université PARIS-SACLAY

université évry val-d'essonne

TELECOM ParisTech

NNT: 201XSACLSXXX

A thesis submitted to attain the degree of

Doctor Of Université Paris-Saclay

Prepared at

Université-D'Évry-Val-D'Essonne & Télécom-ParisTech

Presented by
M. Romain Brault*

Entitled

Data Are Not Real!

About

Large-Scale Learning On Structured Input-Output Data With Operator-Valued Kernels

Doctoral School – 580
Sciences et Technologies de l'Information et de la Communication (STIC)
Laboratoire Informatique, Biologie Intégrative et Systèmes Complexes (IBISC)
Specialty Area of Computer Science

Under supervision of Professor (Prof.) Florence d'Alché-Buc†
Presented and defended at Université-D'Évry-Val-D'Essonne , the 1st of March 2017
Jury

Mme.	Florence	d'Alché-Buc	Télécom-ParisTech	Director,
M.	Jean-Marc	Delosme	Laboratoire IBISC	President,
M.	Alexandre	Gramfort	Télécom-ParisTech	Invited,
M.	Arthur	Gretton	University College London	Reporter,
M.	Zoltán	Szabó	École Polytechnique	Reporter,
Mme.	Marie	Szafranski	Laboratoire IBISC	Examinator.

* Email: romain.brault@ibisc.fr
† Email: florence.dalche@telecom-paristech.fr

February 16, 2017 – version 0.1

Romain Brault: *Data Are Not Real!* Large-Scale Learning On Structured Input-Output Data With Operator-Valued Kernels, © February 16, 2017

SUPERVISOR:
Professor (Prof.) Florence d'Alché-Buc

LOCATION:
15, Rue Plumet, 75015 - Paris, France

ABSTRACT

In this thesis we study scalable methods to perform regression with *Operator-Valued Kernels* in order to learn *vector-valued functions*.

When data present structure, or relations between them or their different components, a common approach is to treat the data as a vector living in an appropriate vector space rather a collection of real number. This representation allows to take into account the structure of the data by defining an appropriate space embbeding the underlying structure. Thus many problems in machine learning can be cast into learning vector-valued functions. Operator-Valued Kernels *Operator-Valued Kernels* and *vector-valued Reproducing Kernel Hilbert Spaces* provide a theoretical and practical framework to address that issue, naturally extending the well-known framework of scalar-valued kernels. In the context of scalar-valued function learning, a scalar-valued kernel can be seen a a similarity measure between two data point. A solution of the learning problem has the form of a linear combination of theses similarities with respect to weights to determine in order to have the best "fit" of the data. When dealing with Operator-Valued Kernels, the evalution of the kernel is no longer a scalar similarity, but a function acting on vectors. A solution is then a linear combination of operators with respect to vector weights.

Although Operator-Valued Kernels generalize strictly scalar-valued kernels, large scale applications are usually not affordable with these tools that require an important computational power along with a large memory capacity. In this thesis, we propose and study scalable methods to perform regression with *Operator-Valued Kernels*. To achieve this goal, we extend Random Fourier Features, an approximation technique originally introduced for scalar-valued kernels, to *Operator-Valued Kernels*. The idea is to take advantage of an approximated operator-valued feature map in order to come up with a linear model in a finite dimensional space.

First we develop a general framework devoted to the approximation of shift-invariant Mercer kernels on Locally Compact Abelian groups and study their properties along with the complexity of the algorithms based on them. Second we show theoretical guarantees by bounding the error due to the approximation, with high probability. Third, we study various applications of Operator Random Fourier Features to different tasks of Machine learning such as multi-class classification, multi-task learning, time serie modeling, functional regression and anomaly detection. We also compare the proposed framework with other state of the art methods. Fourth, we conclude by drawing short-term and mid-term perspectives.

PUBLICATIONS

Some ideas and figures have appeared previously in the following publications:

Put your publications from the thesis here. The packages `multibib` or `bibtopic` etc. can be used to handle multiple different bibliographies in your document.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

ACKNOWLEDGEMENTS

Put your acknowledgements here.

Many thanks to everybody who already sent me a postcard!

Regarding the typography and other help, many thanks go to Marco Kuhlmann, Philipp Lehman, Lothar Schlesier, Jim Young, Lorenzo Pantieri and Enrico Gregorio[1], Jörg Sommer, Joachim Köstler, Daniel Gottschlag, Denis Aydin, Paride Legovini, Steffen Prochnow, Nicolas Repp, Hinrich Harms, Roland Winkler, and the whole LaTeX-community for support, ideas and some great software.

Regarding LyX: The LyX port was initially done by *Nicholas Mariette* in March 2009 and continued by *Ivo Pletikosić* in 2011. Thank you very much for your work and the contributions to the original style.

1 Members of GuIT (Gruppo Italiano Utilizzatori di TeX e LaTeX)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

MSE Mean Squared Error.

OVK Operator-Valued Kernel.

RFF Random Fourier Feature.

ORFF Operator-valued Random Fourier Feature.

POVM Positive Operator-Valued Measure.

RKHS Reproducing Kernel Hilbert Space.

$\mathcal{Y}$-RKHS $\mathcal{Y}$-Reproducing Kernel Hilbert Space.

LCA Locally Compact Abelian.

FT Fourier transform.

IFT Inverse Fourier transform.

Part I

INTRODUCTION

You can put some informational part preamble text here. Illo principalmente su nos. Non message *occidental* angloromanic da. Debitas effortio simplificate sia se, auxiliar summarios da que, se avantiate publicationes via. Pan in terra summarios, capital interlingua se que. Al via multo esser specimen, campo responder que da. Le usate medical addresses pro, europa origine sanctificate nos.

1

MOTIVATIONS

2

BACKGROUND

2.1 NOTATIONS

[1] *Commutative.*

We note $\mathbb{K}$ any Abelian[1] field, $\mathbb{R}$ the Abelian field of real numbers and $\mathbb{C}$ the abelian field of complex numbers. The unit pure imaginary number $\sqrt{-1} \in \mathbb{C}$ is denoted i and the euler constant $\exp(1) \in \mathbb{R}$ is denoted e. $\mathbb{N}$ represent the monoid of natural numbers and $\mathbb{N}_n$, $n \in \mathbb{N}$ the set of natural numbers smaller or equal to n. For any space $\mathcal{S}$, $\mathcal{S}^d$, $d \in \mathbb{N}$ represents the cartesian product space $\mathcal{S}^d = \mathcal{S} \times \ldots \times \mathcal{S}$. For any two algebraic structures $\mathcal{S}$ and $\mathcal{S}'$ we write $\mathcal{S} \cong \mathcal{S}'$ is there exist an isomorphism between these two structures.

$\mathcal{B}(\mathbb{R}^d)$ is the Borel σ-algebra on $\mathbb{R}^d$. If $\mathcal{X}$ and $\mathcal{Y}$ are two topological vector spaces, we denote by $\mathcal{F}(\mathcal{X};\mathcal{Y})$ the vector space of functions $f : \mathcal{X} \to \mathcal{Y}$ and $\mathcal{C}(\mathcal{X};\mathcal{Y}) \subset \mathcal{F}(\mathcal{X};\mathcal{Y})$ the subspace of continuous functions. If $\mathcal{H}$ is an Hilbert space on a field $\mathbb{K}$ we denote its scalar product by $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ and its norm by $\|\cdot\|_{\mathcal{H}}$.

If $x \in \mathcal{H}$ we note $x^* \in \mathcal{H}^*$, $x^* : \mathcal{H} \to \mathbb{K}$ the dual vector in the dual vector space $\mathcal{H}^*$ of $\mathcal{H}$. If $\mathcal{H}$ is a Hilbert space, it is reflexive (i. e. $\mathcal{H}^{**} \cong \mathcal{H}$). Let $\mathcal{H}$ be a finite dimensional Hilbert space endowed with a basis $(e_i)_{i=1}^N$. If no specific mention, we consider that $\mathcal{H}^*$ is endowed with the dual basis $(e_i^*)_{i=1}^N$ such that $e_i^* e_j = \delta_{ij}$. If $\mathcal{H} = \mathbb{R}^d$, this allows us to identify x^* with x^T. The dual space $\mathcal{H}^*$ is isomorphic to $\mathcal{H}$ if and only if $\mathcal{H}$ is finite dimensional. For all x and y in $\mathcal{H}$ we have $\langle x, y\rangle_{\mathcal{H}} = x^* y$.

We set $\mathcal{L}(\mathcal{H}) = \mathcal{L}(\mathcal{H};\mathcal{H})$ to be the space of linear operators from $\mathcal{H}$ to itself. If $W \in \mathcal{L}(\mathcal{H})$, Ker W denotes the nullspace, Im W the image and $W^* \in \mathcal{L}(\mathcal{H})$ the adjoint operator.

We call matrix M of size $(m, n) \in \mathbb{N}^2$ on an Abelian field $\mathbb{K}$ a collection of elements $M = (m_{ij})_{1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n}$, $m_{ij} \in \mathbb{K}$. We note $\mathcal{M}_{m,n}(\mathbb{K})$ the vector space of all matrices. If $\mathcal{H}_1$ and $\mathcal{H}_2$ are two finite dimensional Hilbert spaces on an Abelian field $\mathbb{K}$, any bounded linear operator $L \in \mathcal{L}(\mathcal{H}_1;\mathcal{H}_2)$ can be viewed as a matrix. Let $n = \dim(\mathcal{H}_1)$, $m = \dim(\mathcal{H}_2)$ and let $B = (e_i)_{i=1}^n$ and $C = (e'_i)_{i=1}^m$ be the respective canonical bases of $\mathcal{H}_1$ and $\mathcal{H}_2$. We note $\mathrm{mat}_{B,C} : \mathcal{L}(\mathcal{H}_1;\mathcal{H}_2) \to \mathcal{M}_{m,n}(\mathbb{K})$ such that $M = \mathrm{mat}_{B,C}(L) = (\langle e'_j, L e_i\rangle)_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m} \in \mathcal{M}_{m,n}(\mathbb{K})$. Let $M_1 \in \mathcal{M}_{m,n}(\mathbb{K})$ and $M_2 \in \mathcal{M}_{n,l}(\mathbb{K})$. The product between two matrices is written $M_1 M_2 \in \mathcal{M}_{m,l}(\mathbb{K})$ and obey $(M_1 M_2)_{ij} = \sum_{k=1}^n M_{ik} M_{kj}$. Given two linear operator $L_1 \in \mathcal{L}(\mathcal{H}_1;\mathcal{H}_2)$ and $L2 \in \mathcal{L}(\mathcal{H}_2;\mathcal{H}_3)$ we have $L_1 L_2 \in \mathcal{L}(\mathcal{H}_1;\mathcal{H}_3)$ and

$$\mathrm{mat}_{B,D}(L_1 L_2) = \mathrm{mat}_{B,C}(L_1)\mathrm{mat}_{C,D}(L_2).$$

The operator $\mathrm{mat}_{B,C}$ is a vector space isomorphism allowing us to identify $\mathcal{L}(\mathcal{H}_1;\mathcal{H}_2)$ with $\mathcal{M}_{mn}(\mathbb{K})$ where $n = \dim(\mathcal{H}_1)$ and $m = \dim(\mathcal{H}_2)$. All these notations are summarized in table 1.

Table 1: Mathematical symbols used throughout the parper and their signification.

Symbol	Meaning
$\mathbb{K}$	Any Abelian field.
$\mathbb{R}$	The Abelian field of real numbers.
$\mathbb{C}$	The Abelian field of complex numbers.
$i \in \mathbb{C}$	Unit pure imaginary number $\sqrt{-1}$.
$e \in \mathbb{R}$	Euler constant.
δ_{ij}	Kronecker delta function. $\delta_{ij} = 0$ if $i \neq j$, 1 otherwise.
$\langle \cdot, \cdot \rangle$	Euclidean inner product.
$\|\cdot\|$	Euclidean norm.
$\mathcal{X}$	Input space.
$\widehat{\mathcal{X}}$	The Pontryagin dual of $\mathcal{X}$.
$\mathcal{Y}$	Output space (Hilbert space).
$\mathcal{H}$	Feature space (Hilbert space).
$\langle \cdot, \cdot \rangle_{\mathcal{Y}}$	The canonical inner product of the Hilbert space $\mathcal{Y}$.
$\|\cdot\|_{\mathcal{Y}}$	The canonical norm induced by the inner product of the Hilbert space $\mathcal{Y}$.
$\mathcal{F}(\mathcal{X}; \mathcal{Y})$	Vector space of function from $\mathcal{X}$ to $\mathcal{Y}$.
$\mathcal{C}(\mathcal{X}; \mathcal{Y})$	The vector subspace of $\mathcal{F}$ of continuous function from $\mathcal{X}$ to $\mathcal{Y}$.
$\mathcal{L}(\mathcal{H}; \mathcal{Y})$	The set of bounded linear operator from a Hilbert space $\mathcal{H}$ to a Hilbert space $\mathcal{Y}$.
$\mathcal{M}_{m,n}(\mathbb{K})$	The set of matrices of size (m, n).
$\mathcal{L}(\mathcal{Y})$	The set of bounded linear operator from a Hilbert space $\mathcal{H}$ to itself.
$\mathcal{L}_+(\mathcal{Y})$	The set of non-negative bounded linear operator from a Hilbert space $\mathcal{H}$ to itself.
$\mathcal{B}(\mathcal{X})$	Borel σ-algebra on $\mathcal{X}$.
$\mu(\mathcal{X})$	A scalar positive measure of $\mathcal{X}$.
$L^p(\mathcal{X}, \mu)$	The Banach space of $\|\cdot\|^p$-integrable function from $(\mathcal{X}, \mathcal{B}(\mathcal{X}), \mu)$ to $\mathbb{C}$.
$L^p(\mathcal{X}, \mu; \mathcal{Y})$	The Banach space of $\|\cdot\|_{\mathcal{Y}^p}$ (Bochner)-integrable function from $(\mathcal{X}, \mathcal{B}(\mathcal{X}), \mu)$ to $\mathcal{Y}$.
$\bigoplus_{j=1}^{D} x_i$	The direct sum of D vectors x_i's in $\mathcal{H}$.

2.2 ABOUT STATISTICAL LEARNING

2.3 ON LARGE-SCALE LEARNING

2.4 HISTORY AND STATE OF THE ART OF LARGE SCALE LEARNING WITH KERNELS

2.4.1 *Introduction to kernel methods*

2.4.2 *Quadratic programing, subsampling*

2.4.3 *Gradient descents*

2.4.4 *Mercer Theorem, Nyström method and feature maps*

2.4.5 *Recent extensions*

2.5 ELEMENTS OF ABSTRACT HARMONIC ANALYSIS

2.5.1 *Locally compact Abelian groups*

Definition 2.1 *Locally Compact Abelian group. A group $\mathcal{X}$ endowed with a binary operation $\star$ is said to be Locally Compact Abelian if $\mathcal{X}$ is a topological commutative group w. r. t. $\star$ for which every point has a compact neighborhood and is Hausdorff.*

2.5.2 *The Haar measure*

Definition 2.2 (The Haar measure). *There is, up to a positive multiplicative constant, a unique countably additive, nontrivial measure* **Haar** *on the Borel subsets of $\mathcal{X}$ satisfying the following properties* [2] *:*

[2] *For more details and constructive proofs see [1, 14, 17].*

1. *The measure* **Haar** *is translation-invariant.*

   $$\mathbf{Haar}(x \star E) = \mathbf{Haar}(E),$$

   for every x in $\mathcal{X}$ and Borel set $E \in \mathcal{B}(\mathcal{X})$.

2. *The measure* **Haar** *is finite on every compact set.*

   $$\mathbf{Haar}(K) < \infty$$

   for any compact set $K \in \mathcal{B}(\mathcal{X})$.

3. *The measure* **Haar** *is outer regular on any Borel sets E.*

   $$\mathbf{Haar}(E) = \inf\{\, \mathbf{Haar}(U) \mid E \subseteq U \,\}$$

   For any open set U.

4. *The measure* **Haar** *is inner regular on open sets* E.

$$\mathbf{Haar}(E) = \sup\{\, \mathbf{Haar}(K) \mid K \subseteq E \,\},$$

for any compact set K.

Such a measure on G is called a Haar measure[3] . An immediate consequence of the invariance is that for any $c \in \mathcal{X}$,

$$\int_{\mathcal{X}} f(s \star x)\mathrm{d}\mathbf{Haar}(x) = \int_{\mathcal{X}} f(x)\mathrm{d}\mathbf{Haar}(x).$$

It can be shown that $\mathbf{Haar}(U) > 0$ for every non-empty open subset U. In particular, if $\mathcal{X}$ is compact then $\mathbf{Haar}(\mathcal{X})$ is finite and positive, so we can uniquely specify a left Haar measure on $\mathcal{X}$ by adding the normalization condition $\mathbf{Haar}(\mathcal{X}) = 1$. Another useful property is that given a continuous function $f \in \mathcal{C}(\mathcal{X})$,

$$\int_{\mathcal{X}} f(x)\mathrm{d}\mathbf{Haar}(x) = \int_{\mathcal{X}} f\left(x^{-1}\right)\mathrm{d}\mathbf{Haar}(x).$$

We call measured space the space $(\mathcal{X}, \mathcal{B}(\mathcal{X}), \mathbf{Haar})$ the space $\mathcal{X}$ endowed with its Borel sigma algebra and some measure **Haar**. If $\mathbf{Haar}(\mathcal{X}) = 1$ then the space $(\mathcal{X}, \mathcal{B}(\mathcal{X}), \mathbf{Haar})$ is called a probability space.

[3] *If $\mathcal{X}$ was not supposed to be Abelian, we should have defined a left Haar measure and a right Haar measure. In our case both measure are the same, so we refer to both of them as Haar measure*

2.5.3 *Even and odd functions*

We say that a function $f : \mathcal{X} \to \mathcal{C}$ is even if for all $x \in \mathcal{X}$, $f(x) = f\left(x^{-1}\right)$ and odd if $f(x) = -f\left(x^{-1}\right)$. The definition can be extended to operator-valued functions.

Definition 2.3 (Even and odd function on a LCA group). *Let $\mathcal{X}$ be a measured LCA group. A function $f : \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ is (weakly) even if for all $x \in \mathcal{X}$ and all $y, y' \in \mathcal{Y}$*

$$\langle y, f\left(x^{-1}\right) y' \rangle = \langle y, f(x) y' \rangle \tag{2.1}$$

and (weakly) odd if

$$\langle y, f\left(x^{-1}\right) y' \rangle = -\langle y, f(x) y' \rangle \tag{2.2}$$

It is easy to check that if f is odd then $\int_{\mathcal{X}} \langle y, f(x) y' \rangle \mathrm{d}\mathbf{Haar}(x) = 0$.

Proof

$$\begin{aligned}
\int_{\mathcal{X}} \langle y, f(x) y' \rangle \mathrm{d}\mathbf{Haar}(x) &= \frac{1}{2}\left(\int_{\mathcal{X}} \langle y, f\left(x^{-1}\right) y' \rangle \mathrm{d}\mathbf{Haar}(x) \right. \\
&\qquad \left. + \int_{\mathcal{X}} \langle y, f(x) y' \rangle \mathrm{d}\mathbf{Haar}(x) \right) \\
&= \frac{1}{2}\left(-\int_{\mathcal{X}} \langle y, f(x) y' \rangle \mathrm{d}\mathbf{Haar}(x) \right. \\
&\qquad \left. + \int_{\mathcal{X}} \langle y, f(x) y' \rangle \mathrm{d}\mathbf{Haar}(x) \right) \\
&= 0.
\end{aligned}$$

Besides the product of an even and an odd function is odd. Indeed for all f, $g \in \mathcal{F}(\mathcal{X}; \mathcal{L}(\mathcal{Y}))$, where f is even and g odd. Define $h(x) = \langle y, f(x)g(x)y' \rangle$. Then we have

$$\begin{aligned} h\left(x^{-1}\right) &= \langle y, f\left(x^{-1}\right) g\left(x^{-1}\right) y' \rangle \\ &= \langle y, f(x)\left(-g(x)\right) y' \rangle \\ &= -h(x). \end{aligned} \tag{2.3}$$

2.5.4 *Characters*

Locally Compact Abelian (LCA) groups are central to the general definition of Fourier Transform which is related to the concept of Pontryagin duality [17]. Let $(\mathcal{X}, \star)$ be a LCA group with e its neutral element and the notation, x^{-1}, for the inverse of $x \in \mathcal{X}$. A *character* is a complex continuous homomorphism $\omega : \mathcal{X} \to \mathbb{U}$ from $\mathcal{X}$ to the set of complex numbers of unit module $\mathbb{U}$. The set of all characters of $\mathcal{X}$ forms the Pontryagin *dual group* $\widehat{\mathcal{X}}$. The dual group of an LCA group is an LCA group so that we can endow $\widehat{\mathcal{X}}$ with a "dual" Haar measure noted $\widehat{\textbf{Haar}}$. Then the dual group operation is defined by

$$(\omega_1 \star \omega_2)(x) = \omega_1(x)\omega_2(x) \in \mathbb{U}.$$

The Pontryagin duality theorem states that $\widehat{\widehat{\mathcal{X}}} \cong \mathcal{X}$. I. e. there is a canonical isomorphism between any LCA group and its double dual. To emphasize this duality the following notation is usually adopted

$$\omega(x) = (x, \omega) = (\omega, x) = x(\omega), \tag{2.4}$$

where $x \in \mathcal{X} \cong \widehat{\widehat{\mathcal{X}}}$ and $\omega \in \widehat{\mathcal{X}}$. Another important property involves the complex conjugate of the pairing which is defined as

$$\overline{(x, \omega)} = \left(x^{-1}, \omega\right) = \left(x, \omega^{-1}\right). \tag{2.5}$$

Table 3: Classification of Fourier transforms in terms of their domain and transform domain.

$\mathcal{X} =$	$\widehat{\mathcal{X}} \cong$	Operation	Pairing
$\mathbb{R}^d$	$\mathbb{R}^d$	$+$	$(x, \omega) = \exp\left(i\langle x, \omega\rangle\right)$
$\mathbb{R}^d_{*,+}$	$\mathbb{R}^d$	$\cdot$	$(x, \omega) = \exp\left(i\langle \log(x), \omega\rangle\right)$
$(-c; +\infty)^d$	$\mathbb{R}^d$	$\odot$	$(x, \omega) = \exp\left(i\langle \log(x + c), \omega\rangle\right)$

We notice that for any pairing depending of ω, there exists a function $h_\omega : \mathcal{X} \to \mathbb{R}$ such that $(x, \omega) = \exp(ih_\omega(x))$ since any pairing maps into $\mathbb{U}$. Moreover,

$$\begin{aligned}\left(x \star z^{-1}, \omega\right) &= \omega(x)\omega\left(z^{-1}\right) \\ &= \exp\left(+ih_\omega\left(x\right)\right)\exp\left(+ih_\omega\left(z^{-1}\right)\right) \\ &= \exp\left(+ih_\omega\left(x\right)\right)\exp\left(-ih_\omega\left(z\right)\right).\end{aligned}$$

Example 2.1 $\widehat{\mathbb{R}} \cong \mathbb{R}$ *with the duality pairing* $(x, \omega) = \exp(ix\omega)$ *for all* $x \in \mathbb{R}$ *and all* $\omega \in \mathbb{R}$.

Proof *If* $\omega \in \widehat{\mathbb{R}}$ *then* $\omega(0) = 1$ *since* ω *is an homeomorphism from* $\mathbb{R}$ *to* $\mathbb{U}$. *Therefore there exists* $a > 0$ *such that* $\int_0^a \omega(t)d\mathbf{Leb}(t) \neq 0$. *Setting* $A\omega = \int_0^a \omega(t)d\mathbf{Leb}(t)$ *we have*

$$(A\omega)(x) = \int_0^a \omega(x+t)d\mathbf{Leb}(t) = \int_x^{a+x} \omega(t)d\mathbf{Leb}(t).$$

so ω *is differentiable and*

$$\omega'(x) = A^{-1}(\omega(a+x) - \omega(x)) = c\omega(x) \quad \textit{where} \quad c = A^{-1}(\omega(a) - 1).$$

It follow that $\omega(x) = e^{cx}$, *and since* $|\omega| = 1$, *one can take* $c = i\xi$ *for some* $\xi, \in \mathbb{R}$. *Hence we can identify* ω *with* ξ *and* $\widehat{\mathbb{R}}$ *with* $\mathbb{R}$ *since* ξ *uniquely determines* ω. *With mild abuse of notation we identify* $\omega = \xi$. □

Table 3 provide an explicit list of pairings for various groups based on $\mathbb{R}^d$ or its subsets. We especially mention the duality pairing associated to the skewed multiplicative LCA product group $\mathcal{X} = (-c_k; +\infty)_{k=1}^d$ endowed with the group operation $\odot$ defined component-wise for all $x, z \in \mathcal{X}$ as follow.

$$x \odot z := ((x_k + c)(z_k + c) - c)_{k=1}^d.$$

Then the duality pairing is defined by

$$(x, \omega) = \exp\left(i\sum_{k=1}^d \omega_k \log(x_k + c_k)\right).$$

Hence $h_\omega(x) = \sum_{k=1}^d \omega_k \log(x_k + c_k)$. This group together with the operation $\odot$ has been proposed by [25] to handle histograms features especially useful in image recognition applications.

2.5.5 *The Fourier transform*

For a function with values in a separable Hilbert space $f \in L^1(\mathcal{X}, \mathbf{Haar}; \mathcal{Y})$, we denote $\mathcal{F}[f]$ its Fourier transform (FT) which is defined by

$$\forall \omega \in \widehat{\mathcal{X}}, \quad \mathcal{F}[f](\omega) = \int_{\mathcal{X}} \overline{(x, \omega)} f(x) d\mathbf{Haar}(x).$$

For a measure **Haar** defined on $\mathcal{X}$, there exists a unique suitably normalized measure $\widehat{\mathbf{Haar}}$ on $\widehat{\mathcal{X}}$ such that for all $f \in L^1(\mathcal{X}, \mathbf{Haar}; \mathcal{Y})$ and if $\mathcal{F}[f] \in L^1(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}}; \mathcal{Y})$ we have

$$\forall x \in \mathcal{X}, \quad f(x) = \int_{\widehat{\mathcal{X}}} \mathcal{F}[f](\omega)(x, \omega) d\widehat{\mathbf{Haar}}(\omega). \tag{2.6}$$

Moreover if $\widehat{\mathbf{Haar}}$ is normalized, $\mathcal{F}$ extends to a unitary operator from $L^2(\mathcal{X}, \mathbf{Haar}, \mathcal{Y})$ onto $L^2(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}}; \mathcal{Y})$. Then the Inverse Fourier transform (IFT) of a function $g \in L^1(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}}; \mathcal{Y})$ (where $\widehat{\mathbf{Haar}}$ is suitably normalized w. r. t. **Haar**) is noted $\mathcal{F}^{-1}[g]$ defined by

$$\forall x \in \mathcal{X}, \quad \mathcal{F}^{-1}[g](x) = \int_{\widehat{\mathcal{X}}} (x, \omega) g(\omega) d\widehat{\mathbf{Haar}}(\omega),$$

Table 3 gives some examples of real Abelian groups with their associated dual and pairing. The interested reader can refer to Folland [17] for a more detailed construction of LCA, Pontryagin duality and Fourier transforms on LCA. For the familiar case of a scalar-valued function f on the LCA group $(\mathbb{R}^d, +)$, we have for all $\omega \in \widehat{\mathcal{X}}$

$$\begin{aligned} \mathcal{F}[f](\omega) &= \int_{\widehat{\mathcal{X}}} \overline{(x, \omega)} f(x) d\mathbf{Haar}(x) \\ &= \int_{\mathbb{R}^d} e^{-i\langle x, \omega \rangle} f(x) d\mathbf{Leb}(x), \end{aligned} \tag{2.7}$$

the Haar measure being here the Lebesgue measure.

2.5.6 *Representation of Groups*

Representations of groups are convenient are tools that allows group-theoretic problems to be replaced by linear algebra problems. Let $Gl(\mathcal{H})$ be the group of continuous isomorphism of $\mathcal{H}$, a Hilbert space, onto itself. A representation π of a LCA group $\mathcal{X}$ in $\mathcal{H}$ is an homomorphism π:

$$\pi : \mathcal{X} \to Gl(\mathcal{H})$$

for which all the maps $\mathcal{X} \to \mathcal{H}$ defined for all $v \in \mathcal{H}$ as $x \mapsto \pi(x)v$, are continuous. The space $\mathcal{H}$ in which the representation takes place is called the representation space of π. A representation π of a group $\mathcal{X}$ in a vector space $\mathcal{H}$ defines an action defined for all $x \in \mathcal{X}$ by

$$\pi_x : \begin{cases} \mathcal{H} & \to \mathcal{H} \\ v & \mapsto \pi(x)v. \end{cases}$$

If for all $x \in \mathcal{X}$, $\pi(x)$ is a unitary operator, then the group representation π is said to be unitary (i. e. $\forall x \in \mathcal{X}$, $\pi(x)$ is isometric and surjective). Thus π is unitary when for all $x \in \mathcal{X}$

$$\pi(x)^* = \pi(x)^{-1} = \pi\left(x^{-1}\right).$$

The representation π of $\mathcal{X}$ in $\mathcal{H}$ is said to be irreducible when $\mathcal{H} \neq \{0\}$ and are the only two stable invariant subspaces under all operators $\pi(x)$ for all $x \in \mathcal{X}$. I. e. $\forall U \subset \mathcal{H}$, $U \neq \{0\}, \{\pi(x)v \mid \forall x \in \mathcal{X}, \forall v \in U\} \neq U$.

To study LCA groups we also introduce the left regular representation of $\mathcal{X}$ acting on a Hilbert space of function $\mathcal{H} \subset \mathcal{F}(\mathcal{X}; \mathcal{Y})$. For all $x, z \in \mathcal{X}$ and for all $f \in \mathcal{H}$,

$$(\lambda_z f)(x) := f(z^{-1} \star x).$$

The representation λ of $\mathcal{X}$ defines an action λ_x on $\mathcal{H}$ which is the translation of $f(x)$ by z^{-1}. With this definition one has for all $x, z \in \mathcal{X}$, $\lambda_x \lambda_z = \lambda_{x^{-1} \star z}$. Such representations are faithful, that is $\lambda_x = 1 \iff x = e$.

2.6 ON OPERATOR-VALUED KERNELS

We now introduce the theory of $\mathcal{Y}$-Reproducing Kernel Hilbert Space ($\mathcal{Y}$-RKHS) that provides a flexible framework to study and learn vector-valued functions.

2.6.1 *Definitions and properties*

A $\mathcal{Y}$-Reproducing Kernel Hilbert Space ($\mathcal{Y}$-RKHS) is a functional vector space defined as follow.

Definition 2.4 ($\mathcal{Y}$-Reproducing Kernel Hilbert Space [12, 28]). *Let $\mathcal{Y}$ be a (real or complex) Hilbert space. A $\mathcal{Y}$-Reproducing Kernel Hilbert Space on $\mathcal{X}$, a locally compact second countable topological space is a Hilbert space $\mathcal{H}$ such that*

1. *the elements of $\mathcal{H}$ are functions from $\mathcal{X}$ to $\mathcal{Y}$ (i. e. $\mathcal{H} \subset \mathcal{F}(\mathcal{X}, \mathcal{Y})$);*
2. *for all $x \in \mathcal{X}$, there exists a positive constant C_x such that for all $f \in \mathcal{H}$*

$$\|f(x)\|_{\mathcal{Y}} \leqslant C_x \|f\|_{\mathcal{H}}. \tag{2.8}$$

An operator-valued kernel is defined here as a Operator-Valued Kernel of positive type Carmeli et al. [13].

Definition 2.5 (Operator-Valued Kernel of positive type acting on a complex space). *Given $\mathcal{X}$ a locally compact second countable topological space and $\mathcal{Y}$ a complex Hilbert Space, a map $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ is called an Operator-Valued Kernel of positive type kernel if*

$$\sum_{i,j=1}^{N} \langle K(x_i, x_j) y_j, y_i \rangle_{\mathcal{Y}} \geqslant 0, \tag{2.9}$$

for all $N \in \mathbb{N}$ and for all sequences of points $(x_i)_{i=1}^N$ in $\mathcal{X}^N$ and all sequences of points $(y_i)_{i=1}^N$ in $\mathcal{Y}^N$.

if $\mathcal{Y}$ is a complex Hilbert space, an Operator-Valued Kernel of positive type is always Hermitian, i. e. $K(x,z) = K(z,x)^*$. This gives rise to the following definition of Operator-Valued Kernel of positive type acting on a real Hilbert space.

Definition 2.6 (Operator-Valued Kernel of positive type acting on a real Hilbert space). *Given $\mathcal{X}$ a locally compact second countable topological space and $\mathcal{Y}$ a real Hilbert Space, a map $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ is called an Operator-Valued Kernel of positive type kernel if*

$$K(x,z) = K(z,x) \tag{2.10}$$

and

$$\sum_{i,j=1}^{N} \langle K(x_i,x_j)y_j, y_i\rangle_{\mathcal{Y}} \geqslant 0, \tag{2.11}$$

for all $N \in \mathbb{N}$ and for all sequences of points $(x_i)_{i=1}^N$ in $\mathcal{X}^N$, and all sequences of points $(y_i)_{i=1}^N$ in $\mathcal{Y}^N$.

As in the scalar case any $\mathcal{Y}$-Reproducing Kernel Hilbert Space defines a unique Operator-Valued Kernel of positive type and conversely an Operator-Valued Kernel of positive type defines a unique $\mathcal{Y}$-Reproducing Kernel Hilbert Space.

Proposition 2.1 *Given a $\mathcal{Y}$-Reproducing Kernel Hilbert Space there is a unique Operator-Valued Kernel $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ of positive type.*

Proof *Given $x \in \mathcal{X}$, equation 2.8 ensure that the evaluation map at x defined as*

$$ev_x : \begin{cases} \mathcal{H} \to \mathcal{Y} \\ f \mapsto f(x) \end{cases}$$

is a bounded operator and the Operator-Valued Kernel K associated to $\mathcal{H}$ is defined as

$$K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y}) \qquad K(x,z) = ev_x ev_z^*.$$

Since for all $(x_i)_{i=1}^N$ in $\mathcal{X}^N$ and all $(y_i)_{i=1}^N$ in $\mathcal{Y}^N$,

$$\begin{aligned} \sum_{i,j=1}^{N} \langle K(x_i,x_j)y_j, y_i\rangle_{\mathcal{Y}} &= \sum_{i,j=1}^{N} \langle ev_{x_j}^* y_j, ev_{x_i}^* y_i\rangle_{\mathcal{Y}} \\ &= \left\langle \sum_{i=1}^{N} ev_{x_i}^* y_i, \sum_{i=1}^{N} ev_{x_i}^* y_i \right\rangle_{\mathcal{Y}} \\ &= \left\| \sum_{i=1}^{N} ev_{x_i}^* y_i \right\|_{\mathcal{Y}} \geqslant 0, \end{aligned}$$

the map K is of positive type. □

Given $x \in \mathcal{X}$, $K_x : \mathcal{Y} \to \mathcal{F}(\mathcal{X};\mathcal{Y})$ denotes the linear operator whose action on a vector y is the function $K_x y \in \mathcal{F}(\mathcal{X};\mathcal{Y})$ defined for all $z \in \mathcal{X}$ by $K_x = ev_x^*$. As a consequence we have that

$$K(x,z)y = ev_x ev_z^* y = K_x^* K_z y = (K_z y)(x). \tag{2.12}$$

Some direct consequences follows from the definition.

1. The kernel reproduces the value of a function $f \in \mathcal{H}$ at a point $x \in \mathcal{X}$ since for all $y \in \mathcal{Y}$ and $x \in \mathcal{X}$, $\mathrm{ev}_x^* y = K_x y = K(\cdot, x)y$ so that

$$\langle f(x), y\rangle_{\mathcal{Y}} = \langle f, K(\cdot, x)y\rangle_{\mathcal{H}} = \langle K_x^* f, y\rangle_{\mathcal{Y}}. \tag{2.13}$$

2. The set $\{ K_x y \mid \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \}$ is total in $\mathcal{H}$. Namely,

$$\left(\bigcup_{x\in\mathcal{X}} \operatorname{Im} K_x \right)^{\perp} = \{ 0 \}.$$

   If $f \in (\cup_{x\in\mathcal{X}} \operatorname{Im} K_x)^{\perp}$, then for all $x \in \mathcal{X}$, $f \in (\operatorname{Im} K_x)^{\perp} = \operatorname{Ker} K_x^*$, hence $f(x) = 0$ for all $x \in \mathcal{X}$ that is $f = 0$.

3. Finally for all $x \in \mathcal{X}$ and all $f \in \mathcal{H}$, $\|f(x)\|_{\mathcal{Y}} \leqslant \sqrt{\|K(x,x)\|_{\mathcal{Y},\mathcal{Y}}}\|f\|_{\mathcal{H}}$. This come from the fact that $\|K_x\|_{\mathcal{Y},\mathcal{H}} = \|K_x^*\|_{\mathcal{H},\mathcal{Y}} = \sqrt{\|K(x,x)\|_{\mathcal{Y},\mathcal{Y}}}$ and the operator norm is sub-multiplicative.

Additionally given an Operator-Valued Kernel of positive type, it defines a unique $\mathcal{Y}$-RKHS.

