Kernel methods for uncertainty propagation and sensitivity analysis

HIPERWIND Summer School, August 28 – 31, 2023,
DTU Risø Campus, Roskilde, Denmark

Vincent Chabridon[1,3], Elias Fekhari[1,2], Julien Pelamatti[1,3]

[1]EDF R&D, 6 quai Watier, 78401 Chatou, France
[2]Université Côte d'Azur, 28 Avenue de Valrose, Nice, 06000, France
[3]SINCLAIR AI Lab., Saclay, France

Foreword and acknowledgements

◆ Foreword

❑ This course is given in the context of the HIPERWIND Summer School

❑ The HIPERWIND project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 101006689

◆ Acknowledgements

❑ To Drs. **Gabriel Sarazin** (CEA), **Amandine Marrel** (CEA) and Prof. **Sébastien Da Veiga** (ENSAI) for the fruitful discussions about RKHSs, HSIC indices and UQ in general

❑ To Dr. **Anaïs Lovera** (EDF R&D) for her support as a Project Manager

About the French UQ community

◆ Main organizations and networks

- ❑ **GdR MASCOT-NUM** ➪ a French Research Group dealing with stochastic methods for the analysis of numerical codes
- ❑ **GIS LARTISSTE** ➪ a French Scientific Consortium about UQ @ Paris-Saclay
- ❑ **SINCLAIR AI Lab.** ➪ the Saclay INdustrial Collaborative Laboratory for Artificial Intelligence Research (SINCLAIR), gathering researchers from EDF, Thales and TotalEnergies
- ❑ **frENBIS** ➪ the French local network of the European Network for Business and Industrial Statistics (ENBIS)

◆ Main seminars and scientific events

- ❑ **UQSay seminars** ➪ series of seminars on the broad area of UQ, ML and related topics
- ❑ **ETICS Annual Research Schools** ➪ Thematic Research School on Uncertainty in Scientific Computing

Doing Research and Development @ EDF R&D

- **Our research group**
  - Department ➪ "Performance, Industrial Risk, Monitoring for Maintenance and Operating" (~ 130 people)
  - Group ➪ "Asset Management, Uncertainty Quantification and Statistical Learning" (~ 20 permanent researchers, 4 to 8 PhD students and Master students)

- **The main tools we develop for UQ/ML**
  - OpenTURNS ➪ an open source library for UQ (Python / C++)
  - Persalys ➪ a friendly GUI (open source) for performing UQ, data visualization, ...
  - `sensitivity` ➪ R package for sensitivity analysis of model outputs
  - `mistral` ➪ R package for structural reliability methods

Introduction

Industrial context and motivations

◆ **A road trip through uncertainties!**

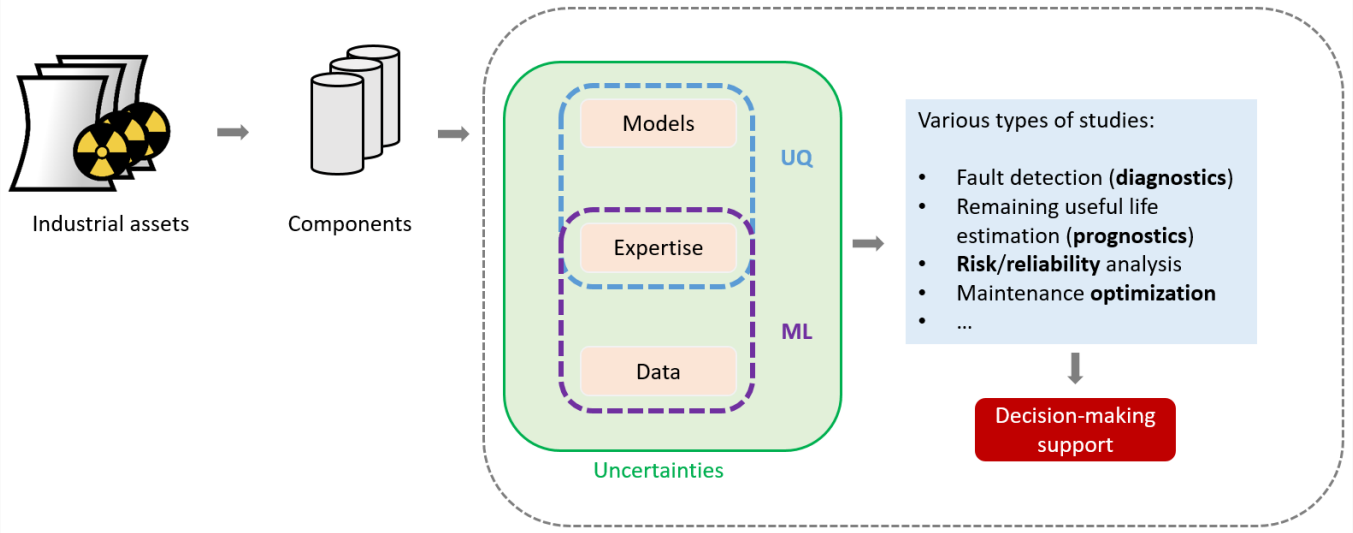


Figure 1: Dealing with uncertainties in an industrial process (©EDF).

☞ About UQ in industrial practice: [DRDT08, DR12]

Uncertainty Quantification in a nutshell!

◆ Verification, Validation & Uncertainty Quantification

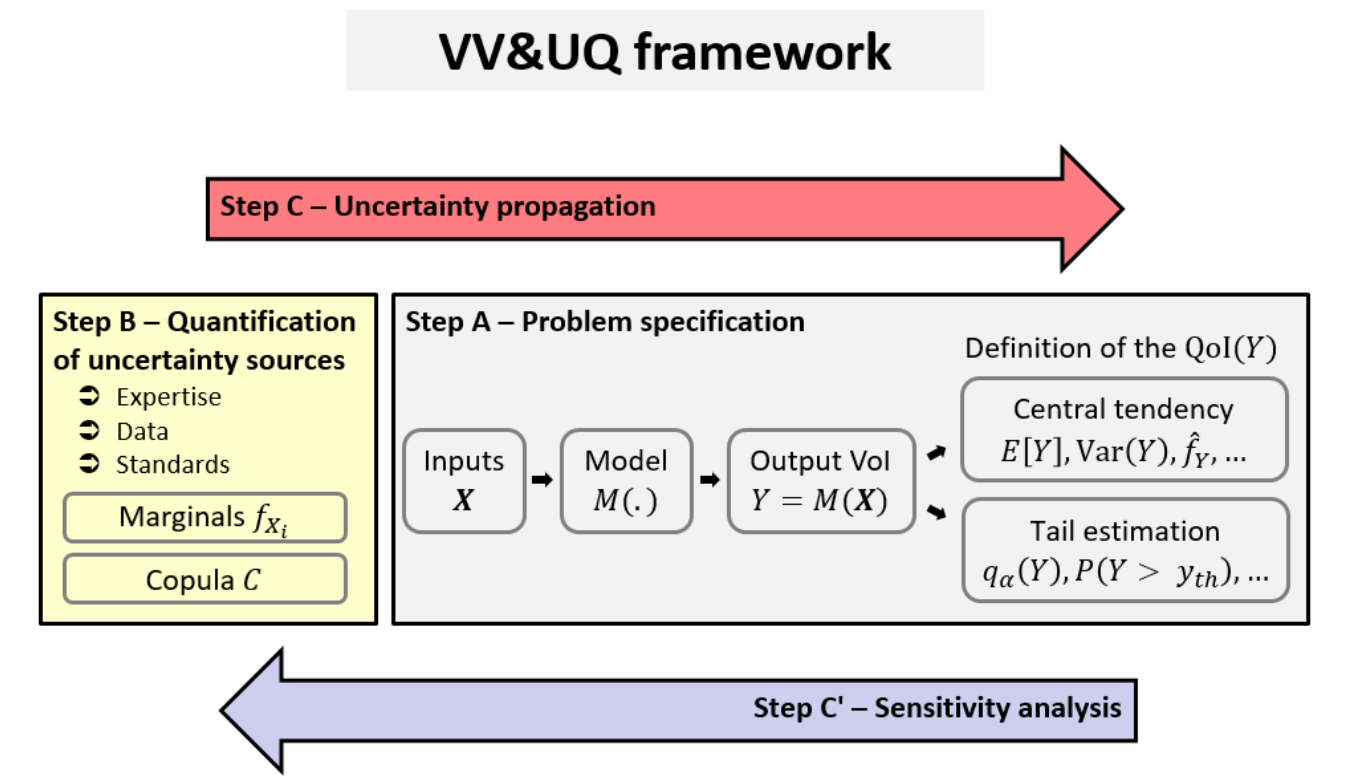


Figure 2: VV&UQ framework (©EDF).

☞ To go further into UQ in general: [Smi13, Sul15]
☞ To go further into Uncertainty Propagation: [MB15, DK22]
☞ To go further into Surrogate Modeling: [Bou18]
☞ To go further into SA: [SRA+08, DGIP21]

Supervised Machine Learning in a nutshell!

◆ **Supervised Machine Learning ...**
(not that much different from UQ)

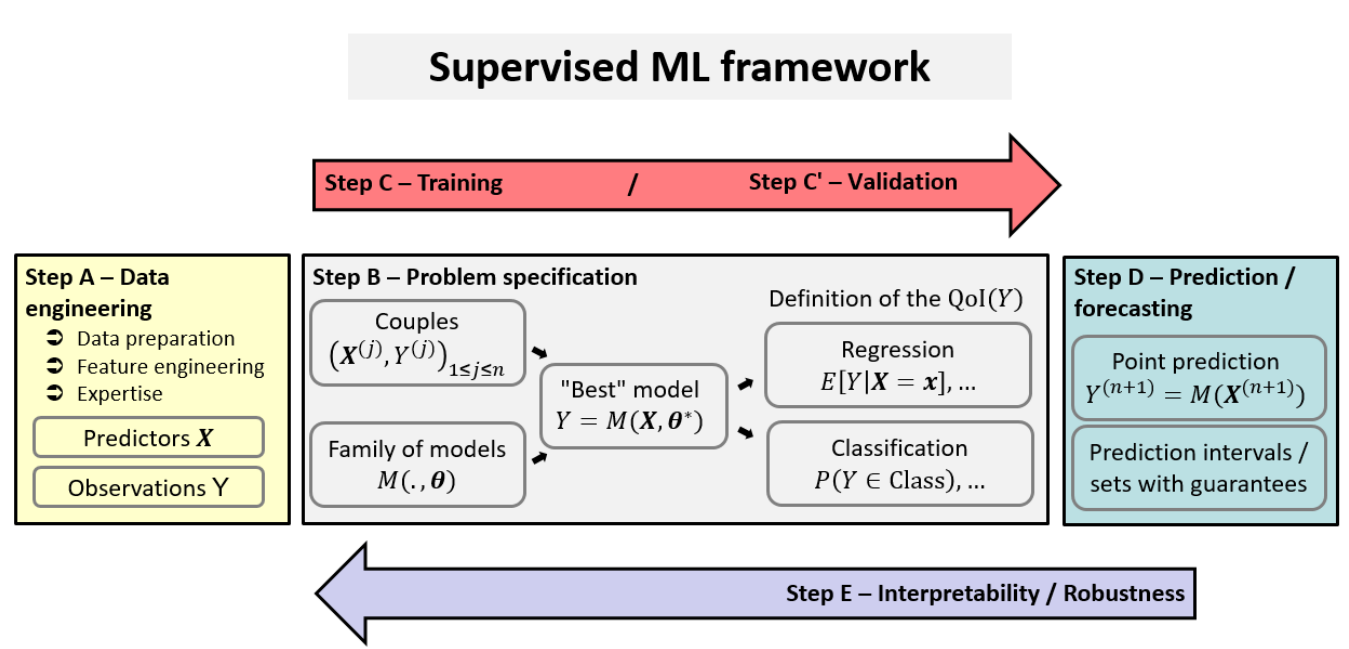


Figure 3: Supervised ML (©EDF).

☞ To further into ML in general: [Bis06, HTF09]
☞ To go further into the links between SQ (UQ) and Interpretability/Explainability (ML): [Mol22, ICT22]

Main objectives of this course

◆ **What about the possible links between UQ and ML?**

❑ Historically, people thought UQ and ML were two distinct disciplines:

- ➤ UQ ➪ dedicated to computer experiments, based on physical / numerical models
- ➤ ML ➪ dedicated to real data, no model available

❑ Recently (> 2008-2009), people tend to see more and more similarities between UQ and ML, e.g., in the fields of:

- ➤ surrogate modeling ≡ supervised statistical learning
- ➤ global sensitivity analysis ≡ explainability/interpretability
- ➤ ...

❑ Even more recently (> 2015), people try more to adopt "hybrid approaches" by combining them!

Kernel methods

Kernel methods are one of the cornerstones between these two disciplines.

Main objectives of this course

- **With this course, you probably will ...**
  - ✔ Have an overview about "why kernels are so popular?"
  - ✔ Have a first glimpse about the mathematical foundations of these methods
  - ✔ Have access to a panel of references to go further into some topics
  - ✔ A good opportunity for discovering (only a few) kernel methods and to **use them for real!**
  - ✔ Discover the (great) **OpenTURNS software**!
- **With this course, you probably won't ...**
  - ✘ Be an expert in kernel methods
  - ✘ Be able to solve all your complicated problems with kernel methods

Table of contents

1. Introduction

2. Mathematical background about kernels for UQ and ML

3. Kernel-based uncertainty propagation

4. Kernel-based sensitivity analysis

2. Mathematical background

Why should we play with “kernels”?

◆ A few motivations

❑ Imagine you have this 1D dataset with two different labels • and •

Figure 4: A 1D dataset (source: [Sut22]).

❑ You want to do some “ML stuff” in order to have a **good classification model**!

❑ Assume a linear model: $f(x) = \beta_0 + \beta_1 x$ such that $\widehat{y}(x) = \text{sign}\left(f(x)\right)$

❑ What do you typically obtain?

Why should we play with “kernels”?

◆ **A few motivations (cont’d)**

Figure 5: 1D dataset : a first classification model (source: [Sut22]).

❑ OK! That’s clearly not a good model!

❑ Let’s extend x such that:

$$f(x) = \boldsymbol{\beta}^\top (1, x, x^2) = \boldsymbol{\beta}^\top \psi(\mathbf{x})$$

❑ What do we obtain?

Why should we play with “kernels”?

◆ **A few motivations (cont’d)**

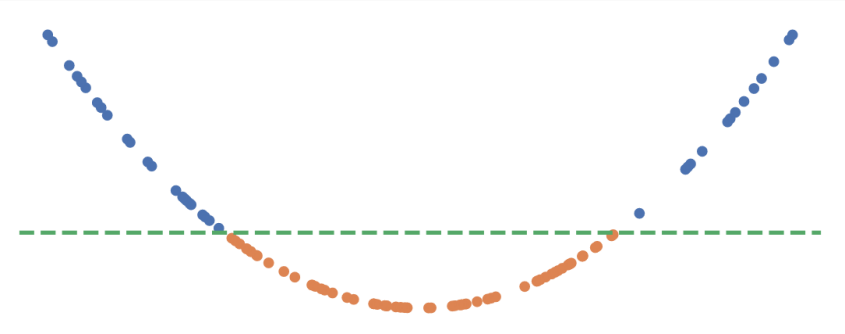


Figure 6: 1D dataset : a second classification model (source: [Sut22]).

- ❏ OK! The model does the job!
- ❏ **Kernels** are tools that enable to do this job with any (potentially) very complicated $\psi(\cdot)$ functions
- ❏ Kernels are used in modern (supervised and unsupervised) ML in order to treat very complicated data with, e.g., time series, images, graphs, text, videos, ...
- ❏ $\psi(\cdot)$ will leave in a specific mathematical space called a **reproducing kernel Hilbert space**

Which “kernels” are we talking about?

- **The word “kernel” is widely used and can be misunderstood** ⚠
  - The concept treated in this course: “positive definite kernel”, “RKHS kernel”, (“Mercer kernel”)
  - A semi-related concept: “kernel density estimation” or “kernel smoothing” (nonparametric statistics)
  - Unrelated notions[1]:
    - The kernel (null space) of a linear map (linear algebra)
    - The kernel of a convolution (image processing)
    - CUDA and Linux kernels
    - Popcorn kernels

[1] This brilliant list is directly borrowed from the wonderful course given by Danica J. Sutherland, available here: https://github.com/djsutherland/etics-kernels-22.

Kernels: definition, properties and links to RKHSs

◆ What is a kernel? A first definition!

- ❑ Let $\mathcal{Z}$ be a space of interest, e.g., such that $\mathcal{Z} \subseteq \mathbb{R}^d$
- ❑ A function $K : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ is said to be a **kernel** if it is symmetric and positive definite (pd):
  - ➤ **Symmetry**: $\forall\, z, z' \in \mathcal{Z}, K(z, z') = K(z', z)$
  - ➤ **Positive definiteness**[2]:

$$\forall\, n \geq 1, \forall\, z_1, \ldots, z_n \in \mathcal{Z}, \forall\, c_1, \ldots, c_n \in \mathbb{R}, \sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j K(z_i, z_j) \geq 0$$

- ❑ Equivalently, the kernel function can be written following a "matrix viewpoint":

$$\mathbf{c}^\top \mathbf{K}_n \mathbf{c} \geq 0 \quad \text{with} \quad \mathbf{K}_n := [K(z_i, z_j)]_{1 \leq i,j \leq n} \quad \text{and} \quad \mathbf{c} := [c_i]_{1 \leq i \leq n}$$

- ❑ The matrix $\mathbf{K}_n$ is usually called the **Gram matrix** (or **kernel matrix**)

[2] ⚠ K is pd $\Leftrightarrow$ all Gram matrices from K are positive semi-definite (psd).

