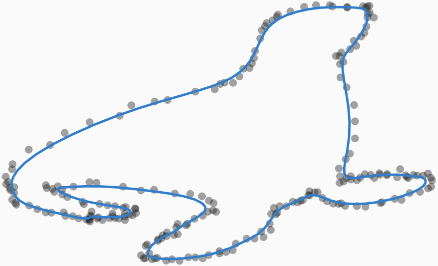
Splines périodiques, fused lasso et reconstruction de courbes fermées

Olivier Binette

24 avril 2018

Université du Québec à Montréal



Introduction

Introduction

Problème de la régression globale

- $\mathbb{M}$ une surface courbe ;
- famille $\{y(x) \mid x \in \mathbb{M}\}$ de variables aléatoires à valeurs dans $\mathbb{R}^k$;
- observations $\{(x_i, y(x_i))\}_{i=1}^n$ pour $\{x_i\}$ fixé.

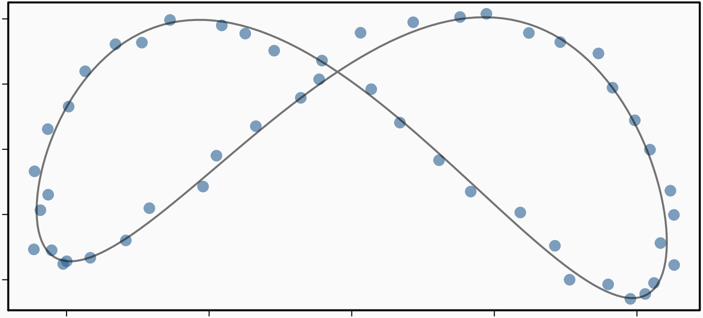
On cherche alors à estimer le plongement

$$\mathbb{E}[y] : \mathbb{M} \to \mathbb{R}^k : x \mapsto \mathbb{E}[y(x)],$$

i.e. on veut prédire $y(x)$ de façon optimale relativement à une perte L^2 et pour $x \in \mathbb{M}$ arbitraire.

Exemple

- $\mathbb{M} = \mathbb{S}^1 = \mathbb{R}/2\pi\mathbb{Z}$ le cercle et y un bruit blanc gaussien centré sur
  $\mathbb{E}[y(u)] = (2\sin(u)+\sin(u+1.6), \cos(2u)+\cos(u)+\cos(u+3))$.
  On observe $y(x_i) \in \mathbb{R}^2$ pour $x_i = 2\pi i/n$, $i = 1, 2, \ldots, n$.



Introduction

Approche

- base de fonctions $\{\phi_j : \mathbb{M} \to \mathbb{R}\}_{j=1}^d$ et modèle

$$\mathcal{H} = \{f_\beta : x \mapsto \Phi(x)^T \beta \mid \beta \in \mathbb{R}^d\}$$

  où $\Phi(x) = [\phi_1(x) \cdots \phi_d(x)]^T$.
- on estime $\mathbb{E}[y]$ par

$$\operatorname*{argmin}_{f \in \mathcal{H}} \sum_i (y_i - f(x_i))^2 + \pi(f)$$

  où π est une fonction de pénalisation sur $\mathcal{H}$.

Construction des espaces de fonctions

Principes généraux

Rappel : $\mathcal{H} = \{f_\beta : x \mapsto \Phi(x)^T\beta \mid \beta \in \mathbb{R}^d\}$, où $\Phi(x) = (\phi_1(x), \ldots, \phi_d(x))$.

- On prend pour $\{\phi_j\}$ une partition de l'unité :
  - $\phi_j \geq 0$;
  - $\sum_j \phi_j(x) = 1$ pour tout $x \in \mathbb{M}$.
- Une partition de l'unité *localisée* sur de petites fenêtres de support.

Exemple

On partitionne $\mathbb{M}$ en régions $R_j \ni z_j$ et on pose $\phi_j = 1_{R_j}$. Alors, l'estimation des moindres carrés est

$$\hat{f}_{\texttt{MCO}} = \sum_{j=1}^{d} \texttt{moy}\{y(x_i) \in R_j\} 1_{R_j},$$

où $\texttt{moy}\{y(x_i) \in R_j\} = \sum_{y(x_i) \in R_j} y_i / \#\{y_i \in R_j\}$.

