

Classification bayésienne non supervisée de données fonctionnelles

Title: Unsupervised bayesian clustering for functional data

Damien Juery¹, Christophe Abraham¹ et Bénédicte Fontez¹

Résumé : Nous nous intéressons à la classification bayésienne non supervisée de données fonctionnelles. Nous généralisons un modèle de classification de données basé sur le processus de Dirichlet, pour les données fonctionnelles. Contrairement à d'autres articles qui utilisent la dimension finie en projetant les courbes dans des bases de fonctions, ou en considérant les courbes aux temps d'observation, les calculs sont ici réalisés sur les courbes complètes en dimension infinie. Le cadre des espaces de Hilbert à noyau reproduisant nous permet alors d'exprimer les densités, en dimension infinie, des courbes par rapport à une mesure gaussienne. Nous proposons un algorithme généralisant l'algorithme *Gibbs with Auxiliary Parameters* (Neal, 2000) dans le cas de processus. Les performances sont comparées à celles d'une autre méthode déjà existante, puis discutées.

Abstract: We are interested in unsupervised bayesian clustering for functional data. We generalize a data clustering model based on the Dirichlet process, to functional data. Contrary to other papers making use of finite dimension, by decomposing curves into arbitrary basis functions, or by considering curves at their observation times, calculations are here realized onto complete curves in infinite dimension. The reproducing kernel Hilbert space theory permits us to derive densities of curves in respect to a gaussian measure. We thus propose a generalization to the algorithm *Gibbs with Auxiliary Parameters* (Neal, 2000), to the functional case. Performances are compared to those of an already existing method, and then discussed.

Mots-clés : statistique bayésienne, données fonctionnelles, processus de Dirichlet, classification de courbes, MCMC *Keywords:* bayesian statistics, functional data, Dirichlet process, curve clustering, MCMC *Classification AMS 2000 :* 62M99, 62P12, 62H30

1. Introduction

Beaucoup de domaines d'application font appel à des courbes et des signaux, comme en spectrométrie ou lors de l'étude de courbes de croissance. Avec le développement actuel du phénotypage où les données sont recueillies en temps continu, de plus en plus d'utilisateurs ont besoin d'outils capables de classer des courbes. Dans cet article, nous nous intéressons à la classification bayésienne non supervisée de données fonctionnelles. Différentes méthodes de classification de données fonctionnelles ont été proposées en statistique non bayésienne (Bouveyron and Jacques, 2011; Chiou and Li, 2007; Jacques and Preda, 2013; James and Sugar, 2003; Luan and Li, 2004; Ma et al., 2006; Shi and Wang, 2008; Yi et al., 2011). En statistique bayésienne, les méthodes de classification utilisent couramment le processus de Dirichlet (Escobar, 1994; Escobar and West,

Journal de la Société Française de Statistique, Vol. 155 No. 2 185-201 http://www.sfds.asso.fr/journal

© Société Française de Statistique et Société Mathématique de France (2014) ISSN: 2102-6238

 $^{^1\,}$ UMR MISTEA, Montpellier SupAgro, 2 place Pierre Viala, 34060 Montpellier cedex 2.

E-mail: juery@supagro.inra.fr,

E-mail: abraham@supagro.inra.fr

E-mail:fontez@supagro.inra.fr

Juery, Abraham, Fontez

1995; Ishwaran and Zarepour, 2002; Liu, 1996; MacEachern and Müller, 1998; Neal, 2000), mais peu de travaux ont été réalisés avec les données fonctionnelles. Le processus de Dirichlet est utile en classification car il permet de choisir le nombre de classes de manière automatique. Nous nous intéressons à la généralisation de ces approches bayésiennes aux courbes. Les approches actuelles de classification de données fonctionnelles traitent les courbes ou bien de manière multivariée, en les considérant aux temps d'observation (Jackson et al., 2007), ou bien par décomposition dans des bases de fonctions telles que les splines ou les ondelettes (Crandell and Dunson, 2011; Gelfand et al., 2005; Ray and Mallick, 2006). Dans ce dernier cas, Ray and Mallick (2006) proposent l'utilisation de bases d'ondelettes, l'inférence étant alors réalisée sur les coefficients de décomposition. Une approche similaire est celle de Gelfand et al. (2005), mais dans un objectif de prédiction de données spatiales. Plus récemment, Jackson et al. (2007) ont proposé un modèle faisant intervenir les processus gaussiens, calculés aux temps d'observation. Les méthodes dans lesquelles les courbes sont discrétisées aux instants d'observation font intervenir le déterminant de matrices de variance-covariance de lois normales de très grande dimension, ce qui peut induire une instabilité numérique. De plus, les calculs nécessaires dans ces algorithmes seront d'autant plus longs que le nombre de temps d'observation augmente. Dans le cas de la décomposition dans une base de fonctions, l'utilisateur est soumis au problème du choix de la base et à l'adéquation entre modèle d'approximation et données. Nous généralisons le modèle DPM (Dirichlet Process Mixture) (Antoniak, 1974), couramment employé pour classer des observations normales multivariées, à des observations fonctionnelles modélisées par des processus gaussiens. Sur le plan théorique, notre approche se distingue des méthodes précédentes en considérant les courbes complètes en dimension infinie. Notre méthodologie requiert le calcul de densités de processus gaussiens. Ces densités sont calculées à l'aide de la théorie des espaces de Hilbert à noyau reproduisant (Parzen, 1959). Ainsi, le passage à la dimension finie, inévitable pour toute implémentation est relégué à un problème numérique consistant simplement en un calcul d'intégrale intervenant dans le produit scalaire de deux fonctions. Ce calcul est numériquement facile à réaliser. En contrepartie, notre méthode nécessite d'exprimer des densités de processus gaussiens, de paramètres différents, relativement à une même mesure de référence. L'estimation est réalisée grâce à une méthode MCMC selon l'algorithme Gibbs with Auxiliary Parameters de Neal (2000). Le plan de l'article est le suivant. Dans la section 2, nous rappelons le modèle DPM et présentons le modèle de classification proposé dans cet article, ainsi qu'un algorithme de calcul. Dans la section 3, nous présentons les principaux résultats théoriques nécessaires pour l'implémentation. Enfin, dans la section 4, nous analysons les performances de notre méthode sur données simulées et données réelles. Dans ce cadre, nous comparons l'approche proposée à une approche plus habituelle de calcul aux temps d'observation (Jackson et al., 2007).