A detour through Hilbert spaces

◆ **What is an Hilbert space $\mathcal{H}$?**

- ❑ A complete (real or ~~complex~~) inner product space
- ❑ Inner product space ➪ a vector space with an **inner product**:
  - ➤ **Linearity**: $\forall\, h_1, h_2, h_3 \in \mathcal{H}, \forall\, a, b \in \mathbb{R}, \langle ah_1 + bh_2, h_3\rangle_{\mathcal{H}}$
  - ➤ **Symmetry**: $\forall\, h_1, h_2 \in \mathcal{H}, \langle h_1, h_2\rangle_{\mathcal{H}} = \langle h_2, h_1\rangle_{\mathcal{H}}$
  - ➤ **Positive definiteness**:
    $\forall\, h \in \mathcal{H}, \langle h, h\rangle_{\mathcal{H}} \geq 0$ and $\langle h, h\rangle_{\mathcal{H}} = 0 \quad \Rightarrow \quad h = 0$
- ❑ This inner product induces a **norm** on $\mathcal{H}$: $||h||_{\mathcal{H}} := \sqrt{\langle h, h\rangle_{\mathcal{H}}}$
- ❑ Completeness ➪ "well-behaved" (i.e., any Cauchy sequence in $\mathcal{H}$ converges and has a limit in $\mathcal{H}$)

◆ **Examples of well-known (and applications of) Hilbert spaces:**

➥ Lebesgue spaces, Sobolev spaces, ...

➥ Applications in Quantum Physics, Signal Processing, Probability & Statistics, ML, Computational Engineering, ...

A detour through Hilbert spaces

◆ **Do you have a map? Because I'm lost ...**

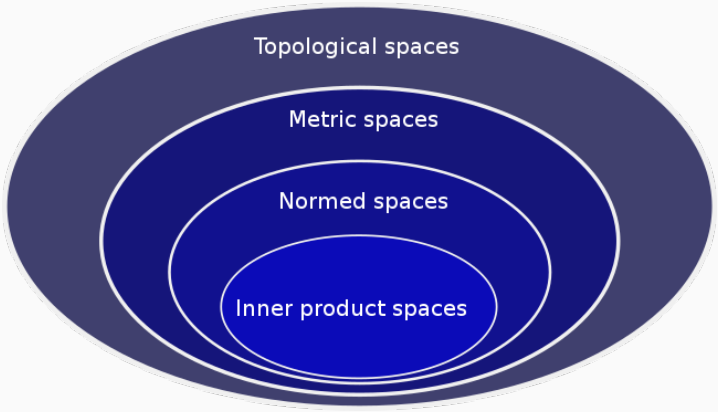


Figure 7: Hierarchy of mathematical spaces (source: Wikipedia).

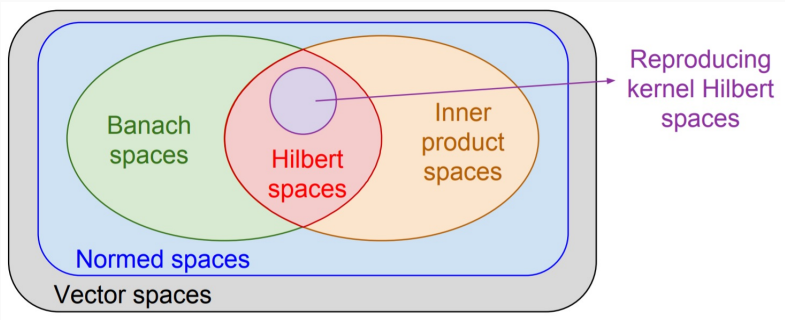


Figure 8: A more refined Venn diagram of mathematical spaces (source: https://ngilshie.github.io).

Focus in this talk

A special class of spaces will be of interest in this course:
the so-called "**reproducing kernel Hilbert spaces**" (**RKHSs**)

Kernels: definition and first properties

◆ Kernel: an inner product between feature maps

- ❑ One considers a set of interest $\mathcal{Z}$ (e.g., $\mathbb{R}^d$, functions, distributions of graphs / images, ...)
- ❑ $K : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ is a **kernel** on $\mathcal{Z}$ if there exists a Hilbert space $\mathcal{G}$ and a **feature map** $\psi : \mathcal{Z} \to \mathcal{G}$ so that:

$$\forall\, z, z' \in \mathcal{Z}, K(z, z') = \langle \psi(z), \psi(z') \rangle_{\mathcal{G}} \tag{1}$$

- ❑ Vocabulary:
  - ➤ ψ is called a **feature map**
  - ➤ $\mathcal{G}$ is called a **feature space**
  - ➤ $\psi(z)$ with $z \in \mathcal{Z}$ are the **feature functions** (or **features**)

◆ A few first properties

- ❑ $\forall\, K, \exists\, \psi$ **such that Eq.** (1) **holds, but** ψ **can be implicit**
- ❑ $\forall\, \psi : \mathcal{Z} \to \mathcal{G}$ **(with** $\mathcal{G}$ **an HS), the function** K **given by Eq.** (1) **is always a kernel**

Kernels: definition and first properties

◆ Kernel and feature map: the "hidden mechanism"

- ❑ Kernel K and feature map ψ are intimately related to each other
- ❑ The feature map ψ offers an alternative viewpoint on the kernel K
- ❑ Given two points z and z' living in an Euclidean space $\mathcal{Z}$, evaluating K leads to:
  - ➤ creating two feature functions $\psi(z), \psi(z')$ in a (possibly very sophisticated) feature space $\mathcal{G}$
  - ➤ comparing them using the existing metric in $\mathcal{G}$
- ❑ When $\mathcal{G}$ is infinite-dimensional ⇨ a much richer and subtle information is captured in $\mathcal{G}$ than in $\mathcal{Z}$
- ❑ The "kernel trick":
  - ➤ mathematically speaking ⇨ comparing $\psi(z)$ and $\psi(z')$ captures a rich information
  - ➤ numerically speaking ⇨ comparing $\psi(z)$ and $\psi(z')$ **only requires to compute** $K(z, z')$

Back to spaces: the "reproducing kernel Hilbert spaces"

◆ What is a "reproducing kernel Hilbert space" (RKHS)?

- ❑ Some basic ingredients first: let $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}}, ||\cdot||_{\mathcal{H}})$ be a Hilbert space (with inner product and associated norm) of functions mapping from some nonempty set $\mathcal{Z}$ to $\mathbb{R}$ ➪ denoted by $\mathcal{H} \subset \mathbb{R}^{\mathcal{Z}}$
- ❑ A first (desirable) property: if two functions $f \in \mathcal{H}$ and $g \in \mathcal{H}$ are close in the norm of $\mathcal{H}$, then, $f(z)$ and $g(z)$ are close for all $z \in \mathcal{Z}$
- ❑ Evaluation functional:

$$L_z : \left| \begin{array}{ccc} \mathcal{H} & \longrightarrow & \mathbb{R} \\ h & \longmapsto & L_z[h] = h(z) \end{array} \right.$$

- ❑ All those functionals are actually **linear forms**

☆ Definition (RKHS)

$\mathcal{H}$ RKHS if all the evaluation functionals (i.e., $\forall\, z \in \mathcal{Z}$) are bounded, i.e.,
$\forall\, z \in \mathcal{Z}, \exists C_z > 0 \quad$ such that $\quad |h(z)| \leq C_z ||h||_{\mathcal{H}}$.

☞ To go further: [SG15] and [PR16, Ch. 1, p. 4]

Back to spaces: the "reproducing kernel Hilbert spaces"

◆ **What is a "reproducing kernel Hilbert space" (RKHS)?**

❑ Consequences:

- ➪ functions in the RKHS are "smooth" in the sense of the previous definition
- ➪ RKHSs are particularly "well-behaved" spaces (relative to other Hilbert spaces)

☆ Corollary (norm convergence in $\mathcal{H} \Rightarrow$ pointwise convergence)

If two functions converge in RKHS norm, then they converge pointwise, i.e., if $\lim_{n\to\infty} ||h_n - h||_{\mathcal{H}} = 0$, then $\lim_{n\to\infty} h_n(z) = h(z), \forall\ z \in \mathcal{Z}$.

☞ To go further: [BTA04, Ch. 1, p. 10]

Reproducing kernel and feature maps

◆ **What about the "reproducing property"?**

❑ Riesz representation theorem: one applies it to all evaluation functionals (bounded linear operators)

$$\forall\, z \in \mathcal{Z}, \exists!\, \kappa_z \in \mathcal{H}, \quad \text{such that} \quad \forall\, h \in \mathcal{H}, L_z[h] = \langle h, \kappa_z \rangle_{\mathcal{H}}$$

➪ ⚠ this theorem ensures that, $\forall\, z \in \mathcal{Z}$, we can find a **unique** element in $\mathcal{H}$ that is a **representer** of the evaluation $h(z)$

❑ Point evaluation property: equipped with this representer, one has:

$$\forall\, z, z' \in \mathcal{Z}, K(z, z') := \langle \kappa_{z'}, \kappa_z \rangle_{\mathcal{H}} = \kappa_z(z') = \kappa_{z'}(z)$$

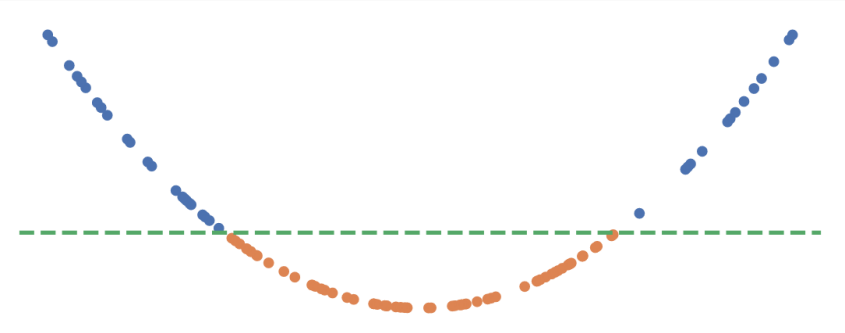
❑ Reproducing property: bringing what precedes together leads to:

$$\forall\, h \in \mathcal{H}, \forall\, z \in \mathcal{Z}, h(z) = \langle h, K(\cdot, z) \rangle_{\mathcal{H}}$$

➪ ⚠ K is the only kernel (**reproducing kernel**) that verifies this property!

Reproducing kernel and feature maps

◆ **Illustration of the reproducing property: back to the initial motivating example**



- Recall that: $\mathcal{Z} = \mathcal{X} = \mathbb{R}$ and $\psi(x) = (1, x, x^2) \in \mathbb{R}^3$
- Kernel is given by: $K(x, y) = \langle \psi(x), \psi(y) \rangle_{\mathcal{H}} = 1 + xy + x^2 y^2$
- Classifier based on a linear function f such that: $f(x) = \langle \boldsymbol{\beta}, \psi(x) \rangle_{\mathcal{H}}$
  - with $f(\cdot)$ the function itself (which corresponds to the vector $\boldsymbol{\beta} \in \mathbb{R}^3$
  - $f(x) \in \mathbb{R}$ which is the function evaluated at a point x
- Reproducing property: $f(x) = \langle f(\cdot), \psi(x) \rangle_{\mathcal{H}}$ for $f \in \mathcal{H}$

Reproducing kernel and feature maps

◆ The canonical feature map

❑ We just saw (reproducing property):

$$\forall\, h \in \mathcal{H}, \forall\, z \in \mathcal{Z}, h(z) = \langle h, K(\cdot, z)\rangle_{\mathcal{H}}$$

❑ Canonical feature map:

$$\forall\, z, z' \in \mathcal{Z}, K(z, z') = \langle \theta(z'), \theta(z)\rangle_{\mathcal{H}}$$

➪ $\theta(z) := \kappa_z$ is called the **canonical feature map**

➪ the associated RKHS is precisely the feature space $\mathcal{H}$

❑ ⚠ **This canonical feature map may be (often) not very instructive for several reasons** (not discussed here, unless questions ...)

➪ OK! Never mind! Luckily, $\exists$ many other feature maps and associated feature spaces (with very $\neq$ mathematical natures)

➪ ⚠ **There's no common rationale to establish / extract feature maps that are easy to understand!**

Reproducing kernel and feature maps

◆ **Kernel and feature maps: what do they look like?**

The dot-product (a.k.a. linear) kernel ☞ [SMDVC23]

❑ Kernel:

$$K_{\text{lin}} : \left| \begin{array}{ccc} \mathbb{R}^2 & \longrightarrow & \mathbb{R} \\ (z, z') & \longmapsto & K_{\text{lin}}(z, z') = zz' \end{array} \right.$$

❑ Feature map: ⚠ It is nonunique!

$$\begin{aligned} \forall\, z, z' \in \mathbb{R},\, K_{\text{lin}}(z, z') &= \langle \psi_1(z'), \psi_1(z) \rangle_{\mathbb{R}} \text{ with } \psi_1(z) := z \\ &= \langle \psi_2(z'), \psi_2(z) \rangle_{\mathbb{R}^2} \text{ with } \psi_2(z) := \frac{1}{\sqrt{2}}[z, z]^\top \\ &= \langle \psi_d(z'), \psi_d(z) \rangle_{\mathbb{R}^d} \text{ with } \psi_d(z) := \frac{1}{\sqrt{d}}[z, \ldots, z]^\top \end{aligned}$$

❑ Interpretation: $3 \neq$ feature spaces $(\mathbb{R}, \mathbb{R}^2, \mathbb{R}^d)$ but 3 feature maps that convey similar information ➪ here, meaningless!

Reproducing kernel and feature maps

The polynomial kernel ☞ [SMDVC23]

- Kernel:

$$K_{\text{poly}} : \left| \begin{array}{ccc} \mathbb{R}^2 & \longrightarrow & \mathbb{R} \\ (z, z') & \longmapsto & K_{\text{poly}}(z, z') = (zz' + c)^m \end{array} \right. \quad \text{with } c \in \mathbb{R}_+ \text{ and } m \in \mathbb{N}^*$$

Using binomial theorem, one has:

$$\begin{aligned} K_{\text{poly}}(z, z') &= \sum_{k=0}^{m} \binom{m}{k} (xx')^k c^{m-k} \\ &= \sum_{k=0}^{m} \left(\sqrt{\binom{m}{k}} (\sqrt{c})^{m-k} x^k \right) \left(\sqrt{\binom{m}{k}} (\sqrt{c})^{m-k} (x')^k \right) \end{aligned}$$

- Feature map: $K_{\text{poly}}(z, z') = \langle \psi_{\text{poly}}(z'), \psi_{\text{poly}}(z) \rangle_{\mathbb{R}^{m+1}}$

$$\text{with } \psi_{\text{poly}} : \left| \begin{array}{ccc} \mathbb{R} & \longrightarrow & \mathbb{R}^{m+1} \\ z & \longmapsto & \left[\sqrt{\binom{m}{k}} (\sqrt{c})^{m-k} x^k \right]_{0 \leq k \leq m} \end{array} \right.$$

- Interpretation: ψ_{poly} ➪ $m + 1$ polynomial features with increasing degrees

Reproducing kernel and feature maps

◆ Techniques for constructing new kernels

- ❑ Kernels are like "building blocks" ➪ they can be combined in order to make new kernels!
- ❑ Building valid kernels:
  - ➤ Scaling: if $\alpha \geq 0$, $K_\alpha(z, z') := \alpha K(z, z')$ is a kernel
  - ➤ Sum: $K_+(z, z') = K_1(z, z') + K_2(z, z')$ is a kernel
  - ➤ Limit: if $K_\infty(z, z') = \lim_{m \to \infty} K_m(z, z')$ exists, then K_∞ is a kernel
  - ➤ Product: $K_\times(z, z') = K_1(z, z')K_2(z, z')$ is a kernel
  - ➤ Power: $K_p(z, z') = K(z, z')^p$ is a kernel for any $p \geq 0$
  - ➤ Exponential: $K_{\exp}(z, z') = \exp[K(z, z')]$ is a kernel
  - ➤ "Sandwhich": if $f : \mathcal{Z} \to \mathbb{R}$, $K_f(z, z') = f(z)K(z, z')f(z')$ is a kernel
  - ➤ ...
- ❑ But ⚠ expliciting the feature map can be very difficult!

✎ Exercise

Starting from a very simple kernel, try to rebuild the Gaussian one!

☞ To go further: [Sut22] and [Bis06, Ch. 6, p. 296]

Links between kernels and RKHSs

◆ One-to-one mapping between kernels and RKHS

❑ Let's quickly recap:

- ➤ a **unique** reproducing kernel can be extracted from any given RKHS
- ➤ any given kernel K verifies the reproducing property for **one and only one** RKHS

☆☆ Theorem (shortened version of "Moore-Aronszajn")

For every pd function $K(\cdot, \cdot)$ on $\mathcal{Z} \times \mathcal{Z}$, there exists a unique RKHS with K as its reproducing kernel. Conversely, the reproducing kernel of an RKHS is unique and pd.