Proposition 2.2 *Given an Operator-Valued Kernel of positive type there* $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$*, there is a unique* $\mathcal{Y}$*-Reproducing Kernel Hilbert Space* $\mathcal{H}$ *on* $\mathcal{X}$ *with reproducing kernel* K*.*

Proof *Let* $K_{x,y} = K(\cdot, x)y \in \mathcal{F}(\mathcal{X}; \mathcal{Y})$ *and let*

$$\mathcal{H}_0 = \textit{span}\{ K_{x,y} \mid \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \} \subset \mathcal{F}(\mathcal{X}; \mathcal{Y}).$$

If $f = \sum_{i=1}^N c_i K_{x_i,y_i}$ *and* $g = \sum_{i=1}^N d_i K_{z_i,y_i'}$ *are elements of* $\mathcal{H}_0$ *we have that*

$$\sum_{j=1}^N \overline{d_j}\langle f(z_j), y_j'\rangle_{\mathcal{Y}} = \sum_{i,j=1}^N c_i\overline{d_j}\langle K(z_j, x_i)y_i, y_j'\rangle_{\mathcal{Y}} = \sum_{i=1}^N c_i\langle y_i, g(x_i)\rangle_{\mathcal{Y}},$$

so that the sesquilinear form on $\mathcal{H}_0 \times \mathcal{H}_0$

$$\langle f, g\rangle_{\mathcal{H}_0} = \sum_{i,j=1}^N c_i\overline{d_j}\langle K(z_j, x_i)y_i, y_j'\rangle_{\mathcal{Y}}$$

is well defined. Since K *is an Operator-Valued Kernel of positive type, we have for all* $f \in \mathcal{H}_0$ *that* $\langle f, f\rangle \geqslant 0$*. Moreover the sesquilinear form is positive, so that it is also Hermitian. Choosing* $g = K_{x,y}$ *in the above definition yields for all* $x \in \mathcal{X}$*, all* $f \in \mathcal{H}_0$ *and all* $y \in \mathcal{Y}$

$$\langle f, K_{x,y}\rangle_{\mathcal{H}_0} = \langle f(x), y\rangle_{\mathcal{Y}}.$$

Besides if $f \in \mathcal{H}_0$ *for all unitary vector* $y \in \mathcal{Y}$*, by the Cauchy-Schwartz inequality we have*

$$\begin{aligned}|\langle f(x), y\rangle_{\mathcal{Y}}| &= \left|\langle f, K_{x,y}\rangle_{\mathcal{H}_0}\right| \leqslant \sqrt{\langle f, f\rangle_{\mathcal{H}_0}}\sqrt{\langle K_{x,y}, K_{x,y}\rangle_{\mathcal{Y}}} \\ &= \sqrt{\langle f, f\rangle_{\mathcal{H}_0}}\sqrt{\langle K(x,x)y, y\rangle_{\mathcal{Y}}} \leqslant \sqrt{\langle f, f\rangle_{\mathcal{H}_0}}\sqrt{\|K(x,x)\|_{\mathcal{Y},\mathcal{Y}}},\end{aligned}$$

which implies that

$$\|f(x)\|_{\mathcal{Y}} \leqslant \|f\|_{\mathcal{H}_0} \sqrt{\|K(x,x)\|_{\mathcal{Y},\mathcal{Y}}}$$

Therefore if $\langle f, f\rangle_{\mathcal{H}_0}$ *then* $f = 0$. *Eventually we deduce that* $\langle\cdot,\cdot\rangle_{\mathcal{H}_0}$ *is a scalar product on* $\mathcal{H}_0$. *Hence* $\mathcal{H}_0$ *is a pre-Hilbert space. To make it a (complete) Hilbert space we need to take the completion of this space. Let* $\mathcal{H}$ *be the completion of* $\mathcal{H}_0$. *Moreover let* $K_x : \mathcal{Y} \to \mathcal{H}$ *where* $K_x y = K_{x,y}$. *By construction* K_x *is bounded and let* $W : \mathcal{H} \to \mathcal{F}(\mathcal{X};\mathcal{Y})$ *where* $(Wf)(x) = K_x^* f$. *The operator* W *is injective. Indeed if* $Wf = 0$ *then for all* $x \in \mathcal{X}$, $f \in$ *Ker* $K_x^* =$ (*Im* $f)^{\perp}$. *since the set* $\cup_{x\in\mathcal{X}}$ *Im* $K_x = \{ K_x y \mid \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \}$ *generates by definition* $\mathcal{H}_0$, *we have* $f = 0$. *Besides, as* W *is injective, we have for all* $f_1, f_2 \in \mathcal{H}_0$ $(Wf_1)(x) = (Wf_2)(x) \Longrightarrow f_1(x) = f_2(x)$ *pointwise in* $\mathcal{H}$ *so that we can identify* $\mathcal{H}$ *with a subspace of* $\mathcal{F}(\mathcal{X};\mathcal{Y})$. *Hence* $K_x^* f = (Wf)(x) = f(x) = ev_x f$, *showing that* $\mathcal{H}$ *is a* $\mathcal{Y}$*-Reproducing Kernel Hilbert Space with reproducing kernel*

$$K_{\mathcal{H}}(x,z)y = (ev_z^* y)(x) = K(x,z)y.$$

The uniqueness of $\mathcal{H}$ *comes from the uniqueness of the completion of* $\mathcal{H}_0$ *up to an isometry.* □

The above theorem holds also if $\mathcal{Y}$ is a real vector space provided we add the assumption that K is symmetric, i. e. $K(x,z) = K(z,x)$.

Since an Operator-Valued Kernel of positive type defines a unique $\mathcal{Y}$-RKHS and conversely a $\mathcal{Y}$-RKHS defines a unique Operator-Valued Kernel, we denotes the Hilbert space $\mathcal{H}$ endowed with the scalar product $\langle\cdot,\cdot\rangle$ respectively $\mathcal{H}_K$ and $\langle\cdot,\cdot\rangle_K$. From now we refer to Operator-Valued Kernel of positive type as Operator-Valued Kernel whether they act on complex or real Hilbert spaces. As a consequence, given K an Operator-Valued Kernel, define $K_x = K(\cdot, x)$ we have

$$K(x,z) = K_x^* K_z \quad \forall x, z \in \mathcal{X}, \tag{2.14a}$$

$$\mathcal{H}_K = \overline{\text{span}} \{ K_x y \mid \forall x \in \mathcal{X}, \ \ \forall y \in \mathcal{Y} \}. \tag{2.14b}$$

Another way to describe functions of $\mathcal{H}_K$ consists in using a suitable feature map.

Proposition 2.3 (Feature Operator [13]). *Let* $\mathcal{H}$ *be any Hilbert space and* $\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{Y};\mathcal{H})$, *with* $\Phi_x := \Phi(x)$. *Then the operator* $W : \mathcal{H} \to \mathcal{F}(\mathcal{X};\mathcal{Y})$ *defined for all* $g \in \mathcal{H}$, *and for all* $x \in \mathcal{X}$ *by* $(Wg)(x) = \Phi_x^* g$ *is a partial isometry from* $\mathcal{H}$ *onto the* $\mathcal{Y}$*-RKHS* $\mathcal{H}_K$ *with reproducing kernel*

$$K(x,z) = \Phi_x^* \Phi_z, \quad \forall x, z \in \mathcal{X}.$$

W^*W *is the orthogonal projection onto*

$$(Ker\ W)^{\perp} = \overline{\text{span}} \{ \Phi_x y \mid \forall x \in \mathcal{X}, \ \ \forall y \in \mathcal{Y} \}.$$

Then

$$\|f\|_K = \inf\{ \|g\|_{\mathcal{H}} \mid \forall g \in \mathcal{H}, \ \ Wg = f \}. \tag{2.15}$$

Proof *The operator* $(Wg)(x) = \Phi(x)^* g$ *ensure that the nullspace of* W *is* $\mathcal{N} = Ker\ W = \cap_{x\in\mathcal{X}} Ker\ \Phi(x)^*$. *Since* $\Phi(x)$ *is bounded,* $\Phi(x)$ *is a continuous operator, thus for all* $x \in \mathcal{X}$, *Ker* $\Phi(x)^*$ *is closed so that* $\mathcal{N}$ *is closed. Moreover,*

$$\mathcal{N} = Ker\ W = \bigcap_{x\in\mathcal{X}} Ker\ \Phi(x)^* = \bigcap_{x\in\mathcal{X}} (Im\ \Phi(x))^\perp = \left(\bigcup_{x\in\mathcal{X}} Im\ \Phi(x)\right)^\perp$$

So that $\mathcal{N}^\perp = \cup_{x\in\mathcal{X}} Im\ \Phi(x)$ *and the restriction of* W *to* $\mathcal{N}^\perp$ *is injective.*

Let $\mathcal{H}_K = Im\ W$ *be a vector space. Define the unique inner product on* $\mathcal{H}_K$ *such that* W *becomes a partial isometry from* $\mathcal{H}$ *onto* $\mathcal{H}_K$. *We call this new partial isometry (again)* W. *We show that* $\mathcal{H}_K$ *is a* $\mathcal{Y}$*-Reproducing Kernel Hilbert Space. Since* W^*W *is a projection on* $\mathcal{N}^\perp$, *given* $f \in \mathcal{H}_K$, *where* $f = Wg$ *and* $g \in \mathcal{N}^\perp$ *we have for all* $x \in \mathcal{X}$

$$f(x) = (Wg)(x) = \Phi(x)^* g = \Phi(x)^* W^* W g = (W\Phi(x))^* f.$$

Since Ker W *is closed,* W *is bounded, and* $\Phi(x)$ *is bounded by definition so that the evaluation map*

$$ev_x = (W\Phi(x))^*$$

is bounded so continuous. Then the reproducing kernel is given for all $x, z \in \mathcal{X}$ *by*

$$K(x,z) = ev_x ev_z^* = (W\Phi(x))^*(W\Phi(z)) = \Phi(x)^* W^* W \Phi(z) = \Phi(x)^*\Phi(z),$$

Since W^*W *is the identity on Im* $\Phi(z)$. *Hence* $\mathcal{H}_K$ *is a* $\mathcal{Y}$*-RKHS (see proof of proposition 2.1).* □

We call Φ a *feature map*, W a *feature operator* and $\mathcal{H}$ a *feature space*. Since W is an isometry from $(\text{Ker } W)^\perp$ onto $\mathcal{H}_K$, the map W allows us to identify $\mathcal{H}_K$ with the closed subspace $(\text{Ker } W)^\perp$ of $\mathcal{H}$. Notice that W is a partial isometry, meaning that there can exist multiple functions $g \in \mathcal{H}$, the redescription space, such that $Wg = f$ where f is a function of the $\mathcal{Y}$-RKHS $\mathcal{H}_K$. However equation 2.15 shows that there is a unique function $g \in \mathcal{H}$ such that $Wg = f$, *and* $\|g\|_{\mathcal{H}} = \|f\|_{\mathcal{H}_K}$. Among all functions $g \in \mathcal{H}$ such that $Wg = f$, the only one making the norm in the $\mathcal{Y}$-RKHS and the redescription space is the one with minimal norm.

In this work we mainly focus on the class kernels inducing a $\mathcal{Y}$-RKHS of continuous functions. Such kernels are named $\mathcal{Y}$-Mercer kernels.

Definition 2.7 ($\mathcal{Y}$-Mercer kernel). *A reproducing kernel* $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ *is called* $\mathcal{Y}$*-Mercer kernel if* $\mathcal{H}_K$ *is a subspace of* $\mathcal{C}(\mathcal{X}; \mathcal{Y})$.

The following proposition characterize $\mathcal{Y}$-Mercer kernel in terms of the properties of a kernels rather than properties of the $\mathcal{Y}$-RKHS.

Proposition 2.4 (Characterization of $\mathcal{Y}$-Mercer kernel [13]). *Let* K *be a reproducing kernel. The kernel* K *is Mercer if and only if the function* $x \mapsto \|K(x,x)\|$ *is locally bounded and for all* $x \in \mathcal{X}$ *and all* $y \in \mathcal{Y}$, $K_x y \in \mathcal{C}(\mathcal{X}; \mathcal{Y})$.

Proof *If* $\mathcal{H}_K \subset \mathcal{C}(\mathcal{X};\mathcal{Y})$*, then for all* $x \in \mathcal{X}$ *and all* $y \in \mathcal{Y}$*,* $K_x y$ *is an element of* $\mathcal{C}(\mathcal{X};\mathcal{Y})$ *(see equation 2.14b). In addition for all* $f \in \mathcal{H}_K$*,* $\|K_x^* f\| = \|f(x)\| \leqslant \|f\|_\infty$*. Hence there exist a constant* $M \in \mathbb{R}_+$ *such that for all* $x \in \mathcal{X}$*,* $\|K_x\| \leqslant M$*. Therefore from equation 2.14a, for all* $x \in \mathcal{X}$*,* $\|K(x,x)\| = \|K_x^*\|^2 \leqslant M^2$*. Conversely assume that the function* $x \mapsto \|K(x,x)\|$ *is locally bounded and* $K_x y \in \mathcal{C}(\mathcal{X};\mathcal{Y})$*. For all* $f \in \mathcal{H}_K$ *and all* $x \in \mathcal{X}$*,*

$$\|f(x)\| = \|f\|_K \sqrt{\|K(x,x)\|} \leqslant M\|f\|_K.$$

Thus convergence in $\mathcal{H}_K$ *implies uniform convergence. Since by assumption*

$$\{ K_x t \mid \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \} \subset \mathcal{C}(\mathcal{X};\mathcal{Y}),$$

then the $\mathcal{Y}$*-Reproducing Kernel Hilbert Space*

$$\mathcal{H}_K = \overline{\text{span}} \{ K_x y \mid \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \} \subset \mathcal{C}$$

is also a subset of $\mathcal{C}(\mathcal{X};\mathcal{Y})$ *by the uniform convergence theorem.* □

Lemma 2.1 *Let* $\mathcal{H}_K$ *be a* $\mathcal{Y}$*-Reproducing Kernel Hilbert Space of continuous function* $f : \mathcal{X} \to \mathcal{Y}$*. If* $\mathcal{X}$ *and* $\mathcal{Y}$ *are separable then* $\mathcal{H}_K$ *is separable.*

Proof *The separability of* $\mathcal{X}$ *assure that there exist a countable dense subset* $\mathcal{X}_0 \subseteq \mathcal{X}$*. Since* $\mathcal{Y}$ *is separable,*

$$\mathcal{S} = \bigcup_{x \in \mathcal{X}_0} \mathit{Im}\ K_x = \{ K_x y \mid \forall x \in \mathcal{X}_0, \forall y \in \mathcal{Y} \} \subset \mathcal{H}_K$$

is separable too. We show that $\mathcal{S}$ *is total in* $\mathcal{H}_K$ *so that* $\mathcal{H}_K$ *is separable. If for all* $x \in \mathcal{X}_0$*,* $f \in \mathcal{S}^\perp$*, then* $f \in \mathit{Ker}\ K_x^*$*. Namely* $f(x) = ev_x f = 0$*. Since* f *is continuous and* $\mathcal{X}_0$ *is dense in* $\mathcal{X}$*, for all* $x \in \mathcal{X}$*,* $f(x) = 0$ *so* $f = 0$*.* □

Proposition 2.5 (Countable orthonormal basis for $\mathcal{Y}$-Mercer kernel [12]). *Let* $K : \mathcal{X} \times \mathcal{X} \to \mathcal{Y}$ *be a reproducing kernel where* $\mathcal{X}$ *and* $\mathcal{Y}$ *are separable spaces. If* K *is a* $\mathcal{Y}$*-Mercer kernel then* $\mathcal{H}_K$ *has a countable orthonormal basis.*

Proof *From proposition 2.4* K *is a* $\mathcal{Y}$*-Mercer kernel if and only if* $\mathcal{H}_K \subset \mathcal{C}(\mathcal{X};\mathcal{Y})$*. Applying lemma 2.1 of [12], we have that* $\mathcal{H}_K$ *is separable. Thus since* $\mathcal{H}_K$ *is also a Hilbert space it has a countable orthonormal basis.* □

An important consequence is that if K is a $\mathcal{Y}$-Mercer and $\mathcal{X}$ and $\mathcal{Y}$ are separable then $\mathcal{H}_K$ and any resdescription is isometrically isomorphic to ℓ^2.

2.6.2 *Shift-Invariant operator-valued kernels*

The main subjects of interest of chapter 3 are shift-invariant Operator-Valued Kernel. When referring to a shift-invariant OVK $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ we assume that $\mathcal{X}$ is a locally compact second countable topological group with identity e.

Definition 2.8 (Shift-invariant OVK). *A reproducing Operator-Valued Kernel* $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ *is called shift-invariant*[4] *if for all* $x, z, t \in \mathcal{X}$,

$$K(x \star t, z \star t) = K(x, z). \tag{2.16}$$

[4] *Also referred to as translation-invariant OVK.*

A shift-invariant kernel can be characterize by a function of one variable K_e called the signature of K. Here e denotes the neutral element of the LCA group $\mathcal{X}$ endowed with the operator $\star$.

We recall the definition of left regular representation of $\mathcal{X}$ acting on $\mathcal{H}_K$ which is useful to study LCA groups. For all $x, z \in \mathcal{X}$ and for all $f \in \mathcal{H}_K$,

$$(\lambda_z f)(x)G := f(z^{-1} \star x).$$

A group representation λ_z describe the group by making it act on a vector space (here $\mathcal{H}_K$) in a linear manner. In other word, the group representation let us see a group as a linear operator which are well studied mathematical object.

Proposition 2.6 (Kernel signature [13]). *Let* $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ *be a reproducing kernel. The following conditions are equivalents.*

1. K *is a shift-invariant of positive type.*
2. *There is a function* $K_e : \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ *of completely positive type such that* $K(x, z) = K_e(z^{-1} \star x)$.

If one of the above conditions is satisfied, then the representation λ *leaves invariant* $\mathcal{H}_K$*, its action on* $\mathcal{H}_K$ *is unitary and*

$$K(x, z) = K_e^* \lambda_{x^{-1} \star z} K_e \qquad \forall (x, z) \in \mathcal{X}^2. \tag{2.17a}$$
$$\|K(x, x)\| = \|K_e(e)\| \qquad \forall x \in \mathcal{X} \tag{2.17b}$$

Proof *Assume* **??** *10 holds true. Given* $x, z \in \mathcal{X}$*, equation 2.12 and equation 2.16 yields*

$$K_e(z^{-1} \star x) = K(z^{-1} \star x, e) = K(x, z).$$

Since K *is a reproducing kernel,* K_e *is of completely positive type, so that* **??** *11 holds true. Besides if* **??** *11 holds true obviously the definition of a reproducing kernel (definition 2.5) is fulfilled so that* **??** *10 holds true.*

Suppose that K *is a shift-invariant reproducing kernel. Given* $t \in \mathcal{X}$ *and* $y \in \mathcal{Y}$*, for all* $x, z \in \mathcal{X}$,

$$(\lambda_x K_t y)(z) = (K_t y)(x^{-1} \star z) = K(x^{-1} \star z, t) = K(z, x \star t) = (K_{x \star t} y)z,$$

that is $\lambda_x K_t = K_{x \star t}$*. Besides for all* $y, y' \in \mathcal{Y}$ *and all* $x, z, t, t' \in \mathcal{X}$,

$$\begin{aligned} \langle \lambda_x K_t y, \lambda_x K_{t'} y' \rangle_K &= \langle K_{x \star t} y, K_{x \star t'} y' \rangle_K = \langle K(x \star t', x \star t) y, y' \rangle \\ &= \langle K(t', t) y, y' \rangle = \langle K_t y, K_{t'} y' \rangle_K \end{aligned}$$

This means that λ *leaves the set* $\{ K_x y \mid \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \}$ *invariant. Since*

$$\{ K_x y \mid \forall x \in \mathcal{X}, \forall y \in \mathcal{Y} \}$$

is total in $\mathcal{H}_K$ *(see equation 2.14b),* λ *is surjective and because it also leaves the inner product invariant, the firs two claims follow.* □

The notation K_e for the function of completely positive type associated with the reproducing kernel K is consistent with the definition given by equation 2.12 since for all $x \in \mathcal{X}$ and all $y \in \mathcal{Y}$

$$(K_e y)(x) = K_e(x)y.$$

2.6.3 *Examples of operator-valued kernels*

Operator-valued kernels have been first introduced in Machine Learning to solve multi-task regression problems. Multi-task regression is encountered in many fields such as structured classification when classes belong to a hierarchy for instance. Instead of solving independently p single output regression task, one would like to take advantage of the relationships between output variables when learning and making a decision.

Definition 2.9 (Decomposable kernel). *Let Γ be a non-negative operator of $\mathcal{L}_+(\mathcal{Y})$. K is said to be a $\mathcal{Y}$-Mercer decomposable kernel*[5] *if for all $(x, z) \in \mathcal{X}^2$,*

$$K(x, z) := k(x, z)\Gamma,$$

where k is a scalar Mercer kernel.

[5] *Some authors also refer to as separable kernels.*

When $\mathcal{Y} = \mathbb{R}^p$, the matrix Γ is interpreted as encoding the relationships between the outputs coordinates. If a graph coding for the proximity between tasks is known, then it is shown in Álvarez, Rosasco, and Lawrence [2], Baldassarre et al. [3], and Evgeniou, Micchelli, and Pontil [16] that Γ can be chosen equal to the pseudo inverse $L^\dagger$ of the graph Laplacian such that the norm in $\mathcal{H}_K$ is a graph-regularizing penalty for the outputs (tasks). When no prior knowledge is available, Γ can be set to the empirical covariance of the output training data or learned with one of the algorithms proposed in the literature [15, 26, 40]. Another interesting property of the decomposable kernel is its universality (a kernel which may approximate an arbitrary continuous target function uniformly on any compact subset of the input space). A reproducing kernel K is said *universal* if the associated $\mathcal{Y}$-RKHS $\mathcal{H}_K$ is dense in the space $\mathcal{C}(\mathcal{X}, \mathcal{Y})$. The conditions for a kernel to be universal have been discussed in Caponnetto et al. [11] and Carmeli et al. [13]. In particular they show that a decomposable kernel is universal provided that the scalar kernel k is universal and the operator Γ is injective.

Proposition 2.7 (Kernels and Regularizers [2]). *Let $K(x, z) := k(x, z)\Gamma$ for all $x, z \in \mathcal{X}$ be a decomposable kernel where Γ is a matrix of size $p \times p$. Then for all $f \in \mathcal{H}_K$,*

$$\|f\|_K = \sum_{i,j=1}^{p} \Gamma^\dagger_{ij} \langle f_i, f_j \rangle_k \tag{2.18}$$

where $f_i = \langle f, e_i \rangle$ (resp $f_j = \langle f, e_j \rangle$), denotes the i-th (resp j-th) component of f.

Curl-free and divergence-free kernels provide an interesting application of operator-valued kernels [4, 27, 29] to *vector field* learning, for which input and output spaces have the same dimensions ($d = p$). Applications cover shape deformation analysis [29] and magnetic fields approximations [49]. These kernels discussed in [18] allow encoding input-dependent similarities between vector-fields.

Definition 2.10 (Curl-free and Div-free kernel). *Assume* $\mathcal{X} = (\mathbb{R}^d, +)$ *and* $\mathcal{Y} = \mathbb{R}^p$ *with* $d = p$. *The divergence-free kernel is defined as*

$$K^{div}(x, z) = K_0^{div}(\delta) = (\nabla\nabla^T - \Delta I)k_0(\delta)$$

and the curl-free kernel as

$$K^{curl}(x, z) = K_0^{curl}(\delta) = -\nabla\nabla^T k_0(\delta),$$

where ∇ *is the gradient operator*[6] , $\nabla\nabla^T$ *is the Hessian operator and* Δ *is the Laplacian operator.*

[6] *See section 4.2.1 for a formal definition of the operator* ∇.

Although taken separately these kernels are not universal, a convex combination of the curl-free and divergence-free kernels allows to learn any vector field that satisfies the Helmholtz decomposition theorem [4, 27].

Part II

CONTRIBUTIONS

You can put some informational part preamble text here. Illo principalmente su nos. Non message *occidental* angloromanic da. Debitas effortio simplificate sia se, auxiliar summarios da que, se avantiate publicationes via. Pan in terra summarios, capital interlingua se que. Al via multo esser specimen, campo responder que da. Le usate medical addresses pro, europa origine sanctificate nos se.

3

OPERATOR-VALUED RANDOM FOURIER FEATURES

3.1 MOTIVATIONS

Random Fourier Features have been proved useful to implement efficiently kernel methods in the scalar case, allowing to learn a linear model based on an approximated feature map. In this work, we are interested to construct approximated operator-valued feature maps to learn vector-valued functions. With an explicit (approximated) feature map, one converts the problem of learning a function f in the vector-valued Reproducing Kernel Hilbert Space $\mathcal{H}_K$ into the learning of a linear model $\tilde{f}$ defined by:

$$\tilde{f}(x) = \widetilde{\Phi}(x)^* \theta,$$

where $\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{H}, \mathcal{Y})$ and $\theta \in \mathcal{H}$. The methodology we propose works for operator-valued kernels defined on any Locally Compact Abelian (LCA) group, noted $(\mathcal{X}, \star)$, for some operation noted $\star$. This allows us to use the general context of Pontryagin duality for Fourier transform of functions on LCA groups. Building upon a generalization of Bochner's theorem for operator-valued measures, an operator-valued kernel is seen as the *Fourier transform* of an operator-valued positive measure. From that result, we extend the principle of Random Fourier Feature for scalar-valued kernels and derive a general methodology to build Operator Random Fourier Feature when operator-valued kernels are shift-invariant according to the chosen group operation. Elements of this chapter have been developped in Brault, Heinonen, and Buc [7].

We present a construction of Operator-valued Random Fourier Feature (ORFF) such that $f : x \mapsto \widetilde{\Phi}(x)^* \theta$ is a continuous function that maps an arbitrary LCA group $\mathcal{X}$ as input space to an arbitrary output Hilbert space $\mathcal{Y}$. First we define a functional *Fourier feature map*, and then propose a Monte-Carlo sampling from this feature map to construct an approximation of a shift-invariant $\mathcal{Y}$-Mercer kernel. Then, we prove the convergence of the kernel approximation $\tilde{K}(x, z) = \widetilde{\Phi}(x)^* \widetilde{\Phi}(z)$ with high probability on *compact* subsets of the LCA $\mathcal{X}$, when $\mathcal{Y}$ is *finite dimensional*. Eventually we conclude with some numerical experiments.

3.2 THEORETICAL STUDY

The following proposition of Carmeli et al. [13] and Zhang, Xu, and Zhang [52] extends Bochner's theorem to any shift-invariant $\mathcal{Y}$-Mercer kernel.

Proposition 3.1 (Operator-valued Bochner's theorem [34, 52]). *If a continuous function K from $\mathcal{X} \times \mathcal{X}$ to $\mathcal{Y}$ is a shift-invariant $\mathcal{Y}$-Mercer kernel on $\mathcal{X}$, then there exists a unique positive projection-valued measure $\widehat{Q} : \mathcal{B}(\mathcal{X}) \to \mathcal{L}_+(\mathcal{Y})$ such that for all $x, z \in \mathcal{X}$,*

$$K(x, z) = \int_{\widehat{\mathcal{X}}} \overline{(x \star z^{-1}, \omega)} \, d\widehat{Q}(\omega), \tag{3.1}$$

where $\widehat{Q}$ *belongs to the set of all the projection-valued measures of bounded variation on the* σ*-algebra of Borel subsets of* $\widehat{\mathcal{X}}$*. Conversely, from any positive operator-valued measure* M*, a shift-invariant kernel* K *can be defined by proposition 3.1.*

Although this theorem is central to the spectral decomposition of shift-invariant $\mathcal{Y}$-Mercer OVK, the following results proved by Carmeli et al. [13] provides insights about this decomposition that are more relevant in practice. It first gives the necessary conditions to build shift-invariant $\mathcal{Y}$-Mercer kernel with a pair $(A, \widehat{\mu})$ where A is an operator-valued function on $\widehat{\mathcal{X}}$ and $\widehat{\mu}$ is a real-valued positive measure on $\widehat{\mathcal{X}}$. Note that obviously such a pair is not unique ad the choice of this paper may have an impact on theoretical properties as well as practical computations. Secondly it also states that any OVK have such a spectral decomposition when $\mathcal{Y}$ is finite dimensional or $\mathcal{X}$.

Proposition 3.2 (Carmeli et al. [13]). *Let* $\widehat{\mu}$ *be a positive measure on* $\mathcal{B}(\widehat{\mathcal{X}})$ *and* $A : \widehat{\mathcal{X}} \to \mathcal{L}(\mathcal{Y})$ *such that* $\langle A(\cdot)y, y' \rangle \in L^1(\mathcal{X}, \widehat{\mu})$ *for all* $y, y' \in \mathcal{Y}$ *and* $A(\omega) \succcurlyeq 0$ *for* $\widehat{\mu}$*-almost all* $\omega \in \widehat{\mathcal{X}}$*. Then, for all* $\delta \in \mathcal{X}$*,*

$$K_e(\delta) = \int_{\widehat{\mathcal{X}}} \overline{(\delta, \omega)} A(\omega) d\widehat{\mu}(\omega) \tag{3.2}$$

is the kernel signature of a shift-invariant $\mathcal{Y}$*-Mercer kernel* K *such that* $K(x, z) = K_e(x \star z^{-1})$*. The* $\mathcal{Y}$*-RKHS* $\mathcal{H}_K$ *is embed in* $L^2(\widehat{\mathcal{X}}, \widehat{\mu}; \mathcal{Y}')$ *by mean of the feature operator*

$$(Wg)(x) = \int_{\widehat{\mathcal{X}}} \overline{(x, \omega)} B(\omega) g(\omega) d\widehat{\mu}(\omega), \tag{3.3}$$

Where $B(\omega)B(\omega)^* = A(\omega)$ *and both integral converges in the weak sense. If* $\mathcal{Y}$ *is finite dimensional or* $\mathcal{X}$ *is compact, any shift-invariant kernel is of the above form for some pair* $(A, \widehat{\mu})$*.*

When $p = 1$ one can always assume A is reduced to the scalar 1, $\widehat{\mu}$ is still a bounded positive measure and we retrieve the Bochner theorem applied to the scalar case (**??**).

Proposition 3.2 shows that a given pair $(A, \widehat{\mu})$ characterize an OVK. Namely given a measure $\widehat{\mu}$ and a function A such that $\langle A(.)y, y' \rangle \in L^1(\mathcal{X}, \widehat{\mu})$ for all $y, y' \in \mathcal{Y}$ and $A(\omega) \succcurlyeq 0$ for $\widehat{\mu}$-almost all ω, it gives rise to an OVK. Since $(A, \widehat{\mu})$ determine a unique kernel we can write $\mathcal{H}_{(A,\widehat{\mu})} \Longrightarrow \mathcal{H}_K$ where K is defined as in equation 3.2. However the converse is to true: Given a $\mathcal{Y}$-Mercer shift invariant Operator-Valued Kernel, there exist infinitely many pairs $(A, \widehat{\mu})$ that characterize an OVK.

The main difference between proposition 3.1 and proposition 3.2 is that the first one characterize an OVK by a unique Positive Operator-Valued Measure (POVM), while the second one shows that the POVM that uniquely characterize a $\mathcal{Y}$-Mercer OVK has an operator-valued density with respect to a *scalar* measure $\widehat{\mu}$; and that this operator-valued density is not unique.

Finally proposition 3.2 does not provide any *constructive* way to obtain the pair $(A, \widehat{\mu})$ that characterize an OVK. The following section 3.2.1 is based on an other proposition of Carmeli, De Vito, and Toigo and show that if the kernel signature $K_e(\delta)$ of an OVK is in L^1 then it is possible to construct *explicitly* a pair $(C, \widehat{\mathbf{Haar}})$ from it. Additionally, we show that we can always extract a scalar-valued *probability* density function from C such that we obtain a pair $(A, \mathbf{Pr}_{\widehat{\mu},\rho})$ where $\mathbf{Pr}_{\widehat{\mu},\rho}$ is a *probability* distribution absolutely continuous with respect to $\widehat{\mu}$ and with associated probability density function (p. d. f.) ρ. Thus for all $\mathcal{Z} \subset \mathcal{B}(\widehat{\mathcal{X}})$,

$$\mathbf{Pr}_{\widehat{\mu},\rho}(\mathcal{Z}) = \int_{\mathcal{Z}} \rho(\omega) \mathrm{d}\widehat{\mu}(\omega).$$

When the reference measure $\widehat{\mu}$ is the Lebesgue measure, we note $\mathbf{Pr}_{\widehat{\mu},\rho} = \mathbf{Pr}_{\rho}$.

3.2.1 *Sufficient conditions of existence*

While proposition 3.2 gives some insights on how to build an approximation of a $\mathcal{Y}$-Mercer kernel, we need a theorem that provides an explicit construction of the pair $(A, \mathbf{Pr}_{\widehat{\mu},\rho})$ from the kernel signature K_e. Proposition 14 in Carmeli et al. [13] gives the solution, and also provide a sufficient condition for proposition 3.2 to apply.

Proposition 3.3 (Carmeli et al. [13]). *Let K be a shift-invariant $\mathcal{Y}$-Mercer kernel. Suppose that for all $z \in \mathcal{X}$ and for all $y, y' \in \mathcal{Y}$, $\langle K_e(.)y, y'\rangle \in L^1(\mathcal{X}, \mathbf{Haar})$ where $\mathcal{X}$ is endowed with the group law $\star$. For all $\omega \in \widehat{\mathcal{X}}$ and for all y, y' in $\mathcal{Y}$, let*

$$\begin{aligned} \langle y', C(\omega)y\rangle &= \int_{\mathcal{X}} (\delta, \omega)\langle y', K_e(\delta)y\rangle \mathrm{d}\delta \\ &= \mathcal{F}^{-1}\left[\langle y', K_e(\cdot)y\rangle\right](\omega). \end{aligned} \tag{3.4}$$

Then

1. *$C(\omega)$ is a bounded non-negative operator for all $\omega \in \widehat{\mathcal{X}}$,*
2. *$\langle y, C(\cdot)y'\rangle \in L^1(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}})$ for all $y, y' \in \mathcal{X}$,*
3. *for all $\delta \in \mathcal{X}$ and for all y, y' in $\mathcal{Y}$,*

$$\begin{aligned} \langle y', K_e(\delta)y\rangle &= \int_{\widehat{\mathcal{X}}} \overline{(\delta, \omega)}\langle y', C(\omega)y\rangle \mathrm{d}\widehat{\mathbf{Haar}}(\omega) \\ &= \mathcal{F}\left[\langle y', C(\cdot)y\rangle\right](\delta). \end{aligned}$$

There have been a lot of confusion in the literature whether a kernel is the Fourier transform or Inverse Fourier transform of a measure. However lemma 3.1 clarify the relation between the Fourier transform and Inverse Fourier transform for a translation invariant Operator-Valued Kernel. Notice that in the real scalar case the Fourier transform and Inverse Fourier transform of a shift-invariant kernel are the same, while the difference is significant for OVK.

The following lemma is a direct consequence of the definition of $C(\omega)$ as the Fourier transform of the adjoint of K_e and also helps simplifying the definition of ORFF.

Lemma 3.1 *Let K_e be the signature of a shift-invariant $\mathcal{Y}$-Mercer kernel and let $\langle y', C(\cdot)y\rangle = \mathcal{F}^{-1}[\langle y', K_e(\cdot)y\rangle]$ for all $y.\ y' \in \mathcal{Y}$. Then*

1. *$C(\omega)$ is self-adjoint and C is even.*
2. *$\mathcal{F}^{-1}[\langle y', K_e(\cdot)y\rangle] = \mathcal{F}[\langle y', K_e(\cdot)y\rangle]$.*
3. *$K_e(\delta)$ is self-adjoint and K_e is even.*

Proof *For any function f on $(\mathcal{X}, \star)$ define the flip operator $\mathcal{R}$ by*

$$\mathcal{R}f(x) := f\left(x^{-1}\right).$$

For any shift invariant $\mathcal{Y}$-Mercer kernel and for all $\delta \in \mathcal{X}$, $K_e(\delta) = K_e\left(\delta^{-1}\right)^*$. Indeed,

$$\begin{aligned}\mathcal{R}\langle y, K_e\left(x \star z^{-1}\right) y'\rangle &= \langle y, K_e\left(\left(x \star z^{-1}\right)^{-1}\right) y'\rangle \\ &= \langle y, K_e\left(x^{-1} \star z\right) y'\rangle \\ &= \langle y, K_e\left(x \star z^{-1}\right)^* y'\rangle.\end{aligned}$$

?? *15: taking the Fourier transform yields,*

$$\mathcal{F}^{-1}\left[\langle y', K_e(\cdot)y\rangle\right] = \mathcal{F}^{-1}\mathcal{R}\left[\langle y', K_e(\cdot)y\rangle\right] = \mathcal{R}\langle y', C(\cdot)y\rangle.$$

Hence $C(\omega) = C\left(\omega^{-1}\right)^*$. Suppose that $\mathcal{Y}$ is a complex Hilbert space. Since for all $\omega \in \widehat{\mathcal{X}}$, $C(\omega)$ is bounded and non-negative so $C(\omega)$ is self-adjoint. Besides we have $C(\omega) = C\left(\omega^{-1}\right)^*$ to C must be pair. Suppose that $\mathcal{Y}$ is a real Hilbert space. Then we have the additional hypothesis that $K_e(\delta) = K_e(\delta)^*$. Taking the Fourier transform yields that $C(\omega) = C(\omega)^*$. Since for any shift invariant $\mathcal{Y}$-Mercer kernel $C(\omega) = C\left(\omega^{-1}\right)^*$ we also conclude that $C\left(\omega^{-1}\right) = C(\omega)$.

?? *16: simply, for all $y, y' \in \mathcal{Y}$, $\langle y, C(\omega^{-1})y'\rangle = \langle y', C(\omega)y\rangle$ thus $\mathcal{F}^{-1}[\langle y', C(\cdot)y\rangle] = \mathcal{F}\mathcal{R}[\langle y', C(\cdot)y\rangle] = \mathcal{F}[\langle y', C(\cdot)y\rangle]$.*

?? *17: from **??** 16 we have $\mathcal{F}^{-1}[\langle y', K_e(\cdot)y\rangle] = \mathcal{F}^{-1}\mathcal{R}\langle y', K_e(\cdot)y\rangle$. By injectivity of the Fourier transform, K_e is even. Since $K_e(\delta) = K_e(\delta^{-1})^*$, we must have $K_e(\delta) = K_e(\delta)^*$.* □

While proposition 3.3 gives an explicit form of the operator $C(\omega)$ defined as the Fourier transform of the kernel K, it is not really convenient to work with the Haar measure $\widehat{\mathbf{Haar}}$ on $\mathcal{B}(\widehat{\mathcal{X}})$. However it is easily possible to turn $\widehat{\mathbf{Haar}}$ into a probability measure to allow efficient integration over an infinite domain.

The following proposition allows to build a spectral decomposition of a shift-invariant $\mathcal{Y}$-Mercer kernel on a LCA group $\mathcal{X}$ endowed with the group law $\star$ with respect to a scalar probability measure, by extracting a scalar probability density function from C.

