

---

Mémoire de stage de Master 1

APPRENTISSAGE NON LINÉAIRE DANS UN  
ESPACE DE HILBERT

---

ABEL DOUZAL  
*École normale supérieure PSL*  
abel.douzal@ens.fr

Sous l'encadrement de  
KENJI FUKUMIZU  
*Institute of Statistical Mathematics (ISM)*  
Tokyo Japon  
fukumizu@ism.ac.jp

5 septembre 2024

# Table des matières

|                                                                        |           |
|------------------------------------------------------------------------|-----------|
| <b>Introduction</b>                                                    | <b>2</b>  |
| <b>1 Apprentissage non linéaire dans un RKHS</b>                       | <b>3</b>  |
| 1.1 Contexte et motivation . . . . .                                   | 3         |
| 1.2 Apprentissage dans un RKHS . . . . .                               | 4         |
| 1.3 Analyse théorique de la borne supérieure . . . . .                 | 4         |
| 1.3.1 Approche générale . . . . .                                      | 5         |
| 1.3.2 Premier membre : partie effectivement apprise . . . . .          | 5         |
| 1.3.3 Second membre : partie orthogonale . . . . .                     | 6         |
| 1.3.4 Bilan . . . . .                                                  | 6         |
| 1.4 Perspective et travaux futurs . . . . .                            | 6         |
| <b>2 Réduction de dimension par représentants : PkPCA</b>              | <b>8</b>  |
| 2.1 Contexte . . . . .                                                 | 9         |
| 2.2 Problème et analyse . . . . .                                      | 10        |
| 2.3 Algorithme . . . . .                                               | 12        |
| 2.3.1 Procédé général . . . . .                                        | 12        |
| 2.3.2 Variantes . . . . .                                              | 13        |
| 2.3.3 Intérêt de la méthode . . . . .                                  | 14        |
| 2.4 Expérimentations . . . . .                                         | 14        |
| 2.5 Cas d'application : variables aléatoires conditionnelles . . . . . | 16        |
| <b>Conclusion</b>                                                      | <b>19</b> |
| <b>Bibliographie</b>                                                   | <b>20</b> |

# Introduction

Les problèmes d'apprentissage non linéaire dans un espace de Hilbert de dimension infinie associé à un noyau fournissent une expressivité importante a même de fournir de bonnes représentations de distributions de données à partir d'échantillons.

La plupart des méthodes à noyaux courantes dont les plus connues sont les versions kernelisées de méthodes telles que les *machines à vecteurs de support* (SVM) et de l'*analyse en composantes principales* (ACP), présentées notamment dans [8], sont des méthodes non linéaires. La non linéarité de ces méthodes provient du noyau (par exemple gaussien) choisi. Toutefois ces méthodes dites kernelisées non linéaires correspondent le plus souvent à la méthode classique linéaire, par exemple une SVM ou une ACP, mais appliquée dans l'espace latent (espace de Hilbert) associé au kernel, aussi appelé RKHS pour *reproducing kernel Hilbert space*. Ces méthodes présentent de bonnes propriétés notamment de lissage des distributions de points. Une autre famille de méthodes permettant de réaliser de l'apprentissage non linéaire sont les réseaux de neurones, ils permettent l'approximation de fonctions complexes par leur structure en graphe de calcul et leur grand nombre de paramètres. On souhaiterait bénéficier de la richesse d'expression et des facilités de calcul dont bénéficient les réseaux de neurones et des bonnes propriétés des kernels vis-à-vis des distributions de données. Pour ce faire on s'intéresse dans la suite à une méthode à noyaux non linéaire dans l'espace latent réalisée en associant à un noyau particulier, un *perceptron multi-couches* (MLP).

Dans ce document nous nous intéressons à la méthode proposée dans [10], nous fournissons une borne supérieure de l'erreur de généralisation utilisée pour le problème de régression posé dans [10], en vue d'améliorer le procédé de détermination des différents paramètres de ladite méthode. Nous proposons ensuite une méthode de sélection de représentants (faisant partie des paramètres) aptes à permettre la meilleure régression sous certaines contraintes.

Ce document est divisé en deux chapitres assez indépendants; le premier porte sur une extension du problème du calcul du *conditional kernel mean embedding* taité dans [10] en le reconsidérant comme un problème d'apprentissage non linéaire dans l'espace latent associé à un kernel, dans le second chapitre nous proposons une méthode générale pour déterminer des représentants dans l'espace initial, en vue de résoudre la question de la représentation issue du problème du chapitre 1. Une application importante dans le cas conditionnel sera notamment développée à cette occasion.

Le travail personnel réalisé est présenté dans les sections 1.2, 1.3, 1.4 et dans chapitre 2. Un lexique des notations suit la conclusion de ce mémoire.

# Chapitre 1

## Apprentissage non linéaire dans un RKHS

### 1.1 Contexte et motivation

On se donne deux variables aléatoires  $X$  et  $Y$  respectivement sur les domaines  $\mathcal{X}$  et  $\mathcal{Y}$  parties d'espaces vectoriels réels de dimension finie ; deux kernels  $k_x, k_y$  sur  $\mathcal{X}$  et  $\mathcal{Y}$  respectivement. Le couple  $(X, Y)$  suit une loi de densité  $\rho$ . L'objectif est de prédire la variable  $Y$  connaissant  $X$ . Pour ce faire, on dispose d'un échantillon fini  $(x_i, y_i)_{1 \leq i \leq N} \in (\mathcal{X} \times \mathcal{Y})^{\{1, \dots, N\}}$  de points-données.

Plus précisément, on va chercher à estimer le *conditional mean embedding* ou CME, c'est à dire l'espérance conditionnelle des plongements par le kernel de  $\mathcal{Y}$  des variables aléatoires  $(X, Y)$ , c'est à dire par l'application  $\Phi : y \in \mathcal{Y} \rightarrow k_y(y, \cdot) \in \mathcal{H}_y$ . Le CME s'écrit

$$\mu_{\rho(Y|X)}(x) = \mathbb{E}_{Y|X} [\Phi(Y)|X = x] \in \mathcal{H}_y \quad (1.1)$$

Que l'on notera par la suite aussi  $\mathbb{E}_Y [\Phi(Y)|X = x]$  dans un souci de légèreté. Le principal intérêt des CME et plus généralement de la moyenne des plongement par des kernels est la représentation parfois complète d'une distribution dans l'espace initial sous la forme d'un vecteur de l'espace latent. Par exemple, dans le cas des kernels gaussiens le *mean embedding* est exactement la fonction caractéristique de la distribution, avec toutes les propriétés liées.

Il existe une méthode classique présentée dans [6] pour approcher cette grandeur à partir de données indépendantes identiquement distribuées (*iid*), en exprimant le CME sous une forme fonctionnelle nécessitant une inversion de matrice. Plus précisément, en notant  $(x_i, y_i)_{1 \leq i \leq N}$  les données dans  $\mathcal{X} \times \mathcal{Y}$ , cette méthode approche le CME par :

$$\hat{\mu}_{\rho(Y|X)}(x) = \sum_{i=1}^N \beta_i(x) \Phi(y_i) = \Phi^T \beta(x) \quad (1.2)$$

où en notant  $K_X$  la matrice kernel, et  $k_x = [k_x(x_1, x) \dots k_x(x_N, x)]^T$ , et  $\lambda$  un hyperparamètre de régularisation,

$$\beta(x) = (K_X + \lambda I)^{-1} k_x \quad (1.3)$$

Cette approche classique d'estimation du CME, telle qu'utilisée dans [12] nécessite une inversion de matrice coûteuse. Pour éviter cela, l'idée présentée dans [10] et inspirée des travaux [14] consiste à utiliser un perceptron multicouches (MLP) couplé avec un kernel évalué en certains points de l'espace. Plus précisément cette dernière approche consiste à remplacer  $\beta$  dans eq. (1.2)

par une application représentée par un MLP  $f^\theta : \mathcal{X} \rightarrow \mathbb{R}^M$  paramétrisé par  $\theta \in \Omega$ . L'estimateur du CME ainsi obtenu est de la forme suivante, où  $(\eta_a)_{1 \leq a \leq M} \in \mathcal{Y}^M$  est une famille de représentants à déterminer.

$$\hat{\mu}_{P(Y|X)} = \sum_{a=1}^M \Phi_y(\eta_a) f_a^\theta \quad (1.4)$$

Le problème d'estimation du CME par la méthode proposée dans [10] point de départ du travail réalisé dans ce mémoire, peut être vue différemment comme un problème d'apprentissage non linéaire dans un RKHS. Ce point de vue plus générique offre plus de souplesse : les réseaux de neurones assurant une expressivité supérieure et notamment la non linéarité, et le kernel permettant de travailler directement dans l'espace latent (le RKHS).

Dans ce qui suit on définira tout d'abord le problème d'apprentissage non linéaire dans un RKHS dans le cadre de l'estimation du CME ([10]), puis on fournira une analyse théorique de la convergence de la méthode proposée par [10] dans le but d'améliorer le processus d'apprentissage. Enfin on conclura ce chapitre par une perspective des travaux futurs.