2. Le modèle fonctionnel et son implémentation

2.1. Mélange de processus de Dirichlet

Supposons que l'on veuille classer un nombre *n* de paramètres quelconques θ_i . Un modèle alors couramment employé est le suivant :

$$\begin{cases} \theta_i | G \stackrel{ind}{\sim} G, \\ G | G_0 \sim DP(\alpha_0, G_0), \end{cases}$$
(1)

- $\theta_1 \sim G_0,$
- θ_2 est égal à θ_1 avec probabilité $\frac{1}{\alpha_0+1}$, et tiré suivant G_0 avec probabilité $\frac{\alpha_0}{\alpha_0+1}$, θ_3 est égal à θ_j , $1 \le j \le 2$, avec probabilité $\frac{1}{\alpha_0+2}$, et tiré suivant G_0 avec probabilité $\frac{\alpha_0}{\alpha_0+2}$, ainsi de suite.

Cet effet de regroupement des θ_i par le processus de Dirichlet a été utilisé par Escobar (1994) dans le cadre des modèles de mélange, afin de classer des données quelconques Y_i . Ces données Y_i sont supposées provenir indépendamment d'une distribution $\mathscr{F}(\theta_i)$ de paramètre θ_i , où la loi de $(\theta_1, ..., \theta_n)$ est donnée par le modèle (1). Ceci nous conduit alors au modèle DPM (Dirichlet Process Mixture):

$$(DPM) \begin{cases} Y_i | \theta_i \stackrel{ind}{\sim} \mathscr{F}(\theta_i), \\ \theta_i | G \stackrel{ind}{\sim} G, \\ G | G_0 \sim DP(\alpha_0, G_0). \end{cases}$$
(2)

D'après ce qui précède, la représentation par urne de Pólya implique la présence de paramètres θ_i ex-æquo. En regroupant alors les données Y_i dont les paramètres θ_i sont égaux, on obtient une classification de ces données. L'objectif consiste donc à estimer les paramètres θ_i . Nous utilisons dans cet article une méthode d'estimation bayésienne.

Comme plusieurs paramètres θ_i sont ex-æquo, on peut proposer une représentation équivalente de la loi jointe de $(Y_1, ..., Y_n)$ dans le modèle (2). D'après la représentation par urne de Pólya, la probabilité pour que θ_j soit égal à une ancienne valeur $\theta_1, ..., \theta_{j-1}$ ne dépend pas des valeurs de ces $\theta_1, ..., \theta_{j-1}$. Définissons alors le vecteur $(c_1, ..., c_n)$ de loi jointe :

$$-c_1 = 1$$

- c_2 est égal à c_1 avec probabilité $\frac{1}{\alpha_0+1}$, et est égal à $k_1 + 1 (= 2)$ avec probabilité $\frac{\alpha_0}{\alpha_0+1}$, c_3 est égal à c_j , $1 \le j \le 2$, avec probabilité $\frac{1}{\alpha_0+2}$, et est égal à $k_2 + 1$ avec probabilité $\frac{\alpha_0}{\alpha_0+2}$, - ainsi de suite,

où k_i désigne le nombre de valeurs distinctes dans le vecteur $(c_1,...,c_i)$. Par la suite, nous notons $CRP(\alpha_0)$ la loi de $(c_1, ..., c_n)$. Nous sommes désormais en mesure d'exprimer un modèle équivalent au modèle (2), qui est le suivant :

$$(DPM) \begin{cases} Y_i | c_i, \phi_c & \stackrel{ind}{\sim} & \mathscr{F}(\phi_{c_i}), \\ c_1, \dots, c_n & \sim & CRP(\alpha_0), \\ \phi_c | G_0 & \stackrel{ind}{\sim} & G_0. \end{cases}$$
(3)

Dans ce dernier modèle, les ϕ_c correspondent aux valeurs distinctes des θ_i , et $(c_1, ..., c_n)$ est le vecteur d'assignation des classes pour chaque observation; ainsi, $\theta_i = \phi_{c_i}$ pour tout entier $i \in \{1, ..., n\}$. La loi CRP (*Chinese Restaurant Process*) est souvent expliquée à partir de la métaphore suivante (Aldous, 1985). Dans cette métaphore, on considère un restaurant avec une infinité de tables, chacune pouvant accueillir une infinité de clients et servant le même plat à tous les clients qui y sont installés. De plus, les tables sont circulaires, de manière à ce que l'ordre des



FIGURE 1. Illustration de la métaphore du restaurant chinois.

clients installés importe peu. Un premier client arrive alors dans le restaurant, s'assied à la table 1 $(c_1 = 1)$ et commande un plat θ_1 . Lorsque le deuxième client arrive, il peut soit s'asseoir à la table du premier client $(c_2 = 1)$ et commander le même plat $(\theta_2 = \theta_1)$, soit s'installer à la table suivante $(c_2 = 2)$ et commander un nouveau plat θ_2 . De manière générale, le i^e client s'assied soit à une table déjà occupée, avec probabilité proportionnelle au nombre de clients qui y sont installés, soit à une table vide, avec probabilité proportionnelle à α_0 . La figure 1 illustre ce procédé.

2.2. Présentation du modèle

Nous modélisons les observations $Y_1, ..., Y_n$ par des processus gaussiens indépendants et identiquement distribués. Rappelons que ces derniers sont la généralisation des lois normales aux espaces de dimension infinie et qu'un processus est gaussien si, et seulement si, toutes ses lois fini-dimensionnelles sont des lois normales multivariées. Un processus gaussien $P_{m,K}$ est alors défini par le biais de deux fonctions qui sont sa fonction moyenne m, et sa fonction de covariance K qui est définie positive. En particulier, la fonction de covariance influe sur la régularité des trajectoires du processus. Dans la littérature, les processus gaussiens sont souvent notés GP(m,K), mais pour des raisons de commodité, nous les noterons $P_{m,K}$ dans toute la suite. Enfin, nous ne considérerons uniquement des processus dont les trajectoires appartiennent à $L^2([0,T])$, espace des fonctions de carré intégrable sur [0,T]. Cet espace étant polonais, il nous garantit l'existence des probabilités conditionnelles.

