
library(TSclust)
library(cluster)

data("synthetic.tseries")
true_cluster=rep(1:6, each=3)
ts.plot(synthetic.tseries)

# distance euclidienne
clus.eucl = pam(diss(synthetic.tseries, "EUCL"), k=6)
# distance basée sur la correlation (Pearson)
clus.cor = pam(diss(synthetic.tseries, "COR"), k=6)
# distance Dynamic Time Warping
clus.dtw = pam(diss(synthetic.tseries, "DTW"), k=6)
# distance basée sur les coefficients de Fourier
clus.four = KMedoids(data=synthetic.tseries, k=6, "fourier")
# distance basée sur les coefficients d'ondelettes
clus.dwt = pam(diss(synthetic.tseries, "DWT"), k=6)
# distance basée sur les coefficients d'un modele ARIMA
clus.ar = pam(diss(synthetic.tseries, "AR.PIC"), k=6)
# distance basée sur la representation SAX
SAX.plot(synthetic.tseries[,1:2],w=50,alpha=7)
clus.sax = pam(diss(synthetic.tseries, "MINDIST.SAX",w=50,alpha=7), k=6)
```

Classification de séries temporelles avec R

```
library(TSdist)
library(TSclust)
library(cluster)

data("synthetic.tseries")
true_cluster=rep(1:6, each=3)
ts.plot(synthetic.tseries)

# distance euclidienne
clus.eucl = KMedoids(data=synthetic.tseries, k=6, "euclidean")
# distance basée sur la correlation (Pearson)
clus.cor = KMedoids(data=synthetic.tseries, k=6, "cor")
# distance Dynamic Time Warping
clus.dtw = KMedoids(data=synthetic.tseries, k=6, "dtw")
# distance basée sur les coefficients de Fourier
clus.four = KMedoids(data=synthetic.tseries, k=6, "fourier")
# distance basée sur les coefficients d'un modele ARIMA
clus.ar = KMedoids(data=synthetic.tseries, k=6, "ar.pic")
# distance basée sur la representation SAX
SAX.plot(synthetic.tseries[,1:2],w=50,alpha=7)
clus.sax = KMedoids(data=synthetic.tseries, k=6, "mindist.sax", w=50,alpha=7)
```