☞ To go further: [Aro50], [MFSS17, Ch. 2, Th. 2.5, p. 24] and [BTA04, Ch. 1, Th. 3, p. 19]

★★ Take-home message #1 ★★

✌ To keep in mind ✌

- We saw 4 main mathematical objects: a kernel K, a feature space $\mathcal{H}$ (RKHS), feature maps ($\psi(\cdot)$) and features ($\psi(z)$)
- If K is a (reproducing) kernel, there **always** exists a RKHS $\mathcal{G}$ and a function $\psi : \mathcal{Z} \to \mathcal{G}$ such that:

$$\forall\, z, z' \in \mathcal{Z}, K(z, z') = \langle \psi(z), \psi(z') \rangle_{\mathcal{G}}$$

- Kernel $\equiv$ inner product between feature maps in a RKHS
- The feature map is not unique! The canonical one is, often, not really informative!
- Expliciting the feature map (when possible) enables to clearly understand what is "captured" by the feature functions!

Kernel mean embedding of distributions

◆ Kernel mean embedding: a first intuition

- ❑ Roughly speaking, kernels can be used to define a sort of "(dis)similarity" between two objects (points, distributions, ...)
  - ➪ Kernels are used to **compare** two elements!
  - ➪ Usually, in UQ, we do not deal with two points, but with **probability distributions**
- ❑ Represent a point $z \in \mathcal{Z}$ as $K(\cdot, z)$: $h(z) = \langle h, K(\cdot, z) \rangle_{\mathcal{H}}$
- ❑ How to represent a distribution ν in this framework?
  - ➪ **Core idea**: to map distributions into a RKHS using a reproducing kernel!

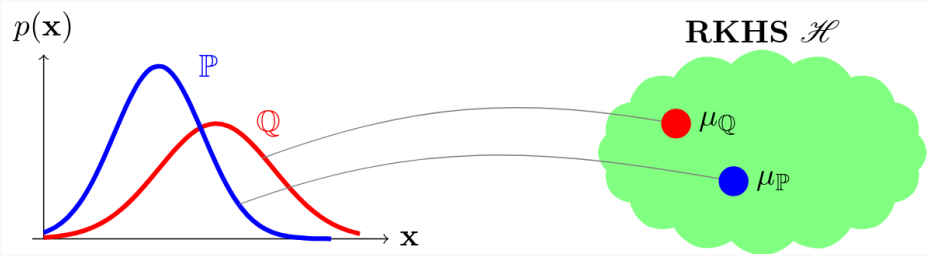


Figure 9: Embedding of two marginal distributions (source: [MFSS17]).

Kernel mean embedding of distributions

◆ Kernel mean embedding (cont'd)

❑ ⚠ I skip deliberately many details, but, only assume that:

- ➤ $\mathcal{Z}$ is always the space of our objects of interest (our **data**)
- ➤ ν is a probability measure with support in $\mathcal{Z}$ ($Z \sim \nu$)
- ➤ $\mathcal{M}_1^+(\mathcal{Z})$ is the space of all Radon probability measures ($\nu \in \mathcal{M}_1^+(\mathcal{Z})$)

❑ Represent a distribution ν:

- ➤ Consider the linear form $L_\nu : h \in \mathcal{H} \mapsto \mathbb{E}_\nu\left[h(Z)\right]$
  ➪ for any $h \in \mathcal{H}$, L_ν computes the mean value wrt. ν
- ➤ ... boundedness of the evaluation functional + Riesz representation theorem:

$$\exists!\ \mu_\nu \in \mathcal{H}, \quad \text{such that} \quad \forall\ h \in \mathcal{H}, L_\nu[h] = \langle h, \mu_\nu \rangle_{\mathcal{H}}$$

- ➤ All this process ensures that: **the initial probability measure ν admits a unique image μ_ν in the RKHS**

❑ Kernel mean embedding of ν:
➪ defined as the Riesz representation of L_ν

Kernel mean embedding of distributions

◆ **An illustration: kernel mean embedding of two probability distributions**

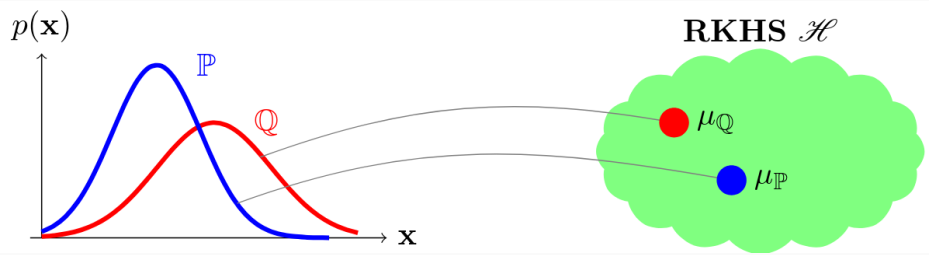


Figure 10: Embedding of two marginal distributions (source: [MFSS17]).

- (Recall) Represent a point $z \in \mathcal{Z}$ as $K(\cdot, z)$: $h(z) = \langle h, K(\cdot, z) \rangle_{\mathcal{H}}$
- Now, suppose one has $X \sim \mathbb{P}$ and $Y \sim \mathbb{Q}$
- Represent a distribution $\mathbb{P}$ as $\mu_{\mathbb{P}}$: $\mathbb{E}_{X \sim \mathbb{P}}[h(X)] = \langle h, \mu_{\mathbb{P}} \rangle_{\mathcal{H}}$
  ➪ $\mathbb{E}_{X \sim \mathbb{P}}[h(X)] = \mathbb{E}_{X \sim \mathbb{P}}[\langle h, K(\cdot, X) \rangle_{\mathcal{H}}] = \langle h, \mathbb{E}_{X \sim \mathbb{P}}[K(\cdot, X)] \rangle_{\mathcal{H}}$
  (+ assumption that $\mathbb{E}_{\mathbb{P}}\left[\sqrt{K(X, X)}\right] < \infty$)
- Thus, one has:

$$\langle \mu_{\mathbb{P}}, \mu_{\mathbb{Q}} \rangle_{\mathcal{H}} = \mathbb{E}_{X \sim \mathbb{P}, Y \sim \mathbb{Q}}[K(X, Y)]$$

- Okay! Why do we do that? ➪ **Comparing distributions!**

Maximum Mean Discrepancy

◆ Comparing two probability distributions

- ❏ Several metrics exist two compare two (or more) probability distributions (metrics, divergences, ...)
- ❏ Among this large panel of distances / divergences / metrics, one is of particular interest:
  ➪ the **Maximum Mean Discrepancy (MMD)**
- ❏ A first expression:

$$\text{MMD} : \left| \begin{array}{rcl} \mathcal{M}_1^+(\mathcal{Z}) \times \mathcal{M}_1^+(\mathcal{Z}) & \longrightarrow & \mathbb{R}_+ \\ (\nu_1, \nu_2) & \longmapsto & ||\mu_{\nu_1} - \mu_{\nu_2}||_{\mathcal{H}} \end{array} \right. \tag{2}$$

- ❏ A second (more useful) expression:

$$\begin{aligned} \text{MMD}^2(\nu_1, \nu_2) = \mathbb{E}_{\nu_1 \otimes \nu_1}\left[K(Z, Z')\right] + \mathbb{E}_{\nu_2 \otimes \nu_2}\left[K(Z, Z')\right] \\ - 2\mathbb{E}_{\nu_1 \otimes \nu_2}\left[K(Z, Z')\right] \end{aligned} \tag{3}$$

Maximum Mean Discrepancy

◆ Properties of the MMD

- ❑ MMD is a very useful tool in UQ and ML due to its interesting properties
- ❑ In short:
  - ➤ If K is a **characteristic**[3] kernel, then the mapping $\nu \mapsto \mu_\nu$ is **injective**, and thus the MMD is a **distance** on $\mathcal{M}_1^+(\mathcal{Z})$
    ➪ It gives us:

$$\mathrm{MMD}(\nu_1, \nu_2) = 0 \quad \Longrightarrow \quad \nu_1 = \nu_2 \tag{4}$$

  - ➤ Can be estimated using only samples and evaluating kernels (i.e., through Gram matrices)

[3] Not discussed in this course.

★★ Take-home message #2 ★★

✌ To keep in mind ✌

- ❏ We saw the mechanism of "kernel mean embedding"
- ❏ We can apply it to many types of data (points, distributions, ...)
- ❏ Such a tool is useful to compare, e.g., two probability distributions!
- ❏ To do so, a relevant metric exists: the **MMD**
- ❏ It has nice properties, among others:
  - ➤ being easy to estimate (because of the "kernel trick")
    - ➪ only expectations of kernels!
  - ➤ if K is **characteristic**, one has the nice property:

$$\mathrm{MMD}(\nu_1, \nu_2) = 0 \quad \Longrightarrow \quad \nu_1 = \nu_2$$

3. Kernel-based uncertainty propagation

Uncertainty propagation: a generic problem

◆ **Goal:** propagate the random vector $\mathbf{X}$ through the function g

$$\mathbb{E}[\,Y\,] = \mathbb{E}[g(\mathbf{X})] = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x})\, f_\mathbf{X}(\mathbf{x})\, \mathrm{d}\mathbf{x} = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x})\, \mathrm{d}\mu(\mathbf{x}) \tag{5}$$

❑ This equation can be written in terms of probability density function $f_\mathbf{X}$ or probability measure μ

❑ The integrand $g : \mathcal{D}_\mathbf{X} \to \mathbb{R}$ can be:

➤ computationally costly, nonlinear, stochastic

❑ The random input vector $\mathbf{X} \in \mathcal{D}_\mathbf{X}$ can:

➤ present a dependence structure

➤ be defined explicitly or only empirically (i.e., through a dataset)

Uncertainty propagation: a generic problem

◆ **Goal:** propagate the random vector $\mathbf{X}$ through the function g

$$\mathbb{E}[Y] = \mathbb{E}[g(\mathbf{X})] = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x})\, f_\mathbf{X}(\mathbf{x})\, \mathrm{d}\mathbf{x} = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x})\, \mathrm{d}\mu(\mathbf{x}) \tag{5}$$

- ❑ This equation can be written in terms of probability density function $f_\mathbf{X}$ or probability measure μ
- ❑ The integrand $g : \mathcal{D}_\mathbf{X} \to \mathbb{R}$ can be:
  - ➤ computationally costly, nonlinear, stochastic
- ❑ The random input vector $\mathbf{X} \in \mathcal{D}_\mathbf{X}$ can:
  - ➤ present a dependence structure
  - ➤ be defined explicitly or only empirically (i.e., through a dataset)

◆ **Remarks:**

- ❑ We can encounter this problem in various steps of the UQ framework
- ❑ It is a numerical integration problem against a density (sometimes called probabilistic integration) [BOG+19]

Approximation by a quadrature rule

A quadrature rule approximates a multivariate integral:

$$\mathbb{E}[g(\mathbf{X})] = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x})\, f_\mathbf{X}(\mathbf{x})\, \mathrm{d}\mathbf{x} \approx \sum_{i=1}^{n} w_i\, g(\mathbf{x}^{(i)}), \quad n \in \mathbb{N} \tag{6}$$

- A quadrature rule is a weighted mean of:
  - the function g evaluated at the set of nodes $\mathbf{X}_n = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$
  - weighted by a set of weights $\mathbf{w}_n = \{w_1, \ldots, w_n\} \in \mathbb{R}^n$
- A "good" quadrature offers an accurate approximation for a restricted number of nodes.
- Beyond performance, convergence guarantees are essential

Approximation by a quadrature rule

A quadrature rule approximates a multivariate integral:

$$\mathbb{E}[g(\mathbf{X})] = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x})\, f_\mathbf{X}(\mathbf{x})\, \mathrm{d}\mathbf{x} \approx \sum_{i=1}^{n} w_i\, g(\mathbf{x}^{(i)}), \quad n \in \mathbb{N} \tag{6}$$

- A quadrature rule is a weighted mean of:
  - the function g evaluated at the set of nodes $\mathbf{X}_n = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$
  - weighted by a set of weights $\mathbf{w}_n = \{w_1, \ldots, w_n\} \in \mathbb{R}^n$
- A "good" quadrature offers an accurate approximation for a restricted number of nodes.
- Beyond performance, convergence guarantees are essential

◆ **Remarks:**

- Most of the quadrature rules are built without any knowledge on the integrand g
- A strong link exists between numerical integration and space-filling design of experiments [FLQZ18]

Quadrature methods and designs of experiments

A large panel of method (see [Sul15]), among which:

- Univariate deterministic quadratures:
  - Gaussian quadrature
  - Féjer quadrature (i.e., Clenshaw-Curtis)
- Multivariate deterministic quadratures:
  - Tensor product (i.e., regular grids)
  - Smolyak grid (i.e., sparse grids)
- Latin Hypercube Sampling (LHS)
- Monte Carlo sampling
- Quasi-Monte Carlo sampling
- Kernel herding: a kernel-based sampling

Quadrature methods and designs of experiments

A large panel of method (see [Sul15]), among which:

- Univariate deterministic quadratures:
  - Gaussian quadrature
  - Féjer quadrature (i.e., Clenshaw-Curtis)
- Multivariate deterministic quadratures:
  - Tensor product (i.e., regular grids)
  - Smolyak grid (i.e., sparse grids)
- Latin Hypercube Sampling (LHS)
- Monte Carlo sampling
- Quasi-Monte Carlo sampling
- Kernel herding: a kernel-based sampling

Monte Carlo

1. Generate n i.i.d samples $\mathbf{X}_n \sim f_{\mathbf{X}}$ using a pseudo-random generator (e.g., Mersenne-Twister algorithm)

Monte Carlo

1. Generate n i.i.d samples $\mathbf{X}_n \sim f_{\mathbf{X}}$ using a pseudo-random generator (e.g., Mersenne-Twister algorithm)
2. Evaluate their images: $g(\mathbf{X}_n) = \{g(\mathbf{x}^{(1)}), \ldots, g(\mathbf{x}^{(n)})\}$

Monte Carlo

1. Generate n i.i.d samples $\mathbf{X}_n \sim f_{\mathbf{X}}$ using a pseudo-random generator (e.g., Mersenne-Twister algorithm)
2. Evaluate their images: $g(\mathbf{X}_n) = \{g(\mathbf{x}^{(1)}), \ldots, g(\mathbf{x}^{(n)})\}$
3. Monte Carlo estimator:

$$\mathbb{E}[g(\mathbf{X})] \approx \bar{y}_n^{\mathrm{MC}} = \frac{1}{n} \sum_{i=1}^{n} g(\mathbf{x}^{(i)}) \tag{7}$$

Monte Carlo

1. Generate n i.i.d samples $\mathbf{X}_n \sim f_{\mathbf{X}}$ using a pseudo-random generator (e.g., Mersenne-Twister algorithm)
2. Evaluate their images: $g(\mathbf{X}_n) = \{g(\mathbf{x}^{(1)}), \ldots, g(\mathbf{x}^{(n)})\}$
3. Monte Carlo estimator:

$$\mathbb{E}[g(\mathbf{X})] \approx \bar{y}_n^{\mathrm{MC}} = \frac{1}{n} \sum_{i=1}^{n} g(\mathbf{x}^{(i)}) \tag{7}$$

Monte Carlo

1. Generate n i.i.d samples $\mathbf{X}_n \sim f_{\mathbf{X}}$ using a pseudo-random generator (e.g., Mersenne-Twister algorithm)
2. Evaluate their images: $g(\mathbf{X}_n) = \{g(\mathbf{x}^{(1)}), \ldots, g(\mathbf{x}^{(n)})\}$
3. Monte Carlo estimator:

$$\mathbb{E}[g(\mathbf{X})] \approx \bar{y}_n^{\mathrm{MC}} = \frac{1}{n} \sum_{i=1}^{n} g(\mathbf{x}^{(i)}) \tag{7}$$

◆ **Properties:**

- ❑ The weights are all uniform
- ❑ The central limit theorem provides:
  - ➤ the variance of this estimator
  - ➤ confidence intervals of this estimator
- ❑ The convergence rate is in $o(1/\sqrt{n})$ (slow rate)
- ❑ This convergence rate does not depend on the dimension

Monte Carlo: illustration vs. regular grid

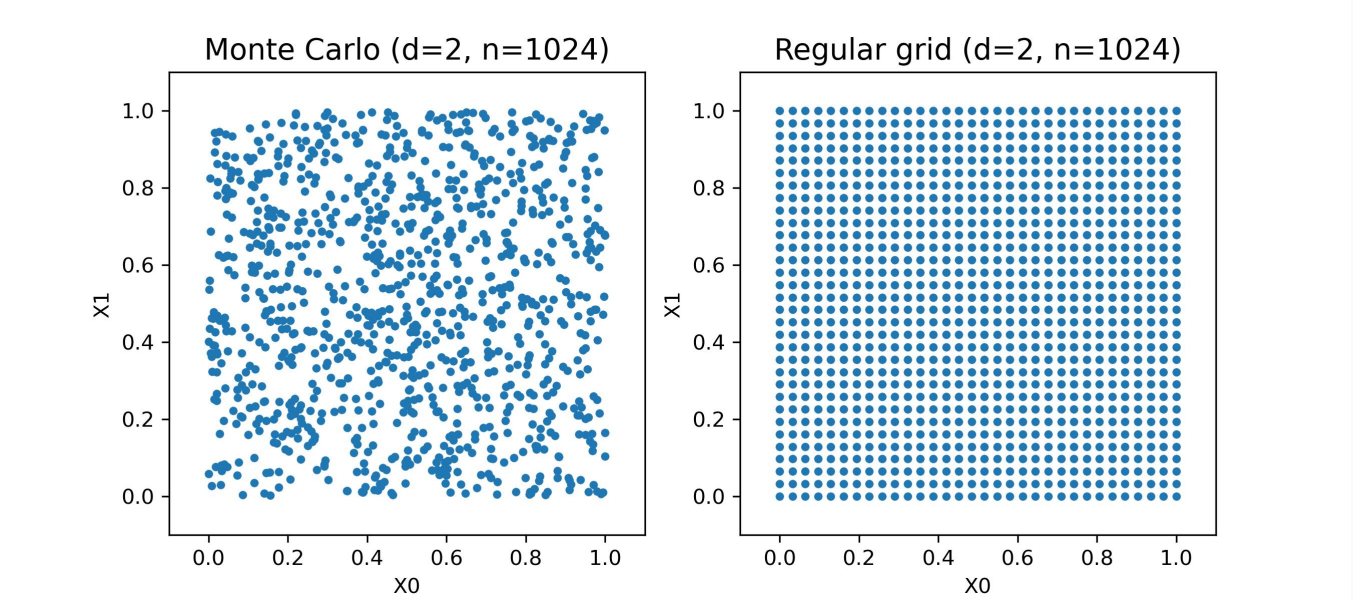


Figure 11: Sub-projection of Monte Carlo sample vs. regular grid in dim. 2 and 5.