Mal défini si une région R_j ne contient pas d'observations !

Principes généraux

Lissage

Soit $T : L^1(\mathbb{M}) \to L^1(\mathbb{M})$ un opérateur linéaire positif tel que $T(1) = 1$ et soit $\mathbb{M}$ compact. Alors :

- $\{T\phi_j\}$ forme encore une partition de l'unité ;
- il existe une famille $\{U(x) \mid x \in \mathbb{M}\}$ de variables aléatoires telles que $Tf(x) = \mathbb{E}[f(U(x))]$ (Thm représentation Riesz (Rudin, 1987)).

Exemple

Si $\mathbb{M} \subset \mathbb{R}$, $U(x) \sim N(x, 1)|_{\mathbb{M}}$, alors $Tf : x \mapsto \mathbb{E}[f(U(x))]$ est une convolution avec un noyau gaussien.

En raffinant la partition $\{R_j\}$, on obtient des modèles emboîtés $\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_d \subset \cdots$ associés à des bases $\{\phi_{j,d}\}_j$, $d \in \mathbb{N}$.

Proposition (mathstatnotes, 2016)

Soit $(\mathbb{M}, d_{\mathbb{M}})$ un espace métrique compact mesuré de mesure finie et soit $\{R_{j,d}\}_{j=1}^{m_d}$ une suite de partitions de $\mathbb{M}$ telle que $\max_j \operatorname{diam}(R_{j,d}) \to 0$ lorsque $d \to \infty$. Si $\{\phi_{j,d}\}_{j=1}^{m_d}$ est une suite de partitions de l'unité *localisée* sur $\{R_{j,d}\}$ au sens où $\sup_{x\,:\, d_{\mathbb{M}}(x, R_{j,d}) \geq \delta} \phi_{j,d}(x) \to 0$ pour tout $\delta > 0$, alors le modèle non paramétrique

$$\mathcal{H}_\infty = \bigcup_{d \geq 0} \left\{ \sum_{j=1}^{m_d} c_j \phi_{j,d} \mid (c_1, \ldots, c_{m_d}) \in \mathbb{R}^{m_d} \right\}$$

est dense dans $(\mathcal{C}(\mathbb{M}), \|\cdot\|_\infty)$ et dans $L^1(\mathbb{M})$.

Le cas du cercle

Le cas du cercle

Prenons $\mathbb{M} = \mathbb{S}^1 = \mathbb{R}/2\pi\mathbb{Z}$ le cercle. On pose :

- $R_{j,d} = \left[\frac{(2j-1)\pi}{d}, \frac{(2j+1)\pi}{d}\right)$ les arcs disjoints de longueur $2\pi/d$ centrés en $z_{j,d} = 2\pi j/d$, $j = 1, \ldots, d$;
- $U_d(x)$ une variable aléatoire uniformément distribuée sur l'arc $x + R_{0,d}$ centré en x ;
- $K_d : f \mapsto (x \mapsto \mathbb{E}[f(U_d(x))])$ l'opérateur de convolution avec $U_d(x)$.

Le cas du cercle

On obtient la base de partition de l'unité

$$\phi_{j,d}^{(\ell)} = K_d^\ell 1_{R_{j,d}}, \qquad j = 1, \ldots, d,$$

où d représente une largeur de fenêtre et ℓ est un paramètre de lissage.

Remarques

- $\phi_{j,d}^{(\ell)}$ est proportionel à la densité de la somme, recentrée en $2\pi j/d$, de $\ell + 1$ variables uniformément distribuées sur $R_{0,d}$.
- Ces fonctions appellées des splines (ici périodiques), et ce sont des polynômes par morceaux de degré ℓ.