Proposition 3.4 (Shift-invariant $\mathcal{Y}$-Mercer kernel spectral decomposition). *Let K_e be the signature of a shift-invariant $\mathcal{Y}$-Mercer kernel. If for all $y, y' \in \mathcal{Y}$, $\langle K_e(.)y, y' \rangle \in L^1(\mathcal{X}, \mathbf{Haar})$ then there exists a positive probability measure $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ and an operator-valued function A an such that for all y, $y' \in \mathcal{Y}$,*

$$\langle y', K_e(\delta)y \rangle = \mathbf{E}_{\widehat{\mathbf{Haar}},\rho}\left[\overline{(\delta,\omega)}\langle y', A(\omega)y\rangle\right], \tag{3.5}$$

with

$$\langle y', A(\omega)y\rangle\rho(\omega) = \mathcal{F}\left[\langle y', K_e(\cdot)y\rangle\right](\omega). \tag{3.6}$$

Moreover

1. *for all $y, y' \in \mathcal{Y}$, $\langle A(.)y, y'\rangle \in L^1(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho})$,*
2. *$A(\omega)$ is non-negative for $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$-almost all $\omega \in \widehat{\mathcal{X}}$,*
3. *$A(\cdot)$ and $\rho(\cdot)$ are even functions.*

Proof *This is a simple consequence of proposition 3.3 and lemma 3.1. By taking $\langle y', C(\omega)y\rangle = \mathcal{F}^{-1}\left[\langle y', K_e(\cdot)y\rangle\right](\omega) = \mathcal{F}\left[\langle y', K_e(\cdot)y\rangle\right](\omega)$ we can write the following equality concerning the OVK signature K_e.*

$$\begin{aligned}\langle y', K_e(\delta)y\rangle(\omega) &= \int_{\widehat{\mathcal{X}}} \overline{(\delta,\omega)}\langle y', C(\omega)y\rangle \mathrm{d}\widehat{\mathbf{Haar}}(\omega) \\ &= \int_{\widehat{\mathcal{X}}} \overline{(\delta,\omega)}\left\langle y', \frac{1}{\rho(\omega)}C(\omega)y\right\rangle \rho(\omega)\mathrm{d}\widehat{\mathbf{Haar}}(\omega).\end{aligned}$$

It is always possible to choose $\rho(\omega)$ such that $\int_{\widehat{\mathcal{X}}}\rho(\omega)\mathrm{d}\widehat{\mathbf{Haar}}(\omega) = 1$. For instance choose

$$\rho(\omega) = \frac{\|C(\omega)\|_{y,y}}{\int_{\widehat{\mathcal{X}}}\|C(\omega)\|_{y,y}\mathrm{d}\widehat{\mathbf{Haar}}(\omega)}$$

Since for all $y, y' \in \mathcal{Y}$, $\langle y', C(\cdot)y\rangle \in L^1(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}})$ and $\mathcal{Y}$ is a separable Hilbert space, by pettis measurability theorem, $\int_{\widehat{\mathcal{X}}}\|C(\omega)\|_{y,y}\mathrm{d}\widehat{\mathbf{Haar}}(\omega)$ is finite and so is $\|C(\omega)\|_{y,y}$ for all $\omega \in \widehat{\mathcal{X}}$. Therefore $\rho(\omega)$ is the density of a probability measure $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$, i. e. conclude by taking

$$\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}(\mathcal{Z}) = \int_{\mathcal{Z}}\rho(\omega)\mathrm{d}\widehat{\mathbf{Haar}}(\omega),$$

for all $\mathcal{Z} \in \mathcal{B}(\widehat{\mathcal{X}})$. □

In the case where $\mathcal{Y} = \mathbb{R}^p$, we rewrite equation 3.6 coefficient-wise by choosing an orthonormal basis $\{ e_j \}_{j \in \mathbb{N}_p^*}$ of $\mathbb{R}^p$.

$$A(\omega)_{ij}\rho(\omega) = \mathcal{F}\left[K_e(\cdot)_{ij}\right](\omega). \tag{3.7}$$

It follows that for all i and j in $\mathbb{N}_p^*$,

$$K_e(x \star z^{-1})_{ij} = \mathcal{F}\left[A(\cdot)_{ij}\rho(\cdot)\right](x \star z^{-1}) \tag{3.8}$$

Remark 3.1 *Note that although the Fourier transform of K_e yields a unique operator-valued function $C(\cdot)$, the decomposition of $C(\cdot)$ into $A(\cdot)\rho(\cdot)$ is again not unique. The choice of the decomposition may be justified by the computational cost or by the nature of the constants involved in the uniform convergence of the estimator.*

Another difficulty arise from the fact that the quantity $\sup_{\omega \in \widehat{\mathcal{X}}} \|A(\omega)\|_{\mathcal{Y},\mathcal{Y}}$ obtained in proposition 3.4 might not bounded. The unboundedness of $\|A(\cdot)\|_{\mathcal{Y},\mathcal{Y}}$ forbid the use of the most simple concentrations inequalities – which require the boundedness of the random variable to be controlled. Therefore in the context of Operator-Valued Kernel concentration inequalities for unbounded random operators should be used. However, as pointed out by Minh [31], under some condition on the trace of $K_e(\delta)$, it is possible to turn $A(\cdot)$ into a bounded random operator for all ω in $\widehat{\mathcal{X}}$. The idea is to define a sum measure $\rho = \sum_{j \in \mathbb{N}^*} \rho_{e_j}$, which gives rise to bounded operator $A(\omega)$ and is idependant of the $\{ e_j \}_{j \in \mathbb{N}^*}$ base, instead of constructing a measure from the operator norm as in proposition 3.4. Additionally with such construction the measure associated to $A(\cdot)$ is *independant* from the basis of $\mathcal{Y}$. In this proof we relax the assumptions of Minh [31] which requires $\int_{\mathcal{X}} |\mathbf{Tr}\, K_e(\delta)| d\mathbf{Haar}(\delta)$ to be well defined. We only require $\mathbf{Tr}\, K_e(e)$ to be well defined.

Proposition 3.5 (Bounded shift-invariant $\mathcal{Y}$-Mercer kernel spectral decomposition). *Let K_e be the signature of a shift-invariant $\mathcal{Y}$-Mercer kernel. If for all y and y' in $\mathcal{Y}$, $\langle K_e(.)y, y' \rangle \in L^1(\mathcal{X}, \mathbf{Haar})$ and $\mathbf{Tr}\, K_e(e) \in \mathbb{R}$, then*

$$\langle y', K_e(\delta)y \rangle = \mathbf{E}_{\widehat{\mathbf{Haar}}, \rho_{\mathbf{Tr}}}\left[\overline{(\delta, \omega)} \langle y, A_{\mathbf{Tr}}(\omega) y' \rangle\right]. \tag{3.9}$$

with

$$\langle y', C(\cdot)y \rangle = \mathcal{F}\left[\langle y', K_e(\cdot)y \rangle\right] \tag{3.10a}$$

$$c_{\mathbf{Tr}} = \mathbf{Tr}\left[K_e(e)\right] \tag{3.10b}$$

$$A_{\mathbf{Tr}}(\omega) = c_{\mathbf{Tr}}\, \mathbf{Tr}\left[C(\omega)\right]^{-1} C(\omega) \tag{3.10c}$$

$$\rho_{\mathbf{Tr}}(\omega) = c_{\mathbf{Tr}}^{-1}\, \mathbf{Tr}\left[C(\omega)\right]. \tag{3.10d}$$

Moreover

1. *For all $y, y' \in \mathcal{Y}$, $\langle y, A_{\mathbf{Tr}}(\cdot) y' \rangle \in L^1(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho_{\mathbf{Tr}}})$.*
2. *$A_{\mathbf{Tr}}(\omega)$ is non-negative for all $\omega \in \widehat{\mathcal{X}}$,*

3. $\sup_{\omega\in\widehat{\mathcal{X}}}\|A_{\mathbf{Tr}}(\omega)\|_{\mathcal{Y},\mathcal{Y}} \leqslant c_p$,
4. $A_{\mathbf{Tr}}(\cdot)$ *and* $\rho_{\mathbf{Tr}}$ *are even functions.*

Proof *Let* $\{ e_j \}_{j\in\mathbb{N}^*}$ *be an orthonormal basis of* $\mathcal{Y}$. *Notice that*

$$\begin{aligned}\int_{\widehat{\mathcal{X}}} \langle e_j, C(\omega)e_j\rangle d\widehat{\mathbf{Haar}}(\omega) &= \int_{\widehat{\mathcal{X}}} \underbrace{\overline{(e,\omega)}}_{=1}\langle e_j, C(\omega)e_j\rangle d\widehat{\mathbf{Haar}}(\omega)\\ &= \langle e_j, K_e(e)e_j\rangle.\end{aligned}$$

Since $C(\omega)$ *is non-negative, all the* $\langle e_j, C(\omega)e_j\rangle$. *Thus using the monotone convergence theorem,*

$$\begin{aligned}\int_{\widehat{\mathcal{X}}} \mathbf{Tr}\left[C(\omega)\right] d\widehat{\mathbf{Haar}}(\omega) &= \int_{\widehat{\mathcal{X}}} \sum_{j\in\mathbb{N}^*} \langle e_j, C(\omega)e_j\rangle d\widehat{\mathbf{Haar}}(\omega)\\ &= \sum_{k\in\mathbb{N}^*} \langle e_j, K_e(e)e_j\rangle\\ &= \mathbf{Tr}\left[K_e(e)\right] = c_{\mathbf{Tr}} < \infty.\end{aligned}$$

Let $A_{\mathbf{Tr}}(\omega)$ *and* $\rho_{\mathbf{Tr}}(\omega)$ *be defined respectively as in equation* 3.10c *and equation* 3.10d. *By definition,* $\int_{\widehat{\mathcal{X}}} \rho_{\mathbf{Tr}}(\omega)d\widehat{\mathbf{Haar}}(\omega) = 1$ *and* $A_{\mathbf{Tr}}(\omega)\rho_{\mathbf{Tr}}(\omega) = C(\omega)$. *Now it remains to check the finiteness of* $\mathbf{Tr}\left[C(\omega)\right]$ *for all* $\omega \in \widehat{\mathcal{X}}$. *Since for all* $\omega \in \widehat{\mathcal{X}}$, $\mathbf{Tr}\left[C(\omega)\right] \geqslant 0$,

$$\mathbf{Tr}\left[C(\omega)\right] \leqslant \int_{\widehat{\mathcal{X}}} \mathbf{Tr}\left[C(\omega)\right] d\widehat{\mathbf{Haar}}(\omega) = \mathbf{Tr}\left[K_e(e)\right] < \infty.$$

Since $\mathbf{Tr}\left[C(\omega)\right]$ *is positive and its integral is finite,* $\rho_{\mathbf{Tr}}$ *is a probability density function. The Schatten norms* $\|\cdot\|_p$ *verifies* $\mathbf{Tr}\left[|\cdot|\right] = \|\cdot\|_1 \geqslant \|\cdot\|_p \geqslant \|\cdot\|_q \geqslant \|\cdot\|_{\mathcal{Y},\mathcal{Y}} = \|\cdot\|_\infty$ *for all* $p, q \in \mathbb{N}^*$ *such that* $1 \leqslant p \leqslant q$. *Therefore since for all* $\omega \in \widehat{\mathcal{X}}$, $C(\omega)$ *is non-negative,*

$$\begin{aligned}\|A_{\mathbf{Tr}}(\omega)\|_{\mathcal{Y},\mathcal{Y}} &= c_{\mathbf{Tr}}\,\mathbf{Tr}\left[C(\omega)\right]^{-1}\|C(\omega)\|_\infty\\ &\leqslant c_{\mathbf{Tr}}\,\mathbf{Tr}\left[C(\omega)\right]^{-1}\|C(\omega)\|_1\\ &= c_{\mathbf{Tr}}\,\mathbf{Tr}\left[C(\omega)\right]^{-1}\mathbf{Tr}\left[|C(\omega)|\right]\\ &= c_{\mathbf{Tr}}\,\mathbf{Tr}\left[C(\omega)\right]^{-1}\mathbf{Tr}\left[C(\omega)\right]\\ &\leqslant c_{\mathbf{Tr}} < \infty.\end{aligned}$$

Thus $\sup_{\omega\in\widehat{\mathcal{X}}}\|A(\omega)\|_{\mathcal{Y},\mathcal{Y}} \leqslant c_{\mathbf{Tr}} < \infty$. *As* C *is an even function, so are* $A_{\mathbf{Tr}}$ *and* $\rho_{\mathbf{Tr}}$. *Eventually* $\langle y', C(\cdot)y\rangle$ *is in* $L^1(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}})$, *thus* $\langle y, A_{\mathbf{Tr}}(\cdot)\rho_{\mathbf{Tr}}(\cdot)y'\rangle$ *is in* $L^1(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}})$, *hence* $\langle y, A_{\mathbf{Tr}}(\cdot)y'\rangle \in L^1(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho_{\mathbf{Tr}}})$. *Since the trace is idenpendent of the basis of* $\mathcal{Y}$, *so is* $\rho_{\mathbf{Tr}}$. □

If $\mathcal{Y}$ is finite dimensional then $\mathbf{Tr}\left[K_e(e)\right]$ is well defined hence proposition 3.5 is valid as long as $K_e(\cdot)_{ij} \in L^1(\mathcal{X}, \mathbf{Haar})$ for all $i, j \in \mathbb{N}_p^*$, where p is the dimension of $\mathcal{Y}$.

3.2.2 *Examples of spectral decomposition*

In this section we give exemple of spectral decomposition of various $\mathcal{Y}$-Mercer kernel, based on proposition 3.4 and proposition 3.5.

3.2.2.1 *Gaussian decomposable kernel*

Recall that a decomposable $\mathbb{R}^p$-Mercer has the form $K(x,z) = k(x,z)\Gamma$, where $k(x,z)$ is a scalar Mercer kernel and $\Gamma \in \mathcal{L}(\mathbb{R}^p)$ is a non-negative operator. Let $K_e^{dec,gauss}(\cdot) = k_e^{gauss}(\cdot)\Gamma$ be the Gaussian decomposable kernel where K_e and k_e are respectively the signature of K and k on the additive group $\mathcal{X} = (\mathbb{R}^d, +)$ – i. e. $\delta = x - z$ and $e = 0$. The scalar Gaussian kernel reads for all $\delta \in \mathbb{R}^d$

$$k_0^{gauss}(\delta) = \exp\left(-\frac{1}{2\sigma^2}\|\delta\|_2^2\right)$$

where $\sigma \in \mathbb{R}_+$ is an hyperparameter corresponding to the bandwith of the kernel. The –Pontryagin– dual group of $\mathcal{X} = (\mathbb{R}^d, +)$ is $\widehat{\mathcal{X}} \cong (\mathbb{R}^d, +)$ with the pairing

$$(\delta, \omega) = \exp\left(i\langle\delta, \omega\rangle\right)$$

where δ and $\omega \in \mathbb{R}^d$. In this case the Haar measures on $\mathcal{X}$ and $\widehat{\mathcal{X}}$ are in both case the Lebesgue measure. However in order to have the property that $\mathcal{F}^{-1}[\mathcal{F}[f]] = f$ and $\mathcal{F}^{-1}[f] = \mathcal{R}\mathcal{F}[f]$ one must normalize both measures by $\sqrt{2\pi}^{-d}$, i. e. for all $\mathcal{Z} \in \mathcal{B}\left(\mathbb{R}^d\right)$,

$$\begin{aligned}\sqrt{2\pi}^{d}\mathbf{Haar}(\mathcal{Z}) &= \mathbf{Leb}(\mathcal{Z}) \text{ and}\\ \sqrt{2\pi}^{d}\widehat{\mathbf{Haar}}(\mathcal{Z}) &= \mathbf{Leb}(\mathcal{Z}).\end{aligned}$$

Then the Fourier transform on $(\mathbb{R}^d, +)$ is

$$\begin{aligned}\mathcal{F}[f](\omega) &= \int_{\mathbb{R}^d} \exp\left(-i\langle\delta, \omega\rangle\right) f(x)\mathrm{d}\mathbf{Haar}(\delta)\\ &= \int_{\mathbb{R}^d} \exp\left(-i\langle\delta, \omega\rangle\right) f(x)\frac{\mathrm{d}\mathbf{Leb}(\delta)}{\sqrt{2\pi}^{d}}.\end{aligned}$$

Since $k_0^{gauss} \in L^1$ and Γ is bounded, it is possible to apply proposition 3.4, and obtain for all $i, j \in \mathbb{N}_p^*$,

$$\begin{aligned}C^{dec,gauss}(\omega)_{ij} &= \mathcal{F}\left[K_0^{dec,gauss}(\cdot)_{ij}\right](\omega)\\ &= \mathcal{F}\left[k_0^{gauss}\right](\omega)\Gamma_{ij}\\ &= \int_{\mathbb{R}^d} \exp\left(-i\langle\omega, x\rangle\right)\exp\left(-\frac{\|\delta\|_2^2}{2\sigma^2}\right)\frac{\mathrm{d}\mathbf{Leb}(\delta)}{\sqrt{2\pi}^{d}}\Gamma_{ij}\\ &= \frac{1}{\sqrt{2\pi\frac{1}{\sigma^2}}^{d}}\exp\left(-\frac{\sigma^2}{2}\|\omega\|_2^2\right)\sqrt{2\pi}^{d}\Gamma_{ij}.\end{aligned}$$

Hence

$$C^{dec,gauss}(\omega) = \underbrace{\frac{1}{\sqrt{2\pi\frac{1}{\sigma^2}}^d} \exp\left(-\frac{\sigma^2}{2}\|\omega\|_2^2\right)\sqrt{2\pi}^d}_{\rho(\cdot)=\mathcal{N}(0,\sigma^{-2}I_d)\sqrt{2\pi}^d} \underbrace{\Gamma}_{A(\cdot)=\Gamma}$$

Therefore the canonical decomposition of $C^{dec,gauss}$ is $A^{dec,gauss}(\omega) = \Gamma$ and $\rho^{dec,gauss} = \mathcal{N}(0, \sigma^{-2}I_d)\sqrt{2\pi}^d$, where $\mathcal{N}$ is the Gaussian probability distribution. Note that this decomposition is done with respect to the *normalized* Lebesgue measure $\widehat{\mathbf{Haar}}$, meaning that for all $\mathcal{Z} \in \mathcal{B}(\widehat{\mathcal{X}})$,

$$\begin{aligned}\mathbf{Pr}_{\widehat{\mathbf{Haar}},\mathcal{N}(0,\sigma^{-2}I_d)\sqrt{2\pi}^d}(\mathcal{Z}) &= \int_{\mathcal{Z}} \mathcal{N}(0, \sigma^{-2}I_d)\sqrt{2\pi}^d \mathrm{d}\widehat{\mathbf{Haar}}(\omega)\\ &= \int_{\widehat{\mathcal{X}}} \mathcal{N}(0, \sigma^{-2}I_d)\mathrm{d}\mathbf{Leb}(\omega)\\ &= \mathbf{Pr}_{\mathcal{N}(0,\sigma^{-2}I_d)}(\mathcal{Z})\end{aligned}$$

Thus, the same decomposition with respect to the usual –non-normalized– Lebesgue measure **Leb** yields

$$A^{dec,gauss}(\cdot) = \Gamma \tag{3.11a}$$
$$\rho^{dec,gauss} = \mathcal{N}(0, \sigma^{-2}I_d) \tag{3.11b}$$

If Γ is a trace class operator, applying proposition 3.5 yields the same decomposition since $\mathbf{Tr}\left[K_0^{dec,gauss}(0)\right] = \mathbf{Tr}\left[\Gamma\right]$ and $\mathbf{Tr}\left[C^{dec,gauss}(\cdot)\right] = \mathcal{N}(0, \sigma^{-2}I_d)\sqrt{2\pi}^d \mathbf{Tr}\left[\Gamma\right]$.

3.2.2.2 *Skewed-χ^2 decomposable kernel*

The skewed-χ^2 scalar kernel is defined on the LCA product group $\mathcal{X} = ((-c_k; +\infty)_{k=1}^d, \odot)$, with $c_k \in \mathbb{R}_+$. Let $(e_k)_{k=1}^d$ be the standard basis of $\mathcal{X}$ and $_k : x \mapsto \langle x, e_k\rangle$. The operator $\odot : \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ is defined by

$$x \odot z = ((x_k + c_k)(z_k + c_k) - c_k)_{k=1}^d .$$

The identity element e is $(1 - c_k)_{k=1}^d$ since $(1 - c) \odot x = x$. Thus the inverse element x^{-1} is $((x_k + c_k)^{-1} - c_k)_{k=1}^d$. The skewed-$\chi^2$ scalar kernel reads

$$k_{1-c}^{skewed}(\delta) = \prod_{k=1}^d \frac{2}{\sqrt{\delta_k + c_k} + \sqrt{\frac{1}{\delta_k + c_k}}}. \tag{3.12}$$

The dual of $\mathcal{X}$ is $\widehat{\mathcal{X}} \cong \mathbb{R}$ with the pairing

$$(\delta, \omega) = \prod_{k=1}^d \exp\left(i\langle \log(\delta_k + c_k), \omega_k\rangle\right) .$$

The Haar measure are defined for all $\mathcal{Z} \in \mathcal{B}((-c;+\infty)^d)$ and all $\widehat{\mathcal{Z}} \in \mathcal{B}(\mathbb{R}^d)$ by

$$\sqrt{2\pi}^d \mathbf{Haar}(\mathcal{Z}) = \int_{\mathcal{Z}} \prod_{k=1}^{d} \frac{1}{z_k + c_k} d\mathbf{Leb}(z)$$
$$\sqrt{2\pi}^d \widehat{\mathbf{Haar}}(\widehat{\mathcal{Z}}) = \mathbf{Leb}(\widehat{\mathcal{Z}}).$$

Thus the Fourier transform is

$$\mathcal{F}[f](\omega) = \int_{(-c;+\infty)^d} \prod_{k=1}^{d} \frac{\exp\left(-i\langle \log(\delta_k + c_k), \omega_k\rangle\right)}{t_k + c_k} f(\delta) \frac{d\mathbf{Leb}(\delta)}{\sqrt{2\pi}^d}.$$

Then, applying Fubini's theorem over product space,

$$\begin{aligned}
\mathcal{F}\left[k_0^{skewed}\right](\omega) &= \prod_{k=1}^{d} \int_{-c_k}^{\infty} \frac{2\exp\left(-i\langle \log(\delta_k + c_k), \omega_k\rangle\right)}{(t_k + c_k)\left(\sqrt{\delta_k + c_k} + \sqrt{\frac{1}{\delta_k + c_k}}\right)} \frac{d\mathbf{Leb}(\delta_k)}{\sqrt{2\pi}^d} \\
&= \prod_{k=1}^{d} \int_{-c_k}^{\infty} \frac{2\exp\left(-i\langle \delta_k, \omega_k\rangle\right)}{\exp\left(\frac{1}{2}\delta_k\right) + \exp\left(-\frac{1}{2}\delta_k\right)} \frac{d\mathbf{Leb}(\delta_k)}{\sqrt{2\pi}^d} \\
&= \sqrt{2\pi}^d \prod_{k=1}^{d} \operatorname{sech}(\pi\omega_k)
\end{aligned}$$

Since $k_{1-c}^{skewed} \in L^1$ and Γ is bounded, it is possible to apply proposition 3.4, and obtain for all $i, j \in \mathbb{N}_p^*$,

$$\begin{aligned}
C^{dec,skewed}(\omega) &= \mathcal{F}\left[k_{1-c}^{skewed}\right](\omega)\Gamma \\
&= \underbrace{\sqrt{2\pi}^d \prod_{k=1}^{d} \operatorname{sech}(\pi\omega_k)}_{\rho(\cdot)=\mathcal{S}(0,2^{-1})^d\sqrt{2\pi}^d} \underbrace{\Gamma}_{A(\cdot)}
\end{aligned}$$

Hence the decomposition with respect to the usual –non-normalized– Lebesgue measure **Leb** yields

$$A^{dec,skewed}(\cdot) = \Gamma \tag{3.13a}$$
$$\rho^{dec,skewed} = \mathcal{S}(0, 2^{-1})^d \tag{3.13b}$$

3.2.2.3 *Curl-free Gaussian kernel*

The curl-free Gaussian kernel is defined as $K_0^{curl,gauss} = -\nabla\nabla^T k_0^{gauss}$. Here $\mathcal{X} = (\mathbb{R}^d, +)$ so the setting is the same than section 3.2.2.1.

$$\begin{aligned}
C^{curl,gauss}(\omega)_{ij} &= \mathcal{F}\left[K_{1-c}^{curl,gauss}(\cdot)_{ij}\right](\omega) \\
&= \mathcal{F}\left[-\frac{d^2}{d\delta_i d\delta_j} k_0^{gauss}\right](\omega) \\
&= -(i\omega_i)(i\omega_j)\mathcal{F}\left[k_0^{gauss}\right](\omega) \\
&= \omega_i\omega_j \mathcal{F}\left[k_0^{gauss}\right](\omega) \\
&= \sqrt{2\pi\frac{1}{\sigma^2}}^d \exp\left(-\frac{\sigma^2}{2}\|\omega\|_2^2\right) \sqrt{2\pi}^d \omega_i\omega_j.
\end{aligned}$$

Hence

$$C^{curl,gauss}(\omega) = \underbrace{\frac{1}{\sqrt{2\pi\frac{1}{\sigma^2}}^d} \exp\left(-\frac{\sigma^2}{2}\|\omega\|_2^2\right)\sqrt{2\pi}^d}_{\mu(\cdot)=\mathcal{N}(0,\sigma^{-2}I_d)\sqrt{2\pi}^d} \underbrace{\omega\omega^T}_{A(\omega)=\omega\omega^T}.$$

Here a canonical decomposition is $A^{curl,gauss}(\omega) = \omega\omega^T$ for all $\omega \in \mathbb{R}^d$ and $\mu^{curl,gauss} = \mathcal{N}(0, \sigma^{-2}I_d)\sqrt{2\pi}^d$ with respect to the normalized Lebesgue measure $d\omega$. Again the decomposition with respect to the usual –non-normalized– Lebesgue measure is for all $\omega \in \mathbb{R}^d$

$$A^{curl,gauss}(\omega) = \omega\omega^T \tag{3.14a}$$
$$\mu^{curl,gauss} = \mathcal{N}(0, \sigma^{-2}I_d) \tag{3.14b}$$

Notice that in this case $\left\|A^{curl,gauss}(\cdot)\right\|_{\mathbb{R}^d,\mathbb{R}^d}$ is not bounded. However applying proposition 3.5 yields a different decomposition where the quantity $\left\|A^{curl,gauss}_{\mathbf{Tr}}(\cdot)\right\|_{\mathbb{R}^d,\mathbb{R}^d}$ is bounded. First we have for all $\delta \in \mathbb{R}^d$ and for all $i, j \in \mathbb{N}_d^*$

$$\frac{d^2}{d\delta_i d\delta_j} k_0^{gauss}(\delta) = \frac{\exp\left(-\frac{1}{2\sigma^2}\|\delta\|_2^2\right)}{\sigma^2} \begin{cases} \frac{\delta_i\delta_j}{\sigma^2} & \text{if } i \neq j \\ \left(1 - \frac{\delta_i\delta_j}{\sigma^2}\right) & \text{otherwise.} \end{cases}$$

Hence

$$-\nabla\nabla^T k_0^{gauss}(\delta) = \left(I_d - \frac{\delta\delta^T}{\sigma^2}\right)\frac{\exp\left(-\frac{1}{2\sigma^2}\|\delta\|_2^2\right)}{\sigma^2}$$

Thus $\mathbf{Tr}\left[K_0^{curl,gauss}(0)\right] = \mathbf{Tr}\left[\nabla\nabla^T k_0^{gauss}(0)\right] = d\sigma^{-2}$ and $\mathbf{Tr}\left[C(\omega)\right] = \|\omega\|_2^2\mathcal{N}(0, \sigma^{-2}I_d)\sqrt{2\pi}^d$. Apply proposition 3.5 to obtain the decomposition $A^{curl,gauss}_{\mathbf{Tr}}(\omega) = \omega\omega^T\|\omega\|_2^{-2}$ and the measure $\mu^{curl,gauss}_{\mathbf{Tr}}(\omega) = \sigma^2 d^{-1}\|\omega\|_2^2\mathcal{N}\left(0, \sigma^{-2}\right)\sqrt{2\pi}^d$ for all $\omega \in \mathbb{R}^d$, with respect to the normalized Lebesgue measure. Therefore the decomposition with respect to the usual non-normalized Lebesgue measure is

$$A^{curl,gauss}_{\mathbf{Tr}}(\omega) = \frac{\omega\omega^T}{\|\omega\|_2^2} \tag{3.15a}$$
$$\mu^{curl,gauss}_{\mathbf{Tr}}(\omega) = \frac{\sigma^2}{d}\|\omega\|_2^2\mathcal{N}\left(0, \sigma^{-2}\right)(\omega) \tag{3.15b}$$

This example also illustrate that there exist many decomposition of $C(\omega)$ into $(A(\omega), \mu(\omega))$.

3.2.2.4 *Divergence-free kernel*

The divegence-free Gaussian kernel is defined as $K_0^{div,gauss} = (\nabla\nabla^T - \Delta)k_0^{gauss}$ on the group $\mathcal{X} = (\mathbb{R}^d, +)$. The setting is the same than section 3.2.2.1. Hence

$$\begin{aligned}
C^{div,gauss}(\omega)_{ij} &= \mathcal{F}\left[K_0^{div,gauss}(\cdot)_{ij}\right](\omega) \\
&= \mathcal{F}\left[\frac{d^2}{d\delta_i d\delta_j}k_0^{gauss} - \delta_{i=j}\sum_{k=1}^{d}\frac{d^2}{d\delta_k d\delta_k}k_0^{gauss}\right](\omega) \\
&= \left(-(i\omega_i)(i\omega_j) - \delta_{i=j}\sum_{k=1}^{d}(i\omega_k)^2\right)\mathcal{F}\left[k_0^{gauss}\right] \\
&= \left(\delta_{i=j}\sum_{k=1}^{d}\omega_k^2 - \omega_i\omega_j\right)\mathcal{F}\left[k_0^{gauss}\right](\omega).
\end{aligned}$$

Hence

$$C^{div,gauss}(\omega) = \underbrace{\frac{1}{\sqrt{2\pi\frac{1}{\sigma^2}}^d}\exp\left(-\frac{\sigma^2}{2}\|\omega\|_2^2\right)\sqrt{2\pi}^d}_{\rho(\cdot)=\mathcal{N}(0,\sigma^{-2}I_d)\sqrt{2\pi}^d}\underbrace{\left(I_d\|\omega\|_2^2 - \omega\omega^T\right)}_{A(\omega)=I_d\|\omega\|_2^2-\omega\omega^T}.$$

Thus the canonical decomposition with respect to the normalized Lebesgue measure is $A^{div,gauss}(\omega) = I_d\|\omega\|_2^2 - \omega\omega^T$ and the measure $\rho^{div,gauss} = \mathcal{N}(0, \sigma^{-2}I_d)\sqrt{2\pi}^d$. The canonical decomposition with respect to the usual Lebesgue measure is

$$A^{div,gauss}(\omega) = I_d\|\omega\|_2^2 - \omega\omega^T \tag{3.16a}$$
$$\rho^{div,gauss} = \mathcal{N}(0, \sigma^{-2}I_d). \tag{3.16b}$$

To obtain the bounded decomposition, again, apply proposition 3.5. For all $\delta \in \mathbb{R}^d$,

$$\sum_{k=1}^{d}\frac{d^2}{d\delta_k d\delta_k}k_0^{gauss}(\delta) = \left(d - \frac{\|\delta\|_2^2}{\sigma^2}\right)\frac{\exp\left(-\frac{1}{2\sigma^2}\|\delta\|_2^2\right)}{\sigma^2}.$$

Thus overall,

$$K_0^{div,gauss}(\delta) = \left(\frac{\delta\delta^T}{\sigma^2} + \left((d-1) - \frac{\|\delta\|_2^2}{\sigma^2}\right)I_d\right)\frac{\exp\left(-\frac{1}{2\sigma^2}\|\delta\|_2^2\right)}{\sigma^2},$$

Eventually $\mathbf{Tr}\left[K_0^{div,gauss}(0)\right] = \mathbf{Tr}\left[(\nabla\nabla^T - \Delta)k_0^{gauss}(0)\right] = d(d-1)\sigma^{-2}$ and $\mathbf{Tr}\left[C(\omega)\right] = (d-1)\|\omega\|_2^2\mathcal{N}(0, \sigma^2 I_d)\sqrt{2\pi}^d$. As a result the decomposition with respect to the normalized Lebesgue measure is $A_{\mathbf{Tr}}^{div,gauss}(\omega) = (I_d - \omega\omega^T\|\omega\|_2^{-2})$ and $\rho_{\mathbf{Tr}}^{div,gauss}(\omega) = d^{-1}\sigma^2\|\omega\|_2^2\mathcal{N}(0, \sigma^2 I_d)\sqrt{2\pi}^d$.

The decomposition with respect to the normalized Lebesgue measure being

$$A_{\mathbf{Tr}}^{div,gauss}(\omega) = I_d - \frac{\omega\omega^T}{\|\omega\|_2^2} \tag{3.17a}$$

$$\rho_{\mathbf{Tr}}^{div,gauss} = \frac{\sigma^2}{d}\|\omega\|_2^2 \mathcal{N}(0, \sigma^{-2} I_d). \tag{3.17b}$$

3.2.3 *Functional Fourier feature map*

Let us introduce a functional feature map, we call here *Fourier Feature map*, defined by the following proposition as a direct consequence of proposition 3.2.

Proposition 3.6 (Functional Fourier feature map). *Let $\mathcal{Y}$ and $\mathcal{Y}'$ be two Hilbert spaces. If there exist an operator-valued function $B : \widehat{\mathcal{X}} \to \mathcal{L}(\mathcal{Y}, \mathcal{Y}')$ such that for all $y, y' \in \mathcal{Y}$,*

$$\langle y, B(\omega)B(\omega)^* y' \rangle_{\mathcal{Y}} = \langle y', A(\omega) y \rangle_{\mathcal{Y}}$$

$\widehat{\mu}$-almost everywhere and $\langle y', A(\cdot)y\rangle \in L^1(\widehat{\mathcal{X}}, \widehat{\mu})$ then the operator Φ_x defined for all y in $\mathcal{Y}$ by

$$(\Phi_x y)(\omega) = (x, \omega) B(\omega)^* y, \tag{3.18}$$

is a feature map[7] of some shift-invariant $\mathcal{Y}$-Mercer kernel K.

[7] *I. e. it satisfies for all $x, z \in \mathcal{X}$, $\Phi_x^* \Phi_z = K(x, z)$ where K is a $\mathcal{Y}$-Mercer OVK.*

Proof *For all $y, y' \in \mathcal{Y}$ and $x, z \in \mathcal{X}$,*

$$\begin{aligned}
\langle y, \Phi_x^* \Phi_z y' \rangle_{\mathcal{Y}} &= \langle \Phi_x y, \Phi_z y' \rangle_{L^2(\widehat{\mathcal{X}}, \widehat{\mu}; \mathcal{Y}')} \\
&= \int_{\widehat{\mathcal{X}}} \overline{(x, \omega)} \langle y, B(\omega)(z, \omega) B(\omega)^* y' \rangle \mathrm{d}\widehat{\mu}(\omega) \\
&= \int_{\widehat{\mathcal{X}}} \overline{(x \star z^{-1}, \omega)} \langle y B(\omega) B(\omega)^* y' \rangle \mathrm{d}\widehat{\mu}(\omega) \\
&= \int_{\widehat{\mathcal{X}}} \overline{(x \star z^{-1}, \omega)} \langle y, A(\omega) y' \rangle \mathrm{d}\widehat{\mu}(\omega),
\end{aligned}$$

which defines a $\mathcal{Y}$-Mercer according to proposition 3.2 of Carmeli et al. [13]. □

With this notation we have $\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{Y}; L^2(\widehat{\mathcal{X}}, \widehat{\mu}; \mathcal{Y}'))$ such that $\Phi_x \in \mathcal{L}(\mathcal{Y}; L^2(\widehat{\mathcal{X}}, \widehat{\mu}; \mathcal{Y}'))$ where $\Phi_x := \Phi(x)$.

3.3 BUILDING OPERATOR-VALUED RANDOM FOURIER FEATURES

As shown in propositions 3.4 and 3.5 it is always possible to find a pair $(A, \mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho})$ from a shift invariant $\mathcal{Y}$-Mercer Operator-Valued Kernel K_e such that $\mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho}$ is a probability measure –i. e. $\int_{\widehat{\mathcal{X}}} \rho \mathrm{d}\widehat{\mathbf{Haar}} = 1$ where ρ is the density of $\mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho}$– and $K_e(\delta) = \mathbf{E}_\rho \overline{(\delta, \omega)} A(\omega)$. In order to obtain an approximation of K from a decomposition $(A, \mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho})$ we turn our attention to a Monte-Carlo estimation of the expectations equation 3.9 and equation 3.5 characterizing a $\mathcal{Y}$-Mercer shift-invariant Operator-Valued Kernel.

However, for efficient computations as motivated in the introduction, we are interested in finding an approximated *feature map* instead of a kernel approximation. Indeed, an approximated feature map will allow to build linear models in regression tasks. The idea is to start from the Monte-Carlo approximation of the expectation and provide a systematic decomposition of the Monte-Carlo sample mean into an approximate feature map. The following proposition provides the general form of an Operator-valued Random Fourier Feature.

Proposition 3.7 (ORFF). *Let $\mathcal{Y}$ and $\mathcal{Y}'$ be two Hilbert spaces. If one can find $B : \widehat{\mathcal{X}} \to \mathcal{L}(\mathcal{Y}, \mathcal{Y}')$ and a probability measure $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ on $\mathcal{B}(\widehat{\mathcal{X}})$, such that for all $y \in \mathcal{Y}$ and all $y' \in \mathcal{Y}'$, $\langle y, B(\cdot)y' \rangle \in L^2(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho})$, then the operator-valued function given for all $y \in \mathcal{Y}$ by*

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} (x, \omega_j) B(\omega_j)^* y, \quad \omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \text{ i. i. d.}, \tag{3.19}$$

is an approximated feature map[8] of a shift-invariant $\mathcal{Y}$-Mercer Operator-Valued Kernel.