## 1.2 Apprentissage dans un RKHS

Le problème que l'on se pose (le même que dans [10] avec un kernel gaussien) est de déterminer une famille finie de représentants  $(\eta_i)_{1 \leq i \leq M}$  de l'espace initial  $\mathcal{Y}$ , et un perceptron multicouches  $f^\theta$  de sorte que la fonction  $\sum_{a=1}^M \Phi_y(\eta_a) f_a^\theta$  approche au mieux le modèle :

$$\text{Modèle global : } \begin{cases} \mathcal{X} \xrightarrow{MLP} \Delta(\mathcal{Y}) \xrightarrow{\Phi_y} \mathcal{H}_y \\ x \longrightarrow \mathbb{P}(Y|x) \longrightarrow \mu_{Y|x} \end{cases} \quad (1.5)$$

Ce qui correspond à résoudre le problème d'optimisation suivant

$$\min_{\theta, \eta_a} \mathbb{E}_X \left[ \left\| \mathbb{E}_Y [\Phi(Y)|X] - \sum_{a=1}^M \Phi(\eta_a) f_a^\theta(X) \right\|_y^2 \right] \quad (1.6)$$

qui se ramène en pratique à minimiser la moyenne empirique :

$$\text{Fonction objectif : } \min_{\theta, \eta_a} \frac{1}{n} \sum_{i=1}^n \left\| \Phi_y(y_i) - \sum_{a=1}^M \Phi_y(\eta_a) f_\theta^a(x_i) \right\|_{\mathcal{H}_y}^2 \quad (1.7)$$

$$\text{Soit : } \min_{\theta, \eta_a} \sum_{i=1}^n \left[ \sum_{a,b=1}^M k_y(\eta_a, \eta_b) f_\theta^a(x_i) f_\theta^b(x_i) - 2 \sum_{a=1}^M k_y(y_i, \eta_a) f_\theta^a(x_i) \right] \quad (1.8)$$

L'objectif de la section suivante est de déterminer une borne supérieure -dépendant des paramètres du problème- suffisamment fine de l'espérance de la vitesse de convergence du modèle vers le CME idéal, avec pour objectif d'en obtenir une méthode pour déterminer autant que possible les bons paramètres du modèle à apprendre.

## 1.3 Analyse théorique de la borne supérieure

Le problème eq. (1.6) se ramène notamment à représenter un vecteur de plongement  $\Phi(y) \in \mathcal{H}_y$ , comme combinaison finie de représentants  $(\Phi(\eta_i))_{1 \leq i \leq M}$ . Pour cela, on décompose tout plongement

$\Phi(y)$  pour  $y \in \mathcal{Y}$  comme somme de deux composantes orthogonales : celle effectivement représentée et l'autre qui est non accessible. On note dans la suite

- $\Phi(\eta)_M^T$  la matrice des représentants, dont les lignes sont les  $\Phi(\eta_i)$
- $P$  est le projecteur orthogonal en base canonique sur le sous-espace vectoriel  $\text{Vect}(\eta_1, \dots, \eta_M)$
- $v(y) = y - P\Phi(y)$  la composante orthogonale à  $\text{Vect}(\Phi(\eta_1), \dots, \Phi(\eta_M))$  de  $\Phi(y)$ .

**Contraintes :** On demande aussi à ce que les critères suivants soient vérifiés :

1. La série  $\sum \|\Phi(\eta_i)\|_y^2$  converge
2. Les représentants sont ordonnés comme suit :  $\|\Phi(\eta_1)\| \geq \|\Phi(\eta_2)\| \geq \dots$

### 1.3.1 Approche générale

L'objectif annoncé précédemment est d'établir une borne supérieure d'une quantité du type  $\mathbb{E}_X \left[ \|g_0(X) - g_\theta(X)\|_y^2 \right]$ . Pour cela, on va chercher à se ramener à des problèmes connus ou généraux dans la mesure du possible.

$$\mathbb{E}_X \left[ \|g_0(X) - g_\theta(X)\|_y^2 \right] = \mathbb{E}_X \left[ \left\| \mathbb{E}_Y [\Phi(Y)|X] - \Phi(\eta)_M^T f^\theta(X) \right\|_y^2 \right] \quad (1.9)$$

$$= \mathbb{E}_X \left[ \left\| \mathbb{E}_Y [\Phi(\eta)_M^T P\Phi_y(Y) - \Phi(\eta)_M^T f^\theta(X)|X] + \mathbb{E}_Y [v(Y)|X] \right\|_y^2 \right] \quad (1.10)$$

Par orthogonalité :

$$= \underbrace{\mathbb{E}_X \left[ \left\| \mathbb{E}_Y [\Phi(\eta)_M^T P\Phi_y(Y) - \Phi(\eta)_M^T f^\theta(X)|X] \right\|_y^2 \right]}_{(1)} + \underbrace{\mathbb{E}_X \left[ \left\| \mathbb{E}_Y [v(Y)|X] \right\|_y^2 \right]}_{(2)} \quad (1.11)$$

### 1.3.2 Premier membre : partie effectivement apprise

$$(1) = \mathbb{E}_X \left[ \left\| \sum_{a=1}^M \Phi(\eta_a) \mathbb{E}_Y [(P\Phi(Y))_a - f_a^\theta(X)] \right\|_y^2 \right] \quad (1.12)$$

$$\begin{aligned} &\leq \mathbb{E}_X \left[ \sum_{a=1}^M \|\Phi(\eta_a)\|_y^2 \left( \mathbb{E}_Y [(P\Phi(Y))_a - f_a^\theta(X)]^2 \right) \right] \\ &\quad + \mathbb{E}_X \left[ \sum_{a \neq b} k(\eta_a, \eta_b) \underbrace{\mathbb{E}_Y [(P\Phi(Y))_a - f_a^\theta(X)] \mathbb{E}_Y [(P\Phi(Y))_b - f_b^\theta(X)]}_{\leq \frac{1}{2} (\mathbb{E}_Y [(P\Phi(Y))_a - f_a^\theta(X)]^2 + \mathbb{E}_Y [(P\Phi(Y))_b - f_b^\theta(X)]^2)} \right] \quad (1.13) \end{aligned}$$

$$\begin{aligned} &\leq \|\Phi(\eta_1)\|_y^2 \mathbb{E}_X \left[ \left\| \mathbb{E}_Y [P\Phi(Y)|X] - f^\theta(X) \right\|_{\text{eucl}}^2 \right] \\ &\quad + \mathbb{E}_X \left[ \sum_{a=2}^M \sum_{b < a} k(\eta_a, \eta_b) \mathbb{E}_Y [(P\Phi(Y))_b - f_b^\theta(X)]^2 \right] \quad (1.14) \end{aligned}$$

Notons  $\delta_{M,a} = \max_{b \neq a, 1 \leq b \leq M} k(\eta_a, \eta_b)$ , la plus grande dissimilarité d'un vecteur parmi les  $M$  premiers vecteurs représentants de  $\eta$  avec un vecteur  $\eta_a$  donné. Il vient :

$$\begin{aligned} \leq \|\Phi(\eta_1)\|_y^2 \mathbb{E}_X \left[ \|\mathbb{E}_Y [P\Phi(Y)|X] - f^\theta(X)\|_{\text{eucl}}^2 \right] \\ + \left( \sum_{a=2}^M \delta_{M,a} \right) \mathbb{E}_X \left[ \|\mathbb{E}_Y [P\Phi(Y)|X] - f^\theta(X)\|_{\text{eucl}}^2 \right] \end{aligned} \quad (1.15)$$

$$= \left[ \|\Phi(\eta_1)\|_y^2 + \sum_{a=2}^M \delta_{M,a} \right] \mathbb{E}_X \left[ \|\mathbb{E}_Y [P\Phi(Y)|X] - f^\theta(X)\|_{\text{eucl}}^2 \right] \quad (1.16)$$

La majoration de la quantité  $\mathbb{E}_X \left[ \|\mathbb{E}_Y [v(Y)|X]\|_y^2 \right]$  étant un problème général pour les réseaux de neurones, et non approfondie dans la suite. On se contente de fournir quelques sources traitant de ce problème dans des cas particuliers, sous des hypothèses de structure [5], ou de classe de fonction à approcher [13]. On note une borne supérieure de cette quantité  $\mathcal{C}(\theta)$  pour un réseau de neurones paramétré par  $\theta$ .

### 1.3.3 Second membre : partie orthogonale

$$(2) = \mathbb{E}_X \left[ \|\mathbb{E}_Y [v(Y)|X]\|_y^2 \right] = \mathbb{E}_X \left[ \|\mathbb{E}_Y [\Phi(Y) - \Phi(\eta)_M^T P\Phi(Y)|X]\|_y^2 \right] \quad (1.17)$$

$$= \mathbb{E}_X \left[ \left\| \mathbb{E}_Y \left[ \underbrace{(\text{Id}_{\mathcal{H}_y} - \Phi(\eta)_M^T P)}_{=F} \Phi(Y) | X \right] \right\|_y^2 \right] = \mathbb{E}_X \left[ \|F \mathbb{E}_Y [\Phi(Y)|X]\|_y^2 \right] \quad (1.18)$$

$$\leq \mathbb{E}_X \left[ \mathbb{E}_Y \left[ \|F\Phi(Y)\|_y^2 | X \right] \right] = \mathbb{E}_Y \left[ \|F\Phi(Y)\|_y^2 \right] \quad \text{Inégalité de Jensen} \quad (1.19)$$

On va étudier plus en détail cette partie dans le chapitre suivant, et proposer un algorithme pour déterminer les représentants permettant de minimiser cette quantité finie. Cette méthode offrant lors du choix des représentants, une borne sur cette quantité notée  $R_M$ .