Le cas du cercle

Forme analytique

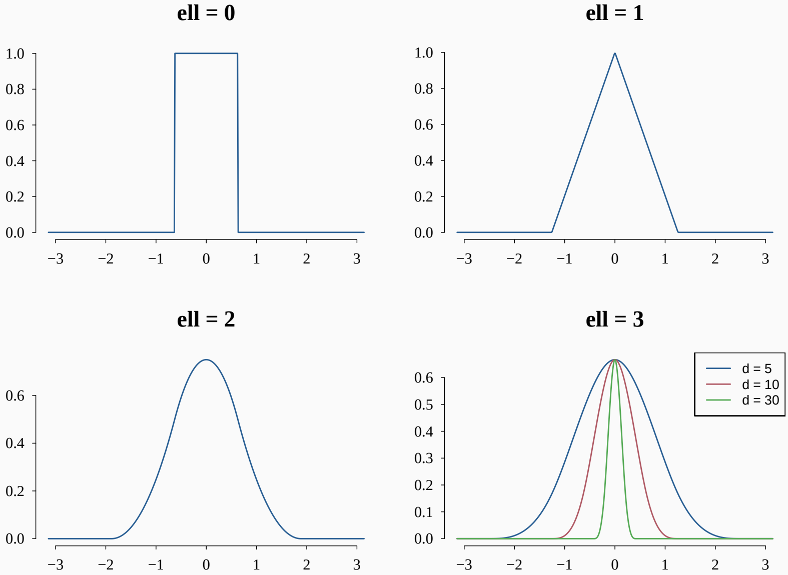
Fonction de répartition F d'une somme de $\ell + 1$ variables uniformément distribuées sur $[0, 1]$:

$$F(x) = \sum_{k=0}^{\lfloor x \rfloor} (-1)^k \frac{(x-k)^{\ell+1}}{k!(\ell+1-k)!}.$$

Avec $u = 2\pi \frac{x-(\ell+1)/2}{d}$ et en dérivant, on obtient

$$\phi_{0,d}^{(\ell)}(u) = \sum_{k=0}^{\lfloor x \rfloor} (-1)^k \binom{\ell+1}{k} \frac{(x-k)^{\ell}}{\ell!}$$

et $\phi_{j,d}^{(\ell)}(u) = \phi_{0,d}^{(\ell)}(u - 2\pi j/d)$.


ell = 0
ell = 1
ell = 2
ell = 3
d = 5
d = 10
d = 30

Diminution de la variation

L'opérateur K_n diminue la variation totale définie par

$$\mathtt{TV}(f) = \sup_{0 \leq t_1 \leq \cdots \leq t_s < 2\pi} \sum_{j=1}^{s} |f(t_{i+1}) - f(t_i)| \qquad (t_{s+1} = t_1),$$

lorsque f est un élément d'un modèle $\mathcal{H}$ de splines (Richards, 1973).

Donc pour $f_\beta^{(\ell)} = \sum_{j=1}^{d} \beta_j \phi_{j,d}^{(\ell)} = K^\ell f_\beta^{(0)}$, on a

$$\mathtt{TV}(f_\beta^{(\ell)}) \leq \mathtt{TV}(f_\beta^{(0)}) = \sum_{j=1}^{d} |\beta_{j+1} - \beta_j| \qquad (\beta_{d+1} = \beta_1). \tag{1}$$

Le cas du cercle

On fait donc la régularisation de la variation totale avec la pénalité *fused lasso*

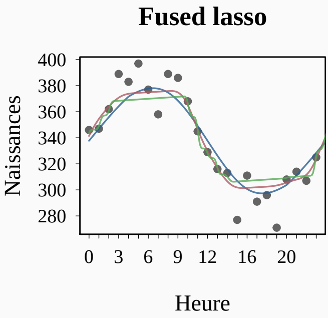
$$\pi_\lambda(f_\beta^{(\ell)}) = \lambda \sum_{j=1}^{d} |\beta_{j+1} - \beta_j|. \tag{2}$$

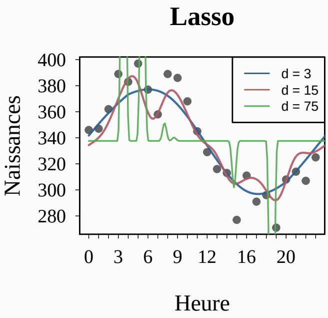
(Tibshirani et al., 2005; Tibshirani and Taylor, 2011)

