Simulator Calibration under Covariate Shift with Kernels

Motonobu Kanagawa

EURECOM

Workshop on Functional Inference and Machine Intelligence 2020
February 2020, EURECOM, France

Contents of This Talk

- Keiichi Kisamori, Motonobu Kanagawa and Keisuke Yamazaki
- Simulator Calibration under Covariate Shift with Kernels
- *AISTATS 2020*, to appear
- arXiv:1809.08159

Outline

Computer Simulators are Everywhere

- Computer Simulator: computer program for modeling a real-world phenomenon.

- e.g., climate, epidemics, natural disasters, cardiology, industrial manufacturing process, etc, etc...

Computer Simulators are Everywhere

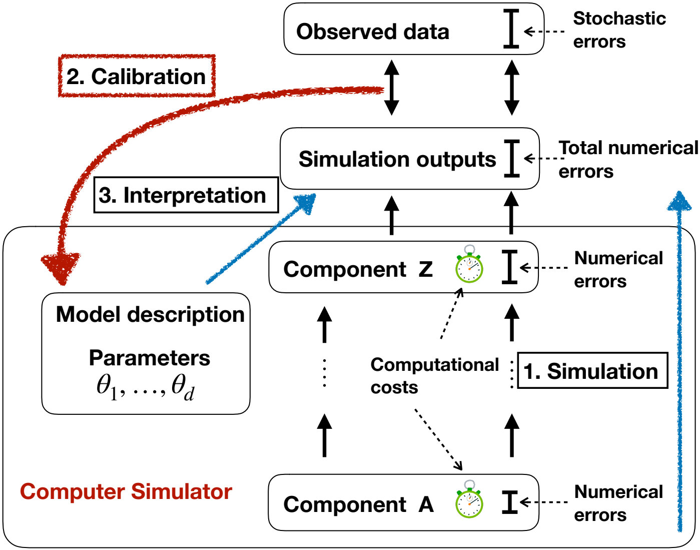
- Computer Simulator: computer program for modeling a real-world phenomenon.

- e.g., climate, epidemics, natural disasters, cardiology, industrial manufacturing process, etc, etc...

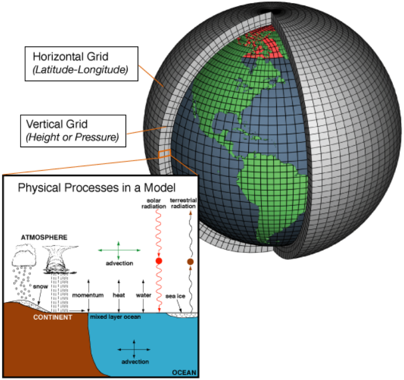
- Simulation provides insights/understanding about the system of interest.

- Enables prediction about the phenomenon in the future / under a hypothetical condition.

Computer Simulation and Related Tasks



Example: Climate Simulator



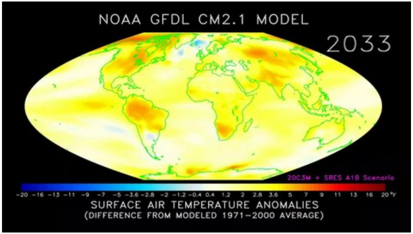


Figure 1: From Wikipedia "General circulation model"

Example: Industrial Manufacturing Process Simulator

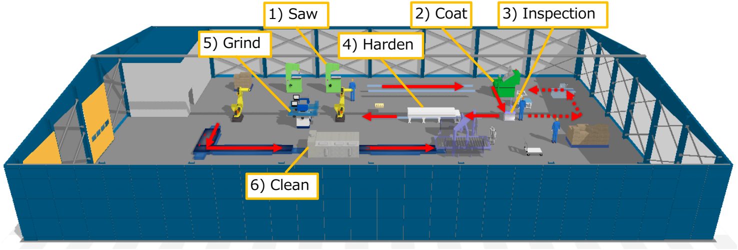


Figure 2: Simulator constructed with *WITNESS*, a popular software package for production simulation (`https://www.lanner.com/en-us/`).

Target System: Formulation as a Regression Model

- We consider a system takes $x \in \mathcal{X} \subset \mathbb{R}^{d_\mathcal{X}}$ as a input, and outputs $y(x) \in \mathbb{R}$.