Nous proposons une généralisation du modèle DPM défini par (2) et (3) au cas infini-dimensionnel. Ce modèle, que nous noterons DPMF (DPM fonctionnel), est défini de la façon suivante :

$$(DPMF) \begin{cases} Y_i | \theta_i & \stackrel{ind}{\sim} & P_{\theta_i, \Sigma}, \\ \theta_i | G & \stackrel{ind}{\sim} & G, \\ G | G_0 & \sim & DP(\alpha_0, G_0), \end{cases}$$
(4)

équivalent d'après ce qui précède au modèle suivant :

$$\begin{cases} Y_i | c_i, \phi_c & \stackrel{ind}{\sim} & P_{\phi_{c_i}, \Sigma}, \\ c_1, \dots, c_n & \sim & CRP(\alpha_0), \\ \phi_c | G_0 & \stackrel{ind}{\sim} & G_0. \end{cases}$$
(5)

Pour des raisons pratiques, nous supposons que $G_0 = P_{\mu,\Sigma_0}$. Dans un premier temps, nous considérons fixes les paramètres μ , Σ , Σ_0 ainsi que α_0 . Dans la section 4, nous envisagerons un modèle plus complet nous permettant de les estimer.

2.3. Implémentation algorithmique

Pour simuler suivant la loi a posteriori de $(\theta_1, ..., \theta_n)$ dans le modèle (2), le plus simple est d'utiliser la représentation par urne de Pólya, que l'on peut écrire de manière formelle :

$$P(d\theta_i|\theta^{-i},\alpha_0,G_0) = \sum_{j\neq i} \frac{1}{\alpha_0 + n - 1} \delta_{\theta_j}(d\theta_i) + \frac{\alpha_0}{\alpha_0 + n - 1} G_0(d\theta_i), \tag{6}$$

où δ_{θ_j} est la mesure de Dirac au point θ_j et θ^{-i} désigne le vecteur $(\theta_1, ..., \theta_n)$ privé de sa i^e composante. L'équation (6) permet de calculer les lois conditionnelles complètes (a posteriori) des θ_i , et ainsi d'élaborer un échantillonneur de Gibbs. Cependant, cette méthode qui fut initialement proposée par Escobar (1994), nécessite de générer chaque θ_i et prend alors souvent trop de temps pour atteindre la convergence vers la loi a posteriori. Afin d'accélérer cette convergence, certains auteurs tirent partie du modèle équivalent (3), afin de s'affranchir de la génération des θ_i . Cette approche permet de gagner du temps en dimension finie et à plus forte raison en dimension infinie. Parmi les algorithmes qui existent, on distingue les échantillonneurs de Gibbs (MacEachern and Müller, 1998; Neal, 2000) des algorithmes de Metropolis-Hastings (Dahl, 2005; Jain and Neal, 2004; Kim et al., 2006; Neal, 2000). Remarquons que Dahl (2005); Jain and Neal (2004); Kim et al. (2006) utilisent des algorithmes de type Split-Merge, où chaque proposition consiste à diviser une classe en deux ou bien réunir deux classes en une seule. Plus récemment, des méthodes variationnelles se sont développées (Blei and Jordan, 2006), mais celles-ci ne faisant pas de simulation a posteriori exacte, nous exclurons ces méthodes par la suite.

Dans un cadre fonctionnel, l'idéal serait de ne pas générer de courbes, ce qui revient à intégrer sur les ϕ_c dans le modèle (3) et inférer uniquement sur les c_i . Pour cela, un algorithme de référence a été proposé par MacEachern (1994) et consiste à simuler les c_i suivant leur loi conditionnelle complète :

$$\mathbb{P}(c_{i} = c | c_{j}, j \neq i, Y_{1}, ..., Y_{n}) \propto \begin{cases} \#\{j \neq i : c_{j} = c\}p(Y_{i} | c, Y^{-i}), \ c = c_{j}, j \neq i, \\ \alpha_{0}p(Y_{i}), \ c \neq c_{j}, j \neq i, \end{cases}$$

où p est la notation générique d'une densité de probabilité et Y^{-i} désigne à nouveau le vecteur $(Y_1, ..., Y_n)$ privé de sa i^e composante. Les quantités $p(Y_i|c, Y^{-i})$ et $p(Y_i)$ correspondent respectivement aux densités de loi prédictive a posteriori sachant la classe et prédictive a priori. Généraliser cet algorithme dans notre cas infini-dimensionnel demanderait donc d'exprimer ces densités relativement à une même mesure de référence. En effet, une mesure différente entraînerait des poids incohérents dans le choix des classes. A notre connaissance, ce problème reste encore non résolu. D'une part, il n'existe pas d'équivalent de la mesure de Lebesgue en dimension infinie. D'autre part, nous n'avons pas réussi à exprimer ces densités par rapport à un même processus gaussien car elles sont associées à des fonctions de covariance différentes.

Plus généralement, parmi les algorithmes que nous avons cités précédemment, ceux inférant uniquement sur les c_i font intervenir les densités (b) et (c) ci-dessous et n'ont pas pu être implémentés :

(a) $p(Y_i | c, \phi_c)$,

Juery, Abraham, Fontez

(b) $p(Y_i)$,

(c) $p(Y_i|c, Y^{-i})$.

Quant aux autres algorithmes faisant intervenir les c_i et les ϕ_c , ils font tous appel à la densité (a) et parfois à la densité (b) également. Seuls ceux ne faisant intervenir que la densité (a) sont implémentables dans notre cas, pour les raisons citées précédemment.

Parmi ces derniers, nous choisissons de généraliser l'algorithme 8 de Neal (2000) qui est, à notre connaissance et d'après l'auteur, l'un des plus performants. Cet algorithme est généralisé en choisissant comme mesure de référence $P_{0,\Sigma}$. Il s'agit d'un algorithme exact de simulation a posteriori, basé sur un échantillonneur de Gibbs. Cet algorithme est une amélioration d'un algorithme proposé par MacEachern and Müller (1998) et qui se base sur les paramètres $(c_1, ..., c_n)$ et $(\phi_1, ..., \phi_n, ..., \phi_{n+m})$. L'entier *m* est fixé arbitrairement et sert à générer *m* valeurs auxiliaires de paramètres. La convergence de cet algorithme est assurée par le même raisonnement que pour celui de MacEachern & Müller, quelle que soit la valeur de *m*. D'après certaines simulations de l'auteur, lorsque *m* est grand, l'algorithme converge rapidement vers sa loi stationnaire. En revanche, chaque itération exigera plus de temps. Étant donné l'entier $m \in \mathbb{N}$, $m \geq 1$, cet algorithme se présente de la façon suivante :