Remarques

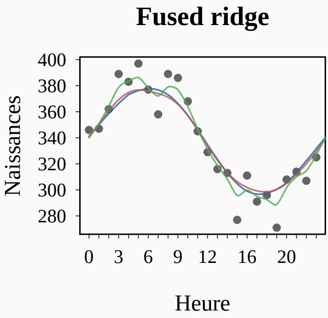
- D'un point de vue heuristique, cette pénalisation cherche à réduire le *run length encoding* de la séquence de paramètres β.
- Variante *fused ridge* :

$$\pi_\lambda^{L^2}(f_\beta^{(\ell)}) = \lambda \sum_{j=1}^{d} |\beta_{j+1} - \beta_j|^2.$$

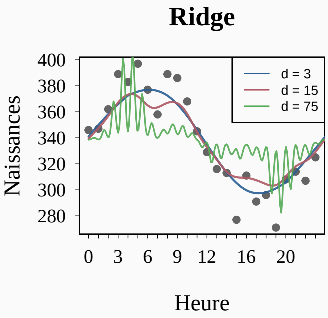

Fused lasso
Naissances
400
380
360
340
320
300
280
0
3
6
9
12
16
20
Heure


Lasso
Naissances
400
380
360
340
320
300
280
0
3
6
9
12
16
20
Heure
d = 3
d = 15
d = 75

Exemples



Fused ridge
Naissances
400
380
360
340
320
300
280
0
3
6
9
12
16
20
Heure

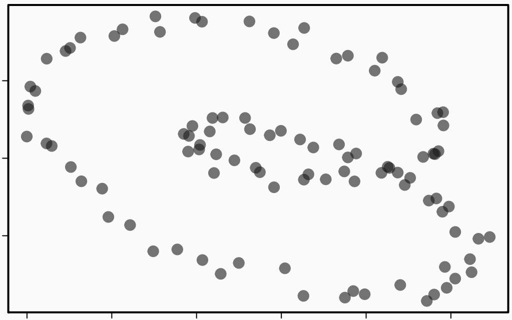


Ridge
d = 3
d = 15
d = 75
Naissances
400
380
360
340
320
300
280
0
3
6
9
12
16
20
Heure

Reconstruction de courbes fermées

Reconstruction de courbes fermées

Soit $\gamma : \mathbb{S}^1 \to \mathbb{R}^2$ une courbe. On observe des points $y(x_i) = \gamma(x_i) + \varepsilon_i$, où ε_i est un bruit de moyenne nulle et où la *paramétrisation* $\{x_i\}$ du nuage de points est inconnue.