### 1.3.4 Bilan

En revenant au problème initial, et en regroupant les deux termes, on obtient la borne supérieure désirée. Pour tout  $n \in \mathbb{N}$  il existe un paramètre  $\theta_n \in \Omega$  à partir de  $n$  points données  $(x_i, y_i)_{1 \leq i \leq n}$  identiquement distribués, tel que

$$\mathbb{E}_X \left[ \|g_0(X) - g_{\theta_n}(X)\|_y^2 \right] \leq \left[ \|\Phi(\eta_1)\|_y^2 + \sum_{a=2}^M \delta_{M,a} \right] \mathcal{C}(\theta_n) + R_M C \quad (1.20)$$

## 1.4 Perspective et travaux futurs

Une première amélioration possible du travail réalisé dans ce premier chapitre est de considérer une meilleure estimation de la matrice de variance-covariance obtenue dans le travail réalisé sur le membre (2) de eq. (1.11). On peut directement chercher à évaluer empiriquement une quantité de la forme

$$\mathbb{E}_X \left[ \mathbb{E}_{Y|X} [\Phi(Y)|X] \mathbb{E}_{Y|X} [\Phi(Y)|X]^T \right] \quad (1.21)$$

On peut toutefois faire mieux : Dans ce premier chapitre, on s'est focalisé sur un choix fixe de représentants quel que soit le  $x \in \mathcal{X}$  donné, ceci découlant de la majoration réalisée par l'inégalité de Jensen dans le membre (2) de eq. (1.11). Un raffinement possible est de déterminer un ensemble fini de  $M$  représentants dépendant de  $x \in \mathcal{X}$ . Pour obtenir une telle détermination conditionnelle des représentants, une approche possible est l'approximation conditionnelle de la matrice de variance-covariance. C'est à dire estimer la matrice de variance-covariance de  $\mathbb{E}_{Y|X} [\Phi(Y)|X = x]$  pour chaque  $x \in \mathcal{X}$  donné, puis de déterminer des représentants pour chacune de ces matrices.

La première difficulté est l'estimation conditionnelle de ces matrices de variance-covariance. En l'absence d'hypothèses supplémentaires sur la distribution de  $\rho$  de  $(X, Y)$ , il existe différentes méthodes non paramétriques que l'on peut trouver dans [3], l'une d'elles est l'estimation empirique de la matrice de variance-covariance avec des points-données situés dans un voisinage du point  $x$  d'intérêt. C'est ce que l'on va notamment développer dans le chapitre 2.

Une autre difficulté apparaît : la nécessité de bénéficier de nombreux points dans le voisinage du point  $x$  pour obtenir une estimation empirique suffisamment bonne de la matrice de variance-covariance et donc des représentants pertinents. On va donc dans cette perspective proposer une méthode de détermination des représentants souffrant le moins possible du manque de points-données.

On distingue deux approches futures qui seraient intéressantes d'étudier :

- L'approche décrite ci-dessus : étant donné un  $x \in \mathcal{X}$ , on va utiliser les données associées au voisinage de ce point pour déterminer une famille de représentants. Une telle approche est une approche par morceaux sur les cellules d'un certain diagramme de Voronoï. C'est cette approche qui est développée dans le cas d'application au chapitre 2.5. Le choix des représentants pouvant se faire avec une méthode ne dépendant que de la matrice de variance-covariance ainsi estimée.
- Détermination directe des représentants par un réseau de neurones : On peut chercher à déterminer directement sachant un  $x \in \mathcal{X}$  donné, une famille de  $M$  points de  $\mathcal{Y}$  tels que l'espace engendré par leurs plongements via  $\Phi$  approche au mieux  $\mathbb{E}_Y [\Phi(Y)|X = x]$ . Le problème d'optimisation associé pouvant s'écrire comme :

$$\arg \min_{\theta, \varphi} \frac{1}{N} \sum_{i=1}^N \|y_i - \Phi(g_M^\varphi(x_i))f_M^\theta(x_i)\|_y^2 \quad (1.22)$$

où  $\Phi$  est la fonction de plongement associée au kernel  $k$  ;  $g_M : \mathcal{X} \rightarrow \mathcal{Y}^M$  et  $f_M : \mathcal{X} \rightarrow \mathbb{R}^M$  sont des fonctions représentées par des MLP paramétrées par  $\varphi$  et  $\theta$  respectivement.

On propose dans le chapitre suivant une méthode de sélection de représentants, correspondant à représenter les données dans un espace de dimension réduite, tout en cherchant à avoir de bonnes propriétés d'estimation même lorsque le nombre de points-données est faible.

## Chapitre 2

# Réduction de dimension par représentateurs : PkPCA

Pour représenter en dimension réduite les plongements d'une variable aléatoire par un kernel, il existe différentes méthodes, dont on distingue en particulier deux types : l'apprentissage de dictionnaire et l'ACP.

L'apprentissage de dictionnaire dans le cas kernel comme présenté dans [15] section 3.1.2. consiste à apprendre un ensemble d'atomes (que l'on nomme dictionnaire) qui sont des combinaisons linéaires des plongements de points-données. Il y a une contrainte de parcimonie qui spécifie (d'autres métriques de contraintes étant employables) le nombre maximal d'atomes utilisables pour représenter les plongements des données. Le problème de cette méthode est que l'on ne sait pas a priori quels seront les atomes nécessaires pour représenter le plongement d'un point-donnée, et nous voulons que le nombre de ces représentants soit le plus faible possible. On cherche donc à fixer un certain nombre  $M$  de représentants qui seront tous utilisés pour décrire des plongements de points-données.

L'ACP, ou plutôt son adaptation au cas des kernel : la kernel ACP [7] (kPCA) permet de déterminer  $M$  vecteurs orthonormaux de représentation dans l'espace latent, correspondant aux directions successives de plus grande variance des données. Ces vecteurs sont des combinaisons linéaires de plongements de points-données.

Cependant, ces méthodes ne fournissent pas directement un ensemble réduit de points dans l'espace initial dont les plongements permettent une bonne représentation des données. Pour obtenir un nombre réduit de ces points, une méthode possible est de combiner une méthode de recherche de pré-image par exemple [4] qui va fournir pour chaque vecteur de l'espace latent -notamment les vecteurs de la base d'ACP- une pré-image dont le plongement en est proche. Toutefois, ceci ne répond pas directement au problème posé et introduit des approximations évitables :

- la pré-image trouvée ne suffit pas nécessairement à bien représenter toute la variance dans la direction d'un vecteur de base d'ACP, il faudra éventuellement en choisir plusieurs.
- Le côté orthonormal de la base d'ACP n'est pas nécessaire, il est suffisant d'avoir un ensemble de plongements de points de l'espace initial dont l'espace vectoriel engendré capture au mieux la variance des plongements des données.
- Par ailleurs, les points dans l'espace initial que l'on utilise sont des points données, on peut éventuellement s'en affranchir pour gagner en performance et en généralisabilité.

Dans la suite, et c'est la méthode présentée, on va chercher à trouver des points directement dans l'espace initial dont l'espace vectoriel engendré par leurs plongements capture au mieux la variance des plongements des données, ceci sous la contrainte du nombre de points utilisables :  $M$ , que l'on nommera représentants.

On a résumé et positionné les méthodes proposées en Tableau 2.1. On les y distingue selon deux critères : selon que on utilise comme vecteurs de représentation dans l'espace latent (RKHS) des combinaisons linéaires de plongements de points l'espace initial (de forme  $\sum_i \alpha_i \Phi(\eta_i)$ ) ou directement des plongements de points de l'espace initial ( $\Phi(\eta_j)$ ) ; et selon que l'ensemble de points utilisés dans l'espace initial est limité aux points-données fournis ou bien laissé à l'espace entier.

| Forme du vecteur de représentation | Données  | Espace entier |
|------------------------------------|----------|---------------|
| $\sum_i \alpha_i \Phi(\eta_i)$     | kPCA     | $\times$      |
| $\Phi(\eta_j)$                     | DATAkPCA | PkPCA         |

TABLE 2.1 – Méthodes proposées et existantes selon la nature du vecteur de représentation dans l'espace latent, dépendant de l'ensemble de points utilisé dans l'espace initial et des opérations algébriques réalisées sur leur plongement

## 2.1 Contexte

On se donne une variable aléatoire  $Y$  à valeurs dans  $\mathcal{Y} \subset \mathbb{R}^d$  ; un kernel  $k$  sur  $\mathbb{R}^d$  dont nous notons  $\mathcal{H}_y$  l'espace latent associé et par  $\Phi : y \mapsto k(y, \cdot)$  la fonction de plongement dans ce même espace.