Monte Carlo: illustration vs. regular grid

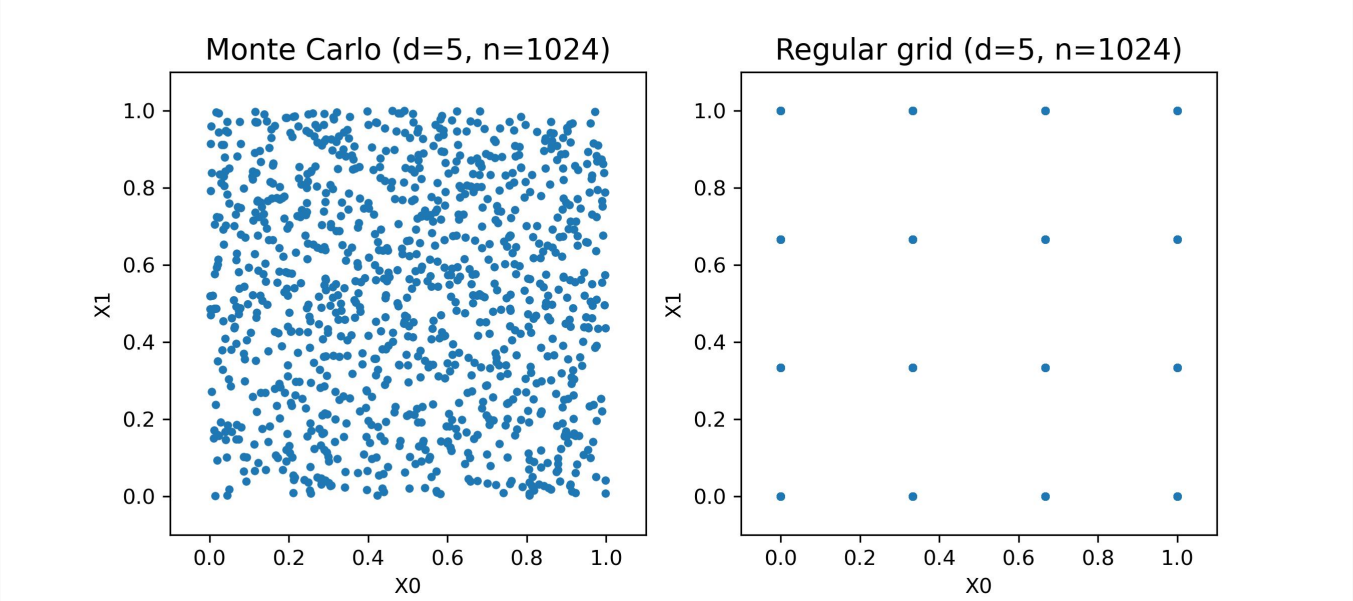


Figure 11: Sub-projection of Monte Carlo sample vs. regular grid in dim. 2 and 5.

Monte Carlo is not subject to the "curse of dimensionality"

Quasi-Monte Carlo: discrepancy and convergence guarantee

◆ Discrepancy

The discrepancy of a set of nodes $\mathbf{X}_n$ is a metric of its uniformity, e.g.:

$$D_p^*(\mathbf{X}_n) = \|\widehat{F}_{\mathbf{X}_n} - F_{\mathbf{U}}\|_p = \left(\int_{[0,1]^d} |\widehat{F}_{\mathbf{X}_n}(\mathbf{x}) - F_{\mathbf{U}}(\mathbf{x})|^p \, \mathrm{d}\mathbf{x} \right)^{1/p} \tag{8}$$

- ❑ The lowest the discrepancy, the "closest" to uniformity
- ❑ Equivalent to Kolmogorov-Smirnov statistic for $p = \infty$
- ❑ Equivalent to Cramér-Von Mises statistic for $p = 2$ (explicit formula)

Quasi-Monte Carlo: discrepancy and convergence guarantee

◆ Discrepancy

The discrepancy of a set of nodes $\mathbf{X}_n$ is a metric of its uniformity, e.g.:

$$D_p^*(\mathbf{X}_n) = \|\widehat{F}_{\mathbf{X}_n} - F_{\mathbf{U}}\|_p = \left(\int_{[0,1]^d} |\widehat{F}_{\mathbf{X}_n}(\mathbf{x}) - F_{\mathbf{U}}(\mathbf{x})|^p \, \mathrm{d}\mathbf{x} \right)^{1/p} \tag{8}$$

- ❑ The lowest the discrepancy, the "closest" to uniformity
- ❑ Equivalent to Kolmogorov-Smirnov statistic for $p = \infty$
- ❑ Equivalent to Cramér-Von Mises statistic for $p = 2$ (explicit formula)

◆ Koksma-Hlawka inequality

If $g : [0,1]^d \to \mathbb{R}$ has a bounded variation, then for any $\mathbf{X}_n \in [0,1]^d$:

$$\left| \int_{[0,1]^d} g(\mathbf{x}) \mathrm{d}\mathbf{x} - \frac{1}{n} \sum_{i=1}^{n} g\left(\mathbf{x}^{(i)}\right) \right| \leq V(g) D_\infty^*(\mathbf{X}_n), \tag{9}$$

- ❑ This bound on the quadrature performance is a composed of:
  - ➤ $V(g)$, quantifying the integrand's complexity
  - ➤ $D_\infty^*(\mathbf{X}_n)$, a uniformity metric of the nodes $\mathbf{X}_n$

Quasi-Monte Carlo: low-discrepancy sequences

For numerous low-discrepancy sequences (Sobol', Halton, Faure, etc.), considering a constant $0 < C < \infty$ and specific sizes $n \in \mathbb{N}$:

$$D_{\infty}^{*}(\mathbf{X}_n) \leq C \frac{\log(n)^d}{n} \tag{10}$$

☞ To go further into quasi-Monte Carlo methods: [LP14]

Quasi-Monte Carlo: low-discrepancy sequences

For numerous low-discrepancy sequences (Sobol', Halton, Faure, etc.), considering a constant $0 < C < \infty$ and specific sizes $n \in \mathbb{N}$:

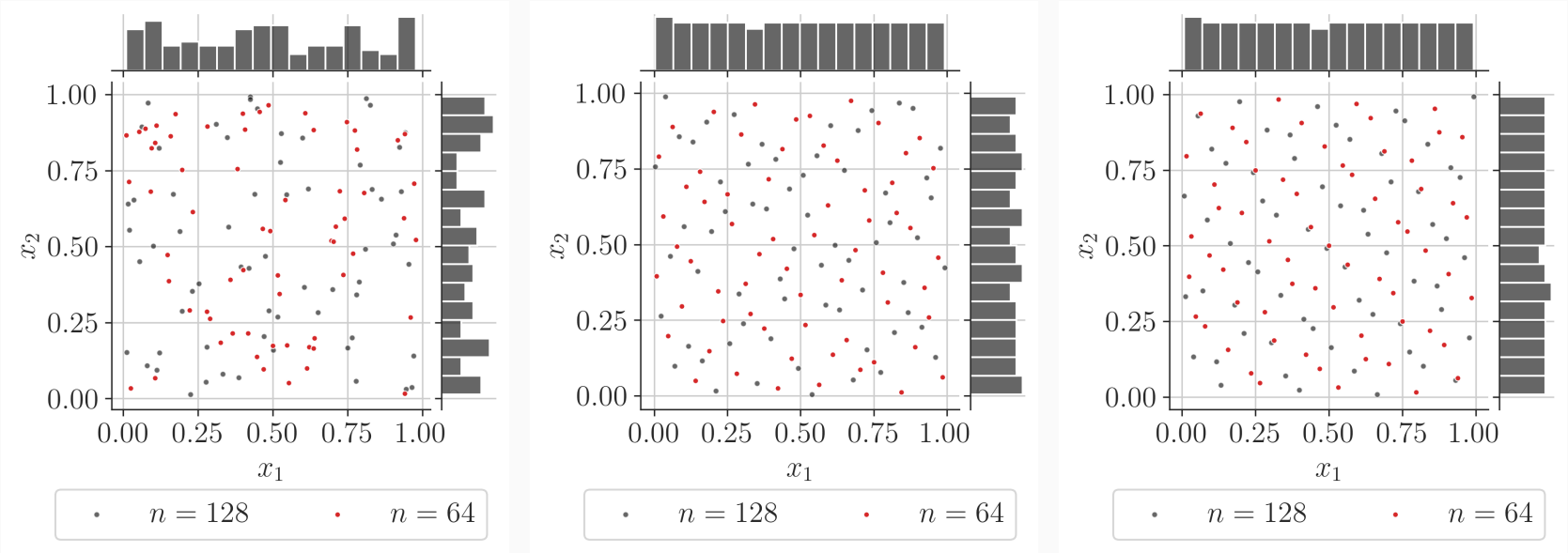
$$D_{\infty}^{*}(\mathbf{X}_n) \leq C \frac{\log(n)^d}{n} \tag{10}$$

☞ To go further into quasi-Monte Carlo methods: [LP14]

◆ Properties and remarks

- ❏ Theses sequences are fully nested and deterministic
- ❏ The samples generated are not independent
- ❏ For non-uniform distribution: use inverse CDF transform
- ❏ Randomized quasi-Monte Carlo introduces a trade-off between exploration and repetitions (allowing to estimate variances [L'E18])

Quasi-Monte Carlo: low-discrepancy sequences

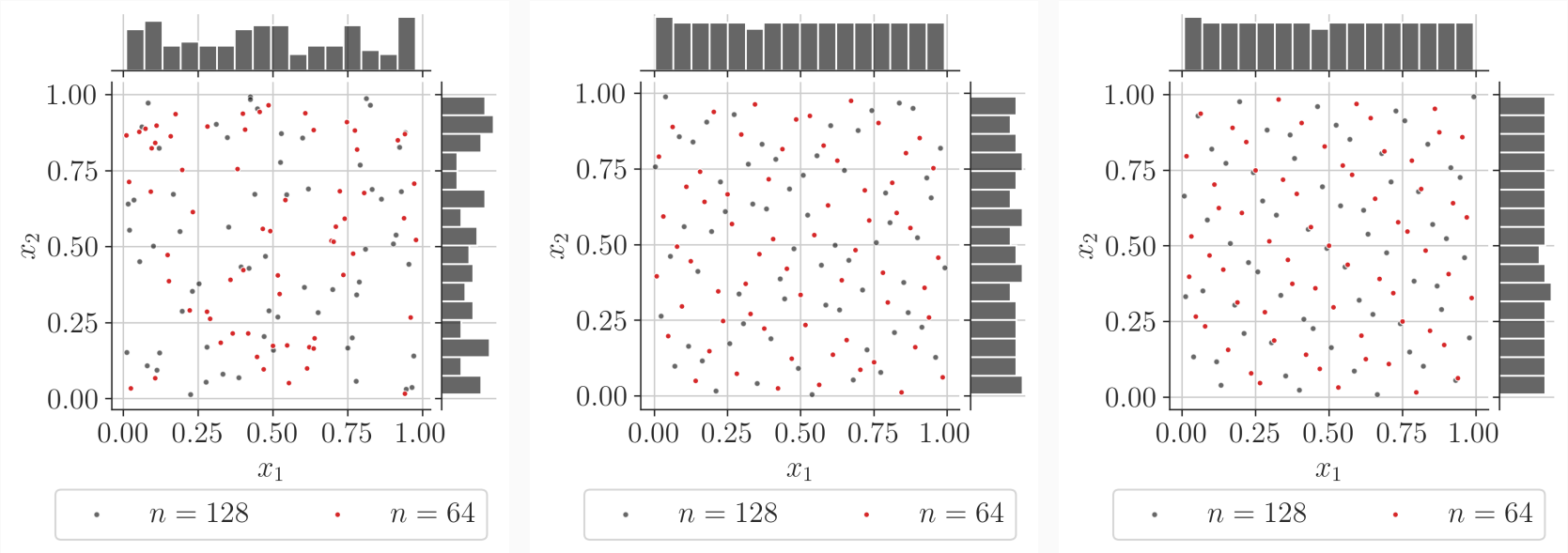


(a) Monte Carlo **(b)** Halton sequence **(c)** Sobol' sequence

Figure 12: Nested Monte Carlo and quasi-Monte Carlo designs ($n = 128$)

[4] *https://openturns.github.io/www/*

Quasi-Monte Carlo: low-discrepancy sequences



(a) Monte Carlo **(b)** Halton sequence **(c)** Sobol' sequence

Figure 12: Nested Monte Carlo and quasi-Monte Carlo designs ($n = 128$)

✳ In OpenTURNS[4]:

- ❏ Sobol', Faure, Halton and Haselgrove low-discrepancy sequences
- ❏ Explicit L2-centered discrepancy

[4] *https://openturns.github.io/www/*

MMD: a generalized discrepancy [SGF+10]

- **Definition:** a discrepancy metric between any distributions μ and ζ

$$\mathrm{MMD}_K(\mu, \zeta) := \sup_{\|g\|_{\mathcal{H}(K)} \leq 1} \left| \int_{\mathcal{D}_{\mathbf{X}}} g(\mathbf{x}) \mathrm{d}\mu(\mathbf{x}) - \int_{\mathcal{D}_{\mathbf{X}}} g(\mathbf{x}) \mathrm{d}\zeta(\mathbf{x}) \right| \tag{11}$$

MMD: a generalized discrepancy [SGF+10]

◆ **Definition:** a discrepancy metric between any distributions μ and ζ

$$\mathrm{MMD}_K(\mu, \zeta) := \sup_{\|g\|_{\mathcal{H}(K)} \leq 1} \left| \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x}) \mathrm{d}\mu(\mathbf{x}) - \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x}) \mathrm{d}\zeta(\mathbf{x}) \right| \tag{11}$$

1. Define a Hilbert space $\mathcal{H}(K)$ (RKHS induced by a kernel $K : \mathcal{D}_\mathbf{X}^2 \mapsto \mathbb{R}_+$)
2. The *maximum mean discrepancy* (MMD) between two probability distributions μ and ζ is given by the worst-case error for any function within $\mathcal{H}(K)$
3. For characteristic kernels, $\mathrm{MMD}_K(\mu, \zeta) = 0 \Leftrightarrow \mu = \zeta$
4. The MMD can be written as a simple difference of potentials $P_\mu(\mathbf{x}) := \int_{\mathcal{D}_\mathbf{X}} K(\mathbf{x}, \mathbf{x}') \mathrm{d}\mu(\mathbf{x}')$ of the distributions $\mathrm{MMD}_K(\mu, \zeta) = \left\| P_\mu - P_\zeta \right\|_{\mathcal{H}(K)}$

MMD: a generalized discrepancy [SGF+10]

◆ **Definition:** a discrepancy metric between any distributions μ and ζ

$$\mathrm{MMD}_K(\mu, \zeta) := \sup_{\|g\|_{\mathcal{H}(K)} \leq 1} \left| \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x}) \mathrm{d}\mu(\mathbf{x}) - \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x}) \mathrm{d}\zeta(\mathbf{x}) \right| \tag{11}$$