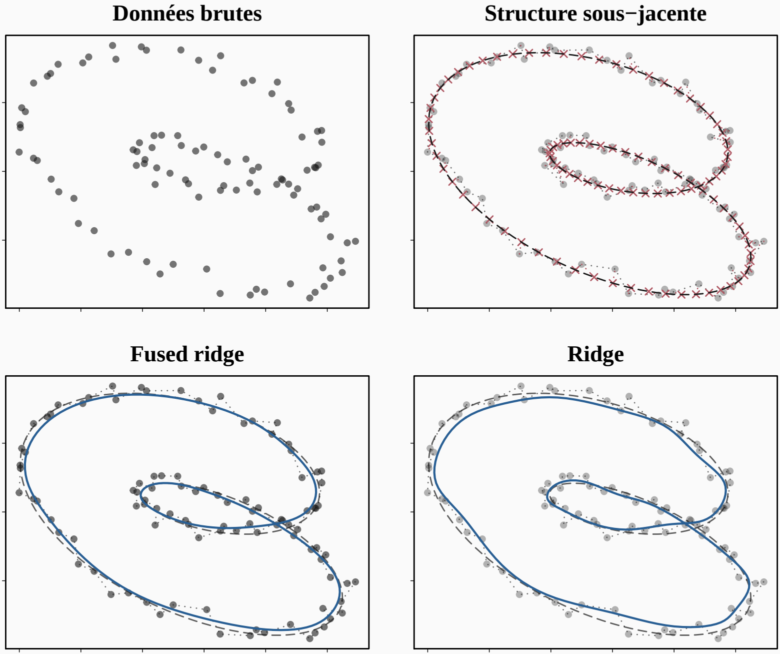
Reconstruction de courbes fermées

Approche

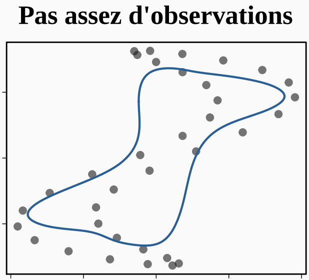
1. Estimer la paramétrisation $\{x_i\} \subset \mathbb{S}^1$.
2. Appliquer nos méthodes de régression.

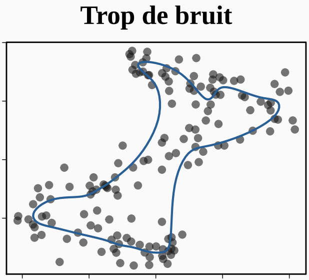
Idée

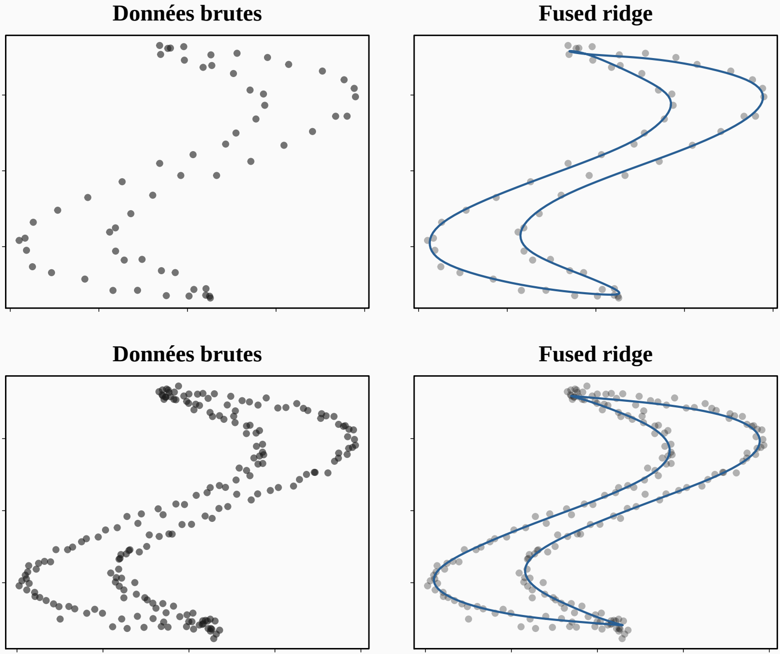
- La caractéristique essentielle de la paramétrisation est l'ordre cyclique des points x_i.
- On estime cet ordre en calculant le cycle hamiltonien minimal (Giesen, 2000; Hahsler et al., 2008).


Données brutes
Structure sous−jacente
Fused ridge
Ridge

Problèmes






Données brutes
Fused ridge
Données brutes
Fused ridge

Conclusion

Conclusion

- Partitions de l'unité et splines : leur interprétabilité permet de faire de la régularisation de façon intelligente $\Leftrightarrow$ spécifier a priori informatif.
- Fused lasso : réduit la sensibilité au choix du nombre d'éléments de base. Pas besoin de tomber dans le nonparamétrique pour obtenir de bons résultats.
- Reconstruction de courbes/surfaces : encore beaucoup de travail à faire !
- `github.com/OlivierBinette/Splinit`

Références

Discussions avec Debdeep Pati / Bayesian closed surface fitting through tensor products.

Binette, O. (2018). Topologie et apprentissage machine. *Notes from the Margin*. Journal étudiant de la Société Mathématique Canadienne.