[8] *I. e. it satisfies $\widetilde{\Phi}(x)^* \widetilde{\Phi}(z) \xrightarrow[D\to\infty]{a.\,s.} K(x, z)$ where K is a $\mathcal{Y}$-Mercer OVK.*

Proof *Let $(\omega_j)_{j=1}^D$ be a sequence of $D \in \mathbb{N}^*$ i. i. d. random vectors, each of them following the law $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$. For all $x, z \in \mathcal{X}$ and all $y, y' \in \mathcal{Y}$,*

$$\begin{aligned}
\left\langle \widetilde{\Phi}(x)y', \widetilde{\Phi}(z)y \right\rangle &= \frac{1}{D} \left\langle \bigoplus_{j=1}^{D} (x, \omega_j) B(\omega_j)^* y', \bigoplus_{j=1}^{D} (z, \omega_j) B(\omega_j)^* y \right\rangle \\
&= \frac{1}{D} \sum_{j=1}^{D} \left\langle y', \overline{(x, \omega_j)} B(\omega_j)(z, \omega_j) B(\omega_j)^* y \right\rangle_{\mathcal{Y}} \\
&= \left\langle y', \left(\frac{1}{D} \sum_{j=1}^{D} \overline{(x \star z^{-1}, \omega_j)} A(\omega_j) \right) y \right\rangle_{\mathcal{Y}},
\end{aligned}$$

where $A(\omega) = B(\omega)B(\omega)^*$. By assumption $\langle y, A(\cdot)y' \rangle \in L^1(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho})$ and ω_j are i. i. d. . Hence from the strong law of large numbers and proposition 3.2 with $\widehat{\mu} = \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$,

$$\frac{1}{D} \sum_{j=1}^{D} \overline{(x \star z^{-1}, \omega_j)} A(\omega_j) \xrightarrow[D\to\infty]{a.\,s.} \mathbf{E}_\rho[\overline{(x \star z^{-1}, \omega_j)} A(\omega)] = K_e(x \star z^{-1})$$

where the integral converges in the weak operator topology. □

Remark 3.2 *The approximate feature map proposed in proposition 3.7 has direct link with the functional Fourier feature map defined in proposition 3.6 since we have for all $y \in \mathcal{Y}$*

$$\begin{aligned}
\widetilde{\Phi}(x)y &= \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} (x, \omega_j) B(\omega_j)^* y, \quad \omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \text{ i. i. d.} \\
&= \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} (\Phi_x y)(\omega_j).
\end{aligned} \tag{3.20}$$

Therefore $\widetilde{\Phi}(x)$ can be seen as an "operator-valued vector" corresponding the "stacking" of D i. i. d. operator-valued random variable $(x, \omega_j)B(\omega_j)^*$. Note that we consider ω to be a *variable* in $\widehat{\mathcal{X}}$ while ω_j are $\widehat{\mathcal{X}}$-valued *random variables*. I. e. ω_j are $(\widehat{\mathcal{X}}, \mathcal{B}(\widehat{\mathcal{X}}))$-valued measurable function and $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ is the distribution of each ω_j. Let

$$\widetilde{\mathcal{H}} = \bigoplus_{j=1}^{D} \mathcal{Y}',$$

be a Hilbert space. Let $\boldsymbol{\omega} \in \widehat{\mathcal{X}}^D$ be a sequence where $\boldsymbol{\omega} = (\omega_j)_{j=1}^D$ (i. e. a realization of the random sequence $(\omega_j)_{j=1}^D$, $\omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$). Notice that for any $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, $\widetilde{\Phi}(x)y \in \widetilde{\mathcal{H}}$ as we can take $g(\omega_j) = (x, \omega_j)B(\omega_j)^* y$. Thus we have for all $y, y' \in \mathcal{Y}$ and all $x, z \in \mathcal{X}$

$$\langle y', \widetilde{K}(x,z)y\rangle_{\mathcal{Y}} = \langle y', \widetilde{\Phi}(x)^*\widetilde{\Phi}(z)y\rangle_{\mathcal{Y}} = \langle \widetilde{\Phi}(x)y', \widetilde{\Phi}(z)y\rangle_{\widetilde{\mathcal{H}}}.$$

Thus $\widetilde{K}(x,z) = \widetilde{\Phi}(x)^*\widetilde{\Phi}(z)$ is a proper random shift-invariant Operator-Valued Kernel as shown in the following proposition.

Proposition 3.8 *Let $\boldsymbol{\omega} \in \widehat{\mathcal{X}}^D$. If for all $y, y' \in \mathcal{Y}$*

$$\begin{aligned}\langle y', \widetilde{K}_e\left(x \star z^{-1}\right) y\rangle_{\mathcal{Y}} &= \langle \widetilde{\Phi}(x)y', \widetilde{\Phi}(z)y\rangle_{\widetilde{\mathcal{H}}} \\ &= \left\langle y', \frac{1}{D}\sum_{j=1}^{D} \overline{(x \star z^{-1}, \omega_j)} B(\omega_j)B(\omega_j)^* y \right\rangle_{\mathcal{Y}},\end{aligned}$$

for all $x, z \in \mathcal{X}$, then $\widetilde{K}$ is a shift-invariant Operator-Valued Kernel.

Proof *Apply proposition 2.3 to $\widetilde{\Phi}$ considering the Hilbert space $\widetilde{\mathcal{H}}$ to show that $\widetilde{K}$ is an OVK. Then proposition 2.6 shows that $\widetilde{K}$ is shift-invariant since $\widetilde{K}(x,z) = \widetilde{K}_e\left(x, \star z^{-1}\right)$.* □

We stress out that if $\boldsymbol{\omega} = (\omega_j)_{j=1}^D \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ i. i. d. is a *random sequence* then

$$\widetilde{K}_e\left(x \star z^{-1}\right) = \widetilde{\Phi}(x)^*\widetilde{\Phi}(z)$$

is not *sensus* stricto an Operator-Valued Kernel since it is a random variable (and $\widetilde{\mathcal{H}}$ is no longer a Hilbert space, since its inner product is a then random variable and not a scalar). However this is not a problem since any realization of the random sequence $\boldsymbol{\omega}$ gives birth to a (different) Operator-Valued Kernel, and $\mathbf{E}_{\widehat{\mathbf{Haar}},\rho}\widetilde{K}$ is an OVK. This illustrated by figure 1 where we represented the same function for different realization of $\widetilde{K} \approx K$. We generated 250 points equally separated on the segment $(-1; 1)$.

We computed the Gram Matrix of the Gaussian decomposable kernel

$$K(x,z)_{ij} = \exp\left(-\frac{1}{2(0.1)^2(x_i - x_j)^2}\right)\Gamma, \quad \text{for } i, j \in \mathbb{N}^*_{250}.$$

We computed a reference function (black line) defined as $(y_1, y_2)^\top = f(x_i) = \sum_{j=1}^{250} K(x_i, x_j)u_j$ where $u_j \sim \mathcal{N}(0, 1)$ i. i. d.. We took $\Gamma = .5I_2 +$

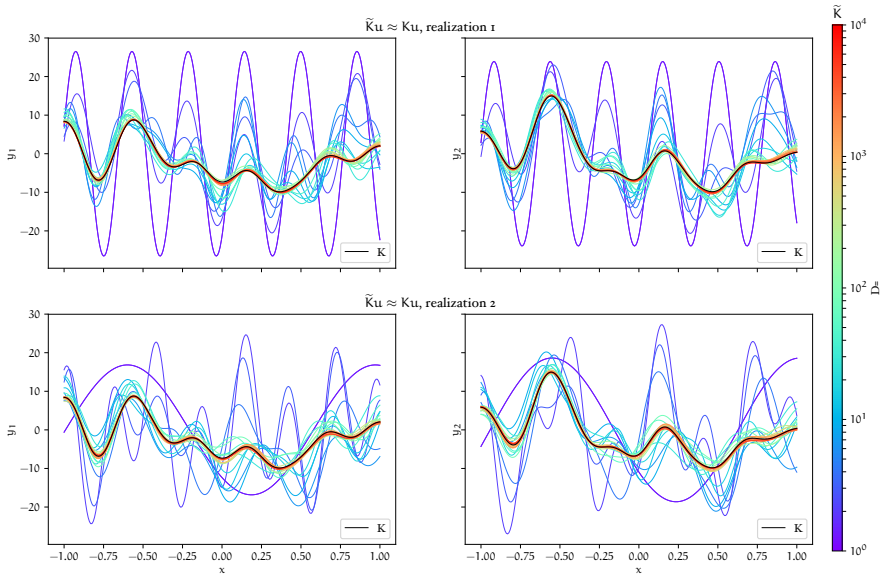

Figure 1: Different realizations of a Gaussian kernel approximation. Top row and bottom row correspond to two different realizations of $\widetilde{K}$, which are *different* Operator-Valued Kernel. However when D tends to infinity, the different realizations of $\widetilde{K}$ yield the samel OVK.

$.51_2$ such that the outputs y_1 and y_2 share some similarities. Then we computed an approximate kernel matrix $\widetilde{K} \approx K$ for 25 increasing values of D ranging from 1 to 10^4. The two graphs on the top row shows that the more the number of features increase the closer the model $\widetilde{f}(x_i) = \sum_{j=1}^{250} \widetilde{K}(x_i, x_j)u_j$ is to f. The bottom row shows the same experiment but for a different realization of $\widetilde{K}$. When D is small the curves of the bottom and top rows are very dissimilar –and sine wave like– while they both converge to f when D increase.

In the same way we defined an ORFF, we can define an approximate feature operator $\widetilde{W}$ which maps $\widetilde{\mathcal{H}}$ onto $\mathcal{H}_{\widetilde{K}}$, where

$$\widetilde{K}(x, z) = \widetilde{\Phi}(x)^* \widetilde{\Phi}(z)$$

for all $x, z \in \mathcal{X}$.

Definition 3.1 (Random Fourier feature operator). *Let* $\omega = (\omega_j)_{j=1}^D \in \widehat{\mathcal{X}}^D$ *and let*

$$\widetilde{K}_e = \frac{1}{D} \sum_{j=1}^{D} \overline{(\cdot, \omega_j)} B(\omega_j) B(\omega_j)^*.$$

We call random Fourier feature operator the linear application $\widetilde{W} : \widetilde{\mathcal{H}} \to \mathcal{H}_{\widetilde{K}}$ *defined as*

$$\left(\widetilde{W}\theta\right)(x) := \widetilde{\Phi}(x)^* \theta = \frac{1}{D} \sum_{j=1}^{D} \overline{(x, \omega_j)} B(\omega_j) g(\omega_j)$$

where $\theta = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} g(\omega_j) \in \widetilde{\mathcal{H}}$. *Then,*

$$\left(Ker\, \widetilde{W}\right)^{\perp} = \overline{\text{span}} \left\{ \widetilde{\Phi}(x)y \;\middle|\; \forall x \in \mathcal{X},\ \forall y \in \mathcal{Y} \right\} \subseteq \widetilde{\mathcal{H}}.$$

The random Fourier feature operator is useful to show the relations between the random Fourier feature map with the functional feature map defined in proposition 3.6. The relationship between the generic feature map (defined for all Operator-Valued Kernel) the functional feature map (defining a shift-invariant $\mathcal{Y}$-Mercer Operator-Valued Kernel) and the random Fourier feature map is presented in figure 2.

Proposition 3.9 *For any* $g \in \mathcal{H} = L^2(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}; \mathcal{Y}')$, *let*

$$\theta := \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} g(\omega_j), \quad \omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \ \textit{i. i. d.}.$$

Then

1. $\left(\widetilde{W}\theta\right)(x) = \widetilde{\Phi}(x)^*\theta \xrightarrow[D\to\infty]{\textit{a. s.}} \Phi_x^* g = (Wg)(x)$,
2. $\|\theta\|^2_{\widetilde{\mathcal{H}}} \xrightarrow[D\to\infty]{\textit{a. s.}} \|g\|^2_{\mathcal{H}}$,

Proof (of ?? 25) *since* $(\omega_j)_{j=1}^{D}$ *are i. i. d. random vectors, for all* $y \in \mathcal{Y}$ *and for all* $y' \in \mathcal{Y}'$, $\langle y, B(\cdot)y' \rangle \in L^2(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho})$ *and* $g \in L^2(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}; \mathcal{Y}')$, *from the strong law of large numbers*

$$\begin{aligned}
(\widetilde{W}\theta)(x) &= \widetilde{\Phi}(x)^*\theta \\
&= \frac{1}{D} \sum_{j=1}^{D} \overline{(x, \omega_j)} B(\omega_j) g(\omega_j), \qquad \omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \ \textit{i. i. d.} \\
&\xrightarrow[D\to\infty]{\textit{a. s.}} \int_{\widehat{\mathcal{X}}} \overline{(x, \omega)} B(\omega) g(\omega) d\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}(\omega) \\
&= (Wg)(x) := \Phi_x^* g. \qquad \square
\end{aligned}$$

Proof (of ?? 26) *again, since* $(\omega_j)_{j-1}^{D}$ *are i. i. d. random vectors and* $g \in L^2(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}; \mathcal{Y}')$, *from the strong law of large numbers*

$$\begin{aligned}
\|\theta\|^2_{\widetilde{\mathcal{H}}} &= \frac{1}{D} \sum_{j=1}^{D} \left\| g(\omega_j) \right\|^2_{\mathcal{Y}'}, \qquad \omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \ \textit{i. i. d.} \\
&\xrightarrow[D\to\infty]{\textit{a. s.}} \int_{\widehat{\mathcal{X}}} \|g(\omega)\|^2_{\mathcal{Y}'} d\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}(\omega) \\
&= \|g\|^2_{L^2\left(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}; \mathcal{Y}'\right)}. \qquad \square
\end{aligned}$$

We write $\widetilde{\Phi}(x)^*\widetilde{\Phi}(x) \approx K(x, z)$ when $\widetilde{\Phi}(x)^*\widetilde{\Phi}(x) \xrightarrow{\text{a. s.}} K(x, z)$ in the weak operator topology when D tends to infinity. With mild abuse of notation

we say that $\widetilde{\Phi}(x)$ is an approximate feature map of Φ_x i. e. $\widetilde{\Phi}(x) \approx \Phi_x$, when for all $y', y \in \mathcal{Y}$,

$$\begin{aligned}\langle y, K(x,z)y'\rangle_{\mathcal{Y}} &= \langle \Phi_x y, \Phi_z y'\rangle_{\mathcal{L}\in(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\subset};\mathcal{Y}')} \\ &\approx \langle \widetilde{\Phi}(x)y, \widetilde{\Phi}(x)y'\rangle_{\widetilde{\mathcal{H}}} := \langle y, \tilde{K}(x,z)y'\rangle_{\mathcal{Y}}\end{aligned}$$

where Φ_x is defined in the sense of proposition 3.6. Then corollary 3.1 exhibit a construction of an ORFF directly from an OVK.

Corollary 3.1 *If $K(x,z)$ is a shift-invariant $\mathcal{Y}$-Mercer kernel such that for all $y, y' \in \mathcal{Y}$, $\langle y', K_e(\cdot)y\rangle \in L^1(\mathcal{X}, \mathbf{Haar})$. Then*

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} (x, \omega_j) B(\omega_j)^* y, \qquad \omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \text{ i. i. d.}, \tag{3.21}$$

where $\langle y, B(\omega)B(\omega)^* y'\rangle \rho(\omega) = \mathcal{F}\left[\langle y', K_e(\cdot)y\rangle\right](\omega)$, is an approximated feature map of K.

Proof *Find $(A, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho})$ from proposition 3.4, find a decomposition of $A(\omega) = B(\omega)B(\omega)^*$ for $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$-almost all ω and apply proposition 3.7.* □

Remark 3.3 *We find a decomposition such that for all $j = 1, \ldots, D$, $A(\omega_j) = B(\omega_j)B(\omega_j)^*$ either by exhibiting an analytic closed-form or using a numerical decomposition.*

Corollary 3.1 allows us to define **??** 1 for constructing ORFF from an operator valued kernel.

Algorithm 1: Construction of ORFF from OVK

Input : $K(x,z) = K_e(\delta)$ a $\mathcal{Y}$-shift-invariant Mercer kernel such that $\forall y, y' \in \mathcal{Y}$, $\langle y', K_e(\cdot)y\rangle \in L^1(\mathbb{R}^d, \mathbf{Haar})$ and D the number of features.

Output : A random feature $\widetilde{\Phi}(x)$ such that $\widetilde{\Phi}(x)^*\widetilde{\Phi}(z) \approx K(x,z)$

1 Define the pairing (x, ω) from the LCA group $(\mathcal{X}, \star)$;

2 Find a decomposition $(B(\omega), \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho})$ such that

$$B(\omega)B(\omega)^*\rho(\omega) = \mathcal{F}^{-1}\left[K_e\right](\omega);$$

3 Draw D random vectors $(\omega_j)_{j=1}^D$ i. i. d. from the probability law $\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$;

4 **return** $\begin{cases} \widetilde{\Phi}(x) \in \mathcal{L}(\mathcal{Y}, \widetilde{\mathcal{H}}) & : y \mapsto \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} (x, \omega_j) B(\omega_j)^* y \\ \widetilde{\Phi}(x)^* \in \mathcal{L}(\widetilde{\mathcal{H}}, \mathcal{Y}) & : \theta \mapsto \frac{1}{\sqrt{D}} \sum_{j=1}^{D} (x, \omega_j) B(\omega_j) \theta_j \end{cases}$;

3.3.1 *Examples of Operator Random Fourier Feature maps*

We now give two examples of operator-valued random Fourier feature map when. First we introduce the general form of an approximated feature map for a matrix-valued kernel on the additive group $(\mathbb{R}^d, +)$.

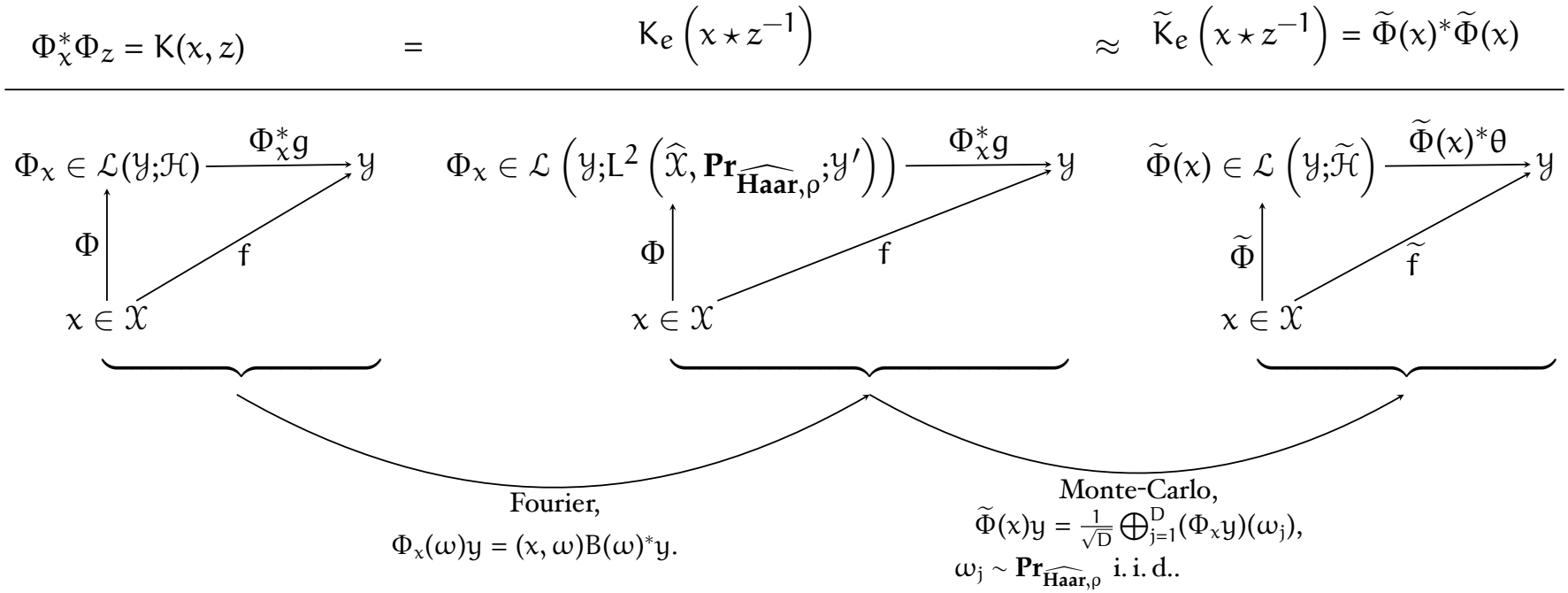

Figure 2: Relationships between feature-maps. For any realization of $\omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ i. i. d. , $\widetilde{\mathcal{H}} = \bigoplus_{j=1}^{D} \mathcal{Y}'$.

Example 3.1 (Matrix-valued kernel on the additive group). *In the following,* $K(x,z) = K_0(x-z)$ *is a* $\mathcal{Y}$*-Mercer matrix-valued kernel on* $\mathcal{X} = \mathbb{R}^d$ *invariant w.r.t. the group operation* $+$*. Then the function* $\widetilde{\Phi}$ *defined as follow is an Operator-valued Random Fourier Feature of* K_0*.*

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle B(\omega_j)^* y \\ \sin\langle x, \omega_j\rangle B(\omega_j)^* y \end{pmatrix}, \quad \omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \ \textit{i. i. d.}.$$

for all $y \in \mathcal{Y}$*.*

Proof *The (Pontryagin) dual of* $\mathcal{X} = \mathbb{R}^d$ *is* $\widehat{\mathcal{X}} \cong \mathbb{R}^d$*, and the duality pairing is* $(x-z,\omega) = \exp(i\langle x-z,\omega\rangle)$*. The kernel approximation yields:*

$$\begin{aligned}
\tilde{K}(x,z) &= \widetilde{\Phi}(x)^* \widetilde{\Phi}(z) \\
&= \frac{1}{D} \sum_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle & \sin\langle x, \omega_j\rangle \end{pmatrix} \begin{pmatrix} \cos\langle z, \omega_j\rangle \\ \sin\langle z, \omega_j\rangle \end{pmatrix} A(\omega_j) \\
&= \frac{1}{D} \sum_{j=1}^{D} \cos\langle x-z, \omega_j\rangle A(\omega_j) \\
&\xrightarrow[D\to\infty]{a.\,s.} \mathbf{E}_\rho \left[\cos\langle x-z,\omega\rangle A(\omega)\right]
\end{aligned}$$

in the weak operator topology. Since for all $x \in \mathcal{X}$*,* $\sin\langle x, \cdot\rangle$ *is an odd function and* $A(\cdot)\rho(\cdot)$ *is even,*

$$\mathbf{E}_\rho \left[\cos\langle x-z,\omega\rangle A(\omega)\right] = \mathbf{E}_\rho \left[\exp(-i\langle x-z,\omega\rangle) A(\omega)\right] = K(x,z).$$

Hence $\tilde{K}(x,z) \xrightarrow[D\to\infty]{a.\,s.} K(x,z)$*.* □

In particular we deduce the following features maps for the kernels proposed in section 3.2.2.

- For the decomposable gaussian kernel $K_0^{dec,gauss}(\delta) = k_0^{gauss}(\delta)\Gamma$ for all $\delta \in \mathbb{R}^d$, let $BB^* = \Gamma$. A bounded –and unbounded– ORFF map is

$$\begin{aligned}
\widetilde{\Phi}(x)y &= \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle B^* y \\ \sin\langle x, \omega_j\rangle B^* y \end{pmatrix} \\
&= (\widetilde{\phi}(x) \otimes B^*) y,
\end{aligned}$$

  where $\omega_j \sim \mathbf{Pr}_{\mathcal{N}(0,\sigma^{-2}I_d)}$ i. i. d. and $\widetilde{\phi}(x) = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle \\ \sin\langle x, \omega_j\rangle \end{pmatrix}$ is a scalar RFF map [37].

- For the curl-free gaussian kernel, $K_0^{curl,gauss} = -\nabla\nabla^T k_0^{gauss}$ an unbounded ORFF map is

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle \omega_j^T y \\ \sin\langle x, \omega_j\rangle \omega_j^T y \end{pmatrix}, \tag{3.22}$$

$\omega_j \sim \mathbf{Pr}_{\mathcal{N}(0,\sigma^{-2}I_d)}$ i. i. d. and a bounded ORFF map is

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle \frac{\omega_j^T}{\|\omega_j\|} y \\ \sin\langle x, \omega_j\rangle \frac{\omega_j^T}{\|\omega_j\|} y \end{pmatrix}, \quad \omega_j \sim \mathbf{Pr}_\rho \text{ i. i. d..}$$

where $\rho(\omega) = \frac{\sigma^2\|\omega\|^2}{d}\mathcal{N}(0, \sigma^{-2}I_d)(\omega)$ for all $\omega \in \mathbb{R}^d$.

- For the divergence-free gaussian kernel $K_0^{div,gauss}(x,z) = (\nabla\nabla^T - \Delta I_d)k_0^{gauss}(x,z)$ an unbounded ORFF map is

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle B(\omega_j)^T y \\ \sin\langle x, \omega_j\rangle B(\omega_j)^T y \end{pmatrix} \tag{3.23}$$

where $\omega_j \sim \mathbf{Pr}_\rho$ i. i. d. and $B(\omega) = \left(\|\omega\| I_d - \omega\omega^T\right)$ and $\rho = \mathcal{N}(0, \sigma^{-2}I_d)$ for all $\omega \in \mathbb{R}^d$. A bounded ORFF map is

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle B(\omega_j)^T y \\ \sin\langle x, \omega_j\rangle B(\omega_j)^T y \end{pmatrix}, \quad \omega_j \sim \mathbf{Pr}_\rho \text{ i. i. d.,}$$

where $B(\omega) = \left(I_d - \frac{\omega\omega^T}{\|\omega\|^2}\right)$ and $\rho(\omega) = \frac{\sigma^2\|\omega\|^2}{d}\mathcal{N}(0, \sigma^{-2}I_d)$ for all $\omega \in \mathbb{R}^d$.

The second example extends scalar-valued Random Fourier Features on the skewed multiplicative group –described in **??** and section 3.2.2.2– to the operator-valued case.

Example 3.2 (Matrix-valued kernel on the skewed multiplicative group). *In the following,* $K(x,z) = K_{1-c}(x \odot z^{-1})$ *is a* $\mathcal{Y}$*-Mercer matrix-valued kernel on* $\mathcal{X} = (-c; +\infty)^d$ *invariant w. r. t. the group operation*[9] $\odot$*. Then the function* $\widetilde{\Phi}$ *defined as follow is an Operator-valued Random Fourier Feature of* K_{1-c}*.*

[9] *The group operation* $\odot$ *is defined in section 3.2.2.2.*

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle \log(x+c), \omega_j\rangle B(\omega_j)^* y \\ \sin\langle \log(x+c), \omega_j\rangle B(\omega_j)^* y \end{pmatrix},$$

$\omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ *iid, for all* $y \in \mathcal{Y}$*.*

Proof *The dual of* $\mathcal{X} = (-c; +\infty)^d$ *is* $\widehat{\mathcal{X}} \cong \mathbb{R}^d$*, and the duality pairing is* $(x \odot z^{-1}, \omega) = \exp(i\langle \log(x \odot z^{-1} + c), \omega\rangle)$*. Following the proof of example 3.1, we have*

$$\tilde{K}(x,z) = \frac{1}{D}\sum_{j=1}^{D} \cos\left\langle \log\left(\frac{x+c}{z+c}\right), \omega_j \right\rangle A(\omega_j).$$

which converges almost surely to

$$\mathbf{E}_\rho[\exp(-i\langle \log(x \odot z^{-1} + c)\rangle)A(\omega)] = \mathbf{E}_\rho[\overline{(x \odot z^{-1}, \omega)}A(\omega)] = K(x,z)$$

when D *tends to infinity, in the weak operator topology.* □

- For the skewed-χ^2 decomposable kernel defined as $K^{dec,skewed}_{1-c}(\delta) = k^{skewed}_{1-c}(\delta)\Gamma$ for all $\delta \in \mathcal{X}$, let $BB^* = \Gamma$. A bounded –and unbounded– ORFF map is

$$\widetilde{\Phi}(x)y = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos \langle \log(x+c), \omega_j \rangle B^* y \\ \sin \langle \log(x+c), \omega_j \rangle B^* y \end{pmatrix}, \quad \omega_j \sim \mathbf{Pr}_\rho \text{ i. i. d.}$$
$$= (\widetilde{\Phi}(x) \otimes B^*)y,$$

  where $\rho = \mathcal{S}(0, 2^{-1})$ and $\widetilde{\Phi}(x) = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos \langle \log(x+c), \omega_j \rangle \\ \sin \langle \log(x+c), \omega_j \rangle \end{pmatrix}$ is a scalar RFF map [25].

3.3.2 *Regularization property*

We have shown so far that it is always possible to construct a feature map that allows to approximate a shift-invariant $\mathcal{Y}$-Mercer kernel. However we could also propose a construction of such map by studying the regularization induced with respect to the Fourier transform of a target function $f \in \mathcal{H}_K$. In other words, what is the norm in $L^2(\widehat{\mathcal{X}}, \widehat{\mathbf{Haar}}; \mathcal{Y}')$ induced by $\|\cdot\|_K$?

Proposition 3.10 *Let K be a shift-invariant $\mathcal{Y}$-Mercer Kernel such that for all y, y' in $\mathcal{Y}$, $\langle y', K_e(\cdot)y \rangle \in L^1(\mathcal{X}, \mathbf{Haar})$. Then for all $f \in \mathcal{H}_K$*

$$\|f\|_K^2 = \int_{\widehat{\mathcal{X}}} \frac{\left\langle \mathcal{F}[f](\omega), A(\omega)^\dagger \mathcal{F}[f](\omega) \right\rangle_{\mathcal{Y}}}{\rho(\omega)} d\widehat{\mathbf{Haar}}(\omega). \tag{3.24}$$

where $\langle y', A(\omega)y \rangle \rho(\omega) := \mathcal{F}[\langle y', K_e(\cdot)y \rangle](\omega)$.

Proof *We first show how the Fourier transform relates to the feature operator. Since $\mathcal{H}_K$ is embed into $\mathcal{H} = L^2(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}; \mathcal{Y})$ by mean of the feature operator W, we have for all $f \in \mathcal{H}_k$, for all $f \in \mathcal{H}$ and for all $x \in \mathcal{X}$*

$$\mathcal{F}\left[\mathcal{F}^{-1}[f]\right](x) = \int_{\widehat{\mathcal{X}}} \overline{(x,\omega)} \mathcal{F}^{-1}[f](\omega) d\widehat{\mathbf{Haar}}(\omega) = f(x)$$
$$(Wg)(x) = \int_{\widehat{\mathcal{X}}} \overline{(x,\omega)} \rho(\omega) B(\omega) g(\omega) d\widehat{\mathbf{Haar}}(\omega) = f(x).$$

By injectivity of the Fourier transform, $\mathcal{F}^{-1}[f](\omega) = \rho(\omega)B(\omega)g(\omega)$. From proposition 2.3 we have

$$\|f\|_K^2 = \inf \left\{ \|g\|_{\mathcal{H}}^2 \;\middle|\; \forall g \in \mathcal{H}, \;\; Wg = f \right\}$$
$$= \inf \left\{ \int_{\widehat{\mathcal{X}}} \|g\|_{\mathcal{Y}}^2 d\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho} \;\middle|\; \forall g \in \mathcal{H}, \;\; \mathcal{F}^{-1}[f] = \rho(\cdot)B(\cdot)g(\cdot) \right\}.$$

The pseudo inverse of the operator $B(\omega)$ – *noted* $B(\omega)^\dagger$ – *is the unique solution of the system* $\mathcal{F}^{-1}[f](\omega) = \rho(\omega)B(\omega)g(\omega)$ *w. r. t.* $g(\omega)$ *with minimal norm*[1]. *Eventually,*

$$\|f\|_K^2 = \int_{\widehat{\mathcal{X}}} \frac{\left\|B(\omega)^\dagger \mathcal{F}^{-1}[f](\omega)\right\|_{\mathcal{Y}}^2}{\rho(\omega)^2} d\mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}(\omega)$$

Using the fact that $\mathcal{F}^{-1}[\cdot] = \mathcal{F}\mathcal{R}[\cdot]$ *and* $\mathcal{F}^2[\cdot] = \mathcal{R}[\cdot]$,

$$\begin{aligned}
\|f\|_K^2 &= \int_{\widehat{\mathcal{X}}} \frac{\left\|\mathcal{R}\left[B(\cdot)^\dagger \rho(\cdot)\right](\omega)\mathcal{F}[f](\omega)\right\|_{\mathcal{Y}}^2}{\rho(\omega)^2} d\widehat{\mathbf{Haar}}(\omega) \\
&= \int_{\widehat{\mathcal{X}}} \frac{\left\|B(\omega)^\dagger \rho(\omega)\mathcal{F}[f](\omega)\right\|_{\mathcal{Y}}^2}{\rho(\omega)^2} d\widehat{\mathbf{Haar}}(\omega) \\
&= \int_{\widehat{\mathcal{X}}} \frac{\langle B(\omega)^\dagger \mathcal{F}[f](\omega), B(\omega)^\dagger \mathcal{F}[f](\omega)\rangle_{\mathcal{Y}}}{\rho(\omega)} d\widehat{\mathbf{Haar}}(\omega) \\
&= \int_{\widehat{\mathcal{X}}} \frac{\langle \mathcal{F}[f](\omega), A(\omega)^\dagger \mathcal{F}[f](\omega)\rangle_{\mathcal{Y}}}{\rho(\omega)} d\widehat{\mathbf{Haar}}(\omega)
\end{aligned}$$

Note that if $K(x,z) = k(x,z)$ is a scalar kernel then for all ω in $\widehat{\mathcal{X}}$, $A(\omega) = 1$. Therefore we recover a well known results for kernels that is for any $f \in \mathcal{H}_k$ we have $\|f\|_k = \int_{\widehat{\mathcal{X}}} \mathcal{F}[k_e](\omega)^{-1}\mathcal{F}[f](\omega)^2 d\widehat{\mathbf{Haar}}(\omega)$ [42, 47, 51]. We also note that the regularization property in $\mathcal{H}_K$ does not depends (as expected) on the decomposition of $A(\omega)$ into $B(\omega)B(\omega)^*$. Therefore the decomposition should be chosen such that it optimizes the computation cost. For instance if $A(\omega) \in \mathcal{L}(\mathbb{R}^p)$ has rank r, one could find an operator $B(\omega) \in \mathcal{L}(\mathbb{R}^p, \mathbb{R}^r)$ such that $A(\omega) = B(\omega)B(\omega)^*$. Moreover, in light of equation 3.24 the regularization property of the kernel with respect to the Fourier transform, it is also possible to define an approximate feature map of an Operator-Valued Kernel from its regularization properties in the $\mathcal{Y}$-RKHS as proposed in **??** 2.

3.4 OPERATOR RANDOM FEATURE ENGINEERING

As in the scalar case, it is possible to construct. We list some examples in the following.

3.4.0.1 *Sum of kernels*

Proposition 3.11 (Sum of kernels). *Let* I *be a countable set and let* $(K^i)_{i\in I}$ *be a familly of* $\mathcal{Y}$*-reproducing kernels such that for all* $y \in \mathcal{Y}$

$$\sum_{i\in I} \langle y, K^i(x,x)y\rangle < \infty.$$

1 Note that since $B(\omega)$ is bounded the pseudo inverse of $B(\omega)$ is well defined for $\widehat{\mathbf{Haar}}$-almost all ω. However if $B(\omega)$ is infinite dimensional, the pseudo inverse is continuous if and only if $A(\omega)$ has closed range. This is always true if $\mathcal{Y}$ is finite dimensional.

Algorithm 2: Construction of ORFF

Input :

- The pairing (x, ω) of the LCA group $(\mathcal{X}, \star)$.
- A probability measure $\mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho}$ with density ρ w. r. t. the haar measure $\widehat{\mathbf{Haar}}$ on $\widehat{\mathcal{X}}$.
- An operator-valued function $B : \widehat{\mathcal{X}} \to \mathcal{L}(\mathcal{Y}, \mathcal{Y}')$ such that for all y $y' \in \mathcal{Y}$, $\langle y', B(\cdot)B(\cdot)^* y \rangle \in L^1(\widehat{\mathcal{X}}, \mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho})$.
- D the number of features.

Output : A random feature $\widetilde{\Phi}(x)$ such that $\widetilde{\Phi}(x)^* \widetilde{\Phi}(z) \approx K(x, z)$.

1 Draw D random vectors $(\omega_j)_{j=1}^D$ i. i. d. from the probability law $\mathbf{Pr}_{\widehat{\mathbf{Haar}}, \rho}$;

2 **return** $\begin{cases} \widetilde{\Phi}(x) \in \mathcal{L}(\mathcal{Y}, \widetilde{\mathcal{H}}) & : y \mapsto \frac{1}{\sqrt{D}} \bigoplus_{j=1}^D (x, \omega_j) B(\omega_j)^* y \\ \widetilde{\Phi}(x)^* \in \mathcal{L}(\widetilde{\mathcal{H}}, \mathcal{Y}) & : \theta \mapsto \frac{1}{\sqrt{D}} \sum_{j=1}^D (x, \omega_j) B(\omega_j) \theta_j \end{cases}$;

Given $x, z \in \mathcal{X}$, the serie $\sum_{i \in I} K^i(x, z)$ converges to a bounded operator $K(x, z)$ in the strong operator topology, and the map $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ defined by

$$K(x, z) y = \sum_{i \in I} K^i(x, z) y$$

is a $\mathcal{Y}$-reproducing kernel. The corresponding space $\mathcal{H}_K$ is embedded in $\oplus_{i \in I} \mathcal{H}_{K^i}$ by means of the feature operator

$$(Wf)(x) = \sum_{i \in I} f_i$$

where $f = \oplus_{i \in I} f_i$, $f_i \in \mathcal{H}_{K^i}$, and the sum converges in norm. Moreover if each K^i is a Mercer kernel –resp. $\mathcal{C}_0$-kernel– and $x \mapsto \sum_{i \in I} \left\| K^i(x, x) \right\|_{\mathcal{Y}, \mathcal{Y}}$ is locally bounded –resp. bounded– then K is Mercer –resp. $\mathcal{C}_0$.