On fixe les contraintes supplémentaires suivantes :

**Famille admissible de kernels** le kernel  $k : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$  doit être de la forme

$$k(y, y') = g(\|y - y'\|) \quad \text{où} \quad g(x) \xrightarrow{x \rightarrow \infty} 0 \quad (2.1)$$

En pratique, les kernels couramment utilisés sont de ce type ; entre autres les kernels gaussiens, triangle, de Cauchy, de Laplace, d'Epanechnikov, tricube, uniforme, de spline cubique, etc...

**Espace latent** L'espace vectoriel engendré par les applications  $\Phi(y)$  est un espace de Hilbert noté  $\mathcal{H}_y$  lorsque muni du produit scalaire  $\langle \Phi(y) | \Phi(y') \rangle = k(y, y')$ . Dans un souci de confort, on notera aussi ce produit scalaire lorsque l'on représente les vecteurs dans une base orthonormée par  $\langle u | v \rangle = u^T v$ .

**Forme de la distribution** On demande que la loi de  $Y$  soit à densité continue sur  $\mathcal{Y}$ . En pratique les données seront constituées d'un échantillon de  $N \in \mathbb{N}$  couples de points suivant cette loi que l'on notera  $(y_i)_{1 \leq i \leq N}$  qui consistent en  $N$  variables aléatoires indépendantes identiquement distribuées *iid*.

On cherche à trouver une famille de représentants  $\eta_1, \dots, \eta_M \in \mathcal{Y}$  offrant la meilleure représentation de  $Y$ . Plus formellement on cherche les paramètres maximisants de :

$$\max_{\eta_1, \dots, \eta_M \in \mathcal{Y}} \mathbb{E}_Y \left[ \left\| F_{\text{Vect}(\Phi(\eta_1), \dots, \Phi(\eta_M))} \Phi(Y) \right\|_y^2 \right] \quad (2.2)$$

où  $F_{\text{Vect}(\Phi(\eta_1), \dots, \Phi(\eta_M))}$  est la projection orthogonale sur  $\text{Vect}(\Phi(\eta_1), \dots, \Phi(\eta_M))$ , que l'on notera  $F$  dans la suite par souci de concision.

## 2.2 Problème et analyse

On cherche à maximiser la quantité eq. (2.2), avec  $F$  défini comme précédemment. Ce problème est bien défini, puisque cette quantité est finie. Plus précisément

$$\mathbb{E}_Y \left[ \|F\Phi(Y)\|_y^2 \right] \leq \mathbb{E}_Y \left[ \|\Phi(Y)\|_y^2 \right] = \mathbb{E}_Y [k(Y, Y)] = g(0) \in \mathbb{R}_+ \quad (2.3)$$

On note que  $g(0)$  est aussi la trace de la matrice kernel des données :  $[k(y_i, y_j)]_{ij}$ .

**Orthogonalisation :** Étant donné une famille  $\eta_1, \dots, \eta_M \in \mathcal{Y}$ , on définit pour  $1 \leq j \leq M$  la fonction  $j$ -ème orthogonalisée c'est à dire la fonction qui à  $\eta \in \mathcal{Y}$  donne l'orthogonalisée par le processus de Gram-Schmidt de  $\Phi(\eta)$  à partir des vecteurs  $\Phi(\eta_1), \dots, \Phi(\eta_{j-1})$  (méthode détaillée [2] page 124) :

$$\Psi_j(\eta) = \Phi(\eta) - \sum_{k < j} \frac{\langle \Phi(\eta) | \Psi_k(\eta_k) \rangle}{\|\Psi_k(\eta_k)\|_y^2} \Psi_k(\eta_k) \quad (2.4)$$

Cela fournit l'existence de coefficients  $(a_{k,j})_{k < j}$  pour  $1 \leq j \leq M$  tels que  $\Psi_j(\eta)$  soit représentée comme combinaison linéaire de  $\Phi(\eta_1), \dots, \Phi(\eta_{j-1}), \Phi(\eta)$  :

$$\Psi_j(\eta) = \Phi(\eta) + \sum_{k < j} \langle \Phi(\eta) | \Phi(\eta_k) \rangle a_{k,j} \Phi(\eta_k) = \Phi(\eta) + \sum_{k < j} k(\eta, \eta_k) a_{k,j} \Phi(\eta_k) \quad (2.5)$$

$$\Psi_j(\eta) = \Phi(\eta) + \sum_{k < j} \tilde{a}_{k,j} \Phi(\eta_k) \quad (2.6)$$

Par ailleurs, la famille  $(\Psi_j(\eta_j))_{1 \leq j \leq M}$  est l'orthogonalisée de Gram-schmidt de  $(\Phi(\eta_j))_{1 \leq j \leq M}$  dans  $\mathcal{H}_y$ , ce qui permet de réécrire  $F$  de la manière suivante :

$$F = \begin{pmatrix} \frac{\Psi_1(\eta_1)^T}{\|\Psi_1(\eta_1)\|_y} \\ \vdots \\ \frac{\Psi_M(\eta_M)^T}{\|\Psi_M(\eta_M)\|_y} \end{pmatrix} \quad (2.7)$$

**Suite :** Reprenons la quantité qui nous intéresse, par propriété de la trace, de l'espérance et de leur linéarité, il vient en notant  $C$  la matrice de variance-covariance de  $\Phi(Y)$  :

$$\mathbb{E}_Y \left[ \|F\Phi(Y)\|_y^2 \right] = \mathbb{E}_Y \left[ \Phi(Y)^T F^T F \Phi(Y) \right] = \text{Tr} \left( \mathbb{E}_Y \left[ F \Phi(Y) \Phi(Y)^T F^T \right] \right) \quad (2.8)$$

$$= \text{Tr} \left( F \underbrace{\mathbb{E}_Y \left[ \Phi(Y) \Phi(Y)^T \right]}_C F^T \right) = \text{Tr} \left( \begin{pmatrix} \frac{\Psi_1(\eta_1)^T}{\|\Psi_1(\eta_1)\|_y} \\ \vdots \\ \frac{\Psi_M(\eta_M)^T}{\|\Psi_M(\eta_M)\|_y} \end{pmatrix} C \begin{pmatrix} \frac{\Psi_1(\eta_1)}{\|\Psi_1(\eta_1)\|_y} & \dots & \frac{\Psi_M(\eta_M)}{\|\Psi_M(\eta_M)\|_y} \end{pmatrix} \right) \quad (2.9)$$

$$= \sum_{j=1}^M \frac{\Psi_j(\eta_j)^T}{\|\Psi_j(\eta_j)\|_y} C \frac{\Psi_j(\eta_j)}{\|\Psi_j(\eta_j)\|_y} = \sum_{j=1}^M \mathbb{E}_Y \left[ \left( \frac{\Psi_j(\eta_j)^T \Phi(Y)}{\|\Psi_j(\eta_j)\|_y} \right)^2 \right] \quad (2.10)$$

**Non équivalece entre maximisation successive et simultanée :** On se retrouve face au problème de maximisation suivant, en rappelant que  $\Psi_j$  dépend du choix des  $\eta_k$  pour  $k < j$  :

$$\max_{\eta_1, \dots, \eta_M} \sum_{j=1}^M \Psi_j(\eta_j)^T C \Psi_j(\eta_j) \quad (2.11)$$

Pour des raisons computationnelles, on ne va pas chercher à maximiser cette quantité, mais à réaliser une maximisation successive :

$$\max_{\eta_M} \left[ \dots \max_{\eta_2} \left[ \max_{\eta_1} \Psi_1(\eta_1)^T C \Psi_1(\eta_1) \right] + \Psi_2(\eta_2)^T C \Psi_2(\eta_2) \right] \quad (2.12)$$

Dont la solution constitue un maximum local de eq. (2.11), mais non global le plus souvent.

Ce point est une différence notable avec le cas des méthodes linéaires que sont l'ACP (ou la kPCA puisque on se ramène à travailler vectoriellement dans un sous-espace de dimension finie de  $\mathcal{H}_y$ ) ou les décompositions en valeurs singulières. Dans ces cas là, puisque l'on travaille dans un sous espace vectoriel, et que l'on cherche à obtenir des directions propres successives, le théorème de séparation de Poincaré ou le théorème d'entrelacement de Cauchy affirment l'équivalence entre la maximisation simultanée et successive. L'invalidité de ceci dans le cas qui nous intéresse est lié à la structure de l'espace des images des points de  $\mathcal{Y}$ , qui est restreint et ne garantit pas nécessairement l'existence de vecteurs propres (base d'ACP) en son sein. Toutefois, la maximisation successive étant calculatoirement plus légère que la maximisation simultanée, et fournissant un maximum local de la fonction de maximisation simultanée, on va considérer seulement le problème eq. (2.12) comme étant une bonne approximation du problème eq. (2.11).