DeVore, R. A. and G. G. Lorentz (1993). *Constructive approximation*. Springer-Verlag, Berlin.

Références II

Giesen, J. (2000, Dec). Curve reconstruction, the traveling salesman problem, and menger's theorem on length. *Discrete & Computational Geometry 24*(4), 577–603.

Hahsler, M., K. Hornik, and C. Buchta (2008). Getting things in order : An introduction to the r package seriation. *Journal of Statistical Software, Articles 25*(3), 1–34.

Richards, F. B. (1973). Best bounds for the uniform periodic spline interpolation operator. *Journal of Approximation Theory 7*(3), 302 – 317.

Rudin, W. (1987). *Real and complex analysis* (Third ed.). New York : McGraw-Hill Book Co.

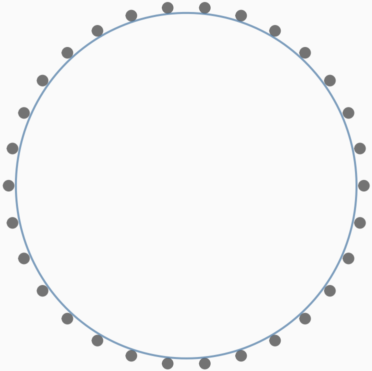
Références III

Tibshirani, R., M. Saunders, S. Rosset, J. Zhu, and K. Knight (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society : Series B (Statistical Methodology) 67*(1), 91–108.

Tibshirani, R. J. and J. Taylor (2011, 06). The solution path of the generalized lasso. *Ann. Statist. 39*(3), 1335–1371.

```r
angles = 2*pi*(1:30)/30
points = cbind(cos(angles), sin(angles))
points = points[sample(1:30, 30), ]

par(mar=c(0,0,0,0));plot(points, col=0, xlab="", ylab="", axes=F)
lines(splinit(points))
```



Code

```r
phi <- function(d, ell) {
  # Base spline function.

  fun <- function(u) {
    x = (d*u/(2*pi) + (ell+1)/2)
    if(x < 0) return(0)
    k = 0:floor(x)
    sum((-1)^k * choose(ell+1, k) * (x-k)^ell / factorial(ell))
  }
  return(Vectorize(fun))
}
```

Code

```
spline.design <- function(X, d, ell) {
  # Design matrix.

  Phi <- function(u, d, ell) {
    u = u - (0:(d-1))*2*pi/d
    u = u %% (2*pi)
    u[u > pi] = u[u > pi] - 2*pi
    phi(d, ell)(u)
  }
  t(sapply(X, function(u) Phi(u, d, ell)))
}

circulant <- function(x, d = length(x)) {
  matrix(x[(1:d - rep(1:d, each=d)) %% d + 1L], ncol=d, byrow=TRUE)
}
```

Code

```
splinit.reg <- function(X, Y, dim=5*ceiling(length(Y)^0.33)+5, degree=3
                        fused = TRUE, type = "ridge",
                        eval.pts=seq(0, 2*pi, length.out=200))
{
  # Regularisation matrix
  rho = if (fused) 1 else 0;
  G =  circulant(c(1, -rho, rep(0, dim-2)))
  G.inv = ginv(G)

  # Spline basis
  M = spline.design(X, dim, degree)
  Mu = spline.design(eval.pts, dim, degree)

  # Cross validated estimation
  alpha = if (type == "ridge") 0 else 1;
  beta = coef(cv.glmnet(M %*% G.inv, Y, alpha=alpha))

  Mu %*% G.inv %*% beta[-1] + beta[1]
}
```

Code

```
splinit <- function(Y, param = NULL,
                    knn = ceiling(nrow(Y)^0.33) + 1, rep = 20, ...)
{
  n = nrow(Y)
  if (missing(param)) {
    # Parameterization estimation
    d = isomapdist(dist(Y), k=knn)
    param = solve_TSP(TSP(d), method="two_opt", rep=rep)
  }

  # Independent regressions on the components of Y
  fun <- function(y) {splinit.reg(2*pi*(1:n)/n, y[param], ...)}
  apply(Y, 2, fun)
}
```