1. Define a Hilbert space $\mathcal{H}(K)$ (RKHS induced by a kernel $K : \mathcal{D}_\mathbf{X}^2 \mapsto \mathbb{R}_+$)
2. The *maximum mean discrepancy* (MMD) between two probability distributions μ and ζ is given by the worst-case error for any function within $\mathcal{H}(K)$
3. For characteristic kernels, $\mathrm{MMD}_K(\mu, \zeta) = 0 \Leftrightarrow \mu = \zeta$
4. The MMD can be written as a simple difference of potentials $P_\mu(\mathbf{x}) := \int_{\mathcal{D}_\mathbf{X}} K(\mathbf{x}, \mathbf{x}') \mathrm{d}\mu(\mathbf{x}')$ of the distributions $\mathrm{MMD}_K(\mu, \zeta) = \left\| P_\mu - P_\zeta \right\|_{\mathcal{H}(K)}$

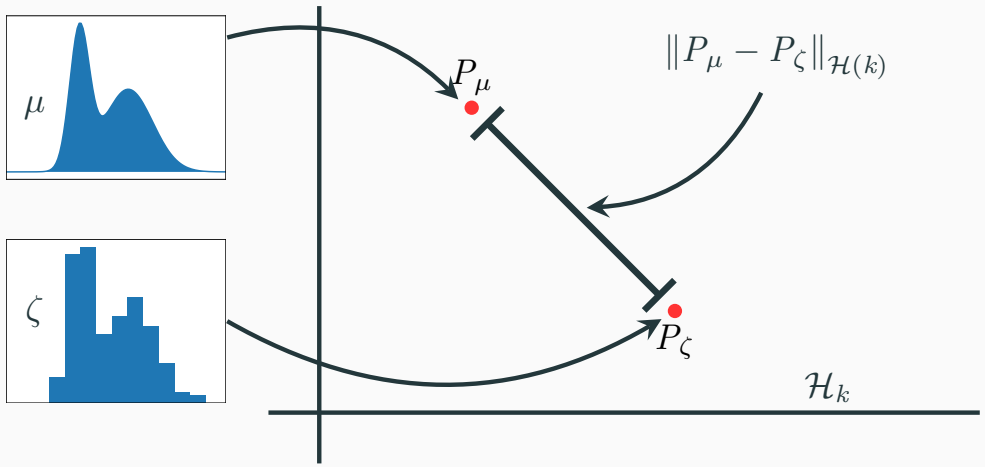


Figure 13: Kernel mean embedding of continuous and discrete distributions

Kernel herding sampling[CWS10]

- **Main idea:** iteratively pick points minimizing a MMD between the:
  - target density μ,
  - current design $\mathbf{X}_n$, represented by the density $\zeta_n = \frac{1}{n}\sum_{i=1}^{n} \delta(\mathbf{x}^{(i)})$
- **Kernel herding criterion:**

$$\mathbf{x}^{(n+1)} \in \underset{\mathbf{x}\in\mathcal{D}_\mathbf{X}}{\arg\min} \left\{ \mathrm{MMD}_K \left(\mu, \frac{1}{n+1}\left(n\zeta_n + \delta(\mathbf{x})\right)\right)^2 \right\} \Rightarrow \underset{\mathbf{x}\in\mathcal{D}_\mathbf{X}}{\arg\min} \left\{ \frac{n}{n+1} P_{\zeta_n}(\mathbf{x}) - P_\mu(\mathbf{x}) \right\} \tag{12}$$

Kernel herding sampling[CWS10]

◆ **Main idea:** iteratively pick points minimizing a MMD between the:

- ❑ target density μ,
- ❑ current design $\mathbf{X}_n$, represented by the density $\zeta_n = \frac{1}{n}\sum_{i=1}^{n} \delta(\mathbf{x}^{(i)})$

◆ **Kernel herding criterion:**

$$\mathbf{x}^{(n+1)} \in \arg\min_{\mathbf{x}\in\mathcal{D}_\mathbf{X}} \left\{ \mathrm{MMD}_K \left(\mu, \frac{1}{n+1}\left(n\zeta_n + \delta(\mathbf{x})\right)\right)^2 \right\} \Rightarrow \arg\min_{\mathbf{x}\in\mathcal{D}_\mathbf{X}} \left\{ \frac{n}{n+1} P_{\zeta_n}(\mathbf{x}) - P_\mu(\mathbf{x}) \right\} \tag{12}$$

◆ **Candidate set $\mathcal{S}$:** is a N-sized sample ($N \gg n$) representative of μ

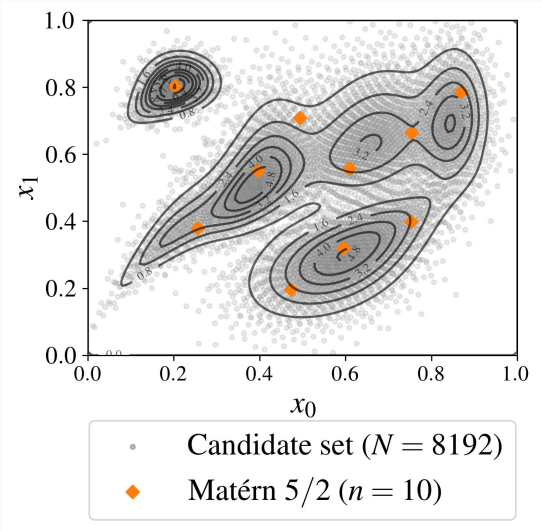
- ❑ use as a discrete domain for a greedy optimization
- ❑ used to estimate the potential $P_\mu(\mathbf{x}), \forall \in \mathcal{S}$

$$\mathbf{x}^{(n+1)} \in \arg\min_{\mathbf{x}\in\mathcal{S}} \left\{ \frac{1}{n+1}\sum_{i=1}^{n} K((\mathbf{x}, \mathbf{x}^{(i)}) - \frac{1}{N}\sum_{j=1}^{N} K(\mathbf{x}, \mathbf{x}^{(j)}) \right\} \tag{13}$$

☞ Theoretical convergence studied in [LJLB15]

Kernel herding: 2D example

- Gaussian mixture with a dependence structure



Kernel herding: 2D example

- Gaussian mixture with a dependence structure

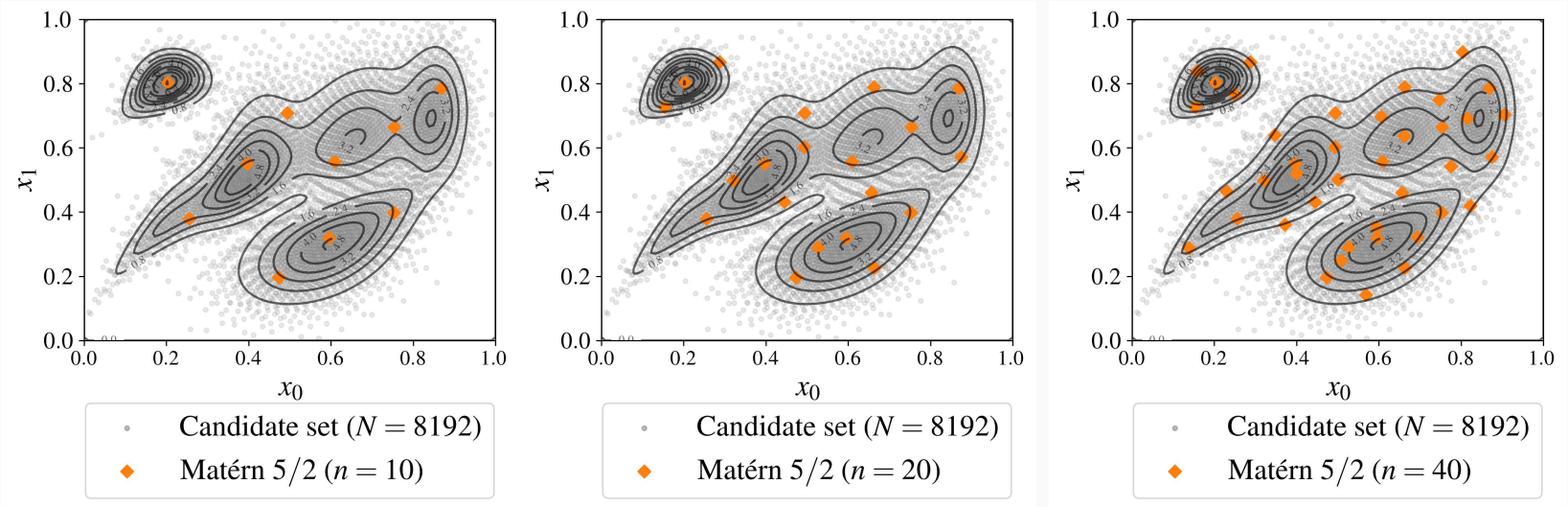


Figure 14: Kernel herding sampling on a Gaussian random mixture

Kernel herding: 2D example

Input random vector: Gaussian mixture $\mathbf{X}$ from Fig.14

Function: $g(\mathbf{x}) = x_1 + x_2$

Quantity of interest: $\mathbb{E}[g(\mathbf{X})] = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x}) f_\mathbf{X}(\mathbf{x}) \, \mathrm{d}\mathbf{x}$

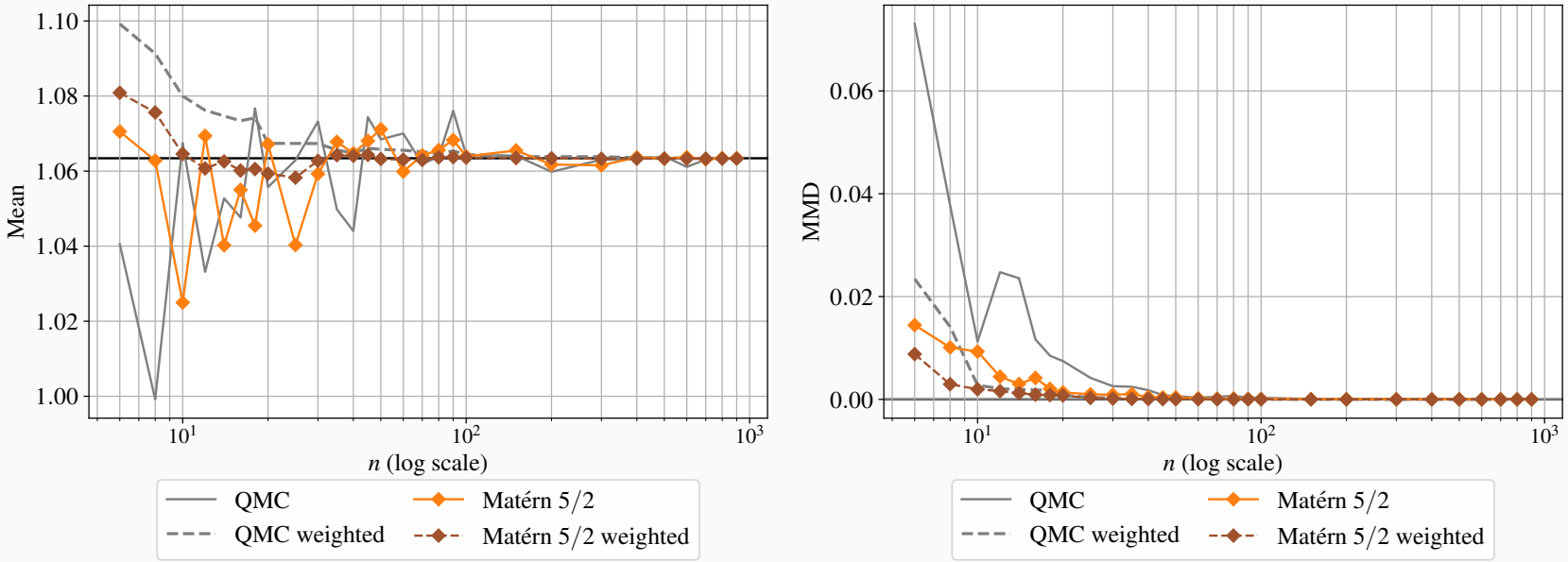


Figure 15: Analytical benchmark results on the toy-case #1

Kernel herding: 2D example

Input random vector: Gaussian mixture $\mathbf{X}$ from Fig.14

Function: $g(\mathbf{x}) = x_1 + x_2$

Quantity of interest: $\mathbb{E}[g(\mathbf{X})] = \int_{\mathcal{D}_\mathbf{X}} g(\mathbf{x}) f_\mathbf{X}(\mathbf{x}) \, \mathrm{d}\mathbf{x}$

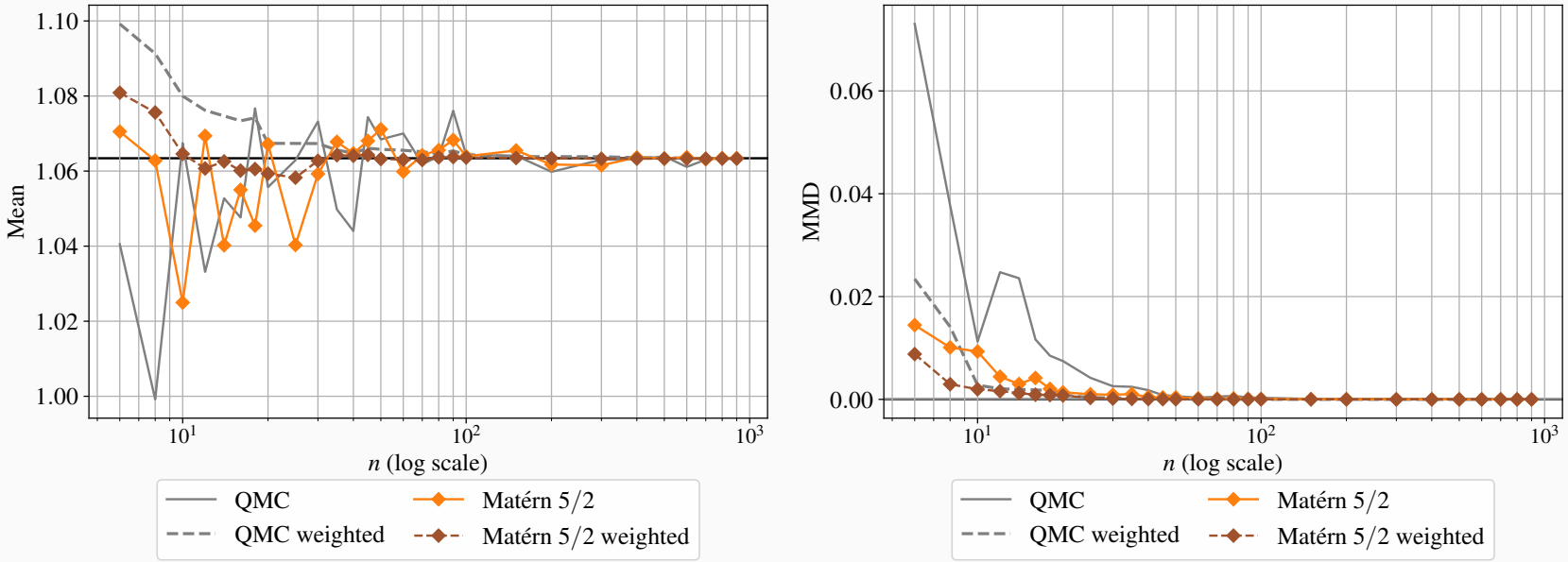


Figure 15: Analytical benchmark results on the toy-case #1

✳ **In OpenTURNS**: New dedicated Python package *otkerneldesign*

```
~$ pip install otkerneldesign
```

Kernel herding: application to probabilistic fatigue of OWT

Kernel herding extracts a representative subset from data [FCMI23]