3.5 CONCLUSIONS

4

LEARNING WITH FEATURE MAPS

4.1 LEARNING WITH ORFF

We now turn our attention to learning function with an ORFF model that approximate an OVK model.

4.1.1 *Warm-up: supervised regression*

Let $\mathbf{s} = (x_i, y_i)_{i=1}^N \in (\mathcal{X} \times \mathcal{Y})^N$ be a sequence of training samples. Given a local loss function $L : \mathcal{X} \times \mathcal{F} \times \mathcal{Y} \to \overline{\mathbb{R}}$ such that L is proper, convex and lower semi-continous in f, we are interested in finding a *vector-valued function* $f_{\mathbf{s}} : \mathcal{X} \to \mathcal{Y}$, that lives in a $\mathcal{Y}$-RKHS and minimize a tradeoff between a data fitting term L and a regularization term to prevent from overfitting. Namely finding $f_{\mathbf{s}} \in \mathcal{H}_K$ such that

$$f_{\mathbf{s}} = \arg\min_{f \in \mathcal{H}_K} \frac{1}{N} \sum_{i=1}^{N} L(x_i, f, y_i) + \frac{\lambda}{2} \|f\|_K^2, \tag{4.1}$$

[10] *Tychonov regularization.*

where $\lambda \in \mathbb{R}_+$ is a regularization[10] parameter. We call the quantity

$$\mathcal{R}(f) = \frac{1}{N} \sum_{i=1}^{N} L(x_i, f, y_i), \quad \forall f \in \mathcal{H}_K, \forall \mathbf{s} \in (\mathcal{X} \times \mathcal{Y})^N.$$

The empirical risk of the model $f \in \mathcal{H}_K$. A common choice of data fitting term for regression is $L : (x_i, f, y_i) \mapsto \|f(x_i) - y_i\|_{\mathcal{Y}}^2$. We introduce a corollary from Mazur and Schauder proposed in 1936 (see Górniewicz [19] and Kurdila and Zabarankin [24]) showing that equation 4.1 –and equation 4.3– attains a unique mimimizer.

Theorem 4.1 (Mazur-Schauder). *Let $\mathcal{H}$ be a Hilbert space and $J : \mathcal{H} \to \overline{\mathbb{R}}$ be a proper, convex, lower semi-continuous and coercive function. Then J is bounded from below and attains a minimizer. Moreover if J is strictly convex the minimizer is unique.*

This is easily verified for Ridge regression. Define

$$J_\lambda(f) = \frac{1}{N} \sum_{i=1}^{N} \|f(x_i) - y_i\|_{\mathcal{Y}}^2 + \frac{\lambda}{2} \|f\|_K^2, \tag{4.2}$$

[11] *Reminder, if $f \in \mathcal{H}_k$, $ev_x : f \mapsto f(x)$ is continuous, see proposition 2.1.*

where $f \in \mathcal{H}_K$ and $\lambda \in \mathbb{R}_{>0}$. J_λ is continuous[11] and strictly convex. Additionally J_λ is coercive since $\|f\|_K$ is coercive, $\lambda \in \mathbb{R}_{>0}$, and all the summands of J_λ are positive. Hence for all positive λ, $f_{\mathbf{s}} = \arg\min_{f \in \mathcal{H}_K} J_\lambda(f)$ exists, is unique and attained.

Remark 4.1 *We condider the optimization problem proposed in equation 4.2 where $L : (x_i, f, y_i) \mapsto \|f(x_i) - y_i\|_{\mathcal{Y}}^2$. If given a training sample $\mathbf{s}$, we have*

$$\frac{1}{N} \sum_{i=1}^{N} \|y_i\|_{\mathcal{Y}}^2 \leqslant \sigma_{\mathcal{Y}}^2,$$

then $\lambda\|f_s\|_K \leqslant 2\sigma_y^2$. *Indeed, since* $\mathcal{H}_K$ *is a Hilbert space,* $0 \in \mathcal{H}_K$, *thus*

$$\begin{aligned}\frac{\lambda}{2}\|f_s\|_K^2 &\leqslant \frac{1}{N}\sum_{i=1}^{N} L(x_i, f_s, y_i) + \frac{\lambda}{2}\|f_s\|_K^2 \\ &\leqslant \frac{1}{N}\sum_{i=1}^{N} L(x_i, 0, y_i) \leqslant \sigma_y^2, \quad \textit{by optimality of } f_s.\end{aligned}$$

Since for all $x \in \mathcal{X}$, $\|f(x)\|_{\mathcal{Y}} \leqslant \sqrt{\|K(x,x)\|_{\mathcal{Y},\mathcal{Y}}}\|f\|_K$, *the maximum value that the solution* $\|f_s(x)\|_{\mathcal{Y}}$ *of equation 4.2 can reach is* $2\sqrt{\|K(x,x)\|}\frac{\sigma_y^2}{\lambda}$. *Thus when solving a Ridge regression problem, given a shift-invariant kernel* K_e, *one should choose*

$$0 < \lambda \leqslant 2\frac{\sqrt{\|K_e(e)\|}\sigma_y^2}{C}.$$

with $C \in \mathbb{R}_{>0}$ *to have a chance to fit all the* y_i *with norm* $\|y_i\|_{\mathcal{Y}} \leqslant C$ *in the train set.*

4.1.2 *Semi-supervised regression*

Regression in $\mathcal{Y}$-Reproducing Kernel Hilbert Space has been well studied [2, 10, 22, 28, 30, 33, 39], and a cornerstone of learning in $\mathcal{Y}$-RKHS is the representer theorem[12] , which allows to replace the search of a minimizer in a infinite dimensional $\mathcal{Y}$-RKHS by a finite number of paramaters $(u_i)_{i=1}^N$, $u_i \in \mathcal{Y}$. We present here the very genreal form of Minh, Bazzani, and Murino [33]. This framework encompass Vector-valued Manifold Regularization [5, 9, 32] and Co-regularized Multi-view Learning [8, 38, 41, 44].

[12] *Sometimes referred to as minimal norm interpolation theorem.*

In the following we suppose we are given a cost function $c : \mathcal{Y} \times \mathcal{Y} \to \overline{\mathbb{R}}$, such that $c(f(x), y)$ returns the error of the prediction $f(x)$ w. r. t. the ground truth y. A loss function of a model f with respect to an example $(x, y) \in \mathcal{X} \times \mathcal{Y}$ can be naturally defined from a cost function as $L(x, f, y) = c(f(x), y)$. Conceptually the function c evaluate the quality of the prediction versus its ground truth $y \in \mathcal{Y}$ while the loss function L evaluate the quality of the model f at a training point $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Moreover we suppose that we are given a training sample $\mathbf{u} = (x_i)_{i=N}^{N+U} \in \mathcal{X}^U$ of unlabelled exemple. We note $\mathbf{z} \in (\mathcal{X} \times \mathcal{Y})^N \times \mathcal{X}^U$ the sequence $\mathbf{z} = \mathbf{su}$ concatenating both labeled ($\mathbf{s}$) and unlabelled ($\mathbf{u}$) training examples.

Theorem 4.2 (Representer [33]). *Let* K *be a* $\mathcal{U}$*-Mercer Operator-Valued Kernel and* $\mathcal{H}_K$ *its corresponding* $\mathcal{U}$*-Reproducing Kernel Hilbert Space.*

Let $V : \mathcal{U} \to \mathcal{Y}$ *be a bounded linear operator and let* $c : \mathcal{Y} \times \mathcal{Y} \to \overline{\mathbb{R}}$ *be a cost function such that* $L(x, f, y) = c(Vf(x), y)$ *is a proper convex lower semi-continuous function in* f *for all* $x \in \mathcal{X}$ *and all* $y \in \mathcal{Y}$.

Eventually let $\lambda_K \in \mathbb{R}_{>0}$ *and* $\lambda_M \in \mathbb{R}_+$ *be two regularization hyperparameters and* $(M_{ik})_{i,k=1}^{N+U}$ *be a sequence of data dependent bounded linear operators in* $\mathcal{L}(\mathcal{U})$*, such that*

$$\sum_{i,j=1}^{N+U} \langle u_i, M_{ik} u_k \rangle \geqslant 0, \quad \forall (u_i)_{i=1}^{N+U} \in \mathcal{U}^{N+U} \text{ and } M_{ik} = M_{ki}^*.$$

The solution $f_z \in \mathcal{H}_K$ *of the regularized optimization problem*

$$\begin{aligned} f_z = \arg\min_{f \in \mathcal{H}_K} & \frac{1}{N} \sum_{i=1}^{N} c(Vf(x_i), y_i) + \frac{\lambda_K}{2} \|f\|_K^2 \\ & + \frac{\lambda_M}{2} \sum_{i,k=1}^{N+U} \langle f(x_i), M_{ik} f(x_k) \rangle_{\mathcal{U}} \end{aligned} \tag{4.3}$$

has the form $f_z = \sum_{j=1}^{N+U} K(\cdot, x_j) u_{z,j}$ *where* $u_{z,j} \in \mathcal{U}$ *and*

$$\begin{aligned} u_z = \arg\min_{u \in \bigoplus_{i=1}^{N+U} \mathcal{U}} & \frac{1}{N} \sum_{i=1}^{N} c\left(V \sum_{k=1}^{N+U} K(x_i, x_j) u_j, y_i \right) \\ & + \frac{\lambda_K}{2} \sum_{k=1}^{N+U} u_i^* K(x_i, x_k) u_k \\ & + \frac{\lambda_M}{2} \sum_{i,k=1}^{N+U} \left\langle \sum_{j=1}^{N+U} K(x_i, x_j) u_j, M_{ik} \sum_{j=1}^{N+U} K(x_k, x_j) u_j \right\rangle_{\mathcal{U}} \end{aligned} \tag{4.4}$$

The first representer theorem was first introduced by Wahba [48] in the case where $\mathcal{Y} = \mathbb{R}$. The extension to an arbitrary Hilbert space $\mathcal{Y}$ has been proved by many authors in different forms [9, 22, 28]. The idea behind the representer theorem is that eventhough we minimize over the whole space $\mathcal{H}_K$, when $\lambda_K > 0$, the solution of equation 4.3 falls inevitably into the set $\mathcal{H}_{K,z} = \left\{ \sum_{j=1}^{N+U} K_{x_j} u_j \;\middle|\; \forall (u_i)_{i=1}^{N+U} \in \mathcal{U}^{N+U} \right\}$. Therefore the result can be expressed as a finite linear combination of basis functions of the form $K(\cdot, x_k)$. Remark that we can perform the kernel expansion of $f_z = \sum_{j=1}^{N+U} K(\cdot, x_j) u_{z,j}$ eventhough $\lambda_K = 0$. However f_z is no longer the solution of equation 4.3 over the whole space $\mathcal{H}_K$ but a projection on the subspace $\mathcal{H}_{K,z}$. While this is in general not a problem for practical applications, it might raise issues for further theoretical investigations. In particular, it makes it difficult to perform theoretical comparison the "exact" solution of equation 4.3 with respect to the ORFF approximation solution given in corollary 4.1.

We present here the proof of the generic formulation proposed by Minh, Bazzani, and Murino [33]. In the mean time we clarify some elements of the proof. Indeed the existence of a global minimizer is not trivial and we must invoke the Mazur-Schauder theorem. Moreover the coercivity of the objective function required by the Mazur-Schauder theorem is

not obvious when we do not require the cost function to take only positive values. However a corollary of Hahn-Banach theorem linking strong convexity to coercivity gives the solution.

Proof *Since* $f(x) = K_x^* f$ *(see equation 2.13), the optimization problem reads*

$$f_z = \arg\min_{f \in \mathcal{H}_K} \frac{1}{N} \sum_{i=1}^{N} c(VK_{x_i}^* f, y_i) + \frac{\lambda_K}{2} \|f\|_K^2 + \frac{\lambda_M}{2} \sum_{i,k=1}^{N+U} \langle K_{x_i}^* f, M_{ik} K_{x_k}^* f \rangle_{\mathcal{U}}$$

Let $W_{V,\mathbf{s}} : \mathcal{H}_K \to \bigoplus_{i=1}^{N} \mathcal{Y}$ *be a linear operator defined as*

$$W_{V,\mathbf{s}} f = \bigoplus_{i=1}^{N} C K_{x_i}^* f,$$

with $VK_{x_i}^* : \mathcal{H}_K \to \mathcal{Y}$ *and* $K_{x_i} V^* : \mathcal{Y} \to \mathcal{H}_K$. *Let* $Y = \bigoplus_{i=1}^{N} y_i \in \mathcal{Y}^N$. *We have*

$$\langle Y, W_{V,\mathbf{s}} f \rangle_{\bigoplus_{i=1}^{N} \mathcal{Y}} = \sum_{i=1}^{N} \langle y_i, VK_{x_i}^* f \rangle_{\mathcal{Y}} = \sum_{i=1}^{N} \langle K_{x_i} V^* y_i, f \rangle_{\mathcal{H}_K}.$$

Thus the adjoint operator $W_{V,\mathbf{s}}^* : \bigoplus_{i=1}^{N} \mathcal{Y} \to \mathcal{H}_K$ *is*

$$W_{V,\mathbf{s}}^* Y = \sum_{i=1}^{N} K_{x_i} V^* y_i,$$

and the operator $W_{V,\mathbf{s}}^* W_{V,\mathbf{s}} : \mathcal{H}_K \to \mathcal{H}_K$ *is*

$$W_{V,\mathbf{s}}^* W_{V,\mathbf{s}} f = \sum_{i=1}^{N} K_{x_i} V^* V K_{x_i}^* f$$

where $V^* V \in \mathcal{L}(\mathcal{U})$. *Let*

$$J_{\lambda_K}(f) = \underbrace{\frac{1}{N} \sum_{i=1}^{N} c(Vf(x_i), y_i)}_{=J_c} + \frac{\lambda_K}{2} \|f\|_K^2 + \underbrace{\frac{\lambda_M}{2} \sum_{i,k=1}^{N+U} \langle f(x_i), M_{ik} f(x_k) \rangle_{\mathcal{U}}}_{=J_M}$$

To ensure that J_{λ_K} *has a global minimizer we need the following technical lemma (which is a consequence of the Hahn-Banach theorem for lower-semicontimuous functional, see Kurdila and Zabarankin [24]).*

Lemma 4.1 *Let* J *be a proper, convex, lower semi-continuous functional, defined on a Hilbert space* $\mathcal{H}$. *If* J *is strongly convex, then* J *is coercive.*

Proof *Consider the convex function function* $G(f) := J(f) - \lambda\|f\|^2$*, for some* $\lambda > 0$*. Since* J *is by assumption proper, lower semi-continuous and strongly convex with parameter* λ*,* G *is proper, lower semi-continuous and convex. Thus Hahn-Banach theorem apply, stating that* G *is bounded by below by an affine functional. I. e. there exists* f_0 *and* $f_1 \in \mathcal{H}$ *such that*

$$G(f) \geqslant G(f_0) + \langle f - f_0, f_1 \rangle, \quad \textit{for all } f \in \mathcal{H}.$$

Then substitute the definition of G *to obtain*

$$J(f) \geqslant J(f_0) + \lambda\left(\|f\| - \|f_0\|\right) + \langle f - f_0, f_1 \rangle.$$

By the Cauchy-Schwartz inequality, $\langle f, f_1 \rangle \geqslant -\|f\|\|f_1\|$*, thus*

$$J(f) \geqslant J(f_0) + \lambda\left(\|f\| - \|f_0\|\right) - \|f\|\|f_1\| - \langle f_0, f_1 \rangle,$$

which tends to infinity as f *tends to infinity. Hence* J *is coercive* □

for all $f \in \mathcal{H}_K$*. Since* c *is proper, lower semi-continuous and convex by assumption, thus the term* J_c *is also proper, lower semi-continuous and convex. Moreover the term* J_M *is always positive for any* $f \in \mathcal{H}_K$ *and* $\frac{\lambda_K}{2}\|f\|_K^2$ *is strongly convex. Thus* J_{λ_K} *is strongly convex. Apply lemma 4.1 to obtain the coercivity of* J_{λ_K}*, and then theorem 4.1 to show that* J_{λ_K} *has a unique minimizer and is attained. Then let* $\mathcal{H}_{K,z} = \left\{ \sum_{j=1}^{N+U} K_{x_j} u_j \;\middle|\; \forall (u_i)_{i=1}^{N+U} \in \mathcal{U}^{N+U} \right\}$*. For* $f \in \mathcal{H}_{K,z}^{\perp}$ [13]*, the operator* $W_{V,s}$ *satisfies*

[13] $\mathcal{H}_{K,z}^{\perp} \oplus \mathcal{H}_{K,z} = \mathcal{H}_{\mathcal{K}}$.

$$\langle Y, W_{V,s} f \rangle_{\bigoplus_{i=1}^{N} \mathcal{Y}} = \langle \underbrace{f}_{\in \mathcal{H}_{K,z}^{\perp}}, \underbrace{\sum_{i=1}^{N+U} K_{x_i} V^* y_i}_{\in \mathcal{H}_{K,z}} \rangle_{\mathcal{H}_K} = 0$$

for all sequences $(y_i)_{i=1}^N$*, since* $V^* y_i \in \mathcal{U}$*. Hence,*

$$(Vf(x_i))_{i=1}^N = 0 \tag{4.5}$$

In the same way,

$$\sum_{i=1}^{N+U} \langle K_{x_i}^* f, u_i \rangle_{\mathcal{U}} = \langle \underbrace{f}_{\in \mathcal{H}_{K,z}^{\perp}}, \underbrace{\sum_{j=1}^{N+U} K_{x_j} u_j}_{\in \mathcal{H}_{K,z}} \rangle_{\mathcal{H}_K} = 0.$$

for all sequences $(u_i)_{i=1}^{N+U} \in \mathcal{U}^{N+U}$*. As a result,*

$$(f(x_i))_{i=1}^{U+N} = 0. \tag{4.6}$$

Now for an arbitrary $f \in \mathcal{H}_{\mathcal{K}}$*, consider the orthogonal decomposition* $f = f^{\perp} + f^{\parallel}$*, where* $f^{\perp} \in \mathcal{H}_{K,z}^{\perp}$ *and* $f^{\parallel} \in \mathcal{H}_{K,z}$*. Then since* $\left\|f^{\perp} + f^{\parallel}\right\|_{\mathcal{H}_K}^2 = \left\|f^{\perp}\right\|_{\mathcal{H}_K}^2 + \left\|f^{\parallel}\right\|_{\mathcal{H}_K}^2$*, equation 4.5 and equation 4.6 shows that if* $\lambda_K \geqslant 0$*, clearly then*

$$J_{\lambda_K}(f) = J_{\lambda_K}\left(f^{\perp} + f^{\parallel}\right) \geqslant J_{\lambda_K}\left(f^{\parallel}\right)$$

The last inequality holds only when $\left\|f^{\perp}\right\|_{\mathcal{H}_K} = 0$*, that is when* $f^{\perp} = 0$*. As a result since the minimizer of* J_{λ_K} *is unique and attained, it must lies in* $\mathcal{H}_{K,z}$*.* □

The representer theorem show that minimizing a functional in a $\mathcal{Y}$-RKHS yields a solution which depends on all the points in the training set. Assuming that for all x_i, $x \in \mathcal{X}$ and for all $u_i \in \mathcal{Y}$ it takes time $O(P)$, to compute $K(x_i, x)u_i$, making a prediction using the representer theorem take $O(2P)$. Obviously If $\mathcal{Y} = \mathbb{R}^p$, Then $P = O(p^2)$ thus making a prediction cost $O(2p^2)$ operations.

Instead learning a model f that depends on all the points of the training set, we would like to learn a parametric model of the form $\widetilde{f}(x) = \widetilde{\Phi}(x)^*\theta$, where θ lives in some redescription space $\widetilde{\mathcal{H}}$. We are interested in finding a parameter vector θ_z such that

$$\begin{aligned}\theta_z = \arg\min_{\theta\in\widetilde{\mathcal{H}}} \frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) + \frac{\lambda_K}{2}\|\theta\|^2_{\widetilde{\mathcal{H}}} \\ + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\langle\theta, \widetilde{\Phi}(x_i)M_{ik}\widetilde{\Phi}(x_k)^*\theta\rangle_{\widetilde{\mathcal{H}}}\end{aligned} \tag{4.7}$$

Corollary 4.1 (ORFF representer). *Let $\widetilde{K}$ be an Operator-Valued Kernel such that for all $x, z \in \mathcal{X}$, $\widetilde{\Phi}(x)^*\widetilde{\Phi}(z) = \widetilde{K}(x, z)$ where $\widetilde{K}$ is a $\mathcal{U}$-Mercer OVK and $\mathcal{H}_{\widetilde{K}}$ its corresponding $\mathcal{U}$-Reproducing kernel Hilbert space.*

Let $V : \mathcal{U} \to \mathcal{Y}$ be a bounded linear operator and let $c : \mathcal{Y} \times \mathcal{Y} \to \overline{\mathbb{R}}$ be a cost function such that $L(x, \widetilde{f}, y) = c(V\widetilde{f}(x), y)$ is a proper convex lower semi-continuous function in $\widetilde{f} \in \mathcal{H}_{\widetilde{K}}$ for all $x \in \mathcal{X}$ and all $y \in \mathcal{Y}$.

Eventually let $\lambda_K \in \mathbb{R}_{>0}$ and $\lambda_M \in \mathbb{R}_+$ be two regularization hyperparameters and $(M_{ik})_{i,k=1}^{N+U}$ be a sequence of data dependent bounded linear operators in $\mathcal{L}(\mathcal{U})$, such that

$$\sum_{i,j=1}^{N+U}\langle u_i, M_{ik}u_k\rangle \geqslant 0, \quad \forall (u_i)_{i=1}^{N+U} \in \mathcal{U}^{N+U} \text{ and } M_{ik} = M_{ki}^*.$$

The solution $f_z \in \mathcal{H}_{\widetilde{K}}$ of the regularized optimization problem

$$\begin{aligned}\widetilde{f}_z = \arg\min_{\widetilde{f}\in\mathcal{H}_{\widetilde{K}}} \frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{f}(x_i), y_i\right) + \frac{\lambda_K}{2}\left\|\widetilde{f}\right\|^2_{\widetilde{K}} \\ + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\langle\widetilde{f}(x_i), M_{ik}\widetilde{f}(x_k)\rangle_{\mathcal{U}}\end{aligned} \tag{4.8}$$

has the form $\widetilde{f}_z = \widetilde{\Phi}(\cdot)^*\theta_z$, where $\theta_z \in (\mathrm{Ker}\ \widetilde{W})^\perp$ and

$$\begin{aligned}\theta_z = \arg\min_{\theta\in\widetilde{\mathcal{H}}} \frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) + \frac{\lambda_K}{2}\|\theta\|^2_{\widetilde{\mathcal{H}}} \\ + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\langle\theta, \widetilde{\Phi}(x_i)M_{ik}\widetilde{\Phi}(x_k)^*\theta\rangle_{\widetilde{\mathcal{H}}}.\end{aligned} \tag{4.9}$$

Proof *Since $\widetilde{K}$ is an operator-valued kernel, from theorem 4.2, equation 4.8 has a solution of the form*

$$\begin{aligned}\widetilde{f}_z &= \sum_{i=1}^{N+U} \widetilde{K}(\cdot, x_i)u_i, \quad u_i \in \mathcal{U}, x_i \in \mathcal{X} \\ &= \sum_{i=1}^{N} \widetilde{\Phi}(\cdot)^*\widetilde{\Phi}(x_i)u_i = \widetilde{\Phi}(\cdot)^* \underbrace{\left(\sum_{i=1}^{N+U} \widetilde{\Phi}(x_i)u_i\right)}_{=\theta \in \left(Ker\ \widetilde{W}\right)^\perp \subset \widetilde{\mathcal{H}}}.\end{aligned}$$

Let

$$\begin{aligned}\theta_z = \underset{\theta \in \left(Ker\ \widetilde{W}\right)^\perp}{\arg\min} &\frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) + \frac{\lambda_K}{2}\left\|\widetilde{\Phi}(\cdot)^*\theta\right\|_{\widetilde{K}}^2 \\ &+ \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \widetilde{\Phi}(x_i)^*\theta, M_{ik}\widetilde{\Phi}(x_k)^*\theta\right\rangle_{\mathcal{U}}.\end{aligned}$$

Since $\theta \in (Ker\ \widetilde{W})^\perp$ and W is an isometry from $(Ker\ \widetilde{W})^\perp \subset \widetilde{\mathcal{H}}$ onto $\mathcal{H}_{\widetilde{K}}$, we have $\left\|\widetilde{\Phi}(\cdot)^*\theta\right\|_{\widetilde{K}}^2 = \|\theta\|_{\widetilde{\mathcal{H}}}^2$. Hence

$$\begin{aligned}\theta_z = \underset{\theta \in \left(Ker\ \widetilde{W}\right)^\perp}{\arg\min} &\frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) + \frac{\lambda_K}{2}\|\theta\|_{\widetilde{\mathcal{H}}}^2 \\ &+ \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\langle \widetilde{\Phi}(x_i)^*\theta, M_{ik}\widetilde{\Phi}(x_k)^*\theta\rangle_{\mathcal{U}}.\end{aligned}$$

Finding a minimizer θ_z over $\left(Ker\ \widetilde{W}\right)^\perp$ is not the same than finding a minimizer over $\widetilde{\mathcal{H}}$. Although in both cases Mazur-Schauder's theorem guarantee that the respective minimizers are unique, they might not be the same. Since $\widetilde{W}$ is bounded, Ker $\widetilde{W}$ is closed, so that we can perform the decomposition $\widetilde{\mathcal{H}} = \left(Ker\ \widetilde{W}\right)^\perp \oplus \left(Ker\ \widetilde{W}\right)$. Then clearly by linearity of W and the fact that for all $\theta^{\|} \in Ker\ \widetilde{W}$, $\widetilde{W}\theta^{\|} = 0$, if $\lambda > 0$ we have

$$\begin{aligned}\theta_z = \underset{\theta \in \widetilde{\mathcal{H}}}{\arg\min} &\frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) + \frac{\lambda_K}{2}\|\theta\|_{\widetilde{\mathcal{H}}}^2 \\ &+ \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \widetilde{\Phi}(x_i)^*\theta, M_{ik}\widetilde{\Phi}(x_k)^*\theta\right\rangle_{\mathcal{U}}\end{aligned}$$

Thus

$$\begin{aligned}\theta_z = \underset{\substack{\theta^\perp \in \left(Ker\ \widetilde{W}\right)^\perp, \\ \theta^\parallel \in Ker\ \widetilde{W}}}{\arg\min} & \frac{1}{N}\sum_{i=1}^{N} c\left(V\left(\widetilde{W}\theta^\perp\right)(x) + \underbrace{V\left(\widetilde{W}\theta^\parallel\right)(x)}_{=0 \ \text{for all} \ \theta^\parallel}, y_i\right) \\ & + \frac{\lambda_K}{2}\left\|\theta^\perp\right\|_{\widetilde{\mathcal{H}}}^2 + \underbrace{\frac{\lambda}{2}\left\|\theta^\parallel\right\|_{\widetilde{\mathcal{H}}}^2}_{=0 \ \text{only if} \ \theta^\parallel = 0} \\ & + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \widetilde{\Phi}(x_i)^*\theta^\perp, M_{ik}\left(\widetilde{W}\theta^\perp\right)(x_k)\right\rangle_{\mathcal{U}} \\ & + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \underbrace{\left(\widetilde{W}\theta^\parallel\right)(x_i)}_{=0 \ \text{for all} \ \theta^\parallel}, M_{ik}\left(\widetilde{W}\theta^\perp\right)(x_k)\right\rangle_{\mathcal{U}} \\ & + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \left(\widetilde{W}\theta^\perp\right)(x_i), M_{ik}\underbrace{\left(\widetilde{W}\theta^\parallel\right)(x_k)}_{=0 \ \text{for all} \ \theta^\parallel}\right\rangle_{\mathcal{U}} \\ & + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \underbrace{\left(\widetilde{W}\theta^\parallel\right)(x_i)}_{=0 \ \text{for all} \ \theta^\parallel}, M_{ik}\underbrace{\left(\widetilde{W}\theta^\parallel\right)(x_k)}_{=0 \ \text{for all} \ \theta^\parallel}\right\rangle_{\mathcal{U}}.\end{aligned}$$

Thus

$$\begin{aligned}\theta_z = \underset{\theta^\perp \in \left(Ker\ \widetilde{W}\right)^\perp}{\arg\min} & \frac{1}{N}\sum_{i=1}^{N} c\left(V\left(\widetilde{W}\theta^\perp\right)(x), y_i\right) + \frac{\lambda_K}{2}\left\|\theta^\perp\right\|_{\widetilde{\mathcal{H}}}^2 \\ & + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \widetilde{\Phi}(x_i)^*\theta^\perp, M_{ik}\left(\widetilde{W}\theta^\perp\right)(x_k)\right\rangle_{\mathcal{U}}.\end{aligned}$$

Hence minimizing over $\left(Ker\ \widetilde{W}\right)^\perp$ *or* $\widetilde{\mathcal{H}}$ *is the same when* $\lambda_K > 0$. *Eventually,*

$$\begin{aligned}\theta_z &= \underset{\theta \in \widetilde{\mathcal{H}}}{\arg\min}\ \frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) + \frac{\lambda_K}{2}\|\theta\|_{\widetilde{\mathcal{H}}}^2 \\ &\quad + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \widetilde{\Phi}(x_i)^*\theta, M_{ik}\widetilde{\Phi}(x_k)^*\theta\right\rangle_{\mathcal{U}} \\ &= \underset{\theta \in \widetilde{\mathcal{H}}}{\arg\min}\ \frac{1}{N}\sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) + \frac{\lambda_K}{2}\|\theta\|_{\widetilde{\mathcal{H}}}^2 \\ &\quad + \frac{\lambda_M}{2}\sum_{i,k=1}^{N+U}\left\langle \theta, \widetilde{\Phi}(x_i)M_{ik}\widetilde{\Phi}(x_k)^*\theta\right\rangle_{\widetilde{\mathcal{H}}}.\end{aligned}$$

This shows that when $\lambda_K > 0$ the solution of equation 4.4 with the approximated kernel $K(x,z) \approx \widetilde{K}(x,z) = \widetilde{\Phi}(x)^*\widetilde{\Phi}(z)$ is the same than the

solution of equation 4.9 up to a transformation. Namely, if u_z is the solution of equation 4.4, θ_z is the solution of equation 4.9 and $\lambda_K > 0$ we have

$$\theta_z = \sum_{j=1}^{N+U} \widetilde{\Phi}(x_j) u_z \in \widetilde{\mathcal{H}}.$$

If $\lambda_K = 0$ we can still find a solution u_z of equation 4.4. By construction of the kernel expansion, we have $u_z \in (\operatorname{Ker} W)^\perp$. However looking at the proof of corollary 4.1 we see that θ_z might *not* g belong to $(\operatorname{Ker} W)^\perp$. Then we can compute a residual vector

$$r_z = \theta_z - \sum_{j=1}^{N+U} \widetilde{\Phi}(x_j) u_z \in \widetilde{\mathcal{H}}.$$

Then if $r_z = 0$, it mkeans that λ_K is large enough for both reprensenter theorem and ORFF representer theorem to apply. If $r_z \neq 0$ but $\widetilde{\Phi}(\cdot)^* r_z = 0$ means that both θ_z and $\sum_{j=1}^{N+U} \widetilde{\Phi}(x_j) u_z$ are in $(\operatorname{Ker} W)^\perp$, thus the representer theorem fails to find the "true" solution over the whole space $\mathcal{H}_K$ but returns a projection onto $\mathcal{H}_{\widetilde{K},z}$ of the solution. If $r_z \neq 0$ and $\widetilde{\Phi}(\cdot)^* r_z \neq 0$ means that θ_z is *not* in $(\operatorname{Ker} W)^\perp$, thus the ORFF representer theorem fails to apply. This remark is illustrated by figure 4.

4.2 SOLUTION OF THE EMPIRICAL RISK MINIMIZATION

4.2.1 *Gradient methods*

In order to find a solution to equation 4.9, we turn our attention to gradient descent methods. In the following we let

$$\begin{aligned} J_{\lambda_K}(\theta) = \frac{1}{N} \sum_{i=1}^{N} c\left(V\widetilde{\Phi}(x_i)^* \theta, y_i\right) + \frac{\lambda_K}{2} \|\theta\|^2_{\widetilde{\mathcal{H}}} \\ + \frac{\lambda_M}{2} \sum_{i,k=1}^{N+U} \langle \theta, \widetilde{\Phi}(x_i) M_{ik} \widetilde{\Phi}(x_k)^* \theta \rangle_{\widetilde{\mathcal{H}}}. \end{aligned} \tag{4.10}$$

Since the solution of equation 4.9 is unique when $\lambda_K > 0$, a sufficient and necessary condition is that the gradient of J_{λ_K} at the minimizer θ_z is zero. We use the Frechet derivative, the strongest notion of derivative in Banach spaces[14] [14, 24] which directly generalizes the notion of gradient to Banach spaces. A function $f : \mathcal{H}_0 \to \mathcal{H}_1$ is call Frechet differentiable at $\theta_0 \in \mathcal{H}_0$ if there exist a bounded linear operator $A : \mathcal{H}_0 \to \mathcal{H}_1$ such that

$$\lim_{h \to 0} \frac{\|f(\theta_0 + h) - f(\theta_0) - Ah\|_{\mathcal{H}_1}}{\|h\|_{\mathcal{H}_0}} = 0$$

We write

$$(D_F f)(\theta_0) = \left.\frac{\partial f(\theta)}{\partial \theta}\right|_{\theta=\theta_0} = A$$

[14] *Here we view the Hilbert space $\mathcal{H}$ (feature space) as a reflexive Banach space.*

and call it Frechet derivative of f with respect to θ at θ_0. With mild abuse of notation we write $\frac{\partial f(\theta_0)}{\partial \theta_0} = \frac{\partial f(\theta)}{\partial \theta}\Big|_{\theta=\theta_0}$. The Frechet derivative is an unbounded linear operator. The chain rule is valid in this context. Namely if a function $f : \mathcal{H}_0 \to \mathcal{H}_1$ is Frechet differentiable at θ and $g : \mathcal{H}_1 \to \mathcal{H}_2$ is Frechet differentiable at $f(\theta)$ then $g \circ f$ is Frechet differentiable at θ and

$$\frac{\partial}{\partial \theta}(g \circ f)(\theta) = \frac{\partial g(f(\theta))}{\partial f(\theta)} \circ \frac{\partial f(\theta)}{\partial \theta}.$$

If $f : \mathcal{H} \to \mathbb{R}$ we define the gradient of f as

$$\nabla_\theta f(\theta) = \left(\frac{\partial f(\theta)}{\partial \theta}\right)^*$$

and in the same way, for a function $f : \mathcal{H}_0 \to \mathcal{H}_1$ the jacobian of f as

$$\mathbf{J}_\theta f(\theta) = \left(\frac{\partial f(\theta)}{\partial \theta}\right)^*.$$

In this context if $f : \mathcal{H}_0 \to \mathcal{H}_1$ and $g : \mathcal{H}_1 \to \mathbb{R}$ the chain rule reads

$$\nabla_\theta (g \circ f)(\theta) = \mathbf{J}_\theta f(\theta) \circ \nabla_{f(\theta)} g(f(\theta)).$$

By linearity and applying the chaine rule to equation 4.9 and since $M^*_{ik} = M_{ki}$ for all $i, k \in \mathbb{N}^*_{N+U}$, we have

$$\begin{aligned}
\nabla_\theta c\left(V\widetilde{\Phi}(x_i)^*\theta, y_i\right) &= \widetilde{\Phi}(x_i)V^*\left(\frac{\partial}{\partial y} c\,(y, y_i)\Big|_{y=V\widetilde{\Phi}(x_i)^*\theta}\right)^*, \\
\nabla_\theta\left\langle \widetilde{\Phi}(x_i)^*\theta, M_{ik}\widetilde{\Phi}(x_k)^*\theta\right\rangle_{\mathcal{U}} &= \widetilde{\Phi}(x_i)\left(M_{ik} + M^*_{ki}\right)\widetilde{\Phi}(x_k)^*\theta, \\
\nabla_\theta \|\theta\|^2_{\widetilde{\mathcal{H}}} &= 2\theta.
\end{aligned}$$

Provided that $c(y, y_i)$ is Frechet differentiable w. r. t. y, for all y and $y_i \in \mathcal{Y}$ we have $\nabla_\theta J_{\lambda_K}(\theta) \in \widetilde{\mathcal{H}}$ and

$$\begin{aligned}
\nabla_\theta J_{\lambda_K}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \widetilde{\Phi}(x_i)V^*\left(\frac{\partial}{\partial y} c\,(y, y_i)\Big|_{y=V\widetilde{\Phi}(x_i)^*\theta}\right)^* \\
+ \lambda_K\theta + \lambda_M \sum_{i,k=1}^{N+U} \widetilde{\Phi}(x_i)M_{ik}\widetilde{\Phi}(x_k)^*\theta
\end{aligned}$$

Therefore after factorization, considering $\lambda_K > 0$,

$$\begin{aligned}
\nabla_\theta J_{\lambda_K}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \widetilde{\Phi}(x_i)V^*\left(\frac{\partial}{\partial y} c\,(y, y_i)\Big|_{y=V\widetilde{\Phi}(x_i)^*\theta}\right)^* \\
+ \lambda_K\left(I_{\widetilde{\mathcal{H}}} + \frac{\lambda_M}{\lambda_K}\sum_{i,k=1}^{N+U} \widetilde{\Phi}(x_i)M_{ik}\widetilde{\Phi}(x_k)^*\right)\theta
\end{aligned}$$

We note the quantity

$$\widetilde{\mathbf{M}}_{(\lambda_K,\lambda_M)} = I_{\widetilde{\mathcal{H}}} + \frac{\lambda_M}{\lambda_K} \sum_{i,k=1}^{N+U} \widetilde{\Phi}(x_i) M_{ik} \widetilde{\Phi}(x_k)^* \in \mathcal{L}(\widetilde{\mathcal{H}}) \tag{4.11}$$

so that

$$\nabla_\theta J_{\lambda_K}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \widetilde{\Phi}(x_i) V^* \left(\frac{\partial}{\partial y} c(y, y_i) \Big|_{y = V\widetilde{\Phi}(x_i)^*\theta} \right)^* + \lambda_K \widetilde{\mathbf{M}}_{(\lambda_K,\lambda_M)} \theta$$

Example 4.1 (Naive close form for the squared error cost). *Consider the cost function defined for all* $y, y' \in \mathcal{Y}$ *by* $c(y, y') = \|y - y\|_{\mathcal{Y}}^2$. *Then*

$$\left(\frac{\partial}{\partial y} c(y, y_i) \Big|_{y = V\widetilde{\Phi}(x_i)^*\theta} \right)^* = 2 \left(V\widetilde{\Phi}(x_i)^*\theta - y_i \right).$$

Thus, since the optimal solution θ_z *verifies* $\nabla_{\theta_z} J_{\lambda_K}(\theta_z) = 0$ *we have*

$$\frac{1}{N} \sum_{i=1}^{N} \widetilde{\Phi}(x_i) V^* \left(V\widetilde{\Phi}(x_i)^*\theta_z - y_i \right) + \lambda_K \widetilde{\mathbf{M}}_{(\lambda_K,\lambda_M)} \theta_z = 0.$$

Therefore,

$$\left(\frac{1}{N} \sum_{i=1}^{N} \widetilde{\Phi}(x_i) V^* V \widetilde{\Phi}(x_i)^* + \lambda_K \widetilde{\mathbf{M}}_{(\lambda_K,\lambda_M)} \right) \theta_z = \frac{1}{N} \sum_{i=1}^{N} \widetilde{\Phi}(x_i) V^* y_i. \tag{4.12}$$

Suppose that $\mathcal{Y} \subseteq \mathbb{R}^p$, $V : \mathcal{U} \to \mathcal{Y}$ *where* $\mathcal{U} \subseteq \mathbb{R}^u$ *and for all* $x \in \mathcal{X}$, $\widetilde{\Phi}(x) : \mathbb{R}^r \to \mathbb{R}^u$ *where all spaces are endowed with the euclidean inner product. From this we can derive* **??** *3 which return the close form solution of equation* 4.10 *for* $c(y, y') = \|y - y'\|_2^2$.