Algorithme 1 : Gibbs with Auxiliary Parameters (pour données fonctionnelles)

pour i=1,...,n faire

fin

 $k^{-} = \#\{c_{j}, j \neq i\};$ $h = k^{-} + m;$ Numéroter les $c_{j}, j \neq i$, dans $\{1, ..., k^{-}\};$ Si $c_{i} = c_{j}$ pour un $j \neq i$, simuler $\phi_{c} \sim GP(\mu, \Sigma_{0})$ pour $k^{-} < c \leq h;$ Si $c_{i} \neq c_{j}$ pour tout $j \neq i$, simuler $\phi_{c} \sim GP(\mu, \Sigma_{0})$ pour $k^{-} + 1 < c \leq h;$ Générer c_{i} à l'aide des probabilités conditionnelles suivantes : $\mathbb{P}(c_{i} = c | c_{j}, j \neq i, \phi_{1}, ..., \phi_{h}, Y_{1}, ..., Y_{n}) \propto \begin{cases} \#\{j \neq i : c_{j} = c\}p(Y_{i} | c, \phi_{c}), \ 1 \leq c \leq k^{-}, \\ \frac{\alpha_{0}}{m}p(Y_{i} | c, \phi_{c}), \ k^{-} + 1 \leq c \leq h. \end{cases}$ fin
pour $c \in \{c_{1}, ..., c_{n}\}$ faire
Simuler ϕ_{c} suivant la loi a posteriori de ϕ sachant les données Y_{i} telles que $c_{i} = c$

3. Résultats sur la vraisemblance et le processus a posteriori

D'après ce qui précède, l'implémentation de notre algorithme 1 requiert :

- les densités $p(Y_i|c, \phi_c)$, pour tous *i* et *c*, relativement à une même mesure de référence,
- la simulation suivant la loi a posteriori de chaque ϕ_c sachant les données qui sont dans la classe *c*.

Pour le premier point, le calcul des densités revient à trouver une mesure de référence P qui permette d'exprimer les dérivées de Radon-Nikodym $\frac{dP_{\phi_c,\Sigma}}{dP}$. Stratonovich and Sosulin (1964) et Kalman & Bucy ont été pionniers pour exprimer $\frac{dP_{m,K}}{dP_{0,K}}$ dans le cas d'un bruit blanc gaussien $P_{0,K}$.

190

D'autres auteurs (Kailath, 1969; Shepp, 1966) ont également travaillé sur ce problème, mais leurs résultats sont souvent peu utilisables car ils font intervenir des fonctions qui n'ont pas de forme explicite et qui ne peuvent pas être approchées facilement numériquement (Kailath and Poor, 1998). La théorie des espaces de Hilbert à noyau reproduisant propose un cadre général dans lequel des densités de processus gaussiens sont explicitables.

Nous rappelons que pour tout noyau de covariance K d'un processus gaussien, il existe un unique espace de Hilbert noté H(K), et appelé *espace de Hilbert à noyau reproduisant de noyau* K, défini comme l'espace des fonctions réelles f sur [0, T] qui vérifient :

(i)
$$\forall t \in [0,T], K(\bullet,t) \in H(K)$$

(ii) $\forall t \in [0,T], \forall f \in H(K), f(t) = (f, K(\bullet, t))_K$,

où $K(\bullet, s)$ est la fonction $s' \mapsto K(s', s)$ et $(f, g)_K$ désigne le produit scalaire sur H(K). Des commentaires sur la régularité des fonctions de H(K) et des écritures explicites de produit scalaire pour différents noyaux de covariance sont proposés dans [3, chap. 7]. Nous donnerons un exemple précis dans la section 4.

Parzen (1963) montre que $\frac{dP_{m,K}}{dP_{0,K}}(X)$ existe si, et seulement si, $m \in H(K)$, et qu'alors on a :

$$\frac{dP_{m,K}}{dP_{0,K}}(X) = e^{(X,m)_K - \frac{1}{2}(m,m)_K},\tag{8}$$

en supposant que $K_N = [K(t_i, t_j)]_{1 \le i,j \le N}$ soit inversible pour tout entier $N \in \mathbb{N}$ et que K soit une fonction faiblement continue sur [0, T] au sens suivant :

(i) $\forall t \in [0, T], K(\bullet, t)$ est continue sur [0, T],

(ii) $\forall t \in [0, T]$, il existe une boule ouverte S(t) contenant t et une constante M(t) telle pour tout $t' \in S(t), K(t', t') \leq M(t)$.

Dans notre cas, chaque processus gaussien $P_{\phi_c,\Sigma}$ admet une densité par rapport au processus $P_{0,\Sigma}$, si, et seulement si, $\phi_c \in H(\Sigma)$. La régularité des trajectoires des ϕ_c étant due, d'après le modèle équivalent (5), à la fonction de covariance Σ_0 , nous considérerons dans toute la suite une fonction de covariance Σ_0 de sorte que la condition $\phi_c \in H(\Sigma)$ soit toujours remplie. Les données étant simulées de manière indépendante, le vecteur $Y_1, \dots, Y_n | c_1, \dots, c_n, \phi_1, \dots, \phi_h$ admet alors une densité par rapport à la mesure produit $P_{0,\Sigma}^{\otimes n}$. En utilisant enfin le théorème de Bayes, qui est valable dans un cadre général (Schervish, 1995), nous pouvons désormais expliciter la loi conditionnelle complète de c_i (7).

Pour le second point, concernant la simulation suivant la loi a posteriori de chaque ϕ_c sachant les données qui sont dans la classe c, nous avons pu démontrer à l'aide de résultats sur la théorie des martingales et des espaces de Hilbert à noyau reproduisant, que la loi conditionnelle de ζ sachant $x_1, ..., x_n$ dans le modèle suivant :

$$\begin{cases} x_1, \dots, x_n | \zeta \stackrel{ind}{\sim} P_{\zeta, \Sigma}, \\ \zeta & \sim P_{\mu, \Sigma_0}, \end{cases}$$

$$\tag{9}$$

est la loi $P_{m,K}$, où les fonctions *m* et *K* sont définies par :

$$m(t) = \mu(t) + \left(\Sigma_0(\bullet, t), \left(\frac{1}{n}\sum_{i=1}^n x_i\right) - \mu\right)_{\Sigma/n+\Sigma_0},$$

$$K(s,t) = \Sigma_0(s,t) - \left(\Sigma_0(\bullet, s), \Sigma_0(\bullet, t)\right)_{\Sigma/n+\Sigma_0}.$$