En appliquant ceci, on ramène à la maximisation successive selon la  $i$ -ème orthogonalisée. Cela se réécrit avec  $\Psi_i$  formé par les  $\eta$  maximisants selon les  $i$  croissants :

$$\max_{\eta_1, \dots, \eta_M} \sum_{j=1}^M \mathbb{E}_Y \left[ \left( \frac{\Psi_j(\eta_j)^T \Phi(Y)}{\|\Psi_j(\eta_j)\|_y} \right)^2 \right] \approx \sum_{j=1}^M \max_{\eta \in \mathcal{Y}} \mathbb{E}_Y \left[ \left( \frac{\Psi_j(\eta)^T \Phi(Y)}{\|\Psi_j(\eta)\|_y} \right)^2 \right] \quad (2.13)$$

**Notion bien définie :** Nous voulons montrer ici que à toute étape  $j$ , le maximum de

$$\mathbb{E}_Y \left[ \left( \frac{\Psi_j(\eta)^T \Phi(Y)}{\|\Psi_j(\eta)\|_y} \right)^2 \right] \quad (2.14)$$

existe et est bien atteint sur  $\mathcal{Y}$ . Pour cela il suffit de montrer que eq. (2.14) tend vers 0 lorsque  $\|\eta\| \rightarrow \infty$  ; le caractère borné découlant de eq. (2.4). Il découle de eq. (2.5), puis de la contrainte eq. (2.1) sur  $g$ , que pour tout  $y \in \mathcal{Y}$

$$\Psi_j(\eta)^T \Phi(y) = k(\eta, y) + \sum_{k < j} k(\eta, \eta_k) a_{k,j} k(\eta_k, y) \xrightarrow{\|\eta\| \rightarrow \infty} 0 \quad (2.15)$$

On obtient de manière analogue  $\|\Phi(\eta)\| \geq \|\Psi_j(\eta)\| \xrightarrow{\|\eta\| \rightarrow \infty} g(0)$ . Par conséquent,

$$\mathbb{E}_Y \left[ \left( \frac{\Psi_j(\eta)^T \Phi(Y)}{\|\Psi_j(\eta)\|_y} \right)^2 \right] = \frac{\mathbb{E}_Y \left[ (\Psi_j(\eta)^T \Phi(Y))^2 \right]}{\|\Psi_j(\eta)\|_y^2} \quad (2.16)$$

On se fixe  $\varepsilon > 0$ , il existe un  $\alpha > 0$  tel que  $\mathbb{P}(\|Y\| \geq \alpha) < \varepsilon$ . Il vient en séparant en deux le domaine d'intégration

$$\mathbb{E}_{\|Y\| \geq \alpha} \left[ (\Psi_j(\eta)^T \Phi(Y))^2 \right] \underset{\text{Cauchy-Schwarz}}{\leq} \mathbb{E}_{\|Y\| \geq \alpha} \left[ \|\Psi_j(\eta)\|^2 \|\Phi(Y)\|^2 \right] \leq \varepsilon \|\Phi(\eta)\|_y^2 g(0)^2 \quad (2.17)$$

Comme la boule  $B(0, \alpha)$  est un compact de  $\mathbb{R}^d$ , la fonction continue  $y \mapsto \Psi_j(\eta)^T \Phi(y)$  atteint son maximum sur  $B(0, \alpha)$  en un point  $y^*$ .

$$\mathbb{E}_{\|Y\| < \alpha} \left[ (\Psi_j(\eta)^T \Phi(Y))^2 \right] \leq (\Psi_j(\eta)^T \Phi(y^*))^2 \xrightarrow{\|\eta\| \rightarrow \infty} 0 \quad (2.18)$$

En regroupant eq. (2.17) et eq. (2.18), et en réinjectant dans eq. (2.16) on obtient finalement

$$\mathbb{E}_Y \left[ \left( \frac{\Psi_j(\eta)^T \Phi(Y)}{\|\Psi_j(\eta)\|_y} \right)^2 \right] \xrightarrow{\|\eta\| \rightarrow \infty} 0 \quad (2.19)$$

Ce qui assure l'existence d'un  $\eta$  maximisant pour tout  $1 \leq j \leq M$ , de plus, le résultat est aussi valable lorsque on réalise une estimation empirique des espérances à partir des données, ce qui clôt ce développement et amène celui de l'estimation à partir des données.

## 2.3 Algorithme

### 2.3.1 Procédé général

On s'est donc ramené à partir de eq. (2.13) à une maximisation successive de fonctions selon la variable  $\eta \in \mathcal{Y}$ ; l'intérêt de ceci est double, en plus de réduire la dimension de l'espace des paramètres devant être optimisés simultanément, la méthode fournit un algorithme en-ligne, ce qui fournit des représentants par degré d'importance décroissant à chaque nouvelle maximisation réalisée. À l'étape  $j$ , on cherche donc

$$\eta_j \in \arg \max_{\eta \in \mathcal{Y}} \frac{\Psi_j(\eta)^T C \Psi_j(\eta)}{\|\Psi_j(\eta)\|_y^2} = \arg \max_{\eta \in \mathcal{Y}} \frac{\mathbb{E}_Y \left[ (\Psi_j(\eta)^T \Phi(Y))^2 \right]}{\|\Psi_j(\eta)\|_y^2} \quad (2.20)$$

**Estimation à partir des données** En pratique comme on dispose d'échantillons  $(y_i)_{1 \leq i \leq N}$ , on ne dispose que d'une estimation empirique de l'espérance. Pour un  $\eta \in \mathcal{Y}$  et  $k$  un entier fixés, il existe des coefficients  $(\tilde{a}_{i,j})_{i < j}$  fournis par l'orthogonalisation de Gram-Schmidt eq. (2.6). Le problème eq. (2.20) devient donc

$$\arg \max_{\eta \in \mathcal{Y}} \frac{1}{\|\Psi_j(\eta)\|_y^2} \sum_{i=1}^N (\Psi_j(\eta)^T \Phi(y_i))^2 \quad (2.21)$$

Que l'on peut réécrire uniquement en fonction du kernel, des données et du paramètre :

$$= \arg \max_{\eta \in \mathcal{Y}} \frac{\sum_{i=1}^N \left[ k(\eta, y_i) + \sum_{k < j} \tilde{a}_{k,j} k(\eta_k, y_i) \right]^2}{k(\eta, \eta) + 2 \sum_{k < j} \tilde{a}_{k,j} k(\eta, \eta_k) + \sum_{l < j} \sum_{k < j} \tilde{a}_{k,j} \tilde{a}_{l,j} k(\eta_k, \eta_l)} \quad (2.22)$$

---

**Algorithm 1: PkPCA**

---

**Data:** points  $(y_i)_{1 \leq i \leq N}$ , kernel  $k$

**Result:** représentants  $(\eta_i)_{1 \leq i \leq M}$

$j = 1$

**while**  $j \leq M$  **do**

$$\left| \begin{array}{l} \Psi_j : \eta \mapsto \Phi(\eta) - \sum_{k < j} \frac{\langle \Phi(\eta) | \Psi_k(\eta_k) \rangle}{\|\Psi_k(\eta_k)\|_y^2} \Psi_k(\eta_k) \\ \eta_j = \arg \max_{\eta \in \mathcal{Y}} \frac{1}{\|\Psi_j(\eta)\|_y^2} \sum_{i=1}^N (\Psi_j(\eta)^T \Phi(y_i))^2 \end{array} \right|$$

**return**  $(\eta_i)_{1 \leq i \leq M}$

---

Différentes approches peuvent-être utilisées pour l'implémentation effective de cet algorithme. On discute les deux cas et on précise ce qui a été réalisé pour l'implémentation.

**Représentation des  $\Psi_j$**  Une bonne approche est de conserver la représentation des  $\Psi_j$  dans la base des  $\Phi(\eta_1), \dots, \Phi(\eta_j)$  (mémoïsation), ce qui évite d'avoir à les recalculer et qui simplifie l'évaluation lors de la recherche de  $\eta_j$ .

**Obtention de l'argmax** Il s'agit d'une recherche de maximum d'une fonction possédant potentiellement plusieurs maxima locaux. Une première idée est de réaliser un recuit-simulé. Une seconde approche est un mélange de recherche aléatoire et de montée de gradient. Enfin, celle qui a été utilisée dans la suite, est de réaliser une montée de gradient issue du point donnée donnant à chaque étape la meilleure contribution ajoutée.

### 2.3.2 Variantes

On peut s'intéresser à des variantes de la méthode PkPCA, les idées pouvant être combinées :

**Travail sur les points données** On limite l'optimisation successive aux points données, la montée de gradient étant réalisée une fois  $M$  représentants choisis parmi les points données. L'intérêt est de réduire le nombre de calculs à réaliser, ceci marchant d'autant mieux que le ratio  $\frac{D}{N}$  est petit, puisque favorisant l'existence d'un point donnée proche de chaque représentant idéal.

**Optimisation simultanée** En revenant sur la simplification effectuée entre eq. (2.11) et eq. (2.12), on peut déterminer les  $M$  représentants simultanément. On résoud donc le problème initial, toutefois le nombre  $M$  de représentants doit être choisi à l'avance (hyperparamètre) et le problème est en dimension  $M \times D$  ce qui complique la recherche du maximum global étant donné la multitude de minima locaux.

**Optimisation combinatoire** Si l'on se fixe un nombre fini  $H$  de points admissibles pour être représentants, une recherche exhaustive des combinaisons de  $M$  de ces points est envisageable, pour peu que la quantité  $C_H^M$  soit raisonnable. Nonobstant, cette méthode est parallélisable.