In the following, if no specific mention, we suppose that for any finite dimensional Hilbert space $\mathcal{H} \cong \mathbb{R}^d$, the adjoint space $\mathcal{H}^*$ is endowed with the dual basis. For any $x \in \mathcal{H}$, this allows us to identify $x^* = x^T$, and the inner product in any $\mathcal{H}$ is given for any $x, y \in \mathcal{H}$ by $\langle x, y \rangle = x^T y$. If one consider a Mahalanobis inner product, evaluation of operators has to be done with extra care since the basis of the dual space is *not* the dual basis (see remark 4.2).

Remark 4.2 *Notice that the evaluation of each operator* $\nabla_\theta J_{\lambda_K}(\theta)$, V^*, $\widetilde{\Phi}(x_i)^*$*'s and* M_{ik}*'s depends on the inner product of the respective spaces they are defined. Namely* $\mathcal{Y}$, $\mathcal{U}$ *and* $\widetilde{\mathcal{H}}$. *For instance if one choose* $\widetilde{\mathcal{H}} = \bigoplus_{j=1}^{D} \mathcal{U}'$, $\mathcal{U}' = \mathbb{R}^{u'}$ *endowed with a Mahalanobis inner product* $\langle x, y \rangle_{\mathcal{U}'} = x^T \Sigma^{-1} z$ *where* Σ *is some positive semi-definite matrix, then for all* $x \in \mathcal{X}$,

$$\widetilde{\Phi}(x)^*\theta = \sum_{j=1}^{D} (\Phi(x)_j)^T \Sigma^{-1} \theta_j.$$

If one choose $\mathcal{U} = \mathbb{R}^u$, $\widetilde{\mathcal{H}} = \mathbb{R}^r$ *with dual vector space endowed with the dual basis and* $\mathcal{Y} = \mathbb{R}^p$ *endowed with a Mahalanobis inner product* $\langle y, y' \rangle_{y'} = y^T \Sigma^{-1} y'$*, then for any* $y \in \mathcal{Y}$, $V^* y = V^T \Sigma^{-1} y$*, thus equation 4.12 reads*

$$\left(\frac{1}{N} \sum_{i=1}^{N} \widetilde{\Phi}(x_i) V^T \Sigma^{-1} V \widetilde{\Phi}(x_i)^T + \lambda_K \widetilde{\mathbf{M}}_{(\lambda_K, \lambda_M)} \right) \theta_z = \frac{1}{N} \sum_{i=1}^{N} \widetilde{\Phi}(x_i) V^T \Sigma^{-1} y_i.$$

where $\widetilde{\mathbf{M}}_{(\lambda_K, \lambda_M)} = I_r + \frac{\lambda_M}{\lambda_K} \sum_{i,k=1}^{N+U} \widetilde{\Phi}(x_i) M_{ik} \widetilde{\Phi}(x_k)^T$.

4.2.2 *Complexity analysis*

?? 3 constitute our first step toward large-scale learning with operator-valued kernels. We can easily compute the time complexity of **??** 3 when all the operators act on finite dimensional Hilbert spaces. Suppose that $u = \dim(\mathcal{U}) < +\infty$ and $u' = \dim(\mathcal{U}') < +\infty$ and for all $x \in \mathcal{X}$, $\widetilde{\Phi}(x) : \mathcal{U}' \to \widetilde{\mathcal{H}}$ where $r = \dim(\widetilde{\mathcal{H}}) < +\infty$ is the dimension of the redescription space $\widetilde{\mathcal{H}} = \mathbb{R}^r$. Since u, u', and $r < +\infty$, we view the operators $\widetilde{\Phi}(x)$, V and $\widetilde{\mathbf{M}}_{(\lambda_K, \lambda_M)}$ as matrices. Computing V^*V cost $O_t(u^2 p)$. Step 1 costs $O_t(r^2 u + r u^2)$. Steps 5 (optional) has the same cost except that the sum is done over all pair of $N + U$ points thus it costs $O_t((N+U)^2(r^2 u + r u^2))$. Steps 7 costs $O_t(N(ru + up))$. For step 8, the naive inversion of the operator costs $O_t(r^3)$. Eventually the overall complexity of **??** 3 is

$$O_t\left(ru(r+u) \begin{cases} (N+U)^2 & \text{if } \lambda_M > 0 \\ N & \text{if } \lambda_M = 0 \end{cases} + r^3 + Nu(r+p) \right),$$

while the space complexity is $O_s(r^2)$. This complexity is to compare with the kernelized solution proposed by Minh, Bazzani, and Murino [33]. Let

$$\mathbf{K} : \begin{cases} \mathcal{U}^{N+U} \to \mathcal{U}^{N+U} \\ u \mapsto \bigoplus_{i=1}^{N+U} \sum_{j=1}^{N+U} K(x_i, x_j) u_j \end{cases}$$

and

$$\mathbf{M} : \begin{cases} \mathcal{U}^{N+U} \to \mathcal{U}^{N+U} \\ u \mapsto \bigoplus_{i=1}^{N+U} \sum_{k=1}^{N+U} M_{ik} u_k. \end{cases}$$

When $\mathcal{U} = \mathbb{R}$,

$$\mathbf{K} = \begin{pmatrix} K(x_1, x_1) & \dots & K(x_1, x_{N+U}) \\ \vdots & \ddots & \vdots \\ K(x_{N+U}, x_1) & \dots & K(x_{N+U}, x_{N+U}) \end{pmatrix}$$

Algorithm 3: Naive close form for the squared error cost.

Input :

- $\mathbf{s} = (x_i, y_i)_{i=1}^N \in (\mathcal{X} \times \mathbb{R}^p)^N$ a sequence of supervised training points,
- $\mathbf{u} = (x_i)_{i=N+1}^{N+U} \in \mathcal{X}^U$ a sequence of unsupervised training points,
- $\widetilde{\Phi}(x_i) \in \mathcal{L}(\mathbb{R}^u, \mathbb{R}^r)$ a feature map defined for all $x_i \in \mathcal{X}$,
- $(M_{ik})_{i,k=1}^{N+U}$ a sequence of data dependent operators (see corollary 4.1),
- $V \in \mathcal{L}(\mathbb{R}^u, \mathbb{R}^p)$ a combination operator,
- $\lambda_K \in \mathbb{R}_{>0}$ the Tychonov regularization term,
- $\lambda_M \in \mathbb{R}_+$ the manifold regularization term.

Output : A model

$$h : \begin{cases} \mathcal{X} \to \mathbb{R}^p \\ x \mapsto \widetilde{\Phi}(x)^T \theta_z, \end{cases}$$

such that θ_z minimize equation 4.10, where $c(y, y') = \|y - y'\|_2^2$.

$\mathbf{P} \leftarrow \frac{1}{N} \sum_{i=1}^N \widetilde{\Phi}(x_i) V^T V \widetilde{\Phi}(x_i)^T \in \mathcal{L}(\mathbb{R}^r, \mathbb{R}^r)$;
if $\lambda_M = 0$ **then**
  $\widetilde{\mathbf{M}}_{(\lambda_K, \lambda_M)} \leftarrow I_r \in \mathcal{L}(\mathbb{R}^r, \mathbb{R}^r)$;
else
  $\widetilde{\mathbf{M}}_{(\lambda_K, \lambda_M)} \leftarrow \left(I_r + \frac{\lambda_M}{\lambda_K} \sum_{i,k=1}^{N+U} \widetilde{\Phi}(x_i) M_{ik} \widetilde{\Phi}(x_k)^T \right) \in \mathcal{L}(\mathbb{R}^r, \mathbb{R}^r)$;
end
$\mathbf{Y} \leftarrow \frac{1}{N} \sum_{i=1}^N \widetilde{\Phi}(x_i) V^T y_i \in \mathbb{R}^r$;
$\theta_z \leftarrow \left(P + \lambda_K \widetilde{\mathbf{M}}_{(\lambda_K, \lambda_M)} \right)^{-1} Y \in \mathbb{R}^r$;
return $h : x \mapsto \widetilde{\Phi}(x)^T \theta_z$;

is called the Gram matrix of K. When $\mathcal{U} = \mathbb{R}^p$, $\mathbf{K}$ is a matrix-valued Gram matrix of size $u(N+U) \times u(N+U)$ where each entry $\mathbf{K}_{ij} \in \mathcal{M}_{u,u}(\mathbb{R})$. When $\mathcal{U} = \mathbb{R}^u$, $\mathbf{M}$ can also be seen as a matrix-valued matrix where each entry is $M_{ik} \in \mathcal{M}_{u,u}(\mathbb{R})$. We also introduce the matrices $\mathbf{C}^T\mathbf{C} := I_{N+U} \otimes (C^T C)$ and

$$\mathbf{P} : \begin{cases} \mathcal{U}^{N+U} \to \mathcal{U}^{N+U} \\ u \mapsto \left(\bigoplus_{j=1}^N u_j \right) \oplus \left(\bigoplus_{j=N+1}^{N+U} 0 \right) \end{cases}$$

The operator $\mathbf{P}$ is a projection that sets all the terms u_j, $N < j \leqslant N + U$ of u to zero. When $\mathcal{U} = \mathbb{R}^u$ it can also be seen as the block matrix of size $u(N + U) \times u(N + U)$ and

$$\mathbf{P} = \begin{pmatrix} & & & 0 & \dots & 0 \\ & I_u \otimes I_N & & \vdots & \ddots & \vdots \\ & & & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 \end{pmatrix}$$

Then the equivalent kernelized solution u_z of theorem 4.2 is given by [33]

$$\left(\frac{1}{N}\mathbf{C}^\top\mathbf{C}\mathbf{P}\mathbf{K} + \lambda_M\mathbf{M}\mathbf{K} + \lambda_K I_{\bigoplus_{i=1}^{N+U} \mathcal{U}}\right) u_z = \left(\bigoplus_{i=1}^{N} C^\top y_i\right) \oplus \left(\bigoplus_{i=N+1}^{N+U} 0\right).$$

which has time complexity $O_t(((N + U)u)^3 + Nup)$ and space complexity $O_s(((N + U)u)^2)$. Hence **??** 3 is better that its kernelized counterpart when $r = 2Du'$ is small compare to $(N + U)u$. Roughly speaking it is better to use **??** 3 when the number of features, r, required is small compared to the number of training point. Notice that computing the data dependent norm (manifold regularization) is expensive. Indeed when $\lambda_M = 0$, **??** 3 has a linear complexity with respect to the number of supervised training points N while the complexity becomes quatratic in the number of supervised and unsupervised training points $N + U$ when $\lambda_M > 0$. Moreover suppose that $\lambda_M = 0$, $\mathcal{U} = \mathbb{R}^p$ and $\mathcal{U}' = \mathbb{R}^p$ and the combination operator is $V = I_p$. Then the complexity of **??** 3 boils down to

$$O_t(p^3(ND^2 + D^3)),$$

which is annoying. Indeed learning p independent models with scalar Random Fourier Features would cost $O_t(p(ND^2 + D^3))$, meaning that learning vector-valued function has increase the (expected) complexity from p to p^3. However in some cases we can drastically reduce the complexity by viewing the feature-maps as linear operators rather than matrices.

4.2.3 *Efficient matrix-free operators*

When developping **??** 3 we considered that the feature map $\widetilde{\Phi}(x)$ was a matrix from $\mathbb{R}^u$ to $\mathbb{R}^r$ for all $x \in \mathcal{X}$, and therefore that computing $\widetilde{\Phi}(x)^\top\theta$ has a time complexity of $O(r^2u)$. While this holds true in the most generic senario, in many cases the feature maps presents some structure or sparsity allowing to reduce the computational cost of evaluating the feature map. We focus on the Operator-valued Random Fourier Feature given by **??** 1, developped in **??** and section 3.3.1 and treat the decomposable kernel, the curl-free kernel and the divergence-free kernel as an example. We recall

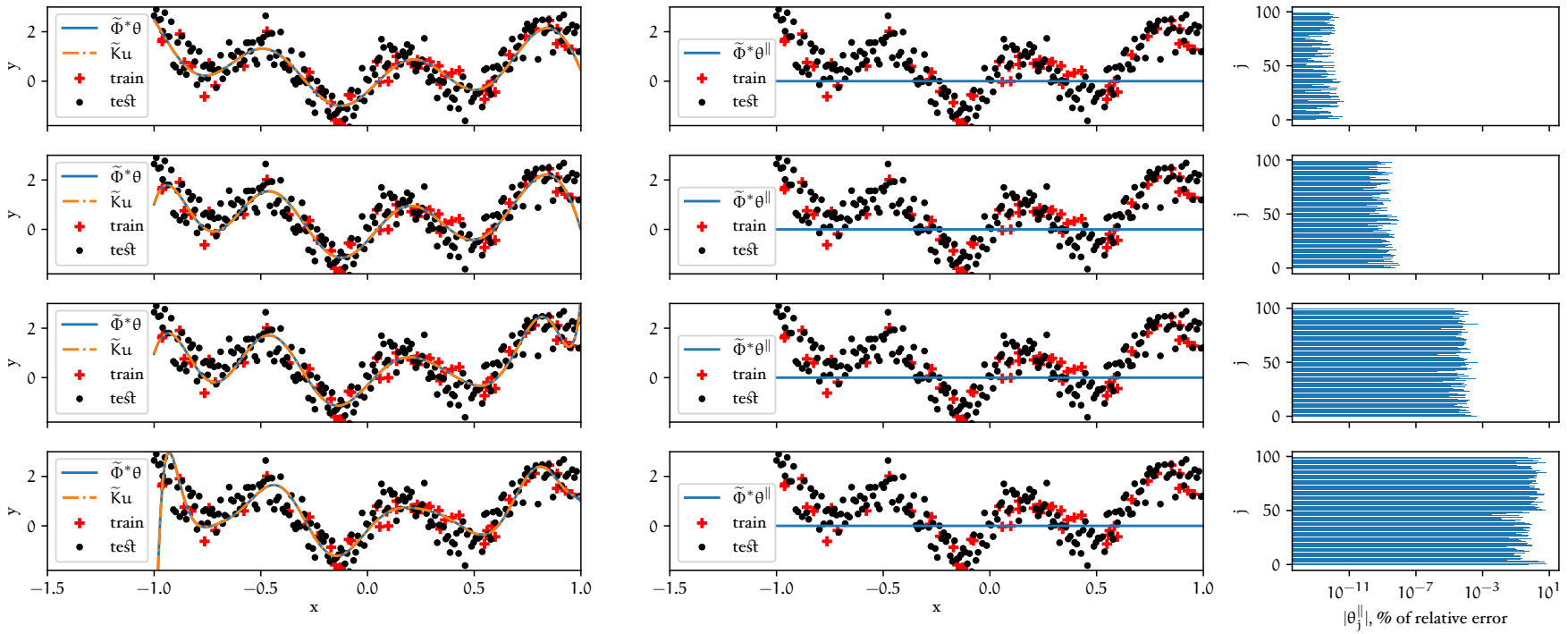

Figure 3: ORFF Representer theorem. We trained a first model named $\widetilde{K}$ following

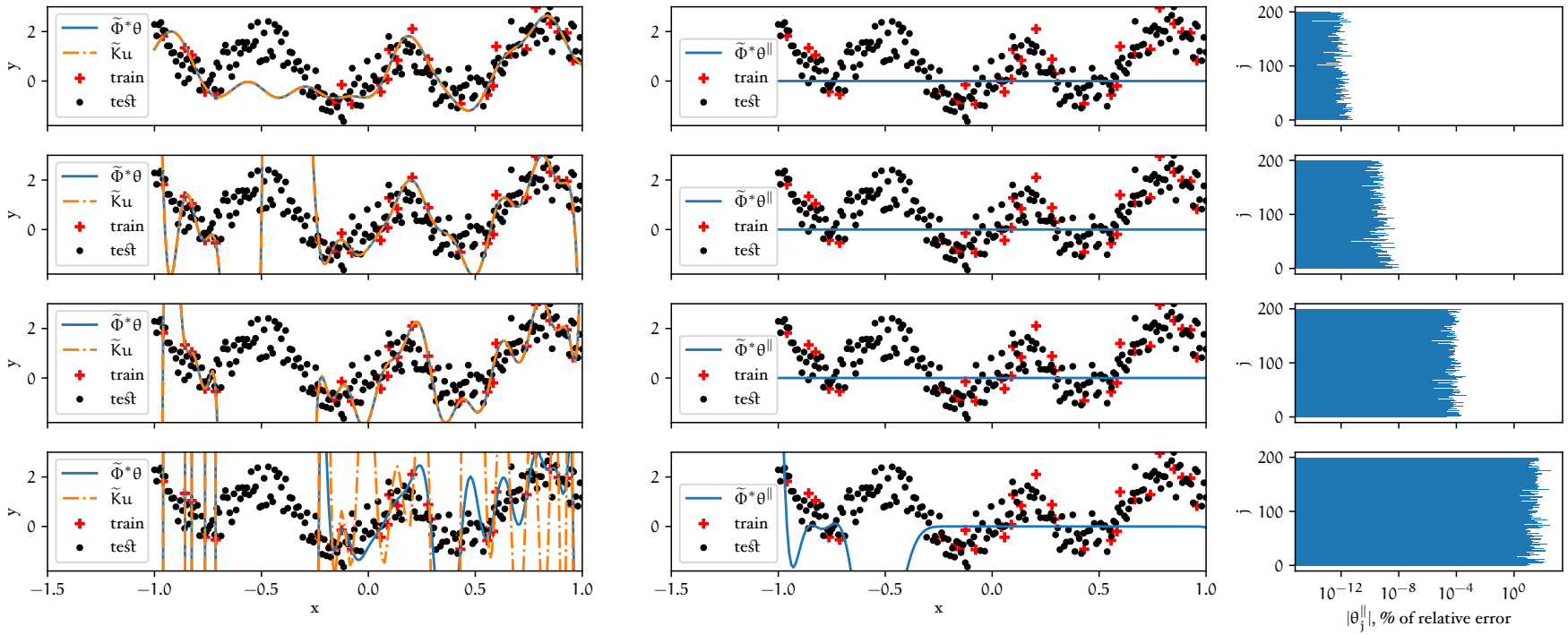

Figure 4: ORFF Representer theorem. We trained a first model named $\widetilde{K}$ following

that if $\mathcal{U}' = \mathbb{R}^{u'}$ and $\mathcal{U} = \mathbb{R}^{u}$, then $\widetilde{\mathcal{H}} = \mathbb{R}^{2Du'}$ thus the Operator-valued Random Fourier Features given in chapter 3 have the form

$$\begin{cases} \widetilde{\Phi}(x) \in \mathcal{L}\left(\mathbb{R}^{u}, \mathbb{R}^{2Du'}\right) & : y \mapsto \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} (x, \omega_j) B(\omega_j)^T y \\ \widetilde{\Phi}(x)^T \in \mathcal{L}\left(\mathbb{R}^{2Du'}, \mathbb{R}^{u}\right) & : \theta \mapsto \frac{1}{\sqrt{D}} \sum_{j=1}^{D} (x, \omega_j) B(\omega_j) \theta_j \end{cases},$$

where $\omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ i. i. d. and $B(\omega_j) \in \mathcal{L}\left(\mathbb{R}^{u}, \mathbb{R}^{u'}\right)$ for all $\omega_j \in \widehat{\mathcal{X}}$. Hence the Operator-valued Random Fourier Feature can be seen as the block matrix

$$\widetilde{\Phi}(x) = \begin{pmatrix} \cos\langle x, \omega_1 \rangle B(\omega_1)^T \\ \sin\langle x, \omega_1 \rangle B(\omega_1)^T \\ \vdots \\ \cos\langle x, \omega_D \rangle B(\omega_D)^T \\ \sin\langle x, \omega_D \rangle B(\omega_D)^T \end{pmatrix} \in \mathcal{M}_{2Du',u}(\mathbb{R}), \tag{4.13}$$

$\omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ i. i. d..

4.2.4 *Case of study: the decomposable kernel*

Troughout this section we show how the mathematical formulation relates to a concrete (Python) implementation. We propose a Python implementation based on NumPy [35], SciPy [21] and Scikit-learn [36]. Following equation 4.13, the feature map associated to the decomposable kernel would be

$$\widetilde{\Phi}(x) = \frac{1}{\sqrt{D}} \begin{pmatrix} \cos\langle x, \omega_1 \rangle B^T \\ \sin\langle x, \omega_1 \rangle B^T \\ \vdots \\ \cos\langle x, \omega_D \rangle B^T \\ \sin\langle x, \omega_D \rangle B^T \end{pmatrix} = \underbrace{\frac{1}{\sqrt{D}} \begin{pmatrix} \cos\langle x, \omega_1 \rangle \\ \sin\langle x, \omega_1 \rangle \\ \vdots \\ \cos\langle x, \omega_D \rangle \\ \sin\langle x, \omega_D \rangle \end{pmatrix}}_{\widetilde{\phi}(x)} \otimes B^T,$$

$\omega_j \sim \mathbf{Pr}_{\widehat{\mathbf{Haar}},\rho}$ i. i. d., which would lead to the following naive python implementation for the Gaussian (RBF) kernel of parameter γ, whose associated spectral distribution is $\mathbf{Pr}_\rho = \mathcal{N}(0, 2\gamma)$.

```
def NaiveDecomposableGaussianORFF(X, A, gamma=1.,
                                  D=100, eps=1e-5, random_state=0):
    r"""Return the Naive ORFF map associated with the data X.

    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    A : {array-like}, shape = [n_targets, n_targets]
        Operator of the Decomposable kernel (positive semi-definite)
    gamma : {float},
```

```
    Gamma parameter of the RBF kernel.
D : {integer}
    Number of random features.
eps : {float}
    Cutoff threshold for the singular values of A.
random_state : {integer}
    Seed of the generator.

Returns
-------
\tilde{\Phi}(X) : array
"""
# Decompose A=BB^T
u, s, v = svd(A, full_matrices=False, compute_uv=True)
B = dot(diag(sqrt(s[s > eps])), v[s > eps, :])

# Sample a RFF from the scalar Gaussian kernel
phi_s = RBFSampler(gamma=gamma, n_components=D, random_state=random_state)
phiX = phi_s.fit_transform(X)

# Create the ORFF linear operator
return matrix(kron(phiX, B))
```

Let $\theta \in \mathbb{R}^{2Du'}$ and $y \in \mathbb{R}$. With such imlementation evaluating a matrix vector product such as $\widetilde{\Phi}(x)^T\theta$ or $\widetilde{\Phi}(x)y$ have $O_t(2Du'u)$ time complexity and $O_s(2Du'u)$ of space complexity, which is utterly inefficient... Indeed, recall that if $B \in \mathcal{M}_{u,u'}\left(\mathbb{R}^{u'}\right)$ is matrix, the operator $\widetilde{\Phi}(x)$ corresponding to the decomposable kernel is

$$\begin{aligned}\widetilde{\Phi}(x)y &= \frac{1}{\sqrt{D}}\bigoplus_{j=1}^{D}\begin{pmatrix}\cos\langle x,\omega_j\rangle B^T y\\ \sin\langle x,\omega_j\rangle B^T y\end{pmatrix}\\ &= \left(\frac{1}{\sqrt{D}}\bigoplus_{j=1}^{D}\begin{pmatrix}\cos\langle x,\omega_j\rangle\\ \sin\langle x,\omega_j\rangle\end{pmatrix}\right)\otimes(B^T y)\end{aligned}$$

and

$$\begin{aligned}\widetilde{\Phi}(x)^T\theta &= \frac{1}{\sqrt{D}}\sum_{j=1}^{D}\cos\langle x,\omega_j\rangle B\theta_j + \sin\langle x,\omega_j\rangle B\theta_j\\ &= B\left(\frac{1}{\sqrt{D}}\sum_{j=1}^{D}\left(\cos\langle x,\omega_j\rangle + \sin\langle x,\omega_j\rangle\right)\theta_j\right).\end{aligned} \tag{4.14}$$

Which requires only on evaluation of B on y abd can be implemented easily in Python thanks to SciPy's LinearOperator. Note that the computation of these expressions can be fully vectorized[15] using the vectorization property of the kronecker product. In the following we consider $\Theta \in \mathcal{M}_{2D,u'}(\mathbb{R})$ and the operator $\mathbf{vec} : \mathcal{M}_{u',2D}(\mathbb{R}) \to \mathbb{R}^{2Du'}$ which turns a matrix into a vector (i. e. $\theta_{u'i+j} = \mathbf{vec}(\Theta_{ij})$, $i \in \mathbb{N}_{2D}$ and $j \in \mathbb{N}^*_{u'}$). Then

[15] *See Walt, Colbert, and Varoquaux [50].*

$$\left(\widetilde{\phi}(x)\otimes B^T\right)^T\theta = \left(\widetilde{\phi}(x)^T\otimes B\right)\mathbf{vec}(\Theta) = \mathbf{vec}\left(B\Theta\widetilde{\phi}(x)\right).$$

which leads us to the following (more efficient) Python implementation

```
def EfficientDecomposableGaussianORFF(X, A, gamma=1.,
                                      D=100, eps=1e-5, random_state=0):
    r"""Return the efficient ORFF map associated with the data X.

    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    A : {array-like}, shape = [n_targets, n_targets]
        Operator of the Decomposable kernel (positive semi-definite)
    gamma : {float},
        Gamma parameter of the RBF kernel.
    D : {integer}
        Number of random features.
    eps : {float}
        Cutoff threshold for the singular values of A.
    random_state : {integer}
        Seed of the generator.

    Returns
    -------
    \tilde{\Phi}(X) : Linear Operator, callable
    """
    # Decompose A=BB^T
    u, s, v = svd(A, full_matrices=False, compute_uv=True)
    B = dot(diag(sqrt(s[s > eps])), v[s > eps, :])

    # Sample a RFF from the scalar Gaussian kernel
    phi_s = RBFSampler(gamma=gamma, n_components=D, random_state=random_state)
    phiX = phi_s.fit_transform(X)

    # Create the ORFF linear operator
    cshape = (D, B.shape[0])
    rshape = (X.shape[0], B.shape[1])
    return LinearOperator((phiX.shape[0] * B.shape[1], D * B.shape[0]),
                          matvec=lambda b: dot(phiX, dot(b.reshape(cshape),
                                                         B)),
                          rmatvec=lambda r: dot(phiX.T, dot(r.reshape(rshape),
                                                            B.T)))
```

4.2.5 *Linear operators in matrix form*

For convinience we give the operators corresponding to the decomposable, curl-free and divergence-free kernels in matrix form. Let $(x_i)_{i=1}^N$, $N \in \mathbb{N}^*$, x_i's in $\mathbb{R}^d$, $d \leqslant \infty$ be a sequence of points in $\mathbb{R}^d$. We note

$$X = \begin{pmatrix} x_1 & \dots & x_N \end{pmatrix} \in \mathcal{M}_{d,N}$$

the data matrix where each column represents a data point[1]. Naturally if $\widetilde{\Phi}(x) : \mathbb{R}^u \to \mathbb{R}^{r_1}$ and $\widetilde{\phi}(x) : \mathbb{R} \to \mathbb{R}^{r_2}$, for all $x \in \mathbb{R}^d$ we define

$$\widetilde{\Phi}(X) = \begin{pmatrix} \widetilde{\Phi}(x_1) & \dots & \widetilde{\Phi}(x_N) \end{pmatrix} \in \mathcal{M}_{r_1,Nu}$$

1 In many programing language, such as Python, C, C++ or Java each data point is traditionally represented by a row in the data matrix (row major formulation). While this is more natural when parsing a data file, it is less common in mathematical formulations. In this document we adopt the *column major* formulation used by Matlab, Fortran or Julia. Moreover although C++ is commonly row major, some libraries such as Eigen are column major. When dealing with row major formulation, one should "transpose" all the equations given in **??** 1.

and

$$\widetilde{\phi}(X) = \begin{pmatrix} \widetilde{\phi}(x_1) & \dots & \widetilde{\phi}(x_N) \end{pmatrix} \in \mathcal{M}_{r_2,N}$$

and

$$U = \begin{pmatrix} u_1 & \dots & u_N \end{pmatrix} \in \mathcal{M}_{u,N}.$$

Given a matrix $X \in \mathcal{M}_{m,n}(\mathbb{R})$, we note $X_{\bullet i}$ the *column* vector coresponding to the i-th column of the matrix X and $X_{i\bullet}$ the *row* vector (covector) corresponding to the i-th line of the matrix X. With these notations, if $X \in \mathcal{M}_{m,n}$ and $Y \in \mathcal{M}_{n,m'}$, $X_{i\bullet}Y_{\bullet j} \in \mathbb{R}$ is the inner product between the i-th row of X and the j-th column of Y and $X_{\bullet i}Y_{j\bullet} \in \mathcal{M}_{m,m'}(\mathbb{R})$ is the outer product between the i-th column of X and j-th row of X.

For the curl-free and divergence-free kernel given in section 3.3.1 we recall the unbounded ORFF maps are respectively for all $u \in \mathcal{U}$

$$\widetilde{\Phi}(x)u = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle \omega_j^T u \\ \sin\langle x, \omega_j\rangle \omega_j^T u \end{pmatrix},$$

and

$$\widetilde{\Phi}(x)u = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle x, \omega_j\rangle \left(\|\omega_j\| I_d - \frac{\omega_j \omega_j^T}{\|\omega_j\|} \right) u \\ \sin\langle x, \omega_j\rangle \left(\|\omega_j\| I_d - \frac{\omega_j \omega_j^T}{\|\omega_j\|} \right) u \end{pmatrix},$$

where $\omega_j \sim \mathbf{Pr}_{\mathcal{N}(0,\sigma^{-2}I_d)}$. To avoid complex index notations we decompose the feature maps $\widetilde{\Phi}(X)$ into two sub feature maps $\widetilde{\Phi}^c$ and $\widetilde{\Phi}^s$ corresponding to the cosine part and the sine part of each feature map. Namely, for the curl-free kernel, for all $u \in \mathcal{U}$

$$\widetilde{\Phi}(x)u = \begin{cases} \widetilde{\Phi}^c(x)u = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \left(\cos\langle x, \omega_j\rangle \omega_j^T u \right), \\ \widetilde{\Phi}^s(x)u = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \left(\sin\langle x, \omega_j\rangle \omega_j^T u \right). \end{cases}$$

In the same way, for the divergence-free kernel,

$$\widetilde{\Phi}(x)u = \begin{cases} \widetilde{\Phi}^c(x)u = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \left(\cos\langle x, \omega_j\rangle \left(\|\omega_j\| I_d - \omega_j \omega_j^T \right) u \right), \\ \widetilde{\Phi}^s(x)u = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \left(\sin\langle x, \omega_j\rangle \left(\|\omega_j\| I_d - \omega_j \omega_j^T \right) u \right). \end{cases}$$

We also introduce $\widetilde{\Phi}^e$, $e \in \{s, c\}$ which denotes either $\widetilde{\Phi}^s$ or $\widetilde{\Phi}^c$. This equivalent formulation allows us to keep the notation "lighter" and closer to a proper Python/Matlab implementation with vectorization. With these notations, a summary of efficient linear operators in matrix form is given in **??** 1. The complexity of evaluating all this operators is given in equation 4.14.

Table 5: Complexity of efficient linear-operator (in matrix form) for different Feature maps given in **??** 1.

Kernel	$\widetilde{\Phi}(X)^*$	$\widetilde{\Phi}(X)$
Decomposable	$O\left((u'r + u'u)N\right)$	$O\left((uN + u'u)r\right)$
Curl-free	$O\left(uND\right)$	$O\left(uND\right)$
Divergence-free	$O\left((u^2 + uN)D\right)$	$O\left((u^2 + uN)D\right)$

Table 7: Efficient linear-operator (in matrix form) for different Feature maps.

Kernel	$\widetilde{\Phi}(X)^*$	$\widetilde{\Phi}(X)$
Decomposable[1]	$\Theta \mapsto B\left(\Theta\widetilde{\phi}(X)\right)$	$U \mapsto B^T\left(U\widetilde{\phi}(X)^T\right)$
Gaussian curl-free[2]	$\Theta^c, \Theta^s \mapsto \sum_{j=1}^{D} \omega_j \left(\Theta^c_j \widetilde{\phi}^c(X)_{j\bullet} + \Theta^s_j \widetilde{\phi}^s(X)_{j\bullet}\right)$	$U \mapsto \Theta^e_j = \omega_j^T \left(U\widetilde{\phi}^e(X)^T_{\bullet j}\right)$
Gaussian divergence-free[2,3]	$\Theta^c, \Theta^s \mapsto \sum_{j=1}^{D} \left(B(\omega_j)\Theta^c_{\bullet j}\right) \widetilde{\phi}^c(X)_{j\bullet} + \left(B(\omega_j)\Theta^s_{\bullet j}\right) \widetilde{\phi}^s(X)_{j\bullet}$	$U \mapsto \Theta^e_{\bullet j} = B(\omega_j)\left(U\widetilde{\phi}^e(X)^T_{\bullet j}\right)$

[1] Where $\widetilde{\phi}(X) = \begin{pmatrix}\widetilde{\phi}(X_{\bullet 1}) & \dots & \widetilde{\phi}(X_{\bullet N})\end{pmatrix} \in \mathcal{M}_{r,N}$ is any design matrix, with scalar feature map $\widetilde{\phi} : \mathbb{R}^d \to \mathbb{R}^r$ such that $\widetilde{\phi}(x)^*\widetilde{\phi}(z) = k(x,z) \in \mathbb{R}$ for all $x, z \in \mathcal{X}$. The input data $X \in \mathcal{M}_{d,N}(\mathbb{R})$, the output data $U \in \mathcal{M}_{u,N}(\mathbb{R})$, the parameter matrices Θ^c and $\Theta^s \in \mathcal{M}_{u',r}(\mathbb{R})$ and the decomposable operator $B \in \mathcal{M}_{u,u'}(\mathbb{R})$.

[2] Where $\widetilde{\phi}^c(X)_{ji} = \cos\langle\omega_j, x_i\rangle$ and $\widetilde{\phi}^s(X)_{ji} = \sin\langle\omega_j, x_i\rangle$, $j \in \mathbb{N}^*_D$ and $i \in \mathbb{N}^*_N$. Thus $\widetilde{\phi}^c(X) \in \mathcal{M}_{D,N}(\mathbb{R})$ and $\widetilde{\phi}^s(X) \in \mathcal{M}_{D,N}(\mathbb{R})$. The input data $X \in \mathcal{M}_{d,N}(\mathbb{R})$, the output data $U \in \mathcal{M}_{d,N}(\mathbb{R})$, the parameter matrices Θ^c and $\Theta^s \in \mathbb{R}^D$, $\omega_j \sim \mathbf{Pr}_{\mathcal{N}(0,\sigma^{-2}I_d)}$ i. i. d. for all $j \in \mathbb{N}^*_D$. Eventually $e \in \{s, c\}$, namely $\Theta^c = \begin{pmatrix}\Theta^{e=c}_1 & \dots & \Theta^{e=c}_D\end{pmatrix}^T$ and $\Theta^s = \begin{pmatrix}\Theta^{e=s}_1 & \dots & \Theta^{e=s}_D\end{pmatrix}^T$.

[3] Here, Θ^c and $\Theta^s \in \mathcal{M}_{d,D}(\mathbb{R})$ thus $\Theta^c = \begin{pmatrix}\Theta^{e=c}_{\bullet 1} & \dots & \Theta^{e=c}_{\bullet D}\end{pmatrix}$, $\Theta^s = \begin{pmatrix}\Theta^{e=s}_{\bullet 1} & \dots & \Theta^{e=s}_{\bullet D}\end{pmatrix}$ and $B(\omega) = \left(\|\omega\|_2 I_d - \frac{\omega\omega^T}{\|\omega\|_2}\right) \in \mathcal{M}_{d,d}$.