- The input-output relationship $x \to y(x)$ can be written as

$$y(x) := R(x) + e(x), \quad x \in \mathcal{X}$$

Target System: Formulation as a Regression Model

- We consider a system takes $x \in \mathcal{X} \subset \mathbb{R}^{d_\mathcal{X}}$ as a input, and outputs $y(x) \in \mathbb{R}$.

- The input-output relationship $x \to y(x)$ can be written as

$$y(x) := R(x) + e(x), \quad x \in \mathcal{X}$$

where

- $R : \mathcal{X} \to \mathbb{R}$: an (unknown) deterministic regression function.

- $e : \mathcal{X} \to \mathbb{R}$: an (unknown) zero-mean stochastic process (representing stochastic error).

Target System: Examples

Climate simulation (How global temperature changes?):

- Input x: time point.

- Output $R(x)$: the global temperature.

Target System: Examples

Climate simulation (How global temperature changes?):

- Input x: time point.

- Output $R(x)$: the global temperature.

Manufacturing process simulation (How production efficiency changes?):

- Input x: the number of products to be manufactured in one day.

- Output $R(x)$: the total time required to manufacture all the products.

Target System: Training Data

- We assume that data $D_n := \{(X_i, Y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathbb{R}$ are available from the target system.

Target System: Training Data

- We assume that data $D_n := \{(X_i, Y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathbb{R}$ are available from the target system.

- Define q_0 as a probability density function on $\mathcal{X}$, and assume that the data are given as

$$X_1, \ldots, X_n \sim q_0 \quad \text{(i.i.d.)}$$
$$Y_i = y(X_i) = R(X_i) + e(X_i), \quad i = 1, \ldots, n,$$

Target System: Training Data

- We assume that data $D_n := \{(X_i, Y_i)\}_{i=1}^n \subset \mathcal{X} \times \mathbb{R}$ are available from the target system.

- Define q_0 as a probability density function on $\mathcal{X}$, and assume that the data are given as

$$
\begin{aligned}
&X_1, \ldots, X_n \sim q_0 \quad \text{(i.i.d.)} \\
&Y_i = y(X_i) = R(X_i) + e(X_i), \quad i = 1, \ldots, n,
\end{aligned}
$$

- The input density q_0 may be known, if it is designed by the user (experimental design).

Simulator

- Let $\Theta \subset \mathbb{R}^{d_\Theta}$ be a parameter space, and $r : \mathcal{X} \times \Theta \to \mathbb{R}$ be a deterministic function.

- For a fixed $\theta \in \Theta$, we define a "simulation model" as the mapping

$$x \in \mathcal{X} \to r(x, \theta) \in \mathbb{R}.$$

Simulator

- Let $\Theta \subset \mathbb{R}^{d_\Theta}$ be a parameter space, and $r : \mathcal{X} \times \Theta \to \mathbb{R}$ be a deterministic function.

- For a fixed $\theta \in \Theta$, we define a "simulation model" as the mapping

$$x \in \mathcal{X} \to r(x, \theta) \in \mathbb{R}.$$

- The "user" designs $r(x, \theta)$ so that it resembles the regression function $R(x)$ of the target system.

Simulator

- Let $\Theta \subset \mathbb{R}^{d_\Theta}$ be a parameter space, and $r : \mathcal{X} \times \Theta \to \mathbb{R}$ be a deterministic function.

- For a fixed $\theta \in \Theta$, we define a "simulation model" as the mapping

$$x \in \mathcal{X} \to r(x, \theta) \in \mathbb{R}.$$

- The "user" designs $r(x, \theta)$ so that it resembles the regression function $R(x)$ of the target system.

- By design, the user can produce the output $r(x, \theta)$ given $(x, \theta) \in \mathcal{X} \times \Theta$.

- However, simulating one output $r(x, \theta)$ for given (x, θ) may be computationally very expensive.

Calibration: Parameter Tuning of a Simulation Model

- The question is how to find a "good" parameter θ in the simulation model $r(x, \theta)$.

- To this end we can use data $D_n := \{(X_i, Y_i)\}_{i=1}^n$ from the target system $y(x) = R(x) + e(x)$.

Calibration: Parameter Tuning of a Simulation Model

- The question is how to find a "good" parameter θ in the simulation model $r(x,\theta)$.

- To this end we can use data $D_n := \{(X_i, Y_i)\}_{i=1}^n$ from the target system $y(x) = R(x) + e(x)$.

- "Good" θ should be such that $r(x,\theta)$ "approximates well" the true (unknown) function $R(x)$.