En particulier, nous supposons que les matrices $[\Sigma(t_i, t_j)]_{1 \le i,j \le N}$ et $[\Sigma_0(t_i, t_j)]_{1 \le i,j \le N}$ sont inversibles pour tout entier $N \in \mathbb{N}$ et que Σ et Σ_0 sont deux fonctions faiblement continues sur [0, T] au sens vu précédemment. A partir de ce résultat, nous déduisons que la loi a posteriori de ϕ_c sachant les données qui sont dans la classe c est la loi $P_{m,K}$ donnée par :

$$m(t) = \mu(t) + \left(\Sigma_0(\bullet, t), \left(\frac{1}{n_c}\sum_{j:c_j=c}Y_j\right) - \mu\right)_{\Sigma/n_c+\Sigma_0},$$

$$K(s,t) = \Sigma_0(s,t) - \left(\Sigma_0(\bullet, s), \Sigma_0(\bullet, t)\right)_{\Sigma/n_c+\Sigma_0},$$
(10)

où $n_c = \#\{j : c_j = c\}.$

4. Résultats et discussion

L'implémentation a été réalisée à partir du logiciel Matlab. Afin d'analyser les performances de notre modèle, nous avons appliqué notre méthode sur trois jeux de données. Comme il est toujours difficile d'évaluer la performance d'un algorithme de classification, nous avons fait le choix, lorsque c'est possible, de comparer le taux de classification correcte (un taux de 10% signifiant que 10% des courbes ont été correctement classées). Dans toute la suite, le taux de classification correcte sera abrégé TCC. Enfin, nous donnons également à titre indicatif le temps moyen requis pour une itération, utile dans le cas du traitement de jeux de données en très grande dimension. Tous les résultats présentés ont été obtenus en produisant 10 000 itérations, avec un temps de chauffe de 1 000 et en retenant 1 itération sur 5. La classification retenue est celle qui apparaît le plus souvent dans les simulations a posteriori. C'est un choix courant dans la littérature, mais précisons qu'il existe d'autres critères tel que celui de Dahl (2006).

4.1. Spécification du modèle

Nous fixons la fonction moyenne μ à zéro et choisissons un noyau de covariance Σ d'un processus d'Ornstein-Uhlenbeck :

$$\Sigma(s,t) = \frac{\sigma^2}{2\beta} e^{-\beta|s-t|},$$

où σ et β sont deux réels strictement positifs. Cela nous permet de travailler avec des produits scalaires simples. En effet, d'après certains résultats (Berlinet and Thomas-Agnan, 2004; Parzen, 1961), l'espace $H(\Sigma)$ est formé des fonctions différentiables sur [0, T] et le produit scalaire sur cet espace est donné par :

$$(f,g)_{\Sigma} = \frac{1}{\sigma^2} \int_0^T (f'(t)g'(t) + \beta^2 f(t)g(t))dt + \frac{\beta}{\sigma^2} (f(0)g(0) + f(T)g(T)).$$
(11)

Rappelons à présent que l'écriture des densités $p(Y_i|c, \phi_c)$ impose, d'après l'équation (8), que chaque $\phi_c \in H(\Sigma)$. Pour assurer cette condition, nous proposons le noyau de covariance Σ_0 suivant :

$$\Sigma_0(s,t) = \frac{\sigma_0^2}{2\beta_0} e^{-\beta_0(s-t)^2},$$
(12)

Journal de la Société Française de Statistique, Vol. 155 No. 2 185-201 http://www.sfds.asso.fr/journal © Société Française de Statistique et Société Mathématique de France (2014) ISSN: 2102-6238

192

lequel garantit des trajectoires de classe C^{∞} .

Sachant que le nombre de classes dans notre modèle dépend fortement du paramètre α_0 (Antoniak, 1974), et afin d'éviter de fixer arbitrairement ce paramètre, nous choisissons de placer une loi a priori *Gamma*(*a*,*b*) sur α_0 , où *a* est le paramètre de forme et *b* le paramètre d'échelle (Escobar and West, 1995). Conditionnellement à α_0 , les lois conditionnelles complètes de $(c_1,...,c_n)$ (7) et de ϕ (10) ne changent pas. Ainsi, il est possible d'adapter notre algorithme 1 afin de rajouter l'inférence sur α_0 . Nous utilisons pour cela la méthode d'Escobar and West (1995), qui introduisent une variable auxiliaire η afin de pouvoir simuler suivant la loi conditionnelle complète de α_0 :

Algorithme 2 : Algorithme 1 avec inférence sur α_0

pour *i*=1,...,*n* **faire** | Simuler *c_i* suivant sa loi conditionnelle complète (7). **fin pour** *c* \in {*c*₁,...,*c_n*} **faire** | Simuler ϕ_c suivant sa loi conditionnelle complète (10). **fin** Simuler une variable η de loi conditionnelle $\eta | \alpha_0 \sim Beta(\alpha_0 + 1, n)$. Poser $\tilde{\pi} = \frac{a+k-1}{a+k-1+n(b-\log(\eta))}$, où *k* est le nombre de classes à l'instant courant. Simuler α_0 suivant sa loi conditionnelle complète qui est :

 $\tilde{\pi}Gamma(a+k,b-\log(\eta)) + (1-\tilde{\pi})Gamma(a+k-1,b-\log(\eta)).$

Le nombre de valeurs auxiliaires est fixé à m = 5. Les hyperparamètres β_0 et σ_0 seront fixés de sorte que la variabilité des données générées à partir de Σ_0 soit de l'ordre de celle des données. Différentes valeurs de β_0 et σ_0 ont été testées, avec pour seule conséquence un temps de convergence de l'algorithme vers la loi stationnaire plus ou moins grand. Concernant les hyperparamètres β et σ , nous proposons de les fixer de manière empirique. En effet, notre expérience montre qu'une modélisation bayésienne complète induirait des problèmes numériques sans améliorer nécessairement les résultats de classification. Ainsi, sachant que les courbes Y_i sont générées à partir des processus gaussiens $P_{\phi_{c_i},\Sigma}$ avec de plus $\Sigma(s,t) = \frac{\sigma^2}{2\beta}e^{-\beta|s-t|}$, les paramètres β et σ sont fixés à partir de l'estimation empirique de la matrice de variance-covariance intra-classe des courbes discrétisées en quelques points.