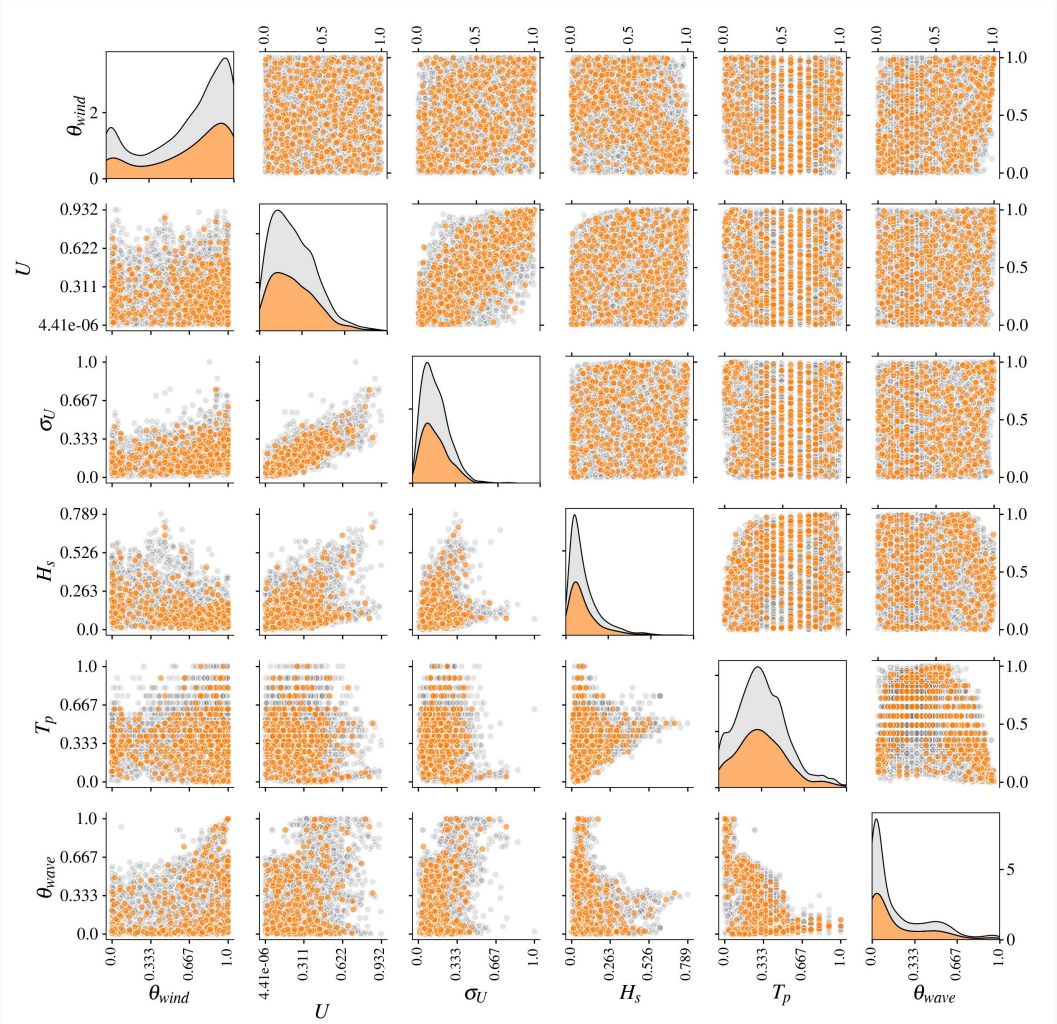


Figure 16: Teesside kernel herding design of experiments

Kernel herding: application to probabilistic fatigue of OWT

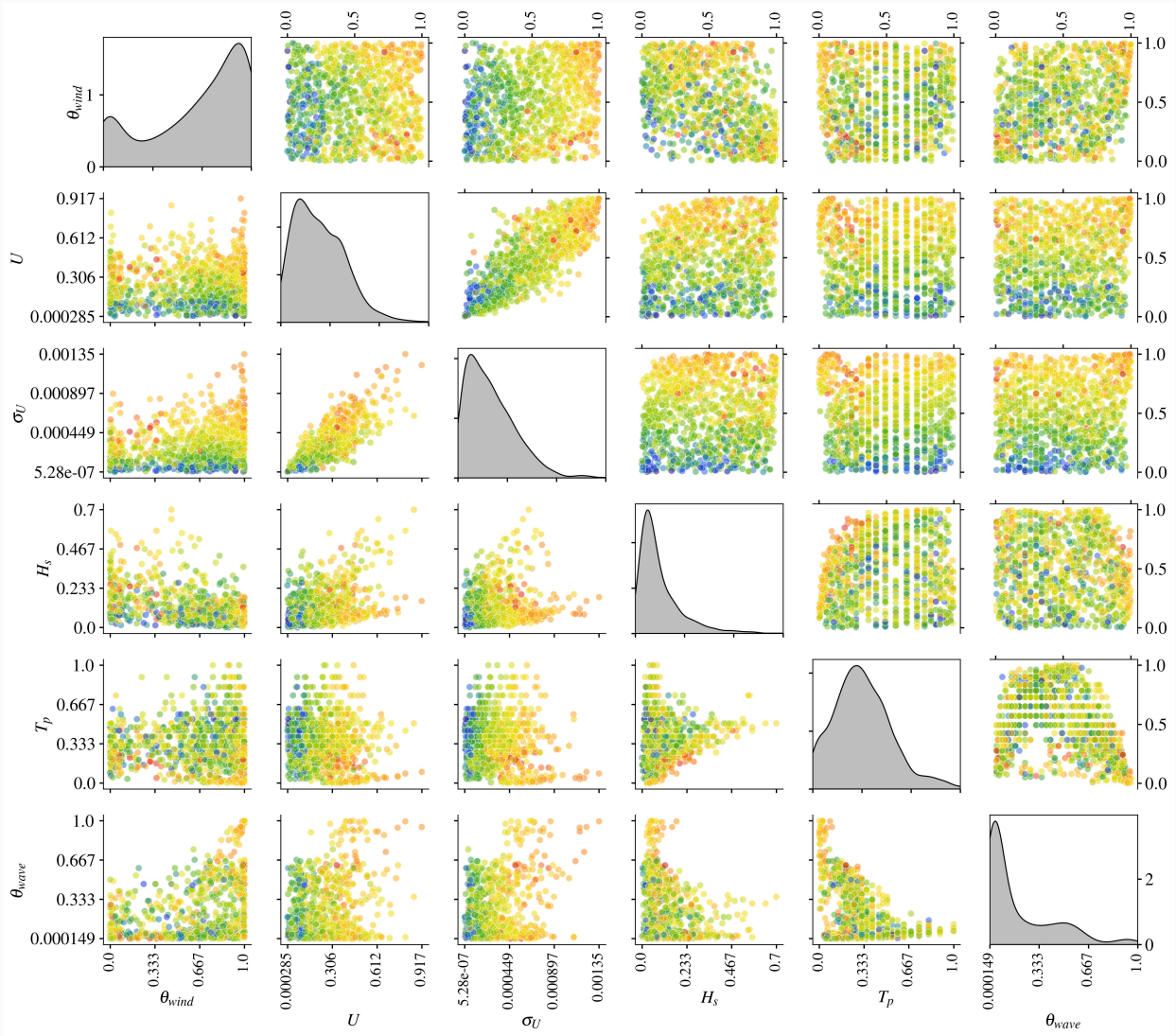


Figure 17: Teesside kernel herding with corresponding outputs (log-scale)

Kernel herding: application to probabilistic fatigue of OWT

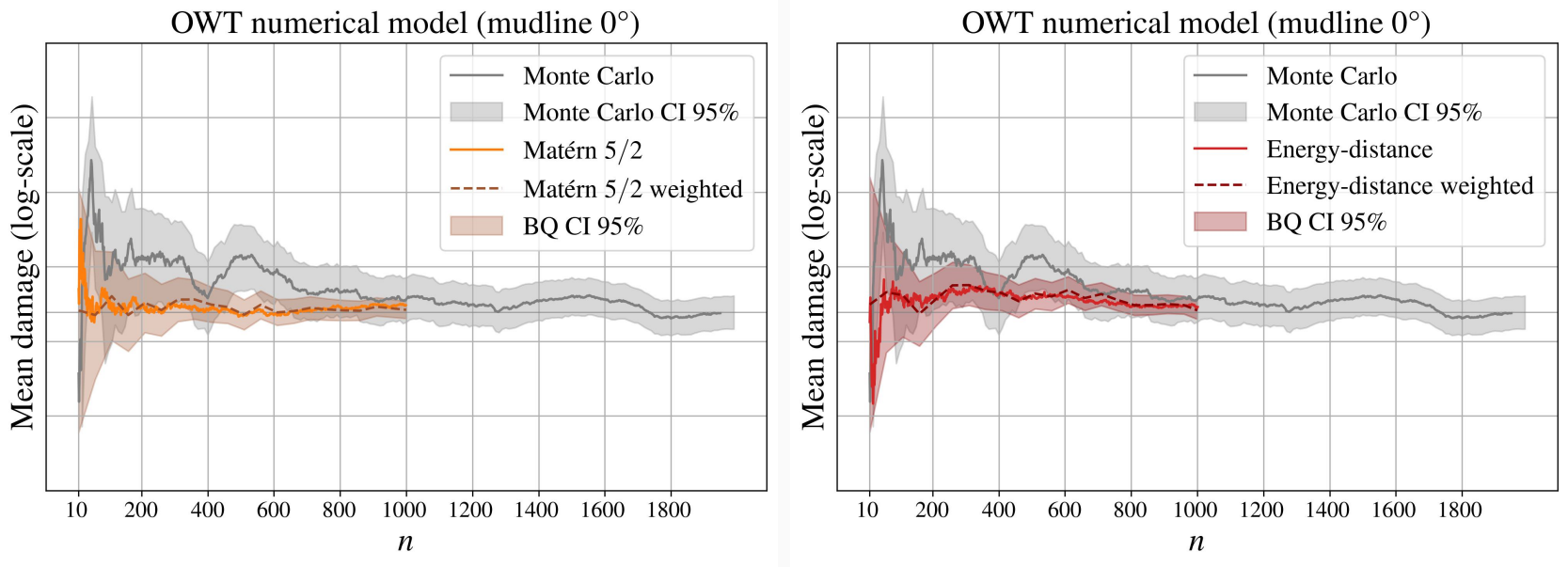


Figure 18: Kernel herding mean convergence (Matérn and energy-distance kernels) (size $n = 500$)

◆ **Remark:** The "BQ IC" is actually a prediction interval

★★ Take-home message #3 ★★

✌ To keep in mind ✌

- **Conclusions**
  - ➤ Kernel herding for quadrature offers a given-data uncertainty propagation method (i.e., without fitting a probabilistic model)
  - ➤ This method can fully exploit parallel computing (i.e., not active)
  - ➤ Possible to compute optimal Bayesian quadrature weights

★★ Take-home message #3 ★★

✌ To keep in mind ✌

- **Conclusions**
  - ➤ Kernel herding for quadrature offers a given-data uncertainty propagation method (i.e., without fitting a probabilistic model)
  - ➤ This method can fully exploit parallel computing (i.e., not active)
  - ➤ Possible to compute optimal Bayesian quadrature weights
- **Limits**
  - ➤ KH is sensitive to the chosen kernel (heuristic tuning in [FIM+23])
  - ➤ In high dimension, $\mathcal{S}$ must be larger, KH becomes costly

4. Kernel-based sensitivity analysis

Challenges in GSA

- **A few issues / challenges in standard GSA practice**
  - Output variance might not be really insightful!
  - High computational cost (e.g., 1 call $\approx$ from a few hours to a day)
  - Large number of inputs (e.g., ≈ 100 inputs)
  - Quantity of interest (QoI) possibly difficult to estimate accurately (e.g., high-order quantile, rare event probability)
  - Input and output variables may not be continuous (e.g., discrete / categorical variables, graphs, distributions)
  - (Input dependency) ➪ Another course!

GSA using the Hilbert-Schmidt Independence Criterion

◆ A few preliminary hypotheses

❑ Probabilistic modeling of d **independent** input physical variables:

$$\mathbf{X} = (X_1, X_2, \ldots, X_d)^\top \sim \mathbb{P}_{\mathbf{X}} \quad \text{over} \quad \mathcal{X} = \bigtimes_{i=1}^{d} \mathcal{X}_i \tag{14}$$

❑ Black-box model (scalar output):

$$\mathcal{M} : \left| \begin{array}{rcl} \mathcal{X} \subseteq \mathbb{R}^d & \longrightarrow & \mathcal{Y} \subseteq \mathbb{R} \\ \mathbf{X} & \longmapsto & Y = \mathcal{M}(\mathbf{X}) \end{array} \right. \tag{15}$$

❑ **Learning sample** ➪ a n-size i.i.d. sample of the couple $(\mathbf{X}, Y)$:

$$\left(\mathbf{X}^{(j)}, Y^{(j)}\right)_{(1 \leq j \leq n)} = \left(X_1^{(j)}, X_2^{(j)}, \ldots, X_d^{(j)}, Y^{(j)}\right)_{(1 \leq j \leq n)} \tag{16}$$

with $\mathbb{P}_{\mathbf{X}^{(j)}} = \mathbb{P}_{\mathbf{X}}$
and $Y^{(j)} = \mathcal{M}\left(X_1^{(j)}, X_2^{(j)}, \ldots, X_d^{(j)}\right), \forall j \in \{1, \ldots, n\}$

GSA using the Hilbert-Schmidt Independence Criterion

◆ Dependence measures for GSA

- ❑ A first idea:
  - ➤ If X_i has **no influence** on $Y \Leftrightarrow X_i$ and Y are **independent**
- ❑ Two possibilities:
  - ➤ By comparing $\mathbb{P}_Y$ and $\mathbb{P}_{Y|X_i} \Rightarrow$ **Dissimilarity measures**
  - ➤ By comparing $\mathbb{P}_{(Y,X_i)}$ and $\mathbb{P}_Y \otimes \mathbb{P}_{X_i} \Rightarrow$ **Dependence measures**
- ❑ Use of a specific dependence measure:
  - ➤ The **Hilbert-Schmidt Independence Criterion** (**HSIC**) proposed in the ML community in ~ 2005 ☞ [GHS+05]
  - ➤ Brought to the GSA community in ~ 2015 ☞ [DV15]

GSA using the Hilbert-Schmidt Independence Criterion

◆ Dependence measures for GSA (cont'd)

❑ We would like to quantify the sensitivity between X_i and Y using a dependence measure, such that:

$$S_i = d(\mathbb{P}_{(Y,X_i)}, \mathbb{P}_Y \otimes \mathbb{P}_{X_i}) \tag{17}$$

where:

- ➤ $\mathbb{P}_{(Y,X_i)}$ is the joint probability distribution
- ➤ $\mathbb{P}_Y \otimes \mathbb{P}_{X_i}$ is the product of marginals distributions

❑ Two desirable properties we are looking for:

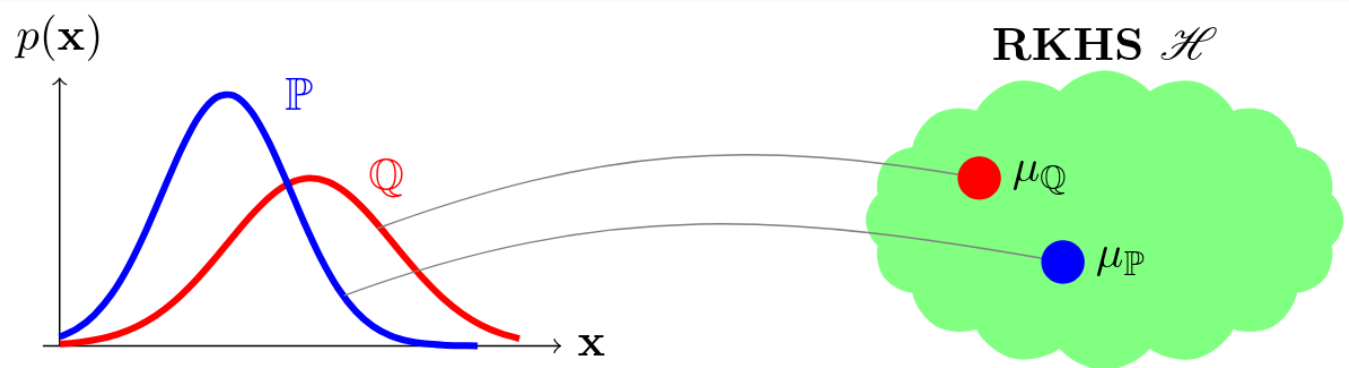
- ➤ large values of S_i imply strong dependence between X_i and Y
- ➤ $S_i = 0$ only if X_i and Y are independent

↪ How to define such a dependence measure between probability distributions?