**Validation croisée** En séparant les données en plusieurs groupes, et en conservant les représentants ayant obtenu les meilleures performances lors d'un test croisé avec les autres groupes, on peut espérer bénéficier de représentants non biaisés par les données.

L'intérêt de la méthode PkPCA étant lorsque les points représentants idéaux sont situés dans des zones de faible densité de points données, ce qui est d'autant plus probable lorsque le ratio  $\frac{D}{N}$  est grand. Une application de ceci est pour l'estimation de représentants pour des variables aléatoires conditionnelles.

### 2.3.3 Intérêt de la méthode

La méthode PkPCA offre des performance comparativement d'autant meilleures lorsque le ratio  $\frac{D}{N}$  est grand, ou dit autrement lorsque la probabilité qu'un point-donnée soit proche d'un véritable représentant de la distribution des données est faible.

## 2.4 Expérimentations

On a utilisé pour les expérimentations quatre jeux de données : Iris composé de 150 éléments, et trois jeux synthétiques : Cones, bidimensionnel de taille 100, Sinus bidimensionnel de taille 200 et Trimodal de dimension 30 et de taille 100. Ces trois ensembles de données sont représentés bidimensionnellement via T-SNE en FIGURE 2.1. Dans tous ces cas, on utilise un kernel gaussien, et on fixe le paramètre de ce kernel comme étant l'écart-type des données.

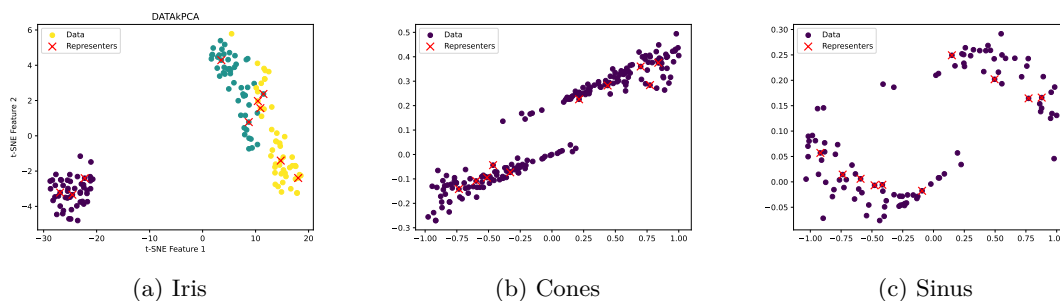


FIGURE 2.1 – Différents représentants obtenus avec DATAkPCA pour différentes données

On va évaluer les performances des méthodes PkPCA et DATAkPCA sur les jeux de données choisis. En particulier, on utilise la trace de la matrice de variance-covariance des données dans la base des vecteurs de représentation choisis comme valeur de comparaison de qualité des méthodes, cela correspondant à la quantité de variance des données capturée par les représentants.

Les méthodes proposées sont au nombre de deux : PkPCA et DATAkPCA. DATAkPCA est l'implémentation de la variante ne travaillant que sur les points-données. On compare ces deux méthodes avec d'autres méthode de sélection des représentants : le *kernel Herding* présenté dans [1] qui approche une fonction objectif analogue mais différente, la méthode **Random** qui choisit  $M$  points uniformément dans l'enveloppe convexe des données, et **Random data** qui choisit aléatoirement et uniformément successivement  $M$  points parmi les données, en sélectionnant ceux qui maximisent la trace représentée (Il s'agit d'une version très grossière de DATAkPCA). Enfin, on compare cette méthode à la quantité de trace représentée par l'ACP kernel, quantité qui est toujours supérieure dans le cas de méthodes limitées aux points données puisque utilisant des combinaisons linéaires de plongements comme vecteurs représentants dans l'espace latent et non plus seulement les plongements seuls.

En FIGURE 2.2 on observe la quantité de variance représentée lors de l'entraînement pour différentes méthodes, au fur et à mesure que l'on augmente le nombre de représentants utilisés (jusqu'à 40 représentants). La FIGURE 2.3 étant analogue pour les données de test.

Les exécutables ayant servis à la génération des figures sont écrits en python et sont disponibles à l'adresse <https://github.com/AbelDouzal/Japon-2024/tree/main/Code>.

La méthode PkPCA offre les meilleures performances de représentation, suivi de proche par DATAkPCA, tant sur les séquences d'entraînement que de test, comme visible en FIGURE 2.2 et FIGURE 2.3. La méthode Random data offre des performances proches puisque optimisant partiellement la même fonction objectif que PkPCA et DATAkPCA. On observe par ailleurs que les résultats obtenus sont d'autant favorables que la dimension de l'espace de données relativement au nombre de points-données utilisés est grand, c'est le cas des données sinus comparativement à Iris. La TABLE 2.2 révèle aussi que les méthodes PkPCA et DATAkPCA offrent les meilleures performances pour tous les ensembles de données utilisés, avec des taux de représentation proches de celles de l'ACP kernel (kPCA).

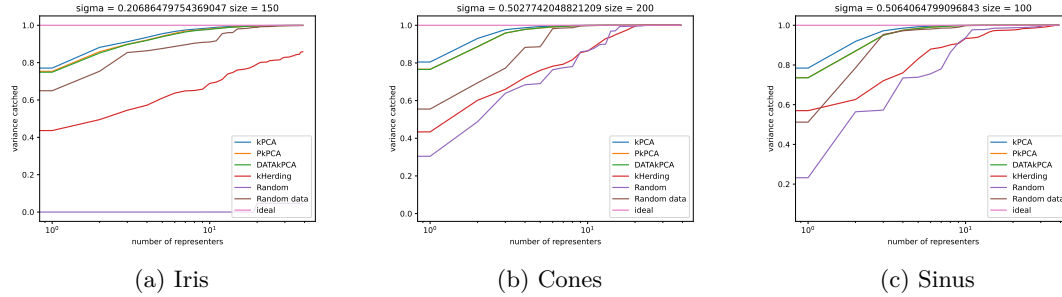


FIGURE 2.2 – Variance représentée à l'entraînement pour différentes méthodes

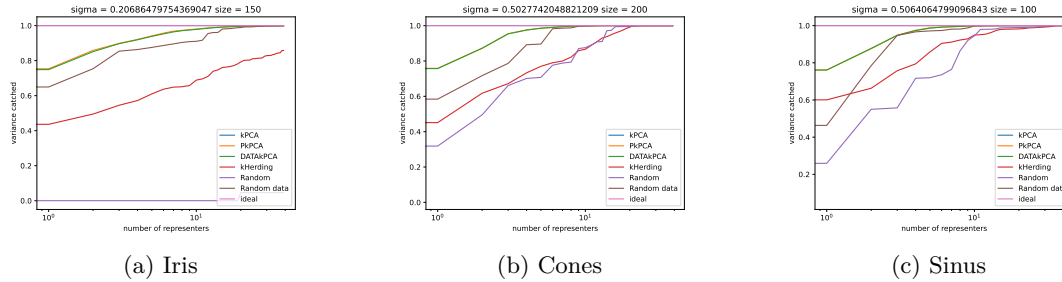


FIGURE 2.3 – Variance représentée sur les données de test pour différentes méthodes

| MODÈLE       | IRIS          |               | CONES         |               | SINUS         |               | TRIMODAL      |               |
|--------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| SÉRIE        | Train         | Test          | Train         | Test          | Train         | Test          | Train         | Test          |
| kPCA (comp.) | 0.9824        | 0.9824        | 0.9992        | 0.9992        | 0.9992        | 0.9992        | 0.8153        | 0.7770        |
| PkPCA        | <b>0.9770</b> | <b>0.9770</b> | <b>0.9985</b> | <b>0.9984</b> | 0.9978        | <b>0.9981</b> | <b>0.7905</b> | <b>0.7360</b> |
| DATAkPCA     | 0.9747        | 0.9747        | 0.9977        | 0.9976        | <b>0.9981</b> | 0.9980        | 0.7589        | 0.7173        |
| kHerding     | 0.6577        | 0.6577        | 0.9014        | 0.8908        | 0.8306        | 0.8498        | 0.3259        | 0.3102        |
| Random       | 0.0657        | 0.0657        | 0.8895        | 0.8994        | 0.8472        | 0.8703        | 0.0409        | 0.0420        |
| Random data  | 0.9597        | 0.9597        | 0.9978        | 0.9978        | 0.9899        | 0.9924        | 0.7261        | 0.6806        |

TABLE 2.2 – Part de la variance totale représentée par  $M = 10$  vecteurs selon l'ensemble d'entraînement et la méthode employée