Enfin, nous pouvons montrer que le processus a posteriori $P_{m,K}$ de l'équation (10) peut être approché par les lois normales fini-dimensionnelles $\mathcal{N}(S,T)$ à l'aide des deux matrices suivantes :

$$S = \mu_N + \Sigma_{0N}^{\top} \left(\frac{\Sigma_N}{n_c} + \Sigma_{0N} \right)^{-1} \left(\frac{\sum_{j:c_j=c} Y_{jN}}{n_c} - \mu_N \right), \tag{13}$$

$$T = \Sigma_{0N} - \Sigma_{0N}^{\top} \left(\Sigma_{0N} + \frac{\Sigma_N}{n_c} \right)^{-1} \Sigma_{0N}, \qquad (14)$$

où l'on note $\Sigma_N = [\Sigma(t_i, t_j)]_{1 \le i, j \le N}$, $\Sigma_{0N} = [\Sigma_0(t_i, t_j)]_{1 \le i, j \le N}$, $Y_{jN} = [Y_j(t_i)]_{1 \le i \le N}$ et enfin $\mu_N = [\mu(t_i)]_{1 \le i \le N}$. Ces matrices correspondent en réalité à l'espérance et à la matrice de variancecovariance de la loi a posteriori de ϕ_c sachant les données qui sont dans la classe *c*, lorsque les lois

Juery, Abraham, Fontez



FIGURE 2. Représentation des courbes simulées, toutes classes confondues.

dans le modèle (10) sont multivariées. Notons finalement que nous pouvons déterminer le nombre de points de discrétisation nécessaires au calcul d'un produit scalaire entre deux fonctions en comparant par exemple des calculs numériques suivant différents pas de discrétisation. Le lecteur pourra se référer à la discussion en fin de cet article pour plus de détails.

4.2. Jeu de données simulées

Le jeu de données simulées est constitué de 40 courbes observées uniformément sur 100 points de l'ensemble [0, 10]. Sur cet ensemble, nous avons généré les quatre polynômes suivants :

$$\begin{cases} s_1(t) = 0.011t^3 - 0.16t^2 + 0.5t, \\ s_2(t) = -0.0075t^4 + 0.149t^3 - 0.91t^2 + 1.7t, \\ s_3(t) = 0.00391t^5 - 0.0977t^4 + 0.854t^3 - 3.05t^2 + 3.7t, \\ s_4(t) = -0.002009t^6 + 0.06026t^5 - 0.6822t^4 + 3.6t^3 - 8.71t^2 + 7.6t, \end{cases}$$

qui ont permis de créer quatre classes. Pour chacune de ces fonctions, nous avons simulé dix processus gaussiens de moyenne s_i et de covariance donnée par celle du processus d'Ornstein-Uhlenbeck, de paramètres $\beta = 10$ et $\sigma = 2.5$. Les données sont générées de manière indépendante et présentées figures 2 et 3.

Nous initialisons notre algorithme avec des valeurs $\beta = \beta_0 = 15$ et $\sigma = \sigma_0 = 1$, fixées de manière arbitraire. Ceci nous permet d'obtenir une première classification, nous permettant alors d'estimer et fixer $\beta = \beta_0 = 14$ et $\sigma = \sigma_0 = 1.58$. La loi a priori Gamma(a, b) sur α_0 est telle que a = 1 et b = 0.5.

En répétant notre algorithme 50 fois, nous obtenons un TCC moyen de 77.90% et un temps moyen par itération de l'ordre de 6s. Les résultats complets sont présentés dans la figure 4 sous forme de diagramme en boîte pour le TCC.

Classification de données fonctionnelles



FIGURE 3. Représentation des courbes simulées, classes séparées.



FIGURE 4. Diagramme en boîte du TCC, obtenu sur 50 répétitions de notre algorithme. La moyenne est de 77.90%.

Enfin, nous avons souhaité connaître le comportement de notre algorithme sur des données simulées avec un noyau de covariance différent de celui utilisé en estimation. Ainsi nous avons simulé des données suivant un noyau de covariance de type Σ_0 ou encore un noyau de covariance exponentiel. Nous obtenons respectivement un TCC de l'ordre de 62% et 40%. Ces résultats sont évidemment moins bons que lorsque les données sont simulées suivant une covariance de type Ornstein-Uhlenbeck. Ceci est du à la nature des données, dont la régularité des trajectoires ne correspond plus à celle du noyau utilisé en estimation.

Juery, Abraham, Fontez



FIGURE 5. Représentation des courbes de croissance.

4.3. Jeu de données de courbes de croissance

Notre second jeu de données est un jeu de données réelles provenant de l'étude Berkeley, reprise par Tuddenham and Snyder (1954). Cette étude a permis de produire des courbes de croissance de garçons et de filles, de la naissance à l'âge adulte. Les données sont disponibles dans le package *fda* de R. Ce jeu de données représente l'évolution temporelle de la taille de 54 filles et de 39 garçons, de 1an à 18ans et à 31 instants variables. Les courbes sont représentées figure 5.

Nous initialisons notre algorithme avec des valeurs $\beta = \beta_0 = 1$ et $\sigma = \sigma_0 = 5$ afin d'obtenir une première classification. Cette classification nous permet alors d'estimer, et de fixer les paramètres $\beta = \beta_0 = 0.7$ et $\sigma = \sigma_0 = 9.5$. La loi a priori Gamma(a,b) sur α_0 est telle que a = 1 et b = 0.5.

Sur 50 répétitions, notre algorithme affiche un TCC moyen de 70.97% et a toujours su retrouver une classification constituée de 2 classes. Le temps moyen par itération est de 10s. Ce jeu de données ayant déjà été étudié précédemment en détails par Jacques and Preda (2013), nous reportons à titre de comparaison les TCC d'autres méthodes dans le tableau 1. Dans leur étude, les auteurs comparent la méthode qu'ils proposent (funclust) à trois autres méthodes fonctionnelles qui sont celles de James and Sugar (2003) (fclust), Chiou and Li (2007) (kCFC) et Bouveyron and Jacques (2011) (funHDDC). Toutes ces méthodes sont parfaitement adaptées au cas de courbes. Les auteurs considèrent également des méthodes fini-dimensionnelles, qui ont été appliquées sur les scores d'une analyse en composantes principales fonctionnelle, sur les observations discrétisées et aussi sur les coefficients d'une décomposition en splines cubiques. Le lecteur pourra se référer à l'article Jacques and Preda (2013) pour plus de détails. Nous reportons tous les TCC dans le tableau 1.

Notons néanmoins que Jacques & Preda ont comparé leur méthode sur d'autres jeux de données et que les résultats sont parfois inversés selon les situations. Les méthodes kCFC et funHDDC peuvent afficher des taux très bons, mais l'inconvénient est qu'elles sont toutes basées sur des modèles qui demandent de choisir une base d'approximation, ce que notre méthode ne requiert pas.