GSA using the Hilbert-Schmidt Independence Criterion

◆ **(Recall) How to compare probability distributions?**

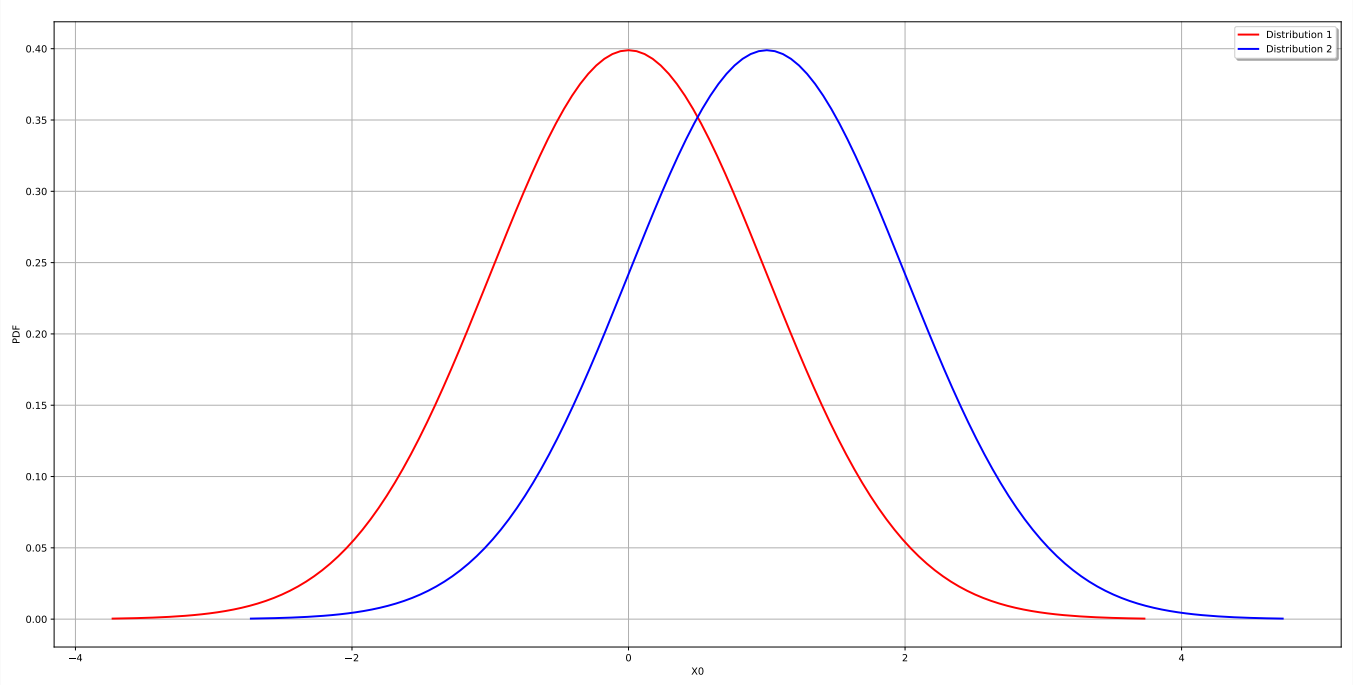
- ❑ By comparing various **features** of the distributions
- ❑ In practice, we map the distributions onto auxiliary spaces characterizing specific features
  - ➤ Feature functions $\psi(\mathbb{P}_{X_i})$, e.g., mean of the distribution $\psi(\mathbb{P}_{X_i}) = \mathbb{E}[X_i]$
- ❑ We can then compute the dependence as a function of the distance between distributions as a distance between their mappings



GSA using the Hilbert-Schmidt Independence Criterion

◆ A first example

Let us consider the following mapping: $\psi(\mathbb{P}_{X_i}) = \mathbb{E}[X_i]$

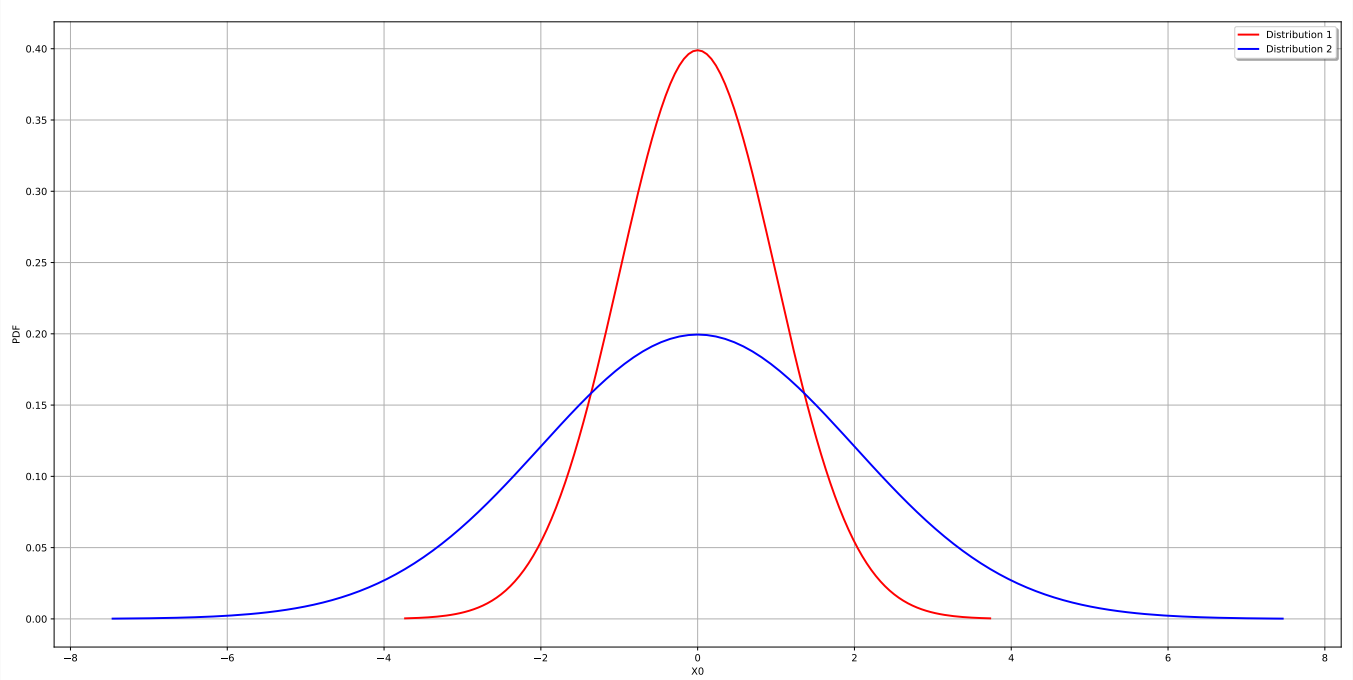


↪ Here, this approach would work ➪ we can distinguish the two distributions ✔

GSA using the Hilbert-Schmidt Independence Criterion

◆ A first example (cont'd)

Let us consider the following mapping: $\psi(\mathbb{P}_{X_i}) = \mathbb{E}[X_i]$

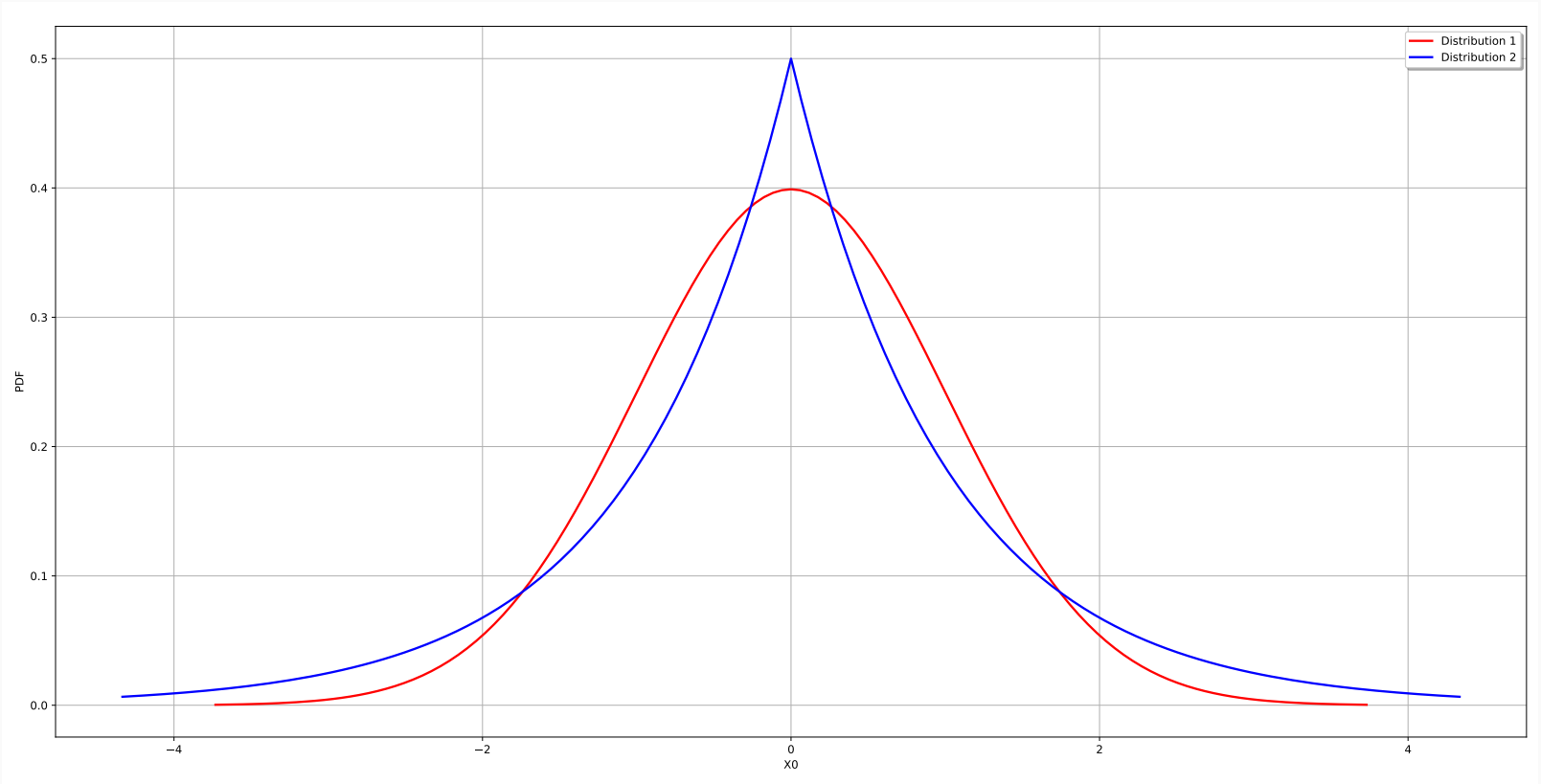


↪ Here, this approach would not work ➪ we cannot distinguish the two distributions ✘

GSA using the Hilbert-Schmidt Independence Criterion

◆ A first example (cont'd)

Let us consider the following mapping: $\psi(\mathbb{P}_{X_i}) = (\mathbb{E}[X_i], \mathrm{Var}\,(X_i))$



↪ Also in this case, we are not able to distinguish between two different distributions ✘

GSA using the Hilbert-Schmidt Independence Criterion

Limitations

For any finite number of features, there will always be, at least, 2 different distributions which appear identical with respect to the considered features!

➪ **Can we look at an infinite collection of features?**

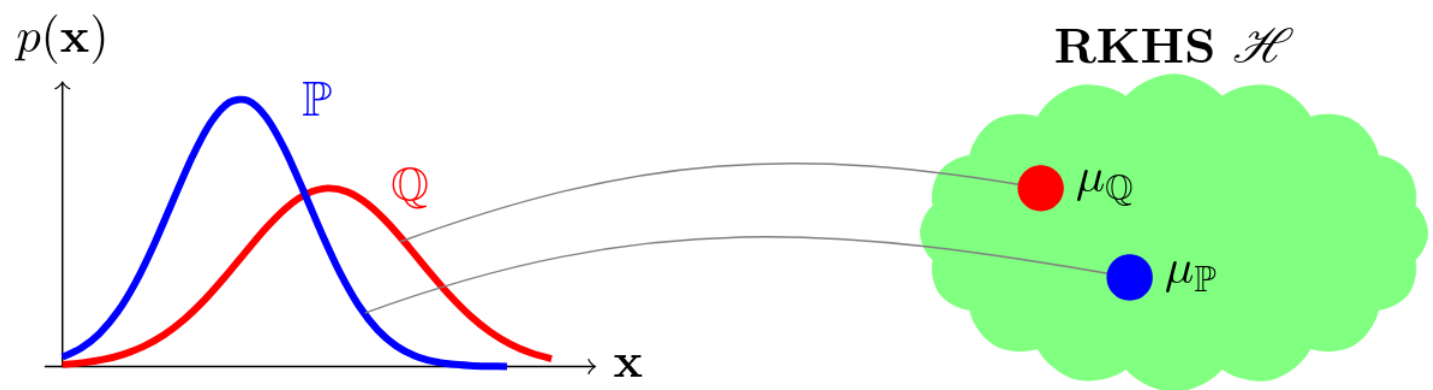
↪ **Yes, using kernel mean embedding!**

GSA using the Hilbert-Schmidt Independence Criterion

◆ Theoretical formulation

- ❑ Let us now consider $\mathcal{V}$, an RKHS over $\mathcal{X} \times \mathcal{Y}$ with kernel $V(\cdot, \cdot)$
- ❑ We consider the mean embedding of $\mathbb{P}_Y \mathbb{P}_{X_i}$ and $\mathbb{P}_{(Y, X_i)}$ in $\mathcal{V}$

$$\Delta := ||\mu_{\mathbb{P}_{(Y,X_i)}} - \mu_{\mathbb{P}_Y \mathbb{P}_{X_i}}|| \tag{18}$$



Characteristic kernels ⇒ Injective feature map

GSA using the Hilbert-Schmidt Independence Criterion

◆ **Theoretical formulation (cont'd)**

❑ Assuming $V((Y, X_i), (Y', X_i')) = K(Y, Y')K_i(X_i, X_i')$, we can show that:

$$\mathrm{HSIC}(X_i, Y)_{\mathcal{F}_i,\mathcal{G}} = \Delta^2 \tag{19}$$

$$\mathrm{HSIC}(X_i, Y)_{\mathcal{F}_i,\mathcal{G}} = \mathbb{E}\left[K_i(X_i, X_i')K(Y, Y')\right] + \mathbb{E}\left[K_i(X_i, X_i')\right]\mathbb{E}\left[K(Y, Y')\right] - 2\mathbb{E}\left[\mathbb{E}[K_i(X_i, X_i')|X_i]\,\mathbb{E}[K(Y, Y')|Y]\right]$$

where (X_i', Y') is an i.i.d. copy of (X_i, Y)

GSA using the Hilbert-Schmidt Independence Criterion

◆ Estimators

❑ A first estimator based on V-statistics ☞ [GHS+05]

$$\widehat{\mathrm{HSIC}}(X_i, Y) = \frac{1}{n^2}\mathrm{Tr}(L_i H L H) \tag{20}$$

where L_i and L are Gram matrices, and H a shift matrix

❑ A second estimator based on the distance correlation estimator ☞ [SRB07]

$$\widehat{\mathrm{HSIC}}(X_i, Y) = \frac{1}{n^2}\sum_{j_1, j_2}^{n} A_{j_1 j_2} B_{j_1 j_2} \tag{21}$$

where $A_{j_1 j_2}$ and $B_{j_1 j_2}$ are linear combinations of Gram matrices

★★ Take-home message #4 ★★

✌ To keep in mind ✌

- **A fundamental property** ➪ if **characteristic** kernels

$$\mathrm{HSIC}(X_i, Y)_{\mathcal{F}_i,\mathcal{G}} = 0 \Leftrightarrow X_i \perp Y \tag{22}$$

- **A plug-in estimator of a normalized sensitivity index** ☞ [DV15]

$$\widehat{R^2_{\mathrm{HSIC},i}} = \frac{\widehat{\mathrm{HSIC}}(X_i, Y)}{\sqrt{\widehat{\mathrm{HSIC}}(X_i, X_i)\,\widehat{\mathrm{HSIC}}(Y, Y)}} \tag{23}$$

- **Choose a characteristic kernel** $\rightarrow$ Type + Hyperparameters' estimation

$$K(z, z') = \exp\left(-\theta \|z - z'\|_2^2\right) \quad \text{(Gaussian kernel)} \tag{24}$$

with $\widehat{\theta} = 1/\widehat{\sigma^2}$ ($\widehat{\sigma^2}$ the empirical variance of the sample z)

Screening and ranking using HSIC

◆ **Screening with HSIC for high-dimensional problems**

❑ A definition of "screening":

➤ Separating the inputs between two distinct groups:
the **significantly influential** ones vs. the **non-influential** ones

❑ In the HSIC framework, one possible solution:

➤ To use some **statistical hypothesis tests** to determine whether the variables X_i and Y are independent or not ☞ [DLM16]

Screening and ranking using HSIC

◆ Statistical hypothesis testing with HSIC

❑ Due to the fundamental property of HSIC (cf. Eq. (9)):

➤ One can construct the following statistical test $\mathcal{T}$

$$\mathcal{T} : \text{Test } (H_{0,i}) : \text{HSIC}(X_i,\, Y) = 0 \quad \text{vs.} \quad (H_{1,i}) : \text{HSIC}(X_i,\, Y) > 0 \tag{25}$$

$H_{0,i}$: X_i and Y are independent

➤ And choose $\widehat{S}_{\mathcal{T}} := n \times \widehat{\text{HSIC}}(X_i,\, Y)$ as the test statistic

❑ To decide, one can use the **p-value** associated to the test $\mathcal{T}$:

➤ It is the probability, under $(\mathcal{H}_{0,i})$, that $\widehat{S}_{\mathcal{T}}$ becomes greater or equal to the observed $\widehat{S}_{\mathcal{T},\text{obs}}$ statistic on the learning sample

$$p_{\text{val}} = \mathbb{P}\left(\widehat{S}_{\mathcal{T}} \geq \widehat{S}_{\mathcal{T},\text{obs}} \mid \mathcal{H}_{0,i}\right) \tag{26}$$

Screening and ranking using HSIC

◆ Statistical hypothesis testing with HSIC

$$p_{\text{val}} = \mathbb{P}\left(\widehat{S}_{\mathcal{T}} \geq \widehat{S}_{\mathcal{T},\text{obs}} \mid H_{0,i}\right) \tag{27}$$

In other words : What is the probability of obtaining a HSIC(X_i, Y) value as large as the observed one under the assumption that Y and X_i are independent?