## 2.5 Cas d'application : variables aléatoires conditionnelles

Lorsque l'on cherche à déterminer des représentants pour des moyennes conditionnelles :  $\mathbb{E}_{Y|X} [\Phi(Y)|X]$  pour  $(X, Y)$  une variable aléatoire sur  $\mathcal{X} \times \mathcal{Y}$ , on ne dispose que de peu de points *a priori* relativement à la taille de l'ensemble de données, au voisinage de tout point  $x \in \mathcal{X}$  pour obtenir de l'information pertinente. On propose l'approche suivante : On dispose d'un ensemble de données  $(x_i, y_i)_{1 \leq i \leq N}$  indépendantes identiquement distribuées. On se donne  $x \in \mathcal{X}$  et on cherche à trouver  $M$  représentants de  $\mathbb{E}_Y [\Phi(Y)|X = x]$ . Pour se faire, on va -puisque sans plus d'hypothèse sur la distribution de  $(X, Y)$ - utiliser une méthode d'estimation non paramétrique de la loi de  $Y|X$  : les  $k$  plus proches voisins (kNN). Ceci permet étant donné un nombre de plus proches voisins à considérer  $k_n$  et un  $x \in \mathcal{X}$  d'extraire la sous famille  $(y_{a_{x,i}})_{1 \leq i \leq k_n}$  de la famille  $(y_i)_{1 \leq i \leq N}$  où les  $a_{x,i}$  sont les indices des  $k_n$  plus proches voisins de  $x$  dans les  $(x_i)_{1 \leq i \leq N}$ . Pour déterminer les bons représentants à partir de cette sous-famille de données, on est ramené au problème traité précédemment eq. (2.2). Comme conséquence de l'estimation conditionnelle, on ne bénéficie que de peu de points-données relativement à la dimension de l'espace  $\mathcal{Y}$ ; le ratio  $\frac{D}{k_n}$  étant grand, et relativement  $\frac{D}{k_n} \gg \frac{D}{N}$ . On peut alors appliquer une des méthodes présentées (PkPCA, DATAkPCA) ou leurs variantes sur l'ensemble de points données sélectionnés.

L'estimation empirique de la matrice de covariance sur la sous famille de données au voisinage de  $x$  est :

$$C_x = \frac{1}{k_n} \sum_{i=1}^{k_n} \Phi(y_{a_{x,i}}) \Phi(y_{a_{x,i}})^T \quad (2.23)$$

On réalise cette estimation pour un échantillon de  $x \in \mathcal{X}$  suivant la distribution de  $X$  (par exemple en sélectionnant une sous-famille des  $(x_i)_{1 \leq i \leq N}$ ), puis en prenant la valeur moyenne de représentation comme évaluation de la qualité du résultat obtenu. On notera  $M$  le nombre de représentants à déterminer pour chacun des  $H$  points sélectionnés dans  $\mathcal{X}$ , le voisinage de chaque point sélectionné dépendant du paramètre  $k$  du KNN.

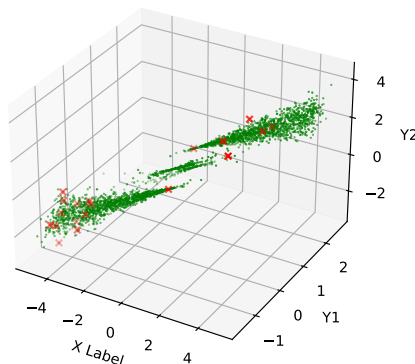


FIGURE 2.4 – Représentateurs pour  $k=10$ ,  $M=5$ ,  $H=10$  et PkPCA avec les points-données utilisés

On dispose de 3000 points-données générés suivant une distribution donnée, la variable  $X$  est unidimensionnelle et  $Y$  est bidimensionnelle. On représente ces données tridimensionnellement en

FIGURE 2.4. Les points-données  $y$  sont en vert et quelques représentants calculés sont en rouge.

On utilise un noyau gaussien, avec un paramètre  $\sigma$  égal à un sixième de l'écart type des données. Un meilleur choix de ce paramètre pourrait être envisagé, par exemple en prenant la moyenne des écarts types sur des voisinages de taille fixe dans l'espace  $\mathcal{X}$ .

On détermine pour chaque méthode : PkPCA, DATAkPCA, kHerd, une famille de représentants à partir des 3000 points données d'entraînement générées selon la loi de  $(X, Y)$ . Pour la partie test, on tire un ensemble de 500 points uniformément sur  $\mathcal{X}$ , puis pour chacun de ces points tirés, on génère un ensemble de points  $y$  selon la loi de  $Y|X$ . Ceci n'étant pas à proprement parler un échantillon test puisque ne suivant pas la loi de  $(X, Y)$ , cela explique pourquoi certaines des valeurs obtenues à l'entraînement sont inférieures aux valeurs de test.

Les différents résultats obtenus sont présentés en TABLE 2.3. On constate que aussi bien à l'entraînement qu'aux test la méthode PkPCA se révèle légèrement meilleure que DATAkPCA, et fournit des taux de représentation meilleurs que le kernel Herding. On constate aussi que les méthodes PkPCA et DATAkPCA sont d'autant meilleures que le nombre de représentants est important et que l'on augmente le nombre de points où l'on détermine des représentants. On constate aussi une certaine variabilité des résultats liée à la méthode d'évaluation utilisée non déterministe.

Cette bonne représentation des données est aussi visible en FIGURE 2.5 et FIGURE 2.6, où l'on a représenté la répartition des taux de représentation pour les 500 points-test utilisés. On voit que le plus faible taux de représentation en un point augmente à mesure que le nombre de points utilisés pour le calcul des représentants augmente, et que le nombre de représentants choisis est grand. Cela donne une certaine borne uniforme sur l'erreur de représentation commise par la méthode choisie. On constate aussi que la méthode PkPCA offre une borne uniforme plus étroite que kHerd, ce qui est une propriété désirable pour une représentation fidèle.

| Paramètres |     |    | PkPCA          |                | DATAkPCA       |                | kHerd   |                |
|------------|-----|----|----------------|----------------|----------------|----------------|---------|----------------|
| M          | H   | k  | TRAIN          | TEST           | TRAIN          | TEST           | TRAIN   | TEST           |
| 2          | 10  | 11 | <b>0.64367</b> | 0.764424       | 0.61340        | 0.72168        | 0.52900 | <b>0.77963</b> |
|            |     | 20 | <b>0.66939</b> | <b>0.91146</b> | 0.56852        | 0.71837        | 0.56707 | 0.82913        |
|            | 20  | 11 | 0.70660        | 0.81616        | <b>0.71572</b> | 0.82707        | 0.55241 | <b>0.84316</b> |
|            |     | 20 | <b>0.64172</b> | 0.91793        | 0.59623        | <b>0.93846</b> | 0.56006 | 0.92122        |
|            | 100 | 11 | 0.67216        | 0.86026        | <b>0.69014</b> | <b>0.87860</b> | 0.59077 | 0.86177        |
|            |     | 20 | <b>0.66753</b> | 0.93859        | 0.62306        | <b>0.93935</b> | 0.54089 | 0.93079        |
| 5          | 10  | 11 | 0.93402        | 0.76421        | <b>0.95296</b> | <b>0.76501</b> | 0.73505 | 0.72859        |
|            |     | 20 | <b>0.89977</b> | <b>0.89263</b> | 0.87034        | 0.82643        | 0.71934 | 0.83612        |
|            | 20  | 11 | <b>0.96062</b> | <b>0.84747</b> | 0.94473        | 0.76135        | 0.72419 | 0.80191        |
|            |     | 20 | <b>0.86074</b> | 0.89442        | 0.85513        | 0.90860        | 0.63951 | <b>0.92972</b> |
|            | 100 | 11 | <b>0.92790</b> | 0.85943        | 0.91510        | <b>0.87516</b> | 0.74134 | 0.82704        |
|            |     | 20 | <b>0.87835</b> | <b>0.94197</b> | 0.85592        | 0.93968        | 0.68347 | 0.93537        |
| 10         | 10  | 11 | 0.99823        | 0.68059        | <b>0.99928</b> | <b>0.79588</b> | 0.96195 | 0.77903        |
|            |     | 20 | <b>0.98573</b> | 0.79748        | 0.96552        | <b>0.84756</b> | 0.83448 | 0.78329        |
|            | 20  | 11 | 0.99435        | <b>0.81482</b> | <b>0.99866</b> | 0.76878        | 0.95149 | 0.79406        |
|            |     | 20 | <b>0.99000</b> | <b>0.93761</b> | 0.98083        | 0.92928        | 0.78330 | 0.93356        |
|            | 100 | 11 | 0.99540        | <b>0.88098</b> | <b>0.99879</b> | 0.86264        | 0.95185 | 0.86284        |
|            |     | 20 | <b>0.98301</b> | <b>0.94235</b> | 0.98275        | 0.94015        | 0.79525 | 0.93767        |

TABLE 2.3 – Variance représentée pour différents paramètres et différentes méthodes pour les données Multicones (3000 points) à l'entraînement et au test