Taux de classification correcte (TCC)					
			Données discrétisées	Splines cubiques	Scores ACPF
DPMF	70.97%	HDDC	56.99%	50.51%	97.85%
fclust	69.89%	MixtPPCA	62.36%	50.53%	97.85%
kCFC	93.55%	GMM	65.59%	63.44%	95.70%
funHDDC	96.77%	k-means	65.59%	66.67%	64.52%
funclust	69.98%	hclust	51.61%	75.27%	68.81%

TABLEAU 1. TCC obtenus sur différentes méthodes pour le jeu de données de courbes de croissance. Hormis pour le DPMF, les résultats ont été obtenus par Jacques and Preda (2013).



FIGURE 6. Représentation des courbes de spectrométrie.

Rappelons en effet que nous n'avons ni à choisir le nombre de classes ni la base d'approximation. Cette spécificité du DPMF permet de le rendre indépendant de toute base d'approximation, ce qui n'est pas le cas des autres méthodes.

4.4. Jeu de données de spectrométrie

Ce dernier jeu de données réelles est issu du projet SpecBio, qui a pour objectif de caractériser des sols à l'aide de la spectrométrie infrarouge. Il est constitué de 78 courbes qui sont des indicateurs spectraux de caractéristiques biologiques de sol, mesurés de 350nm à 2 500nm et avec un pas de 1nm. Tout l'intérêt ici consiste à produire des classes de courbes correspondant à des caractéristiques bien précises du sol. Les courbes sont représentées figure 6.

Nous initialisons notre algorithme avec des valeurs $\beta = \beta_0 = 0.5$ et $\sigma = \sigma_0 = 0.01$ afin d'obtenir une première classification, nous permettant d'estimer et fixer $\beta = \beta_0 = 0.7$ et $\sigma = \sigma_0 = 0.14$. La



FIGURE 7. Résultats de classification, toutes classes confondues.

loi a priori Gamma(a,b) sur α_0 est telle que a = 1 et b = 0.5. La figure 7 présente les résultats de classification.

Pour ce jeu de données, nous ne pouvons pas donner de TCC car nous ne connaissons pas les vraies classes. Notre algorithme DPMF a su trouver une classification constituée de 3 classes et le temps moyen par itération est de 63s. Afin d'étudier la pertinence de ces résultats, nous avons reporté la même classification sur les données de masse de carbone organique présente dans le sol. En effet, à chaque courbe est associée une valeur de masse en carbone organique. La figure 8 présente ces résultats sous forme de diagramme en boîte. Chaque diagramme de couleur correspond à la classe associée. On constate qu'il existe un lien entre nos résultats et les données carbone. Des études plus approfondies pourraient être menées afin de démontrer le lien entre la classification obtenue et les caractéristiques attendues des sols.

4.5. Discussion

Le modèle DPMF que nous proposons offre une approche fonctionnelle de classification de courbes, et s'est révélé capable de classer des courbes observées en un très grand nombre de points. Nous avons également voulu le comparer à la méthode de Jackson et al. (2007). Leur modèle est un DPM appliqué aux processus gaussiens, mais où les auteurs proposent un modèle fini-dimensionnel. En effet, ils ne considèrent pas les courbes en tant qu'objets de dimension infinie mais discrétisées aux temps d'observation. Leur modèle de classification opère à la fois sur la moyenne et le noyau de covariance, avec des covariances entre classes pouvant être différentes. Pour des raisons de temps, nous n'avons pu produire assez d'itérations pour conclure quant à leur méthode, car une itération demande environ 6h39 et presque 1 mois serait nécessaire pour produire une centaine d'itérations.



FIGURE 8. Résultats de la classification obtenue sur la masse de carbone organique (g/kg). Chaque diagramme de couleur correspond à la classe associée.

Notre modèle considère les courbes en dimension infinie. Une étape de discrétisation est nécessaire pour calculer les produits scalaires entre les courbes. L'avantage de notre approche réside dans le fait que le pas de discrétisation peut être choisi relativement large, dans la mesure où le produit scalaire entre les courbes discrétisées donne une approximation satisfaisante. Notre expérience montre que ce pas est souvent beaucoup plus grossier que le pas des courbes réellement observées, d'où un gain substantiel de temps de calcul numérique. Par ailleurs dans notre cas, les densités des courbes sont exprimées relativement à une mesure gaussienne, car il n'existe pas d'analogue à la mesure de Lebesgue en dimension infinie. Ceci permet d'éviter quelques instabilités numériques, notamment dans le calcul du déterminant de matrice de variance-covariance en grande dimension.

Cet article utilise les processus de Dirichlet pour classer des données fonctionnelles, mais d'autres processus comme celui de Pitman-Yor sont aussi utilisés pour la classification non supervisée. Ces processus permettent un choix automatique du nombre de classes, mais ce choix est sensible aux hyperparamètres du processus. Plusieurs articles (Miller and Harrison, 2013a,b) soulignent l'inconsistance des processus de Dirichlet et Pitman-Yor pour inférer sur le nombre de classes, lorsque les hyperparamètres sont fixes. Aussi, pour limiter ces problèmes, on pose une loi a priori sur ces paramètres et ils sont inférés a posteriori. Notons que Kimura et al. (2013) obtiennent de bons résultats de classification en inférant sur α_0 à partir d'un algorithme EM.

Remarquons pour finir qu'il est possible de généraliser notre méthode au cas où chaque courbe Y_i disposerait de sa propre fonction de covariance Σ_i . En généralisant alors le DPMF pour inférer à la fois sur les moyennes et les paramètres du noyau de covariance, on obtiendrait des covariances

 Σ_i égales dans chaque classe. Notre méthode est alors directement généralisable, avec $\prod_{i=1}^n P_{0,\Sigma_i}$ comme mesure commune pour la vraisemblance, et un résultat similaire pour la loi décrite dans l'équation (9).

Remerciements

Les auteurs tiennent à remercier l'éditeur en chef du Journal de la SFdS ainsi que les relecteurs pour leurs commentaires constructifs qui nous ont permis d'améliorer la qualité de l'article.

Références

Aldous, D. (1985). Exchangeability and related topics, pages 1–198. Springer.