It is worth mentioning that the same strategy can be applied in many different language. For instance in C++, the libray Eigen [20] allows to wrap a sparse matrix with a custom type, where the user overload the transpose and dot product operator (as in Python). Then the custom user operator behaves as a (sparse) matrix –see `https://eigen.tuxfamily.org/dox/group__MatrixfreeSolverExample.html`. With this implementation the time complexity of $\widetilde{\Phi}(x)^T\theta$ and $\widetilde{\Phi}(x)y$ falls down to $O_t(2Du' + u'u)$ and the same holds for space complexity.

A quick experiment shows the advantage of seeing the decomposable kernel as a linear operator rather than a matrix. We draw $N = 100$ points $(x_i)_{i=1}^N$ in the interval $(0,1)^{20}$ and use a decomposable kernel with matrix $\Gamma = BB^T \in \mathcal{M}_{p,p}(\mathbb{R})$ where $B \in \mathcal{M}_{p,p}(\mathbb{R})$ is a random matrix with coefficients drawn uniformly in $(0,1)$. We compute $\widetilde{\Phi}(x)^T\theta$ for all x_i's, where $\theta \in \mathcal{M}_{2D,1}(\mathbb{R})$, $D = 100$, with the implementation `EfficientDecomposableGaussianORFF`, equation 4.14, and `NaiveDecomposableGaussianORFF`, equation 4.13. The coefficients of θ were drawn at random uniformly in $(0,1)$. We report the execution time in figure 5 for different values of p, $1 \leqslant p \leqslant 100$. The left plot report the execution time in seconds of the

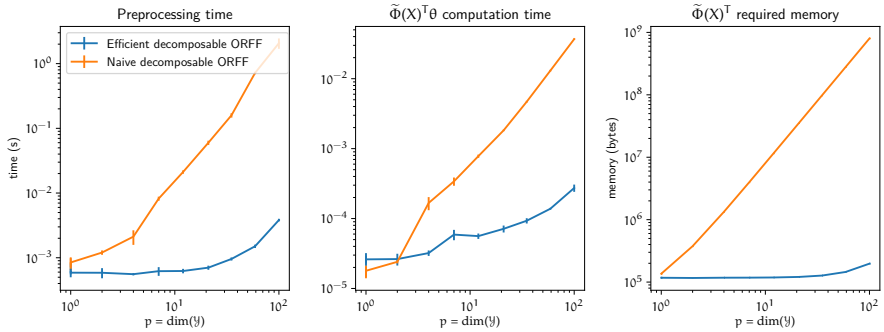

Figure 5: Efficient decomposable gaussian ORFF (lower is better).

construction of the feature. The middle plot report the execution time of $\widetilde{\Phi}(x)^T\theta$, and the right plot the memory used in bytes to store $\widetilde{\Phi}(x)$ for all x_i's. We average the results over ten runs. Full code is given in appendix C.3.

4.2.6 *Curl-free kernel*

We use the unbounded ORFF map presented in equation 3.22. We draw $N = 1000$ points $(x_i)_{i=1}^N$ in the interval $(0,1)^p$ and use a curl-free kernel. We compute $\widetilde{\Phi}(x)^T\theta$ for all x_i's, where $\theta \in \mathcal{M}_{2D,1}(\mathbb{R})$, $D = 500$, with the matrix implementation and the `LinearOperator` implementation. The coefficients of θ were drawn at random uniformly in $(0,1)$. We report the execution time in figure 6 for different values of p, $1 \leqslant p \leqslant 100$. The left plot report the execution time in seconds of the construction of the

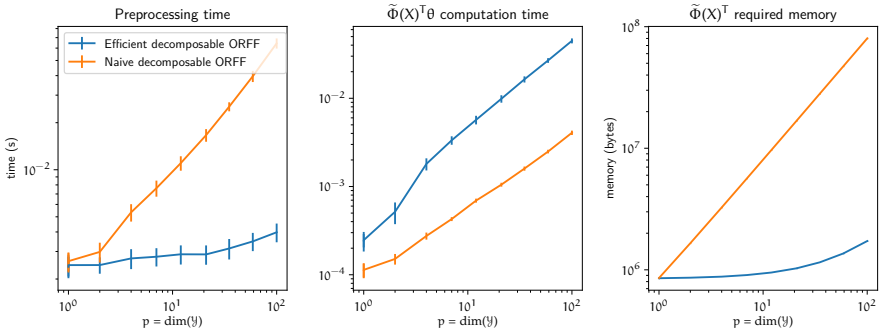

Figure 6: Efficient curl-free gaussian ORFF (lower is better).

feature. The middle plot report the execution time of $\widetilde{\Phi}(x)^T\theta$, and the right plot the memory used in bytes to store $\widetilde{\Phi}(x)$ for all x_i's. We average the results over fifty runs. Full code is given in appendix C.4. As we can see the linear-operator implementation is one order of magnitude slower than its matrix counterpart. However it uses considerably less memory.

4.2.7 *Divergence-free kernel*

We use the unbounded ORFF map presented in equation 3.23. We draw $N = 100$ points $(x_i)_{i=1}^N$ in the interval $(0, 1)^p$ and use a curl-free kernel. We compute $\widetilde{\Phi}(x)^T\theta$ for all x_i's, where $\theta \in \mathcal{M}_{2Dp,1}(\mathbb{R})$, $D = 100$, with the matrix implementation and the `LinearOperator` implementation. The coefficients of θ were drawn at random uniformly in $(0, 1)$. We report the execution time in figure 6 for different values of p, $1 \leqslant p \leqslant 100$. The

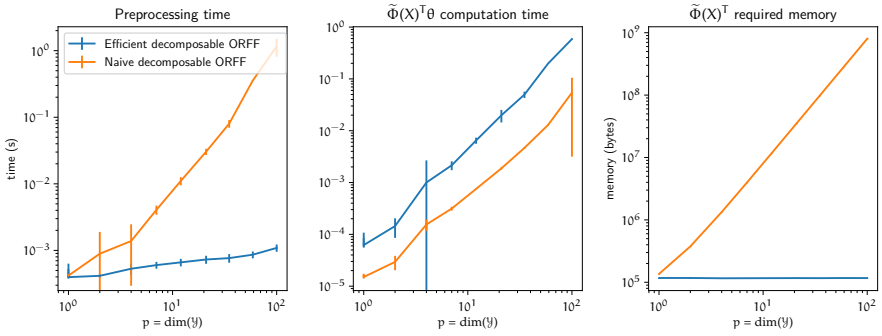

Figure 7: Efficient divergence-free gaussian ORFF (lower is better).

left plot report the execution time in seconds of the construction of the feature. The middle plot report the execution time of $\widetilde{\Phi}(x)^T\theta$, and the right plot the memory used in bytes to store $\widetilde{\Phi}(x)$ for all x_i's. We average the results over ten runs. Full code is given in appendix C.5. We draw the same conclusions as the curl-free kernel.

4.4 CONCLUSIONS

5

CONSISTENCY AND GENERALIZATION

5.1 CONSISTENCY OF THE ORFF ESTIMATOR

We are now interested on a non-asymptotic analysis of the ORFF approximation of shift-invariant $\mathcal{Y}$-Mercer kernels on LCA group $\mathcal{X}$ endowed with the operation group $\star$ with $\mathcal{X} \subset \mathbb{R}^d$. For a given D, we study how close is the approximation $\tilde{K}(x, z) = \widetilde{\Phi}(x)^* \widetilde{\Phi}(z)$ to the target kernel $K(x, z)$ for any x, z in $\mathcal{X}$.

If $A \in \mathcal{L}_+(\mathcal{Y})$ we denote $\|A\|_{\mathcal{Y},\mathcal{Y}}$ its operator norm, which amounts the square root of the largest eigenvalue of A. For x and z in some compact $\mathcal{C} \subset \mathbb{R}^d$, we consider: $F(x \star z^{-1}) = \tilde{K}(x, z) - K(x, z)$ and study how the uniform norm

$$\left\|\tilde{K} - K\right\|_{\mathcal{C}\times\mathcal{C}} = \|F\|_\infty = \sup_{(x,z)\in\mathcal{C}\times\mathcal{C}} \left\|\tilde{K}(x, z) - K(x, z)\right\|_{\mathcal{Y},\mathcal{Y}} \tag{5.1}$$

behaves according to D. Figure 8 empirically shows convergence of three different OVK approximations for x, z sampled from the compact $[-1, 1]^4$ and using an increasing number of sample points D. The log-log plot shows that all three kernels have the same convergence rate, up to a multiplicative factor.

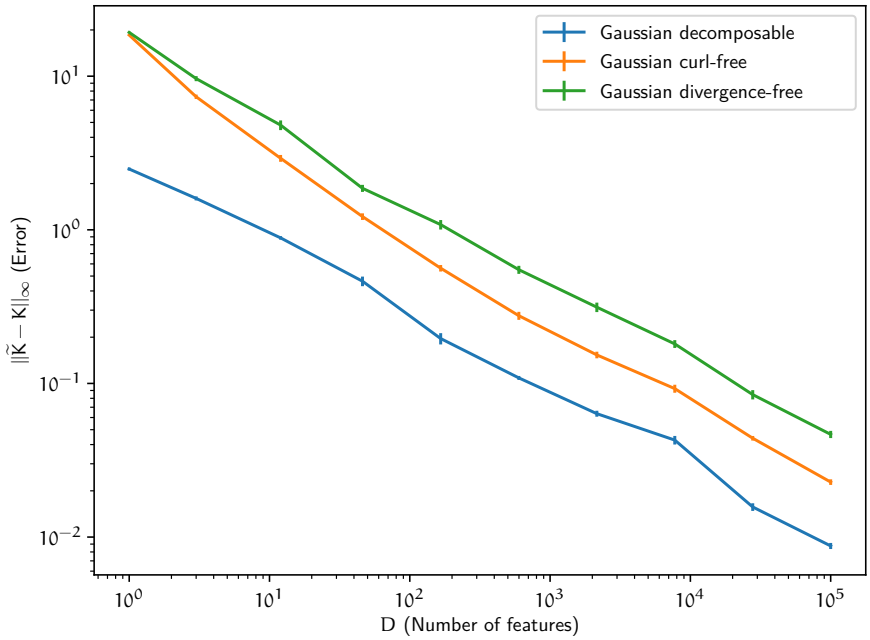

Figure 8: Error reconstructing the target operator-valued kernel K with ORFF approximation $\tilde{K}$ for the decomposable, curl-free and divergence-free kernel.

In order to bound the error with high probability, we turn to concentration inequalities devoted to random matrices [6]. Prior to the presentation

of general results, we briefly recall the uniform convergence of RFF approximation for a scalar shift invariant kernel on the additive LCA group $\mathbb{R}^d$ and introduce a direct corollary about decomposable shift-invariant OVK on the LCA group $(\mathbb{R}^d, +)$.

5.1.1 *Scalar random Fourier Features and decomposable kernel*

Rahimi and Recht [37] proved the uniform convergence of Random Fourier Feature (RFF) approximation for a scalar shift invariant kernel on the LCA group $\mathbb{R}^d$ endowed with the group operation $\star = +$. In the case of the shift-invariant decomposable OVK, an upper bound on the error can be directly obtained as a consequence of the result in the scalar case obtained by Rahimi and Recht [37] and other authors [43, 45] since in this case

Theorem 5.1 (Uniform error bound for RFF, Rahimi and Recht [37]). *Let $\mathcal{C}$ be a compact of subset of $\mathbb{R}^d$ of diameter $|\mathcal{C}|$. Let k be a shift invariant kernel, differentiable with a bounded second derivative and μ its normalized Inverse Fourier transform such that is defines a probability measure. Let D the dimension of the Fourier feature vectors. Then for the mapping $\widetilde{\phi}$ described in Section 1, we have*

$$\mathbf{Pr}_\mu\left\{(\omega_j)_{j=1}^D \;\middle|\; \left\|\tilde{k} - k\right\|_{\mathcal{C}\times\mathcal{C}} \geqslant \epsilon\right\} \leqslant 2^8\left(\frac{d\sigma|\mathcal{C}|}{\epsilon}\right)^2 \exp\left(-\frac{\epsilon^2 D}{4(d+2)}\right)$$

From theorem 5.1, we can deduce the following corollary about the uniform convergence of the ORFF approximation of the decomposable kernel. We recall that for a given pair x, z in $\mathcal{C}$, $\tilde{K}(x,z) = \widetilde{\Phi}(x)^*\widetilde{\Phi}(z) = A\tilde{k}(x,z)$ and $K_0(x-z) = A\mathbf{E}_\mu[\tilde{k}(x,z)]$.

Corollary 5.1 (Uniform error bound for decomposable ORFF). *Let $\mathcal{C}$ be a compact of subset of $\mathbb{R}^d$ of diameter $|\mathcal{C}|$. Let K be a decomposable kernel built from a positive operator A, and k a shift invariant kernel with $\sigma^2 = \int_{\mathbb{R}^d}\|\omega\|^2 d\mu(\omega) < \infty$. Then*

$$\begin{aligned}&\mathbf{Pr}_\mu\left\{(\omega_j)_{j=1}^D \;\middle|\; \left\|\tilde{K} - K\right\|_{\mathcal{C}\times\mathcal{C}} \geqslant \epsilon\right\}\\ &\leqslant 2^8\left(\frac{d\sigma\|A\|_{y,y}|\mathcal{C}|}{\epsilon}\right)^2 \exp\left(-\frac{\epsilon^2 D}{4\|A\|_2^2(d+2)}\right)\end{aligned}$$

Proof *The proof directly extends theorem 5.1 given by [37]. Let $\tilde{k}$ the Random Fourier approximation for the scalar-valued kernel k. Since*

$$\sup_{(x,z)\in\mathcal{C}\times\mathcal{C}}\left\|\tilde{K}(x,z) - K(x,z)\right\|_{y,y} = \sup_{(x,z)\in\mathcal{C}\times\mathcal{C}}\|A\|_{y,y}\left|\tilde{K}(x,z) - k(x,z)\right|,$$

taking $\epsilon' = \|A\|_{y,y}\epsilon$ gives the following result for all positive ϵ':

$$\begin{aligned}&\mathbf{Pr}_\mu\left\{(\omega_j)_{j=1}^D \;\middle|\; \sup_{x,z\in\mathcal{C}}\left\|A\left(\tilde{k}(x,z) - k(x,z)\right)\right\|_{y,y} \geqslant \epsilon'\right\}\\ &\leqslant 2^8\left(\frac{d\sigma\|A\|_{y,y}|\mathcal{C}|}{\epsilon'}\right)^2 \exp\left(-\frac{(\epsilon')^2 D}{4\|A\|_{y,y}^2(d+2)}\right)\end{aligned}$$

which concludes the proof. □

Please note that a similar corollary could have been obtained for the recent result of Sutherland and Schneider [45] who refined the bound proposed by Rahimi and Recht by using a Bernstein concentration inequality instead of the Hoeffding inequality. More recently Sriperumbudur and Szabo [43] showed an optimal bound for Random Fourier Feature,

5.1.2 *Uniform convergence of ORFF approximation on LCA groups*

In this analysis, we assume that $\mathcal{Y}$ is finite dimensional, in remark 5.1, we discuss how the proof could be extended to infinite dimensional output Hilbert spaces. We propose a bound for Operator-valued Random Fourier Feature approximation in the general case. It relies on two main ideas:

1. matrix-Bernstein concentration inequality for random matrices need to be used instead of concentration inequality for scalar random variables,
2. a general theorem valid for random matrices with bounded norms such as decomposable kernel ORFF approximation as well as unbounded norms such as the ORFF aproximation we proposed for curl and divergence-free kernels that behave as subexponential random variables.

Before introducing the new theorem, we give the definition of the Orlicz norm which gives a proxy-bound on the norm of subexponential random variables.

Definition 5.1 (Orlicz norm [46]). *Let $\psi : \mathbb{R}_+ \to \mathbb{R}_+$ be a non-decreasing convex function with $\psi(0) = 0$. For a random variable X on a measured space $(\Omega, \mathcal{T}(\Omega), \mu)$, the quantity*

$$\|X\|_\psi = \inf\{\, C > 0 \mid \mathbf{E}_\mu[\psi(|X|/C)] \leqslant 1 \,\}.$$

is called the Orlicz norm of X.

Here, the function ψ is chosen as $\psi(u) = \psi_\alpha(u)$ where $\psi_\alpha(u) := e^{u^\alpha} - 1$. When $\alpha = 1$, a random variable with finite Orlicz norm is called a *subexponential variable* because its tails decrease at an exponential rate. Let X be a Hermitian random operator. We call *variance* of X the quantity $\mathbf{Var}_\mu[X] = \mathbf{E}_\mu[X - \mathbf{E}_\mu[X]]^2$. With this convention if X is a $p \times p$ Hermitian matrix, $\mathbf{Var}_\mu[X]_{\ell m} = \sum_{r=1}^{p} \mathbf{Cov}_\mu[X_{\ell r}, X_{rm}]$.

Theorem 5.2 *Assume K is a shift-invariant $\mathcal{Y}$-Mercer kernel on $\mathcal{C}$ a compact subset of $\mathbb{R}^d$ of diameter $|\mathcal{C}|$, where $\mathbb{R}^d$ is seen as an Abelian group endowed with the group operation $\star$. Suppose that the decomposition $\mu(\cdot)A(\cdot)$ is equal to the Inverse Fourier transform of the kernel's signature*[16] *. Let $\tilde{K}$ be the ORFF approximation*[17] *of K, K_0 be the kernel signature of K. Define the constants b, σ_p^2, $m \in \mathbb{R}_+$ as*

[16] *in the sense of proposition 3.3*

[17] *$\tilde{K}$ depends on D.*

1. $b = D\left\|\mathbf{Var}_\mu\left[\tilde{K}\right]\right\|_{\mathcal{C}\times\mathcal{C}} < \infty$,
2. $\sigma_p^2 = \mathbf{E}_\mu\left[H_\omega^2\|A(\omega)\|_{y,y}^2\right] < \infty$,
3. $m = 4\left(\left\|\|A(\omega)\|_{y,y}\right\|_{\psi_1} + \|K\|_{\mathcal{C}\times\mathcal{C}}\right) < \infty$,

where $\omega \sim \mu$. *Then for all* ϵ *in* $\mathbb{R}_+$,

$$\mathbf{Pr}_\mu\left\{(\omega_j)_{j=1}^D \;\middle|\; \left\|\tilde{K}-K\right\|_{\mathcal{C}\times\mathcal{C}} \geqslant \epsilon\right\} \leqslant C_{d,p}\left(\frac{\sigma_p|\mathcal{C}|}{\epsilon}\right)^{\frac{2}{1+2/d}} \begin{cases} \exp\left(-\frac{\epsilon^2 D}{8(d+2)\left(b+\frac{\epsilon\bar{u}}{6}\right)}\right) & \text{if } \epsilon\bar{u} \leqslant 2(e-1)b \\ \exp\left(-\frac{\epsilon D}{(d+2)(e-1)\bar{u}}\right) & \text{otherwise,} \end{cases}$$

where $\bar{u} = 2m\log\left(2^{\frac{3}{2}}\left(\frac{m}{b}\right)^2\right)$ *and* $C_{d,p} = p\left(\left(\frac{d}{2}\right)^{\frac{-d}{d+2}} + \left(\frac{d}{2}\right)^{\frac{2}{d+2}}\right)2^{\frac{6d+2}{d+2}}$ *and* H_ω *is the Lipschitz constant*[18] *of* $h_\omega(x)$.

[18] *Remember that* $\exp(ih_\omega(x)) = (x,\omega)$, *so* H_ω *depends on the group operation.*

Sketch of proof *In the following, let* $F(\delta) = F(x\star z^{-1}) = \tilde{K}(x,z) - K(x,z)$. *Let* $\mathcal{C}_\Delta = \left\{x\star z^{-1} \;\middle|\; x,z\in\mathcal{C}\right\}$. *Since* $\mathcal{C}$ *is supposed compact, so is* $\mathcal{C}_\Delta$. *Its diameter is at most* $2|\mathcal{C}|$ *where* $|\mathcal{C}|$ *is the diameter of* $\mathcal{C}$. *It is then possible to find an* ϵ*-net covering* $\mathcal{C}_\Delta$ *with at most* $T = (4|\mathcal{C}|/r)^d$ *balls of radius* r. *We call* δ_i *for* $i\in\{1,\dots,T\}$ *the center of the* i*-th ball, called anchors of the* ϵ*-net. Denote* L_F *the Lipschitz constant of* F. *We introduce the following lemma proved in [37].*

Lemma 5.1

To apply the lemma, we must check assumptions (H1) and (H2).

Sketch of proof (H1) *We bound the Lipschitz constant by noticing that* F *is differentiable, so*

$$L_F = \left\|\frac{\partial F}{\partial\delta}(\delta^*)\right\|_{y,y} \text{ where } \delta^* = \arg\max_{\delta\in\mathcal{C}_\Delta}\left\|\frac{\partial F}{\partial\delta}(\delta)\right\|_{y,y}.$$

Using Jensen's inequality and applying Markov's inequality yields

$$\mathbf{Pr}_\mu\left\{\omega \;\middle|\; L_F \geqslant \frac{\epsilon}{2r}\right\} = \mathbf{Pr}_\mu\left\{\omega \;\middle|\; L_F^2 \geqslant \left(\frac{\epsilon}{2r}\right)^2\right\} \leqslant \mathbf{E}_\mu\left[\frac{\partial}{\partial\delta}h_\omega(\delta)\|A(\omega)\|_{y,y}^2\right]\left(\frac{2r}{\epsilon}\right)^2.$$

We set $\sigma_p^2 = \mathbf{E}_\mu\left[H_\omega^2\|A(\omega)\|_2^2\right]$ *where* H_ω *is the Lipschitz-constant of* h_ω *and suppose its existence.* □

Sketch of proof (H2) *To obtain a bound on the anchors we apply theorem 4 of Koltchinskii [23]. We suppose the existence of the two constants*

$$b = \sup_{\delta\in\mathcal{C}_\Delta} D\left\|\mathbf{Var}_\mu\left[\tilde{K}_e(\delta)\right]\right\|_{y,y}$$

and

$$\bar{u} = \log\left(2\left(\frac{m}{b}\right)^2 + 1\right),$$

where $m = 4\left(\|\|A(\omega)\|_{y,y}\|_{\psi_1} + \sup_{\delta\in\mathcal{C}_\Delta}\|K_\epsilon(\delta)\|_{y,y}\right)$ *and* $\omega \sim \mu$. *Then* $\forall i \in \{1,\dots,T\}$,

$$\mathbf{Pr}_\mu\left\{(\omega_j)_{j=1}^D \mid \|F(\delta_i)\|_{y,y} \geqslant \epsilon\right\} \leqslant 2p\begin{cases}\exp\left(-\frac{D\epsilon^2}{4b+2\epsilon\bar{u}/3}\right) & \text{if } \epsilon\bar{u} \leqslant 2(e-1)b\\ \exp\left(-\frac{D\epsilon}{(e-1)\bar{u}}\right) & \text{otherwise.}\end{cases}$$

Combining (H1) and (H2). Now applying the lemma and taking the union bound over the centers of the ϵ*-net yields*

$$\mathbf{Pr}_\mu\left\{(\omega_j)_{j=1}^D \;\middle|\; \sup_{\delta\in\mathcal{C}_\Delta}\|F(\delta)\|_{y,y} \leqslant \epsilon\right\} \geqslant 1 - \kappa_1 r^{-d} - \kappa_2 r^2,$$

with $\kappa_2 = 4\sigma_p^2\epsilon^{-2}$ *and*

$$\kappa_1 = 2p(4|\mathcal{C}|)^d\begin{cases}\exp\left(-\frac{\epsilon^2 D}{16\left(b+\frac{\epsilon}{6}\bar{u}\right)}\right) & \text{if } \epsilon\bar{u} \leqslant 2(e-1)b\\ \exp\left(-\frac{\epsilon D}{2(e-1)\bar{u}}\right) & \text{otherwise.}\end{cases}$$

We choose r *such that* $d\kappa_1 r^{-d-1} - 2\kappa_2 r = 0$, *i.e.* $r = \left(\frac{d\kappa_1}{2\kappa_2}\right)^{\frac{1}{d+2}}$. *The bound becomes*

$$\mathbf{Pr}_\mu\left\{(\omega_j)_{j=1}^D \;\middle|\; \sup_{\delta\in\mathcal{C}_\Delta}\|F(\delta)\| \geqslant \epsilon\right\}$$

$$\leqslant pC'_d 2^{\frac{6d+2}{d+2}}\left(\frac{\sigma_p|\mathcal{C}|}{\epsilon}\right)^{\frac{2}{1+2/d}}\begin{cases}\exp\left(-\frac{\epsilon^2}{8(d+2)\left(b+\frac{\epsilon}{6}\bar{u}\right)}\right) & \text{if } \epsilon\bar{u} \leqslant 2(e-1)b\\ \exp\left(-\frac{\epsilon}{(d+2)(e-1)\bar{u}}\right) & \text{otherwise.}\end{cases}$$

where $C'_d = \left(\left(\frac{d}{2}\right)^{\frac{-d}{d+2}} + \left(\frac{d}{2}\right)^{\frac{2}{d+2}}\right)$. *Conclude by taking* $C_{d,p} = pC'_d 2^{\frac{6d+2}{d+2}}$. □

We give a comprehensive full proof of the theorem in the appendix. It follows the usual scheme derived in Rahimi and Recht [37] and Sutherland and Schneider [45] and involves Bernstein concentration inequality for unbounded symmetric matrices (theorem 3.3 in the supplements).

Remark 5.1 (dealing with infinite dimensional operators). *We studied the concentration of ORFFs under the assumption that* $\mathcal{Y}$ *is finite dimensional. This assumption is required in the proof when bounding the Lipschitz constant. The proof require the quantity* $\tilde{K}$ *to be smooth enough to be able to interchange integral and derivative such that* $\mathbf{E}\frac{\partial}{\partial\delta}\tilde{K} = \frac{\partial}{\partial\delta}\mathbf{E}\tilde{K}$. *In the proof we show, component-wise, that that is* K *is twice differentiable, then the interchange is possible. To do the same for infinite dimensional operator-valued kernels, one could use Hille's theorem*[19] *and could deduce the necessary condition on* K *to allow the interchange. One could also deduced a bound on the Lipchitz constant without computing the derivative of* $\tilde{K} - K$. *Additionally one should study whether Koltchinskii [23, theorem 4] could apply to infinite dimensional operators and under which assumption.*

[19] *See* http://fa.its.tudelft.nl/~neerven/publications/papers/ISEM.pdf.

5.1.3 *Variance of the ORFF approximation*

We now provide a bound on the norm of the variance of $\tilde{K}$, required to apply theorem 5.2.

Proposition 5.1 (Bounding the *variance* of $\tilde{K}$). *Let K be a $\mathcal{Y}$-Mercer kernel on $\mathcal{C}$, a compact subset of $\mathbb{R}^d$. K is supposed to be translation invariant for the group operation $\star$ on $\mathbb{R}^d$. Let $\tilde{K}$ be the ORFF approximation of K (as defined in **??**) and $\mathcal{C}_\Delta = \left\{ x \star z^{-1} \;\middle|\; (x, z) \in \mathcal{C} \times \mathcal{C} \right\}$. Then for all δ in $\mathcal{C}_\Delta$,*

$$\begin{aligned} &\left\| \mathbf{Var}_\mu \left[\tilde{K}_e(\delta) \right] \right\|_{\mathcal{Y},\mathcal{Y}} \\ &\leqslant \frac{\left\| (K_e(2\delta) + K_e(0)) \mathbf{E}_\mu[A(\omega)] - 2K_e(\delta)^2 \right\|_{\mathcal{Y},\mathcal{Y}} + 2\|\mathbf{Var}_\mu[A(\omega)]\|_{\mathcal{Y},\mathcal{Y}}}{2D}. \end{aligned}$$

Proof *It relies on the i. i. d. property of the random vectors ω_j and trigonometric identities (see the proof in **??** of the supplementary material).* □

An empirical illustration of these bounds is shown in **??**. We generated a random point in $[-1, 1]^4$ and computed the empirical variance of the estimator (blue line). We also plotted (red line) the theoretical bound in proposition 5.1.

5.1.4 *Application on decomposable, curl-free and div-free OVKs*

First, the two following example discuss the form of H_ω for the additive group and the skewed-multiplicative group.

Example 5.1 (Additive group). *On the additive group, $h_\omega(\delta) = \langle \omega, \delta \rangle$. Hence $H_\omega = \|\omega\|_2$.*

Example 5.2 (Skewed-multiplicative group). *On the skewed multiplicative group, $h_\omega(\delta) = \langle \omega, \log(\delta + c) \rangle$. Therefore*

$$\sup_{\delta \in \mathcal{C}} \|\nabla h_\omega(\delta)\| = \sup_{\delta \in \mathcal{C}} \|\omega/(\delta + c)\|_2.$$

Eventually $\mathcal{C}$ is compact and finite dimensional thus $\mathcal{C}$ is closed and bounded. Thus $H_\omega = \|\omega\|_2 / (\min_{\delta \in \mathcal{C}} \|\delta\|_2 + c)$.

Now we compute upper bounds on the norm of the variance and Orlicz norm of the three ORFFs we took as examples.

DECOMPOSABLE KERNEL: notice that in the case of the Gaussian decomposable kernel, i.e. $A(\omega) = A$, $e = 0$, $K_0(\delta) = A k_0(\delta)$, $k_0(\delta) \geqslant 0$ and $k_0(\delta) = 1$, then we have

$$D \left\| \mathbf{Var}_\mu \left[\tilde{K}_0(\delta) \right] \right\|_{\mathcal{Y},\mathcal{Y}} \leqslant (1 + k_0(2\delta)) \|A\|_{\mathcal{Y},\mathcal{Y}}/2 + k_0(\delta)^2.$$

CURL-FREE AND DIV-FREE KERNELS: recall that in this case $p = d$. For the (Gaussian) curl-free kernel, $A(\omega) = \omega\omega^*$ where $\omega \in \mathbb{R}^d \sim \mathcal{N}(0, \sigma^{-2} I_d)$ thus $\mathbf{E}_\mu[A(\omega)] = I_d/\sigma^2$ and $\mathbf{Var}_\mu[A(\omega)] = (d+1)I_d/\sigma^4$. Hence,

$$D\left\|\mathbf{Var}_\mu\left[\tilde{K}_0(\delta)\right]\right\|_{y,y} \leqslant \frac{1}{2}\left\|\frac{1}{\sigma^2}K_0(2\delta) - 2K_0(\delta)^2\right\|_{y,y} + \frac{(d+1)}{\sigma^4}.$$

This bound is illustrated by figure 8 B, for a given datapoint. Eventually for the Gaussian divergence-free kernel, $A(\omega) = I\|\omega\|_2^2 - \omega\omega^*$, thus $\mathbf{E}_\mu[A(\omega)] = I_d(d-1)/\sigma^2$ and $\mathbf{Var}_\mu[A(\omega)] = d(4d-3)I_d/\sigma^4$. Hence,

$$D\left\|\mathbf{Var}_\mu\left[\tilde{K}_0(\delta)\right]\right\|_{y,y} \leqslant \frac{1}{2}\left\|\frac{(d-1)}{\sigma^2}K_0(2\delta) - 2K_0(\delta)^2\right\|_{y,y} + \frac{d(4d-3)}{\sigma^4}.$$

Eventually, we ensure that the random variable $\|A(\omega)\|$ has a finite Orlicz norm with $\psi = \psi_1$ in these three cases.

COMPUTING THE ORLICZ NORM: for a random variable with strictly monotonic moment generating function (MGF), one can characterize its inverse ψ_1 Orlicz norm by taking the functional inverse of the MGF evaluated at 2 (see **??** of the supplementary material). In other words $\|X\|_{\psi_1}^{-1} = \mathrm{MGF}(x)_X^{-1}(2)$. For the Gaussian curl-free and divergence-free kernel,

$$\left\|A^{\mathrm{div}}(\omega)\right\|_{y,y} = \left\|A^{\mathrm{curl}}(\omega)\right\|_{y,y} = \|\omega\|_2^2,$$

where$\omega \sim \mathcal{N}(0, I_d/\sigma^2)$, hence $\|A(\omega)\|_2 \sim \Gamma(p/2, 2/\sigma^2)$. The MGF of this gamma distribution is $\mathrm{MGF}(x)(t) = (1 - 2t/\sigma^2)^{-(p/2)}$. Eventually

$$\left\|\left\|A^{\mathrm{div}}(\omega)\right\|_{y,y}\right\|_{\psi_1}^{-1} = \left\|\left\|A^{\mathrm{curl}}(\omega)\right\|_{y,y}\right\|_{\psi_1}^{-1} = \frac{\sigma^2}{2}\left(1 - 4^{-\frac{1}{p}}\right).$$

6

APPLICATIONS

Part III

FINAL WORDS AND PERSPECTIVES

You can put some informational part preamble text here. Illo principalmente su nos. Non message *occidental* angloromanic da. Debitas effortio simplificate sia se, auxiliar summarios da que, se avantiate publicationes via. Pan in terra summarios, capital interlingua se que. Al via multo esser specimen, campo responder que da. Le usate medical addresses pro, europa origine sanctificate nos se.