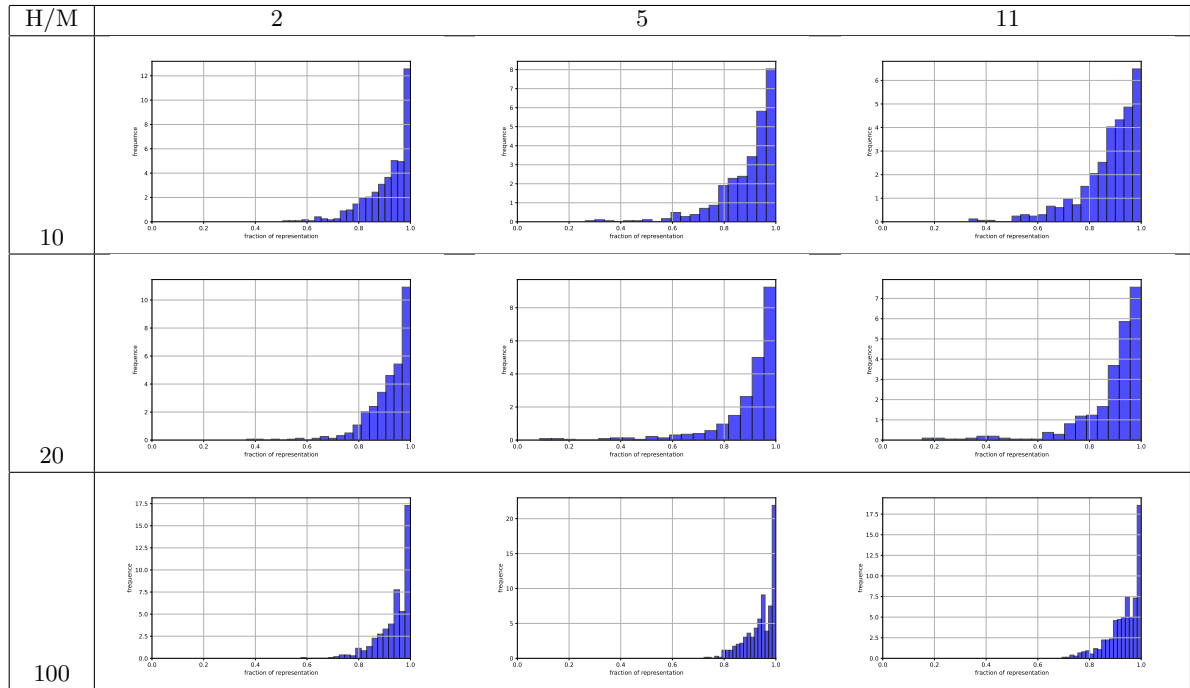


FIGURE 2.5 – Distribution des variances obtenues pour PkPCA et  $k=20$

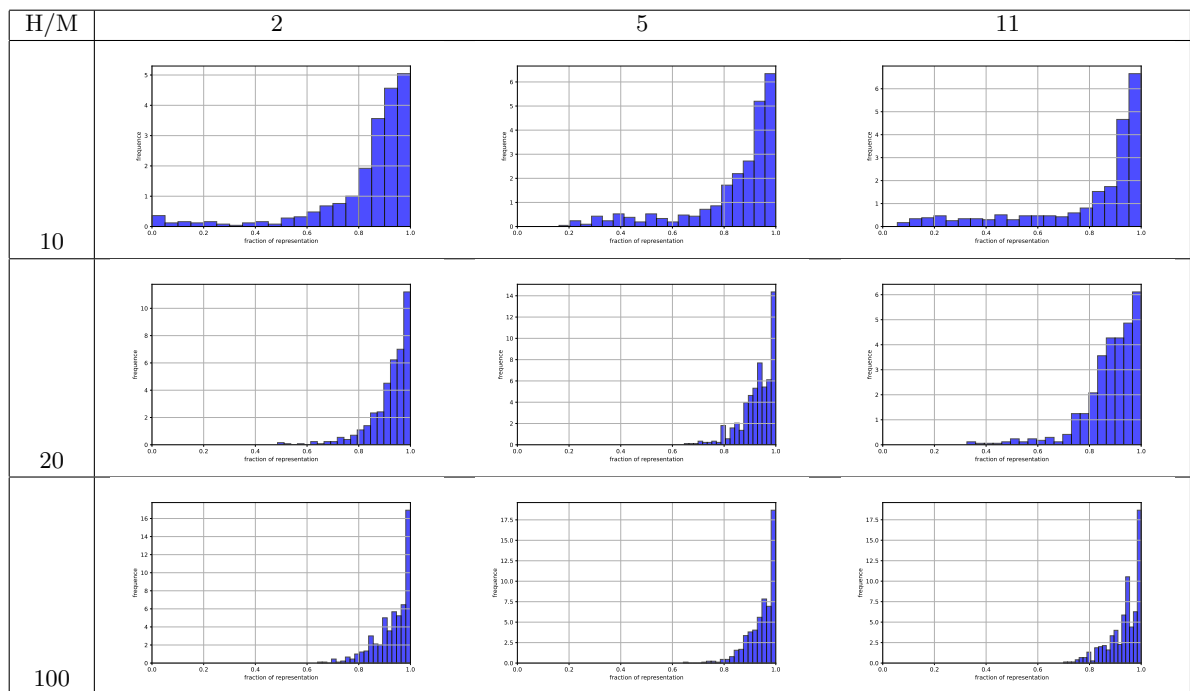


FIGURE 2.6 – Distribution des variances obtenues pour kHerding et  $k=20$

# Conclusion

Pour conclure ce mémoire, on a proposé une analyse de la borne théorique de convergence du Neural CME pour un kernel gaussien, après l'avoir reformulé comme un problème d'apprentissage non linéaire dans l'espace latent d'un kernel. On a fourni une méthode pour représenter les données en dimension réduite, via le choix d'un ensemble de représentants dans l'espace initial. On a par la suite fait le lien entre cette méthode proposée et le problème initial dans le cas de données conditionnelles. Enfin on a donné une perspective d'évolution de la méthode du Neural CME en proposant une architecture offrant une détermination directe de représentants conditionnellement.

## Notations

- On parle indistinctement de noyau ou de kernel
- $\mathcal{X}$  et  $\mathcal{Y}$  parties d'espaces vectoriels de dimension finie
- $(X, Y)$  couple de variables aléatoires sur  $\mathcal{X} \times \mathcal{Y}$  de loi  $\rho$
- $(x_i, y_i)_{1 \leq i \leq n}$  points données dans  $\mathcal{X}$  et  $\mathcal{Y}$
- $\eta_i \in \mathcal{Y}$  sont des représentants
- $k_x, k_y$  kernels
- $\Phi_x, \Phi_y$  fonctions de plongement associés aux kernels
- $\mathcal{H}_x, \mathcal{H}_y$  espaces latents (RKHS) associés aux kernels  $k_x, k_y$
- $\Omega$  est l'espace des paramètres d'un réseau de neurones donné
- $f_\theta^a$   $a$ -ième composante ( $1 \leq a \leq M$ ) du réseau de neurones  $f$  paramétré par  $\theta \in \Omega$
- $\|\cdot\|_y$  est la norme associée au produit scalaire dans le RKHS associé à  $k_y$
- On note  $\mathbb{E}_X$  l'espérance obtenue en intégrant selon la mesure correspondant à la loi de probabilité de  $X$ . de même pour  $Y, Y|X...$

# Bibliographie

- [1] Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. arXiv preprint arXiv :1203.3472, 2012.
- [2] Nello Cristianini and John Shawe-Taylor. Kernel methods for pattern analysis, volume 173. Cambridge University Press Cambridge, 2004.
- [3] László Györfi, Michael Kohler, Adam Krzyzak, Harro Walk, et al. A distribution-free theory of nonparametric regression, volume 1. Springer, 2002.
- [4] Paul Honeine and Cedric Richard. Preimage problem in kernel-based machine learning. IEEE Signal Processing Magazine, 28(2) :77–88, 2011.
- [5] Sophie Langer. Analysis of the rate of convergence of fully connected deep neural network regression estimates with smooth activation function. Journal of Multivariate Analysis, 182 :104695, 2021.
- [6] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel mean embedding of distributions : A review and beyond. Foundations and Trends® in Machine Learning, 10(1-2) :1–141, 2017.
- [7] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In International conference on artificial neural networks, pages 583–588. Springer, 1997.
- [8] Bernhard Schölkopf and Alexander J Smola. Learning with kernels : support vector machines, regularization, optimization, and beyond. MIT press, 2002.
- [9] John Shawe-Taylor and Nello Cristianini. Kernel methods for pattern analysis. Cambridge university press, 2004.
- [10] Eiki Shimizu, Kenji Fukumizu, and Dino Sejdinovic. Neural-kernel conditional mean embeddings. arXiv preprint arXiv :2403.10859, 2024.
- [11] John C Snyder, Sebastian Mika, Kieron Burke, and Klaus-Robert Müller. Kernels, pre-images and optimization. Empirical Inference : Festschrift in Honor of Vladimir N. Vapnik, pages 245–259, 2013.
- [12] Le Song, Jonathan Huang, Alex Smola, and Kenji Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In Proceedings of the 26th Annual International Conference on Machine Learning, pages 961–968, 2009.
- [13] Taiji Suzuki. Adaptivity of deep relu network for learning in besov and mixed smooth besov spaces : optimal rate and curse of dimensionality. arXiv preprint arXiv :1810.08033, 2018.
- [14] Liyuan Xu, Yutian Chen, Siddarth Srinivasan, Nando de Freitas, Arnaud Doucet, and Arthur Gretton. Learning deep features in instrumental variable regression. arXiv preprint arXiv :2010.07154, 2020.
- [15] Qiang Zhang and Baoxin Li. Dictionary learning in visual computing. Springer Nature, 2022.