- ❑ To discriminate, one uses the **significance level** α (a.k.a. Type I error):
  - ➤ Typically, one can fix this level: $\alpha = 0.05$ (or $\alpha = 0.10$)
  - ➤ **Decision rule**: if $p_{\text{val}} < \alpha \Rightarrow H_{0,i}$ (i.e., independence) is rejected

  The probability of obtaining the observed HSIC(X_i, Y) value is low
  $\rightarrow$ The independence hypothesis is most likely wrong
  $\rightarrow$ X_i **has a non negligible influence on** Y

Screening and ranking using HSIC

◆ Asymptotic test for large sample size

In order to estimate how unlikely a value of $n\widehat{\mathrm{HSIC}}(X_i, Y)$ is, we can use its probability distribution

- ❑ Asymptotic results about the HSIC estimators:
  - ➤ The $n\widehat{\mathrm{HSIC}}(X_i, Y)$ estimator can be reasonably approached by a **Gamma distribution** parametrized by (γ, β) ☞ [GFT+08]
- ❑ **Asymptotic test $\mathcal{T}_{\mathrm{asymp}}$:**
  - ➤ Therefore, the p-value can be approximated as follows:

$$p_{\mathrm{val}} \approx 1 - F_{\mathrm{Ga}}\left(n \times \widehat{\mathrm{HSIC}}(X_i, Y)_{\mathrm{obs}}\right) \tag{28}$$

where $F_{\mathrm{Ga}}(\cdot)$ is the CDF of the Gamma distribution and $\widehat{\mathrm{HSIC}}(X_i, Y)_{\mathrm{obs}}$ is the observed value of the random variable $\widehat{\mathrm{HSIC}}(X_i, Y)$

Screening and ranking using HSIC

- Asymptotic test for large sample size

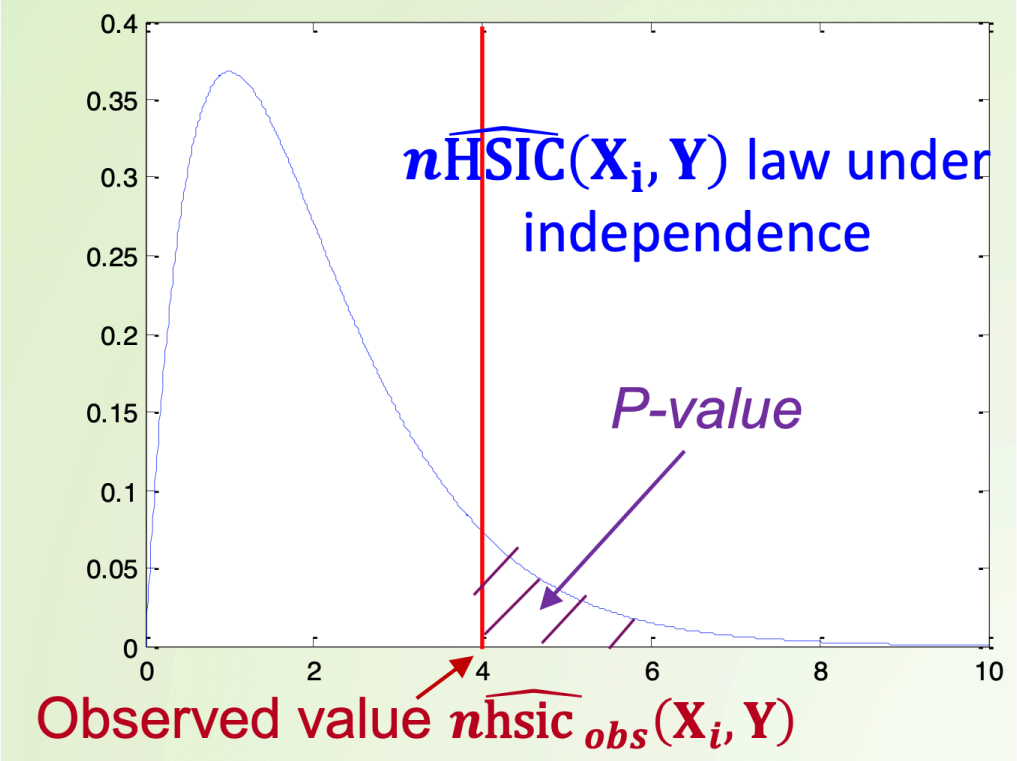


Figure 19: Asymptotic law of the HSIC estimator (source: [Mar21]).

Screening and ranking using HSIC

◆ **Non-asymptotic test for small sample size**

❑ **Permutation-based test $\mathcal{T}_{\text{perturb}}$** ☞ [DLM16]

- ➤ $Z_n := \left(X^{(j)}, Y^{(j)}\right)_{(1 \leq j \leq n)}$ an initial n-sample and $\widehat{\text{HSIC}}(Z_n)$
- ➤ $\{\tau_1, \ldots, \tau_B\}$, a set of B independent random permutations of $\{1, \ldots, n\}$
- ➤ We consider $\widehat{H}^{*b} := \widehat{\text{HSIC}}\left(X^{(j)}, Y^{(\tau_b(j))}\right)_{(1 \leq j \leq n)}$
- ➤ And the order statistic:

$$\widehat{H}^{*(1)} \leq \widehat{H}^{*(2)} \leq \cdots \leq \widehat{H}^{*(B)} \tag{29}$$

- ➤ The p-value (under $\mathcal{H}_{0,i}$) can be approximated as follows:

$$p_{\text{val}} \approx \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}_{\widehat{H}^{*b} > \widehat{\text{HSIC}}(X,Y)} \tag{30}$$

If X_i and Y are independent, considering a permutation of Y should have no impact

Screening and ranking using HSIC

- **A heuristic procedure for ranking the variables**
  - Ranking the input variables w.r.t. their influence using the p-values:
    - **The lower $p_{\mathbf{val}}$ is, the stronger $(\mathcal{H}_{0,i})$ is rejected**
      $\Rightarrow$ the higher the influence of X_i
  - Rule for ranking:
    - **Increasing order of influence by decreasing order of p-values**
    - **If p-values are equal**
      $\rightarrow$ variables are ordered by increasing order of $\widehat{R^2_{\mathrm{HSIC},i}}$

★★ Take-home message #5 ★★

✌ To keep in mind ✌

- **Screening**
  - If n is large ($n > n_{\text{th}}$), $\rightarrow$ **asymptotic test**
  - If not ($n \leq n_{\text{th}}$), $\rightarrow$ **permutation-based test**
- "**Tuning parameters**"

  $n_{\text{th}} = 10^3$, $B = 200$, $\alpha = 0.05$
- **Ranking**
  - $\nearrow$ influence, $\searrow$ p-values
  - If $p_{\text{val},i} = p_{\text{val},j}$, check $\widehat{R^2_{\text{HSIC},i}}$ and $\widehat{R^2_{\text{HSIC},j}}$
- Details in this paper ☞ [DLM16]

Target SA using HSIC

◆ Target Sensitivity Analysis (TSA)

❑ Basic underlying idea of TSA:

➤ To measure the influence of the inputs on a **critical domain** of the model output, and in particular, over the **occurrence** of the critical phenomenon ☞ [RM18, MC21]

❑ **Target $\rightarrow$ critical domain/event on the model output**:

➤ $\mathcal{C} \subset \mathcal{Y}$ a critical domain such that $\mathbb{P}(Y \in \mathcal{C})$ is small

➤ Critical measure of $Y \rightarrow \mathcal{R}(Y)$ (risk measure)

➤ High-order quantile $\rightarrow \mathcal{C} := \{y \in \mathcal{Y} \mid y > \mathcal{R}(y)\}$ and $\mathcal{R}(Y) = q_{0.9}(Y)$

❑ TSA with HSIC:

➤ **Indicator function** $\mathbb{1}_{\mathcal{C}} : \mathcal{Y} \rightarrow \{0, 1\}, y \mapsto 1$ if $y \in \mathcal{C}$, 0 otherwise ☞ [DV15]

$\Rightarrow$ Use Dirac kernel

➤ **Smooth weight function** $w_{\mathcal{C}} : \mathcal{Y} \rightarrow [0, 1], y \mapsto \exp\left(-d_{\mathcal{C}}(y)/s\right)$

$\Rightarrow$ Use Gaussian kernel

★★ Take-home message #6 ★★

✌ To keep in mind ✌

❑ **Focusing on the critical domain**

- ➤ Either an indicator function or a weith function on Y
- ➤ Typical weight function "around" the binary threshold

$$w_{\mathcal{C}} : \left| \begin{array}{ccc} \mathcal{Y} & \longrightarrow & [0,1] \\ y & \longmapsto & w_{\mathcal{C}}(y) = \exp\left(-\frac{\max(c-y,\,0)}{s\,\sigma_Y}\right) \end{array} \right. \tag{31}$$

with $s = \frac{1}{5}$ a smoothness parameter and σ_Y the empirical estimation of the std of Y

❑ **A Target-HSIC index under** $w_{\mathcal{C}}(\cdot)$

$$\mathrm{T-HSIC}_{w_{\mathcal{C}}}(X_i,\, Y) = \mathrm{HSIC}(X_i,\, w_{\mathcal{C}}(\,Y)) \tag{32}$$

❑ More details in this paper ☞ [MC21]

Conditional SA using HSIC

◆ Conditional Sensitivity Analysis (CSA)

❑ Basic underlying idea of CSA:

➤ To measure the influence of the inputs within a **critical domain** of the model output, ignoring what happens outside ☞ [MC21]

❑ **Conditional $\rightarrow$ conditional to a critical event**:

➤ Consider $P_{Y|\, Y\in\mathcal{C}}$ with $\{\, Y \in \mathcal{C}\}$ characterized either by $\not\Vdash_{\mathcal{C}}$ (hard thresholding) or by $w_{\mathcal{C}}$ (smooth)

➤ Implicitly, $P_{\mathbf{X}|\, Y\in\mathcal{C}}$ is no more equal to the product of marginals

★★ Take-home message #7 ★★

✌ To keep in mind ✌

- **Characterize the critical domain**
  - Either with a hard conditioning (indicator function) or a smooth one
  - Introduction of a weight matrix W
- **A Conditional-HSIC index**

$$\mathrm{C}-\widehat{\mathrm{HSIC}}_{w_C}(X_i,\, Y) = \frac{1}{n^2}\mathrm{Tr}(\, WL_i\, WH_1\, LH_2) \tag{33}$$

- More details in this paper ☞ [MC21]

In practice: the proposed methodology

◆ The cooking recipe

1. Define the goal of the study:
   - ➤ Screening vs. Ranking
   - ➤ Type of analysis: Global / Target / Conditional
   - ➤ Maximum allowable budget (sample size)
2. Generate input samples such that $\mathbf{X} \sim P_{\mathbf{X}}$ according to $P_{\mathbf{X}}$ and get corresponding output samples (e.g., by crude Monte Carlo sampling)
3. Perform the analysis:
   - ➤ Choose kernels for inputs and output
   - ➤ Choose an estimator (U-stat vs. V-stat)
   - ➤ If necessary (TSA / CSA): define a weight function
   - ➤ Choose a test strategy (asymptotic vs. permutation-based)
   - ➤ Get HSIC & R^2_{HSIC} indices and p-values
4. Aggregate results from several analyses (e.g., GSA and TSA)
   - ➤ Use p-values first
   - ➤ Then, use R^2_{HSIC} indices

Thank your for your attention!

Any question?

References i

N. Aronszajn.

Theory of reproducing kernels.

Transactions of the American Mathematical Society, 68(3):337–404, 1950.

C. M. Bishop.

Pattern Recognition and Machine Learning.

Information Science and Statistics. Springer Science + Business Media, 2006.

F.X. Briol, C. Oates, M. Girolami, M. Osborne, and D. Sejdinovic.

Probabilistic Integration: A Role in Statistical Computation?

Statistical Science, 34:1 – 22, 2019.

J.-M. Bourinet.

Reliability analysis and optimal design under uncertainty – Focus on adaptive surrogate-based approaches.

HDR (French Accreditation to Supervise Research), Université Clermont Auvergne, 2018.

245 pages.

A. Berlinet and C. Thomas-Agnan.

Reproducing Kernel Hilbert Spaces in Probability and Statistics.

Springer New York, NY, 2004.

Y. Chen, M. Welling, and A. Smola.

Super-samples from kernel herding.

In Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, pages 109 – 116. AUAI Press, 2010.

S. Da Veiga, F. Gamboa, B. Iooss, and C. Prieur.

Basics and Trends in Sensitivity Analysis. Theory and Practice in R.

Society for Industrial and Applied Mathematics, Philadelphia, PA, 2021.

References ii

A. Der Kiureghian.

Structural and System Reliability.

Cambridge University Press, 2022.

M. De Lozzo and A. Marrel.

New improvements in the use of dependence measures for sensitivity analysis and screening.

Journal of Statistical Computation and Simulation, 86(15):3038–3058, 2016.

E. De Rocquigny.

Modelling Under Risk and Uncertainty: An Introduction to Statistical, Phenomenological and Computational Methods.

Wiley series in Probability and Statistics. Wiley, 2012.

E. De Rocquigny, N. Devictor, and S. Tarantola.

Uncertainty in industrial practice: a guide to quantitative uncertainty management.

Wiley, 2008.

S. Da Veiga.

Global sensitivity analysis with dependence measures.

Journal of Statistical Computation and Simulation, 85(7):1283–1305, 2015.

E. Fekhari, V. Chabridon, J. Mure, and B. Iooss.

Fast given-data uncertainty propagation in offshore wind turbine simulator using Bayesian quadrature.

preprint: hal-04052859v1, 2023.

E. Fekhari, B. Iooss, J. Muré, L. Pronzato, and J. Rendas.

Model predictivity assessment: incremental test-set selection and accuracy evaluation.

In N. Salvati, C. Perna, S. Marchetti, and R. Chambers, editors, Studies in Theoretical and Applied Statistics, pages 315–347. Springer, 2023.

References iii

K. Fang, M-Q Liu, H. Qin, and Y-D Zhou.

Theory and application of uniform experimental designs, volume 221.

Springer, 2018.

A. Gretton, K. Fukumizu, C. H. Teo, L. Song, B. Schölkopf, and A. J. Smola.

A Kernel Statistical Test of Independence.

In Advances in Neural Information Processing Systems, pages 585–592, 2008.

A. Gretton, R. Herbrich, A. J. Smola, O. Bousquet, and B. Schölkopf.

Kernel Methods for Measuring Independence.

Journal of Machine Learning Research, 6:2075–2129, 2005.

T. Hastie, R. Tibshirani, and J. Friedman.

The Elements of Statistical Learning: Data mining, Inference, and Prediction.

Springer-Verlag New York, 2nd ed. edition, 2009.

B. Iooss, V. Chabridon, and V. Thouvenot.

Variance-based importance measures for machine learning model interpretability.

In Proc. of the 23rd Congrès de Maîtrise des Risques et de Sûreté de Fonctionnement, Institut pour la Maîtrise des Risques (Lambda-Mu 23), Paris Saclay, France, October 2022.

P. L'Ecuyer.

Randomized Quasi-Monte Carlo: An Introduction for Practitioners.

In Monte Carlo and Quasi-Monte Carlo Methods, pages 29–52, Cham, 2018. Springer International Publishing.

S. Lacoste-Julien, F. Lindsten, and F. Bach.

Sequential Kernel Herding: Frank-Wolfe Optimization for Particle Filtering.

In Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, volume 38, pages 544–552, 2015.

References iv

G. Leobacher and F. Pillichshammer.

Introduction to quasi-Monte Carlo integration and applications.

Springer, 2014.

A. Marrel.

Sensitivity Analysis based on HSIC Dependence Measures (OpenTURNS Users' Day #14), 2021.

J. Morio and M. Balesdent.

Estimation of Rare Event Probabilities in Complex Aerospace and Other Systems: A Practical Approach.

Woodhead Publishing, Elsevier, 2015.

A. Marrel and V. Chabridon.

Statistical developments for target and conditional sensitivity analysis: Application on safety studies for nuclear reactor.

Reliability Engineering and System Safety, 214:107711, 2021.

K. Muandet, K. Fukumizu, B. Sriperumbudur, and B. Schölkopf.

Kernel Mean Embedding of Distributions: A Review and Beyond.

Foundations and Trends® in Machine Learning, 46(1-2):1–141, 2017.

C. Molnar.

Interpretable machine learning: A guide for making black-box models explainable.

GitHub, 2nd ed. edition, 2022.

V. I. Paulsen and M. Raghupathi.

An Introduction to the Theory of Reproducing Kernel Hilbert Spaces.

Cambridge studies in advanced mathematics. Cambridge University Press, 2016.

References v

H. Raguet and A. Marrel.

Target and conditional sensitivity analysis with emphasis on dependence measures.

ArXiv e-prints, pages 1–48, 2018.

D. Sejdinovic and A. Gretton.

What is an RKHS? (Lecture Notes), 2015.

B.K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G.R.G. Lanckriet.

Hilbert space embeddings and metrics on probability measures.

J. Mach. Learn. Res., 11:1517–1561, 2010.

G. Sarazin, A. Marrel, S. Da Veiga, and V. Chabridon.

Towards more interpretable global sensitivity analysis: New insights into Sobolev kernels and their feature maps.

ArXiv e-prints, pages 1–72, 2023.

R. C. Smith.

Uncertainty Quantification: Theory, Implementation, and Applications.

Society for Industrial and Applied Mathematics, 2013.

A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola.

Global Sensitivity Analysis. The Primer.

Wiley, 2008.

G. J. Székely, M. L. Rizzo, and N. K. Bakirov.

Measuring and testing dependence by correlation of distances.

The Annals of Statistics, 35(6):2769–2794, 2007.

References vi

T. J. Sullivan.

Introduction to Uncertainty Quantification, volume 63 of Texts in Applied Mathematics.

Springer International Publishing Switzerland, 2015.

D. J. Sutherland.

Modern Kernel Methods in Machine Learning: Part I (ETICS Summer School Course), 2022.