7

CONLUSIONS

Part IV

APPENDIX

A

OPERATOR-VALUED FUNCTIONS AND INTEGRATION

B

PROOFS OF THEOREMS

B.1 PROOF OF

C

RELEVANT PIECE OF CODE

C.1 PYTHON CODE FOR FIGURE 1

```
r"""Plot figure: Different outcomes of a Gaussian kernel approximation."""

import matplotlib
import numpy as np
import matplotlib.pyplot as plt

from sklearn.metrics import pairwise_kernels


def phi(x, w, D):
    r"""RFF map."""
    Z = np.dot(x, w)
    return np.hstack((np.cos(Z), np.sin(Z))) / np.sqrt(D)


def createColorbar(lwr, upr, fig, axes):
    r"""Create colorbar for multiple Pyplot plot."""
    cax = fig.add_axes([.92, 0.1, 0.01, 0.8])
    norm = matplotlib.colors.LogNorm(vmin=lwr, vmax=upr, clip=False)
    c = matplotlib.colorbar.ColorbarBase(cax, cmap=plt.get_cmap('rainbow'),
                                         norm=norm, label='D=')
    plt.title(r'$\widetilde{K}$')
    return c


def main():
    r"""Plot figure: Different outcomes of a Gaussian kernel approximation."""
    T = 25  # Number of curves

    cm_subsection = np.linspace(0, 1, T + 1)
    colors = [matplotlib.cm.rainbow(x) for x in cm_subsection]

    d = 1  # Dimension of the input
    N = 250  # Number of points per curves

    # Generate N data in (-1, 1) and exact Gram matrix
    np.random.seed(0)
    X = np.linspace(-1, 1, N).reshape((N, d))
    K = pairwise_kernels(X, metric='rbf', gamma=1. / (2. * .1 ** 2))

    # A Matrix for the decomposable kernel. Link the outputs to some mean value
    c = np.random.randn(N, 2)
    A = .5 * np.eye(2) + .5 * np.ones((2, 2))

    plt.close()
    plt.rc('text', usetex=True)
    plt.rc('font', family='serif')
    f, axes = plt.subplots(2, 2, figsize=(12, 8), sharex=True, sharey=True)

    # For each curve with different D
    for k, D in enumerate(np.logspace(0, 4, T)):
        D = int(D)
        np.random.seed(0)

        w = np.random.randn(d, D) / .1
```

```
            phiX = phi(X, w, D)
            Kt = np.dot(phiX, phiX.T)

            # Generate outputs with the exact Gram matrix
            pred = np.dot(np.dot(Kt, c), A)
            axes[0, 0].plot(X, pred[:, 0], c=colors[k], lw=.5, linestyle='-')
            axes[0, 0].set_ylabel(r'$y_1$')
            axes[0, 1].plot(X, pred[:, 1], c=colors[k], lw=.5, linestyle='-')
            axes[0, 1].set_ylabel(r'$y_2$')

            # Generate outputs with the a realization of the random Gram matrix
            w = np.random.randn(d, D) / .1
            phiX = phi(X, w, D)
            Kt = np.dot(phiX, phiX.T)

            pred = np.dot(np.dot(Kt, c), A)
            axes[1, 0].plot(X, pred[:, 0], c=colors[k], lw=.5, linestyle='-')
            axes[1, 0].set_xlabel(r'$x$')
            axes[1, 0].set_ylabel(r'$y_1$')
            axes[1, 1].plot(X, pred[:, 1], c=colors[k], lw=.5, linestyle='-')
            axes[1, 1].set_xlabel(r'$x$')
            axes[1, 1].set_ylabel(r'$y_2$')

        axes[0, 0].plot(X, np.dot(np.dot(K, c), A)[:, 0], c='k', lw=.5, label='K')
        axes[0, 1].plot(X, np.dot(np.dot(K, c), A)[:, 1], c='k', lw=.5, label='K')
        axes[1, 0].plot(X, np.dot(np.dot(K, c), A)[:, 0], c='k', lw=.5, label='K')
        axes[1, 1].plot(X, np.dot(np.dot(K, c), A)[:, 1], c='k', lw=.5, label='K')

        axes[0, 0].set_title(r'$\widetilde{K}u \approx Ku$, realization 1', x=1.1)
        axes[1, 0].set_title(r'$\widetilde{K}u \approx Ku$, realization 2', x=1.1)

        for xx in axes.ravel():
            xx.legend(loc=4)

        createColorbar(1, D, f, axes)
        plt.savefig('not_Mercer.pgf', bbox_inches='tight')

if __name__ == "__main__":
    main()
```

C.2 PYTHON CODE FOR FIGURE 3

```
r"""Plot figure: ORFF Representer theorem."""

import numpy as np
import matplotlib.pyplot as plt
import matplotlib


def phi(x, w, D):
    r"""RFF map."""
    Z = np.dot(x, w)
    return np.hstack((np.cos(Z), np.sin(Z))) / np.sqrt(D)


def main():
    r"""Plot figure: ORFF Representer theorem."""
    d = 1  # dimensionality of the inputs
    D = 50  # number of random features

    N = 50
    Nt = 200

    # N training points in (0,1)
    np.random.seed(0)
```

```
x = 2 * np.random.rand(N, d) - 1
y = np.sin(10 * x)
y += .5 * np.random.randn(y.shape[0], y.shape[1]) + 2. * x ** 2

# Nt testing points in (0,1)
xt = np.linspace(-1, 1, Nt).reshape((-1, 1))
yt = np.sin(10 * xt) + 2. * xt ** 2
yt += .5 * np.random.randn(yt.shape[0], yt.shape[1])

sigma = .3
w = np.random.randn(d, D) / sigma  # Realization of (\omega_j)_{j=1}^D

phiX = phi(x, w, D)  # Train RFF
phiXt = phi(xt, w, D)  # Test RFF

# Create plot
plt.close()
plt.rc('text', usetex=True)
plt.rc('font', family='serif')
f, axis = plt.subplots(4, 3, gridspec_kw={'width_ratios': [3, 3, 1.5]},
                       figsize=(16, 6), sharex='col', sharey='col')
f.subplots_adjust(hspace=.25)
formatter = matplotlib.ticker.ScalarFormatter()
formatter.set_powerlimits((-3, 4))

# For different hyperparameters \lambda
for k, lbda in enumerate([1e-2, 5e-6, 1e-10, 0]):
    # Train with ORFF with kernel approximation (dual)
    ck = np.linalg.lstsq(np.dot(phiX, phiX.T) + lbda * np.eye(N),
                         y, rcond=-1)[0]
    # Train with ORFF without kernel approximation (primal)
    c = np.linalg.lstsq(np.dot(phiX.T, phiX) + lbda * np.eye(2 * D),
                        np.dot(phiX.T, y), rcond=-1)[0]
    cc = np.sum((phi(x, w, D) * ck), axis=0)
    # Link dual coefficient with primal coefficients
    cr = (cc - c.ravel()) / np.linalg.norm(c) * 100
    err = np.array([np.linalg.norm(np.dot(phiXt, c) - yt) ** 2 / Nt,
                    np.linalg.norm(np.dot(np.dot(phiXt,
                                                 phiX.T),
                                          ck) - yt) ** 2 / Nt,
                    np.linalg.norm(np.dot(phiXt, cr)) ** 2 / Nt,
                    np.linalg.norm(cr)])

    # Plot
    lmin = -1.8
    lmax = 3.
    axis[k, 0].set_xlim([-1.5, 1])
    axis[k, 0].set_ylim([lmin, lmax])
    axis[k, 0].plot(xt, np.dot(phiXt, c),
                    label=r'$\widetilde{\Phi}^* \theta$')
    axis[k, 0].plot(xt, np.dot(np.dot(phiXt, phiX.T), ck),
                    label=r'$\widetilde{K}u$', linestyle='-.')
    axis[k, 0].scatter(x, y, c='r', marker='+', label='train', lw=2)
    axis[k, 0].scatter(xt, yt, c='k', marker='.', label='test')
    axis[k, 0].legend(loc=3)
    axis[k, 0].set_ylabel('y')
    if k == 3:
        axis[k, 0].set_xlabel('x')

    lmin = -1.8
    lmax = 3.
    pred = np.dot(phi(xt, w, D), cr)
    axis[k, 1].set_xlim([-1.5, 1])
    axis[k, 1].set_ylim([lmin, lmax])
    axis[k, 1].plot(xt, pred,
                    label=r'$\widetilde{\Phi}^* \theta^{\parallel}$')
```

```
            axis[k, 1].scatter(x, y, c='r', marker='+', label='train', lw=2)
            axis[k, 1].scatter(xt, yt, c='k', marker='.', label='test')
            axis[k, 1].legend(loc=3)
            if k == 3:
                axis[k, 1].set_xlabel('x')

            xs = np.arange(cr.size)
            axis[k, 2].barh(xs, np.abs(cr), edgecolor="none", log=True)
            axis[k, 2].set_ylabel(r'$j$')
            if k == 3:
                axis[k, 2].set_xlabel(
                    r'$|\theta^{\parallel}_j|$, \% of relative error')
        plt.savefig('representer.pgf', bbox_inches='tight')

        return err

    if __name__ == "__main__":
        main()
```

C.3 PYTHON CODE FOR FIGURE 4

```
r"""Efficient implementation of the Gaussian ORFF decomposable kernel."""

from time import time

from pympler.asizeof import asizeof

from numpy.linalg import svd
from numpy.random import rand, seed
from numpy import (dot, diag, sqrt, kron, zeros,
                   logspace, log10, matrix, eye, int, float)
from scipy.sparse.linalg import LinearOperator
from sklearn.kernel_approximation import RBFSampler
from matplotlib.pyplot import savefig, subplots, tight_layout


def NaiveDecomposableGaussianORFF(X, A, gamma=1.,
                                  D=100, eps=1e-5, random_state=0):
    r"""Return the Naive ORFF map associated with the data X.

    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    A : {array-like}, shape = [n_targets, n_targets]
        Operator of the Decomposable kernel (positive semi-definite)
    gamma : {float},
        Gamma parameter of the RBF kernel.
    D : {integer},
        Number of random features.
    eps : {float},
        Cutoff threshold for the singular values of A.
    random_state : {integer},
        Seed of the generator.

    Returns
    -------
    \tilde{\Phi}(X) : array
    """
    # Decompose A=BB^T
    u, s, v = svd(A, full_matrices=False, compute_uv=True)
    B = dot(diag(sqrt(s[s > eps])), v[s > eps, :])

    # Sample a RFF from the scalar Gaussian kernel
    phi_s = RBFSampler(gamma=gamma, n_components=D, random_state=random_state)
```

```
    phiX = phi_s.fit_transform(X)

    # Create the ORFF linear operator
    return matrix(kron(phiX, B))


def EfficientDecomposableGaussianORFF(X, A, gamma=1.,
                                      D=100, eps=1e-5, random_state=0):
    r"""Return the Efficient ORFF map associated with the data X.

    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    A : {array-like}, shape = [n_targets, n_targets]
        Operator of the Decomposable kernel (positive semi-definite)
    gamma : {float},
        Gamma parameter of the RBF kernel.
    D : {integer},
        Number of random features.
    eps : {float},
        Cutoff threshold for the singular values of A.
    random_state : {integer},
        Seed of the generator.

    Returns
    -------
    \tilde{\Phi}(X) : Linear Operator, callable
    """
    # Decompose A=BB^T
    u, s, v = svd(A, full_matrices=False, compute_uv=True)
    B = dot(diag(sqrt(s[s > eps])), v[s > eps, :])

    # Sample a RFF from the scalar Gaussian kernel
    phi_s = RBFSampler(gamma=gamma, n_components=D, random_state=random_state)
    phiX = phi_s.fit_transform(X)

    # Create the ORFF linear operator
    cshape = (D, B.shape[0])
    rshape = (X.shape[0], B.shape[1])
    return LinearOperator((phiX.shape[0] * B.shape[1], D * B.shape[0]),
                          matvec=lambda b: dot(phiX, dot(b.reshape(cshape),
                                               B)),
                          rmatvec=lambda r: dot(phiX.T, dot(r.reshape(rshape),
                                                B.T)),
                          dtype=float)


def main():
    r"""Plot figure: Efficient decomposable gaussian ORFF."""
    N = 100  # Number of points
    pmax = 100  # Maximum output dimension
    d = 20  # Input dimension
    D = 100  # Number of random features

    seed(0)
    X = rand(N, d)

    R, T = 10, 10
    time_Efficient, mem_Efficient = zeros((R, T, 2)), zeros((R, T))
    time_naive, mem_naive = zeros((R, T, 2)), zeros((R, T))

    for i, p in enumerate(logspace(0, log10(pmax), T)):
        A = rand(int(p), int(p))
        A = dot(A.T, A) + eye(int(p))
```

```
    # Perform \Phi(X)^T \theta with Efficient implementation
    for j in range(R):
        start = time()
        phiX1 = EfficientDecomposableGaussianORFF(X, A, D)
        time_Efficient[j, i, 0] = time() - start
        theta = rand(phiX1.shape[1], 1)
        start = time()
        phiX1 * theta
        time_Efficient[j, i, 1] = time() - start
        mem_Efficient[j, i] = asizeof(phiX1, code=True)

    # Perform \Phi(X)^T \theta with naive implementation
    for j in range(R):
        start = time()
        phiX2 = NaiveDecomposableGaussianORFF(X, A, D)
        time_naive[j, i, 0] = time() - start
        theta = rand(phiX2.shape[1], 1)
        start = time()
        phiX2 * theta
        time_naive[j, i, 1] = time() - start
        mem_naive[j, i] = asizeof(phiX2, code=True)

# Plot
f, axes = subplots(1, 3, figsize=(10, 4), sharex=True, sharey=False)
axes[0].errorbar(logspace(0, log10(pmax), T).astype(int),
                 time_Efficient[:, :, 0].mean(axis=0),
                 time_Efficient[:, :, 0].std(axis=0),
                 label='Efficient decomposable ORFF')
axes[0].errorbar(logspace(0, log10(pmax), T).astype(int),
                 time_naive[:, :, 0].mean(axis=0),
                 time_naive[:, :, 0].std(axis=0),
                 label='Naive decomposable ORFF')
axes[1].errorbar(logspace(0, log10(pmax), T).astype(int),
                 time_Efficient[:, :, 1].mean(axis=0),
                 time_Efficient[:, :, 1].std(axis=0),
                 label='Efficient decomposable ORFF')
axes[1].errorbar(logspace(0, log10(pmax), T).astype(int),
                 time_naive[:, :, 1].mean(axis=0),
                 time_naive[:, :, 1].std(axis=0),
                 label='Naive decomposable ORFF')
axes[2].errorbar(logspace(0, log10(pmax), T).astype(int),
                 mem_Efficient[:, :].mean(axis=0),
                 mem_Efficient[:, :].std(axis=0),
                 label='Efficient decomposable ORFF')
axes[2].errorbar(logspace(0, log10(pmax), T).astype(int),
                 mem_naive[:, :].mean(axis=0),
                 mem_naive[:, :].std(axis=0),
                 label='Naive decomposable ORFF')
axes[0].set_xscale('log')
axes[0].set_yscale('log')
axes[1].set_xscale('log')
axes[1].set_yscale('log')
axes[2].set_xscale('log')
axes[2].set_yscale('log')
axes[0].set_xlabel(r'$p=\dim(\mathcal{Y})$')
axes[1].set_xlabel(r'$p=\dim(\mathcal{Y})$')
axes[2].set_xlabel(r'$p=\dim(\mathcal{Y})$')
axes[0].set_ylabel(r'time (s)')
axes[2].set_ylabel(r'memory (bytes)')
axes[0].set_title(r'Preprocessing time')
axes[1].set_title(r'$\widetilde{\Phi}(X)^T \theta$ computation time')
axes[2].set_title(r'$\widetilde{\Phi}(X)^T$ required memory')
axes[0].legend(loc=2)
tight_layout()
savefig('efficient_decomposable_gaussian.pgf', bbox_inches='tight')
```

```
if __name__ == "__main__":
    main()
```

C.4 PYTHON CODE FOR FIGURE 5

```
r"""Efficient implementation of the Gaussian curl-free kernel."""

from time import time

from pympler.asizeof import asizeof

from numpy.random import rand, seed
from numpy import dot, zeros, logspace, log10, matrix, int, float
from scipy.sparse.linalg import LinearOperator
from sklearn.kernel_approximation import RBFSampler
from matplotlib.pyplot import savefig, subplots, tight_layout


def NaiveCurlFreeGaussianORFF(X, gamma=1.,
                              D=100, eps=1e-5, random_state=0):
    r"""Return the Naive ORFF map associated with the data X.

    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    gamma : {float},
        Gamma parameter of the RBF kernel.
    D : {integer},
        Number of random features.
    eps : {float},
        Cutoff threshold for the singular values of A.
    random_state : {integer},
        Seed of the generator.

    Returns
    -------
    \tilde{\Phi}(X) : array
    """
    phi_s = RBFSampler(gamma=gamma, n_components=D,
                       random_state=random_state)
    phiX = phi_s.fit_transform(X)
    phiX = (phiX.reshape((phiX.shape[0], 1, phiX.shape[1])) *
            phi_s.random_weights_.reshape((1, -1, phiX.shape[1])))

    return matrix(phiX.reshape((-1, phiX.shape[2])))


def EfficientCurlFreeGaussianORFF(X, gamma=1.,
                                  D=100, eps=1e-5, random_state=0):
    r"""Return the Efficient ORFF map associated with the data X.

    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    gamma : {float},
        Gamma parameter of the RBF kernel.
    D : {integer},
        Number of random features.
    eps : {float},
        Cutoff threshold for the singular values of A.
    random_state : {integer},
        Seed of the generator.
```

```
    Returns
    -------
    \tilde{\Phi}(X) : array
    """
    phi_s = RBFSampler(gamma=gamma, n_components=D,
                       random_state=random_state)
    phiX = phi_s.fit_transform(X)

    return LinearOperator((phiX.shape[0] * X.shape[1], phiX.shape[1]),
                          matvec=lambda b:
                          dot(phiX.reshape((phiX.shape[0], 1, phiX.shape[1])) *
                          phi_s.random_weights_.reshape((1, -1,
                                                         phiX.shape[1])), b),
                          rmatvec=lambda r:
                          dot((phiX.reshape((phiX.shape[0], 1,
                                             phiX.shape[1])) *
                               phi_s.random_weights_.reshape((1, -1,
                                                              phiX.shape
                                                              [1]))).reshape
                          (phiX.shape[0] * X.shape[1], phiX.shape[1]).T, r),
                          dtype=float)


def main():
    r"""Plot figure: Efficient decomposable gaussian ORFF."""
    N = 1000  # Number of points
    dmax = 100  # Input dimension
    D = 500  # Number of random features

    seed(0)

    R, T = 50, 10
    time_Efficient, mem_Efficient = zeros((R, T, 2)), zeros((R, T))
    time_naive, mem_naive = zeros((R, T, 2)), zeros((R, T))

    for i, d in enumerate(logspace(0, log10(dmax), T)):
        X = rand(N, int(d))

        # Perform \Phi(X)^T \theta with Efficient implementation
        for j in range(R):
            start = time()
            phiX1 = EfficientCurlFreeGaussianORFF(X, D)
            time_Efficient[j, i, 0] = time() - start
            start = time()
            phiX1 * rand(phiX1.shape[1], 1)
            time_Efficient[j, i, 1] = time() - start
            mem_Efficient[j, i] = asizeof(phiX1, code=True)

        # Perform \Phi(X)^T \theta with naive implementation
        for j in range(R):
            start = time()
            phiX2 = NaiveCurlFreeGaussianORFF(X, D)
            time_naive[j, i, 0] = time() - start
            start = time()
            phiX2 * rand(phiX2.shape[1], 1)
            time_naive[j, i, 1] = time() - start
            mem_naive[j, i] = asizeof(phiX2, code=True)

    # Plot
    f, axes = subplots(1, 3, figsize=(10, 4), sharex=True, sharey=False)
    axes[0].errorbar(logspace(0, log10(dmax), T).astype(int),
                     time_Efficient[:, :, 0].mean(axis=0),
                     time_Efficient[:, :, 0].std(axis=0),
                     label='Efficient decomposable ORFF')
    axes[0].errorbar(logspace(0, log10(dmax), T).astype(int),
                     time_naive[:, :, 0].mean(axis=0),
```

```
                    time_naive[:, :, 0].std(axis=0),
                    label='Naive decomposable ORFF')
    axes[1].errorbar(logspace(0, log10(dmax), T).astype(int),
                    time_Efficient[:, :, 1].mean(axis=0),
                    time_Efficient[:, :, 1].std(axis=0),
                    label='Efficient decomposable ORFF')
    axes[1].errorbar(logspace(0, log10(dmax), T).astype(int),
                    time_naive[:, :, 1].mean(axis=0),
                    time_naive[:, :, 1].std(axis=0),
                    label='Naive decomposable ORFF')
    axes[2].errorbar(logspace(0, log10(dmax), T).astype(int),
                    mem_Efficient[:, :].mean(axis=0),
                    mem_Efficient[:, :].std(axis=0),
                    label='Efficient decomposable ORFF')
    axes[2].errorbar(logspace(0, log10(dmax), T).astype(int),
                    mem_naive[:, :].mean(axis=0),
                    mem_naive[:, :].std(axis=0),
                    label='Naive decomposable ORFF')
    axes[0].set_xscale('log')
    axes[0].set_yscale('log')
    axes[1].set_xscale('log')
    axes[1].set_yscale('log')
    axes[2].set_xscale('log')
    axes[2].set_yscale('log')
    axes[0].set_xlabel(r'$p=\dim(\mathcal{Y})$')
    axes[1].set_xlabel(r'$p=\dim(\mathcal{Y})$')
    axes[2].set_xlabel(r'$p=\dim(\mathcal{Y})$')
    axes[0].set_ylabel(r'time (s)')
    axes[2].set_ylabel(r'memory (bytes)')
    axes[0].set_title(r'Preprocessing time')
    axes[1].set_title(r'$\widetilde{\Phi}(X)^T \theta$ computation time')
    axes[2].set_title(r'$\widetilde{\Phi}(X)^T$ required memory')
    axes[0].legend(loc=2)
    tight_layout()
    savefig('efficient_curlfree_gaussian.pgf', bbox_inches='tight')

if __name__ == "__main__":
    main()
```

C.5 PYTHON CODE FOR FIGURE 6

```
r"""Efficient implementation of the Gaussian divergence-free kernel."""

from time import time

from pympler.asizeof import asizeof

from numpy.random import rand, seed
from numpy.linalg import norm
from numpy import dot, zeros, logspace, log10, matrix, int, eye, float
from scipy.sparse.linalg import LinearOperator
from sklearn.kernel_approximation import RBFSampler
from matplotlib.pyplot import savefig, subplots, tight_layout


def _rebase(phiX, W, Wn):
    return (phiX.reshape((phiX.shape[0], 1, 1, phiX.shape[1])) *
            (eye(W.shape[1]).reshape(1, W.shape[1], W.shape[1], 1) * Wn -
            W * W.reshape(1, 1, W.shape[1], phiX.shape[1]) / Wn)).reshape(
        (-1, W.shape[1] * Wn.shape[3]))


def NaiveDivergenceFreeGaussianORFF(X, gamma=1.,
                                    D=100, eps=1e-5, random_state=0):
    r"""Return the Naive ORFF map associated with the data X.
```

```
    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    gamma : {float},
        Gamma parameter of the RBF kernel.
    D : {integer},
        Number of random features.
    eps : {float},
        Cutoff threshold for the singular values of A.
    random_state : {integer},
        Seed of the generator.

    Returns
    -------
    \tilde{\Phi}(X) : array
    """
    phi_s = RBFSampler(gamma=gamma, n_components=D,
                       random_state=random_state)

    phiX = _rebase(phi_s.fit_transform(X),
                   phi_s.random_weights_.reshape((1, -1, 1, D)),
                   norm(phi_s.random_weights_, axis=0).reshape((1, 1, 1, -1)))

    return matrix(phiX)


def EfficientDivergenceFreeGaussianORFF(X, gamma=1.,
                                        D=100, eps=1e-5, random_state=0):
    r"""Return the Efficient ORFF map associated with the data X.

    Parameters
    ----------
    X : {array-like}, shape = [n_samples, n_features]
        Samples.
    gamma : {float},
        Gamma parameter of the RBF kernel.
    D : {integer},
        Number of random features.
    eps : {float},
        Cutoff threshold for the singular values of A.
    random_state : {integer},
        Seed of the generator.

    Returns
    -------
    \tilde{\Phi}(X) : array
    """
    phi_s = RBFSampler(gamma=gamma, n_components=D,
                       random_state=random_state)
    phiX = phi_s.fit_transform(X)
    W = phi_s.random_weights_.reshape((1, -1, 1, phiX.shape[1]))
    Wn = norm(phi_s.random_weights_, axis=0).reshape((1, 1, 1, -1))
    return LinearOperator((phiX.shape[0] * X.shape[1],
                           phiX.shape[1] * X.shape[1]),
                          matvec=lambda b: dot(_rebase(phiX, W, Wn), b),
                          rmatvec=lambda r: dot(_rebase(phiX, W, Wn).T, r),
                          dtype=float)


def main():
    r"""Plot figure: Efficient decomposable gaussian ORFF."""
    N = 100  # Number of points
    dmax = 100  # Input dimension
    D = 100  # Number of random features
```

```
seed(0)

R, T = 10, 10
time_Efficient, mem_Efficient = zeros((R, T, 2)), zeros((R, T))
time_naive, mem_naive = zeros((R, T, 2)), zeros((R, T))

for i, d in enumerate(logspace(0, log10(dmax), T)):
    X = rand(N, int(d))

    # Perform \Phi(X)^T \theta with Efficient implementation
    for j in range(R):
        start = time()
        phiX1 = EfficientDivergenceFreeGaussianORFF(X, D)
        time_Efficient[j, i, 0] = time() - start
        theta = rand(phiX1.shape[1], 1)
        start = time()
        phiX1 * theta
        time_Efficient[j, i, 1] = time() - start
        mem_Efficient[j, i] = asizeof(phiX1, code=True)

    # Perform \Phi(X)^T \theta with naive implementation
    for j in range(R):
        start = time()
        phiX2 = NaiveDivergenceFreeGaussianORFF(X, D)
        time_naive[j, i, 0] = time() - start
        theta = rand(phiX2.shape[1], 1)
        start = time()
        phiX2 * theta
        time_naive[j, i, 1] = time() - start
        mem_naive[j, i] = asizeof(phiX2, code=True)

# Plot
f, axes = subplots(1, 3, figsize=(10, 4), sharex=True, sharey=False)
axes[0].errorbar(logspace(0, log10(dmax), T).astype(int),
                 time_Efficient[:, :, 0].mean(axis=0),
                 time_Efficient[:, :, 0].std(axis=0),
                 label='Efficient decomposable ORFF')
axes[0].errorbar(logspace(0, log10(dmax), T).astype(int),
                 time_naive[:, :, 0].mean(axis=0),
                 time_naive[:, :, 0].std(axis=0),
                 label='Naive decomposable ORFF')
axes[1].errorbar(logspace(0, log10(dmax), T).astype(int),
                 time_Efficient[:, :, 1].mean(axis=0),
                 time_Efficient[:, :, 1].std(axis=0),
                 label='Efficient decomposable ORFF')
axes[1].errorbar(logspace(0, log10(dmax), T).astype(int),
                 time_naive[:, :, 1].mean(axis=0),
                 time_naive[:, :, 1].std(axis=0),
                 label='Naive decomposable ORFF')
axes[2].errorbar(logspace(0, log10(dmax), T).astype(int),
                 mem_Efficient[:, :].mean(axis=0),
                 mem_Efficient[:, :].std(axis=0),
                 label='Efficient decomposable ORFF')
axes[2].errorbar(logspace(0, log10(dmax), T).astype(int),
                 mem_naive[:, :].mean(axis=0),
                 mem_naive[:, :].std(axis=0),
                 label='Naive decomposable ORFF')
axes[0].set_xscale('log')
axes[0].set_yscale('log')
axes[1].set_xscale('log')
axes[1].set_yscale('log')
axes[2].set_xscale('log')
axes[2].set_yscale('log')
axes[0].set_xlabel(r'$p=\dim(\mathcal{Y})$')
axes[1].set_xlabel(r'$p=\dim(\mathcal{Y})$')
```

```
    axes[2].set_xlabel(r'$p=\dim(\mathcal{Y})$')
    axes[0].set_ylabel(r'time (s)')
    axes[2].set_ylabel(r'memory (bytes)')
    axes[0].set_title(r'Preprocessing time')
    axes[1].set_title(r'$\widetilde{\Phi}(X)^T \theta$ computation time')
    axes[2].set_title(r'$\widetilde{\Phi}(X)^T$ required memory')
    axes[0].legend(loc=2)
    tight_layout()
    savefig('efficient_divfree_gaussian.pgf', bbox_inches='tight')

if __name__ == "__main__":
    main()
```

BIBLIOGRAPHY

[1] Erik M Alfsen. "A simplified constructive proof of the existence and uniqueness of Haar measure." In: *Mathematica Scandinavica* 12.1 (1964), pp. 106–116 (cit. on p. 8).

[2] M. A. Álvarez, L. Rosasco, and N. D. Lawrence. "Kernels for vector-valued functions: a review." In: *Foundations and Trends in Machine Learning* 4.3 (2012), pp. 195–266 (cit. on pp. 20, 53).

[3] L. Baldassarre, L. Rosasco, A. Barla, and A. Verri. "Vector Field Learning via Spectral Filtering." In: *ECML/PKDD*. Ed. by J. Balcazar, F. Bonchi, A. Gionis, and M. Sebag. Vol. 6321. LNCS. Springer Berlin / Heidelberg, 2010, pp. 56–71 (cit. on p. 20).

[4] L. Baldassarre, L. Rosasco, A. Barla, and A. Verri. "Multi-output learning via spectral filtering." In: *Machine Learning* 87.3 (2012), pp. 259–301 (cit. on p. 21).

[5] Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. "Manifold regularization: A geometric framework for learning from labeled and unlabeled examples." In: *Journal of machine learning research* 7.Nov (2006), pp. 2399–2434 (cit. on p. 53).

[6] S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities*. Oxford Press, 2013 (cit. on p. 80).

[7] Romain Brault, Markus Heinonen, and Florence d'Alché Buc. "Random Fourier Features For Operator-Valued Kernels." In: *Proceedings of The 8th Asian Conference on Machine Learning*. 2016, pp. 110–125 (cit. on p. 26).

[8] Ulf Brefeld, Thomas Gärtner, Tobias Scheffer, and Stefan Wrobel. "Efficient co-regularised least squares regression." In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 137–144 (cit. on p. 53).

[9] C. Brouard, F. d'Alché-Buc, and M. Szafranski. "Semi-supervised Penalized Output Kernel Regression for Link Prediction." In: *Proc. of the 28th Int. Conf. on Machine Learning*. 2011 (cit. on pp. 53, 54).

[10] C. Brouard, F. d'Alché-Buc, and M. Szafranski. "Input Output Kernel Regression." In: *to appear in JMLR* (2016) (cit. on p. 53).

[11] A. Caponnetto, C. A. Micchelli, M., and Y. Ying. "Universal MultiTask Kernels." In: *Journal of Machine Learning Research* 9 (2008), pp. 1615–1646 (cit. on p. 20).

[12] C. Carmeli, E. De Vito, and A. Toigo. "Vector valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem." In: *Analysis and Applications* 4.04 (2006), pp. 377–408 (cit. on pp. 13, 18, 28).

[13] C. Carmeli, E. De Vito, A. Toigo, and V. Umanità. "Vector valued reproducing kernel Hilbert spaces and universality." In: *Analysis and Applications* 8 (2010), pp. 19–61 (cit. on pp. 13, 16, 17, 19, 20, 26–28, 38).

[14] John B Conway. *A course in functional analysis*. Vol. 96. Springer Science & Business Media, 2013 (cit. on pp. 8, 60).

[15] F. Dinuzzo, C.S. Ong, P. Gehler, and G. Pillonetto. "Learning Output Kernels with Block Coordinate Descent." In: *Proc. of the 28th Int. Conf. on Machine Learning*. 2011 (cit. on p. 20).

[16] T. Evgeniou, C. A. Micchelli, and M. Pontil. "Learning Multiple Tasks with kernel methods." In: *JMLR* 6 (2005), pp. 615–637 (cit. on p. 20).

[17] Gerald B Folland. *A course in abstract harmonic analysis*. CRC press, 1994 (cit. on pp. 8, 10, 12).

[18] E. Fuselier. "Refined Error Estimates for Matrix-Valued Radial Basis Functions." PhD thesis. Texas A&M University, 2006 (cit. on p. 21).

[19] Lech Górniewicz. *Topological fixed point theory of multivalued mappings*. Vol. 495. Springer, 1999 (cit. on p. 52).

[20] Gaël Guennebaud, Benoît Jacob, et al. *Eigen v3*. http://eigen.tuxfamily.org. 2010 (cit. on p. 74).

[21] Eric Jones, Travis Oliphant, and Pearu Peterson. "{SciPy}: open source scientific tools for {Python}." In: (2014) (cit. on p. 68).

[22] Hachem Kadri, Emmanuel Duflos, Philippe Preux, Stéphane Canu, Alain Rakotomamonjy, and Julien Audiffren. "Operator-valued kernels for learning from functional response data." In: *Journal of Machine Learning Research* 16 (2015), pp. 1–54 (cit. on pp. 53, 54).

[23] Vladimir Koltchinskii et al. "A remark on low rank matrix recovery and noncommutative Bernstein type inequalities." In: *From Probability to Statistics and Back: High-Dimensional Models and Processes*. Institute of Mathematical Statistics, 2013, pp. 213–226 (cit. on pp. 83, 84).

[24] Andrew J Kurdila and Michael Zabarankin. *Convex functional analysis*. 2006 (cit. on pp. 52, 55, 60).

[25] F. Li, C. Ionescu, and C. Sminchisescu. "Pattern Recognition: 32nd DAGM Symposium, Darmstadt, Germany, September 22-24, 2010. Proc." In: ed. by M. Goesele, S. Roth, A. Kuijper, B. Schiele, and K. Schindler. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. Chap. Random Fourier Approximations for Skewed Multiplicative Histogram Kernels, pp. 262–271. ISBN: 978-3-642-15986-2. DOI: 10.1007/978-3-642-15986-2_27. URL: http://dx.doi.org/10.1007/978-3-642-15986-2_27 (cit. on pp. 11, 47).

[26] N. Lim, F. d'Alché-Buc, C. Auliac, and G. Michailidis. "Operator-valued kernel-based vector autoregressive models for network inference." In: *Machine Learning* 99.3 (2015), pp. 489–513 (cit. on p. 20).

[27] Y. Macedo and R. Castro. *Learning Div-Free and Curl-Free Vector Fields by Matrix-Valued Kernels*. Tech. rep. Preprint A 679/2010 IMPA, 2008 (cit. on p. 21).

[28] C. A. Micchelli and M. A. Pontil. "On Learning Vector-Valued Functions." In: *Neural Computation* 17 (2005), pp. 177–204 (cit. on pp. 13, 53, 54).

[29] M. Micheli and J. Glaunes. *Matrix-valued kernels for shape deformation analysis*. Tech. rep. Arxiv report, 2013 (cit. on p. 21).

[30] H. Q. Minh, L. Bazzani, and V. Murino. "A unifying framework for vector-valued manifold regularization and multi-view learning." In: *Proc. of the 30th International Conference on Machine Learning*. 2013 (cit. on p. 53).

[31] Ha Quang Minh. "Operator-Valued Bochner Theorem, Fourier Feature Maps for Operator-Valued Kernels, and Vector-Valued Learning." In: *arXiv preprint arXiv:1608.05639* (2016) (cit. on p. 31).

[32] Ha Quang Minh, Loris Bazzani, and Vittorio Murino. "A unifying framework for vector-valued manifold regularization and multi-view learning." In: *ICML (2)*. 2013, pp. 100–108 (cit. on p. 53).

[33] Ha Quang Minh, Loris Bazzani, and Vittorio Murino. "A unifying framework in vector-valued reproducing kernel Hilbert spaces for manifold regularization and co-regularized multi-view learning." In: *Journal of Machine Learning Research* 17.25 (2016), pp. 1–72 (cit. on pp. 53, 54, 63, 65).

[34] Karl-Hermann Neeb. "Operator-valued positive definite kernels on tubes." In: *Monatshefte für Mathematik* 126.2 (1998), pp. 125–160 (cit. on p. 26).

[35] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006 (cit. on p. 68).

[36] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. "Scikit-learn: Machine learning in Python." In: *Journal of Machine Learning Research* 12.Oct (2011), pp. 2825–2830 (cit. on p. 68).

[37] A. Rahimi and B. Recht. "Random Features for Large-Scale Kernel Machines." In: *NIPS 2007*. 2007, pp. 1177–1184 (cit. on pp. 45, 81, 83, 84).

[38] D Rosenberg, Vikas Sindhwani, P Bartlett, and Partha Niyogi. "A kernel for semi-supervised learning with multi-view point cloud regularization." In: *IEEE Signal Processing Magazine* 26.5 (2009), pp. 145–150 (cit. on p. 53).

[39] Maxime Sangnier, Olivier Fercoq, and Florence D'Alché-Buc. "Joint quantile regression in vector-valued RKHSs." In: (2016) (cit. on p. 53).

[40] V. Sindhwani, H. Q. Minh, and A.C. Lozano. "Scalable Matrix-valued Kernel Learning for High-dimensional Nonlinear Multivariate Regression and Granger Causality." In: *Proc. of UAI'13, Bellevue, WA, USA, August 11-15, 2013*. AUAI Press, Corvallis, Oregon, 2013 (cit. on p. 20).

[41] Vikas Sindhwani and David S Rosenberg. "An RKHS for multi-view learning and manifold co-regularization." In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 976–983 (cit. on p. 53).

[42] Alex J Smola, Bernhard Schölkopf, and Klaus-Robert Müller. "The connection between regularization operators and support vector kernels." In: *Neural networks* 11.4 (1998), pp. 637–649 (cit. on p. 48).

[43] Bharath Sriperumbudur and Zoltan Szabo. "Optimal Rates for Random Fourier Features." In: *Advances in NIPS 28*. Ed. by C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, and R. Garnett. 2015, pp. 1144–1152 (cit. on pp. 81, 82).

[44] Shiliang Sun. "Multi-view Laplacian support vector machines." In: *International Conference on Advanced Data Mining and Applications*. Springer. 2011, pp. 209–222 (cit. on p. 53).

[45] Dougal J. Sutherland and Jeff G. Schneider. "On the Error of Random Fourier Features." In: *Proc. of UAI 2015, July 12-16, 2015, Amsterdam, The Netherlands*. 2015, pp. 862–871 (cit. on pp. 81, 82, 84).

[46] A. W Van Der Vaart and J. A Wellner. "Weak Convergence." In: *Weak Convergence and Empirical Processes*. Springer, 1996, pp. 16–28 (cit. on p. 82).

[47] Jean-Philippe Vert. "Regularization of Kernel Methods by Decreasing the Bandwidth of the Gaussian Kernel." In: () (cit. on p. 48).

[48] G. Wahba. *Spline model for observational data*. Philadelphia, Society for Industrial and Applied Mathematics, 1990 (cit. on p. 54).

[49] N. Wahlström, M. Kok, T.B. Schön, and Fredrik Gustafsson. "Modeling magnetic fields using Gaussian processes." In: *in Proc. of the 38th ICASSP*. 2013 (cit. on p. 21).

[50] Stéfan van der Walt, S Chris Colbert, and Gael Varoquaux. "The NumPy array: a structure for efficient numerical computation." In: *Computing in Science & Engineering* 13.2 (2011), pp. 22–30 (cit. on p. 69).

[51] T. Yang, Y.-F. Li, M. Mahdavi, R. Jin, and Z. Zhou. "Nyström Method vs Random Fourier Features: A Theoretical and Empirical Comparison." In: *NIPS 25*. Ed. by F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger. 2012, pp. 476–484 (cit. on p. 48).

[52] Haizhang Zhang, Yuesheng Xu, and Qinghui Zhang. "Refinement of Operator-valued Reproducing Kernels." In: *Journal of Machine Learning Research* 13 (2012), pp. 91–136 (cit. on p. 26).

DECLARATION

Put your declaration here.

15, Rue Plumet, 75015 - Paris, France, February 16, 2017

Romain Brault

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

LARGE-SCALE LEARNING ON STRUCTURED INPUT-OUTPUT DATA WITH OPERATOR-VALUED KERNELS

KEYWORDS: Operator-Valued Kernels, Large Scale Learning, Random Fourier Features

ABSTACT: Many problems in Machine Learning can be cast into vector-valued functions approximation. Operator-Valued Kernels *Operator-Valued Kernels* and vector-valued Reproducing Kernel Hilbert Spaces provide a theoretical and practical framework to address that issue, extending nicely the well-known framework of scalar-valued kernels. However large scale applications are usually not affordable with these tools that require an important computational power along with a large memory capacity. In this thesis, we propose and study scalable methods to perform regression with *Operator-Valued Kernels*. To achieve this goal, we extend Random Fourier Features, an approximation technique originally introduced for scalar-valued kernels, to *Operator-Valued Kernels*. The idea is to take advantage of an approximated operator-valued feature map in order to come up with a linear model in a finite-dimensional space.

This thesis is structured as follows. First we develop a general framework devoted to the approximation of shift-invariant Mercer kernels on Locally Compact Abelian groups and study their properties along with the complexity of the algorithms based on them. Second we show theoretical guarantees by bounding the error due to the approximation, with high probability. Third, we study various applications of Operator Random Fourier Features to different tasks of Machine learning such as multi-class classification, multi-task learning, time serie modeling, functional regression and anomaly detection. We also compare the proposed framework with other state of the art methods. Fourth, we conclude by drawing short-term and mid-term perspectives of this work.

RÉGRESSION À NOYAUX À VALEURS OPÉRATEURS POUR GRANDS ENSEMBLES DE DONNÉES

MOTS CLEFS: Noyaux à Valeurs Opérateurs, Passage à l'échelle, Random Fourier Features

RÉSUMÉ: Many problems in Machine Learning can be cast into vector-valued functions approximation. Operator-Valued Kernels *Operator-Valued Kernels* and vector-valued Reproducing Kernel Hilbert Spaces provide a theoretical and practical framework to address that issue, extending nicely the well-known framework of scalar-valued kernels. However large scale applications are usually not affordable with these tools that require an important computational power along with a large memory capacity. In this thesis, we propose and study scalable methods to perform regression with *Operator-Valued Kernels*. To achieve this goal, we extend Random Fourier Features, an approximation technique originally introduced for scalar-valued kernels, to *Operator-Valued Kernels*. The idea is to take advantage of an approximated operator-valued feature map in order to come up with a linear model in a finite-dimensional space.

This thesis is structured as follows. First we develop a general framework devoted to the approximation of shift-invariant Mercer kernels on Locally Compact Abelian groups and study their properties along with the complexity of the algorithms based on them. Second we show theoretical guarantees by bounding the error due to the approximation, with high probability. Third, we study various applications of Operator Random Fourier Features to different tasks of Machine learning such as multi-class classification, multi-task learning, time serie modeling, functional regression and anomaly detection. We also compare the proposed framework with other state of the art methods. Fourth, we conclude by drawing short-term and mid-term perspectives of this work.

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available at `https://bitbucket.org/amiede/classicthesis/` for both LaTeX and LyX.

Université Paris-Saclay
ED STIC – 580
Université Paris Sud, Bâtiment 650 Ada Lovelace, 91405 Orsay Cedex, France.