Calibration: Parameter Tuning of a Simulation Model

- The question is how to find a "good" parameter θ in the simulation model $r(x, \theta)$.

- To this end we can use data $D_n := \{(X_i, Y_i)\}_{i=1}^n$ from the target system $y(x) = R(x) + e(x)$.

- "Good" θ should be such that $r(x, \theta)$ "approximates well" the true (unknown) function $R(x)$.

- But in what sense should $r(x, \theta)$ "approximate well" $R(x)$?

Calibration for Extrapolation: Covariate Shift

- Simulation is very often used for the purpose of extrapolation.

- i.e., prediction on an input region where training data are scarce.

Calibration for Extrapolation: Covariate Shift

- Simulation is very often used for the purpose of extrapolation.

- i.e., prediction on an input region where training data are scarce.

- In machine learning, this situation is known as **Covariate Shift** [Shimodaira, 2000].

Calibration for Extrapolation: Covariate Shift

- Simulation is very often used for the purpose of extrapolation.

- i.e., prediction on an input region where training data are scarce.

- In machine learning, this situation is known as **Covariate Shift** [Shimodaira, 2000].

Examples:

- **Climate Simulation**: Prediction is required for the future, but data are only available from the past.

Calibration for Extrapolation: Covariate Shift

- Simulation is very often used for the purpose of extrapolation.

- i.e., prediction on an input region where training data are scarce.

- In machine learning, this situation is known as **Covariate Shift** [Shimodaira, 2000].

Examples:

- **Climate Simulation**: Prediction is required for the future, but data are only available from the past.

- **Manufacturing Process Simulation**: Prediction is required for mass production (when the factory is deployed), while data are only available from a trial period.

Calibration for Extrapolation: Covariate Shift

- Covarite shift is the setting where input distributions for training $q_0(x)$ and test $q_1(x)$ are **different**:

Calibration for Extrapolation: Covariate Shift

- Covarite shift is the setting where input distributions for training $q_0(x)$ and test $q_1(x)$ are **different**:

- Training inputs locations are generated $X_1, \ldots, X_n \sim q_0$;

Calibration for Extrapolation: Covariate Shift

- Covarite shift is the setting where input distributions for training $q_0(x)$ and test $q_1(x)$ are **different**:

- Training inputs locations are generated $X_1, \ldots, X_n \sim q_0$;

- But test (or prediction) is required for locations $\tilde{X}_1, \ldots, \tilde{X}_m \sim q_1$.

Calibration for Extrapolation: Covariate Shift

- Therefore the generalization error should be defined in terms of the test input density $q_1(x)$.

$$
\begin{aligned}
L(\theta) &:= \int (y(x) - r(x,\theta))^2 \, q_1(x) dx \\
&= \int (y(x) - r(x,\theta))^2 \, \beta(x) q_0(x) dx,
\end{aligned}
$$

Calibration for Extrapolation: Covariate Shift

- Therefore the generalization error should be defined in terms of the test input density $q_1(x)$.

$$\begin{aligned} L(\theta) &:= \int (y(x) - r(x,\theta))^2 \, q_1(x) dx \\ &= \int (y(x) - r(x,\theta))^2 \, \beta(x) q_0(x) dx, \end{aligned}$$

where $\beta : \mathcal{X} \to \mathbb{R}_+$ is the **importance weight** function:

$$\beta(x) := q_1(x)/q_0(x).$$

Calibration for Extrapolation: Covariate Shift

- Therefore the generalization error should be defined in terms of the test input density $q_1(x)$.

$$
\begin{aligned}
L(\theta) &:= \int (y(x) - r(x,\theta))^2 \, q_1(x) dx \\
&= \int (y(x) - r(x,\theta))^2 \, \beta(x) q_0(x) dx,
\end{aligned}
$$

where $\beta : \mathcal{X} \to \mathbb{R}_+$ is the **importance weight** function:

$$\beta(x) := q_1(x)/q_0(x).$$

- One needs to tune the parameter $\theta \in \Theta$ so that this generalization error will be small.

- The generalization error can be approximated by weighted squares:

$$L_n(\theta) := \frac{1}{n} \sum_{i=1}^{n} \beta(X_i) \left(Y_i - r(X_i, \theta)\right)^2.$$

because $X_1, \ldots, X_n \sim q_0$.

Why One Needs to Care About Covariate Shift?