- Antoniak, C. E. (1974). Mixtures of Dirichlet Processes with applications to Bayesian nonparametric problems. *Ann. Stat.*, 2(6):1152–1174.
- Berlinet, A. and Thomas-Agnan, C. (2004). *Reproducing kernel Hilbert spaces in Probability and Statistics*. Kluwer Academic Publishers.
- Blackwell, D. and MacQueen, J. B. (1973). Ferguson Distributions via Polya urn schemes. Ann. Stat., 1(2):353-355.
- Blei, D. M. and Jordan, M. I. (2006). Variational Inference for Dirichlet Process Mixtures. *Bayesian Analysis*, 1(1):121–144.
- Bouveyron, C. and Jacques, J. (2011). Model-based clustering of time series in group-specific functional subspaces. *Advances in Data Analysis and Classification*, 5:281–300.
- Chiou, J. M. and Li, P. L. (2007). Functional clustering and identifying substructures of longitudinal data. J. R. Stat. Soc. Ser. B Stat. Methodol., 69:679–699.
- Crandell, J. L. and Dunson, D. B. (2011). Posterior simulation across nonparametric models for functional clustering. *Sankhya B*, (73) :42–61.
- Dahl, D. B. (2005). Sequentially-Allocated Merge-Split Sampler for Conjugate and Nonconjugate Dirichlet Process Mixture Models. Technical report, Texas A&M University.
- Dahl, D. B. (2006). *Model-Based Clustering for Expression Data via a Dirichlet Process Mixture Model*, chapter 10, pages 201–218. Cambridge University Press.
- Escobar, M. D. (1994). Estimating Normal Means With a Dirichlet Process Prior. J. Am. Stat. Assoc., 89(425):268-277.
- Escobar, M. D. and West, M. (1995). Bayesian Density Estimation and Inference Using Mixtures. J. Am. Stat. Assoc., 90(430) :577–588.
- Ferguson, T. S. (1973). A Bayesian analysis of some nonparametric problems. Ann. Stat., 1(2):209-230.
- Gelfand, A. E., Kottas, A., and MacEachern, S. N. (2005). Bayesian Nonparametric Spatial Modeling With Dirichlet Process Mixing. J. Am. Stat. Assoc., 100(471) :1021–1035.
- Ishwaran, H. and Zarepour, M. (2002). Exact and approximate sum representations for the Dirichlet process. *Can. J. Stat.*, 30(2):269–283.
- Jackson, E., Davy, M., Doucet, A., and Fitzgerald, W. J. (2007). Bayesian Unsupervised Signal Classification by Dirichlet Process Mixtures of Gaussian Processes. In *IEEE Int. Conf. Acoust. Spee.*, 2007, pages 1077–1080.
- Jacques, J. and Preda, C. (2013). Funclust : a curves clustering method using functional random variable density approximation. *Neurocomputing*, (112) :164–171.
- Jain, S. and Neal, R. M. (2004). A Split-Merge Markov Chain Monte Carlo Procedure for the Dirichlet Process Mixture Model. J. Comput. Graph. Stat., 13(1):158–182.
- James, G. M. and Sugar, C. A. (2003). Clustering for sparsely sampled functional data. J. Am. Stat. Assoc., 98(462):397– 408.
- Kailath, T. (1969). A General Likelihood-Ratio Formula for Random Signals in Gaussian Noise. *IEEE T. Inform. Theory*, 15(3):350–361.
- Kailath, T. and Poor, H. V. (1998). Detection of Stochastic Processes. IEEE T. Inform. Theory, 44(6):2230–2259.
- Kim, S., Tadesse, M. G., and Vannucci, M. (2006). Variable selection in clustering via Dirichlet process mixture models. *Biometrika*, 93(4):877–893.
 Kimura, T., Tokuda, T., Nakada, Y., Nokajima, T., Matsumoto, T., and Doucet, A. (2013). Expectation-maximization
- algorithms for inference in Dirichlet processes mixture. *Pattern Analysis and Applications*, 16(1):55 67.

Journal de la Société Française de Statistique, Vol. 155 No. 2 185-201 http://www.sfds.asso.fr/journal

© Société Française de Statistique et Société Mathématique de France (2014) ISSN: 2102-6238

200

- Liu, J. S. (1996). Nonparametric hierarchical Bayes via sequential imputations. Ann. Stat., 24(3):911–930.
- Luan, Y. and Li, H. (2004). Model-based methods for identifying periodically expressed genes based on time course microarray gene expression data. *Bioinformatics*, 20(3):332–339.
- Ma, P., Castillo-Davis, C. I., Zhong, W., and Liu, J. S. (2006). A data-driven clustering method for time course gene expression data. *Nucleic Acids Res.*, 34(4) :1261–1269.
- MacEachern, S. N. (1994). Estimating Normal Means With a Conjugate Style Dirichlet Process Prior. Commun. Stat. Simulat., (23):727 – 741.
- MacEachern, S. N. and Müller, P. (1998). Estimating Mixture of Dirichlet Process Models. J. Comput. Graph. Stat., 7(2):223–238.
- Miller, J. W. and Harrison, M. T. (2013a). A simple example of Dirichlet process mixture inconsistency for the number of components. Technical Report arXiv:1301.2708 [math.ST], Brown University.
- Miller, J. W. and Harrison, M. T. (2013b). Inconsistency of Pitman-Yor process mixtures for the number of components. Technical Report arXiv:1309.0024 [math.ST], Brown University.
- Neal, R. M. (2000). Markov chain sampling methods for Dirichlet Process mixture models. J. Comput. Graph. Stat., 9(2):249–265.
- Parzen, E. (1959). Statistical inference on time series by Hilbert space methods I. Technical report, Stanford University. Parzen, E. (1961). Regression Analysis of Continuous Parameter Time Series. In *Proc. Fourth Berkeley Symp. on*
- Math. Statist. and Prob., pages 469–489, Stanford University. University of California Press. Parzen, E. (1963). Probability Density Functionals and Reproducing Kernel Hilbert Spaces. In *Time Series Analysis*,
- chapter 11, pages 155–169. Wiley.
- Ray, S. and Mallick, B. (2006). Functional clustering by Bayesian wavelet methods. J. R. Stat. Soc. Ser. B Stat. Methodol., 68(2):305–332.
- Schervish, M. J. (1995). Theory of Statistics. Springer-Verlag.
- Shepp, L. A. (1966). Radon-Nikodym Derivatives of Gaussian Measures. Ann. Math. Stat., 37(2):321-354.
- Shi, J. Q. and Wang, B. (2008). Curve prediction and clustering with mixtures of Gaussian process functional regression models. *Stat. Comput.*, 18(3):267–283.
- Stratonovich, R. L. and Sosulin, Y. G. (1964). Optimal Detection of a Markov Process in Noise. *Eng. Cybernet*, 6:7–19.
- Tuddenham, R. D. and Snyder, M. M. (1954). Physical Growth of California Boys and Girls from Birth to Eighteen years, pages 183–364. University of California Press.
- Yi, G., Shi, J. Q., and Choi, T. (2011). Penalized Gaussian Process Regression and Classification for High-Dimensional Nonlinear Data. *Biometrics*, 67(4):1285–1294.