- Simulator $r(x, \theta)$ is a **parametric** model, with a finite degree of freedom.

- As such, $r(x, \theta)$ cannot capture all aspects of the unknown target system $R(x)$ ("All models are wrong").

Why One Needs to Care About Covariate Shift?

- Simulator $r(x, \theta)$ is a **parametric** model, with a finite degree of freedom.

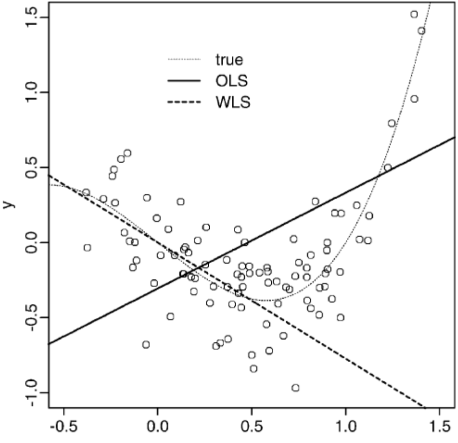
- As such, $r(x, \theta)$ cannot capture all aspects of the unknown target system $R(x)$ ("All models are wrong").

- Under such a model misspecification, the optimal model under covariate shift can be **drastically different** from the one without covariate shift.

Effect of Covariate Shift under Model Misspecification

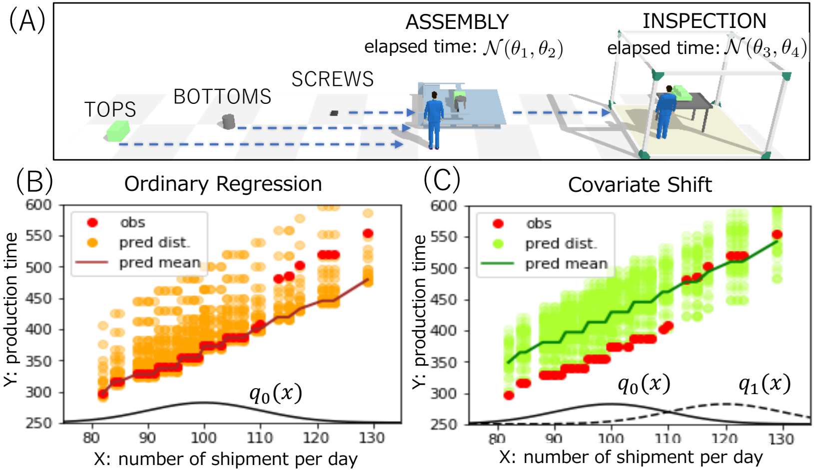
- **True** = 3rd order polynomial (curve); **OLS** = standard linear fit (solid line); **WLS** = linear with importance weighting (dotted line).

- $q_0 = \mathcal{N}(0.5, 2.5^2)$, $q_1 = \mathcal{N}(0.0, 0.3^2)$. [Shimodaira, 2000]



(a)

Covariate Shift in Manufacturing Process Simulation



Challenges in Simulator Calibration

There are several challenges in simulator calibration (in general).

Challenges in Simulator Calibration

There are several challenges in simulator calibration (in general).

- The mapping $(x, \theta) \to r(x, \theta)$ is **usually very complicated**.

e.g. a simulation may involve numerically solving differential equations, or various decision rules of agents (if-else rules).

Challenges in Simulator Calibration

There are several challenges in simulator calibration (in general).

- The mapping $(x, \theta) \to r(x, \theta)$ is **usually very complicated**.

e.g. a simulation may involve numerically solving differential equations, or various decision rules of agents (if-else rules).

- Therefore $r(x, \theta)$ **cannot be written in a simple functional form**.

— This prohibits the standard statistical inference procedures (MLE, Bayes).

Challenges in Simulator Calibration

There are several challenges in simulator calibration (in general).

- The mapping $(x, \theta) \to r(x, \theta)$ is **usually very complicated**.

e.g. a simulation may involve numerically solving differential equations, or various decision rules of agents (if-else rules).

- Therefore $r(x, \theta)$ **cannot be written in a simple functional form**.

— This prohibits the standard statistical inference procedures (MLE, Bayes).

- One can only generate an output $y = r(x, \theta)$, but one such simulation may be **computationally very expensive**.

Our Contributions

- We propose a novel method for simulator calibration, explicitly dealing with covariate shift.

Our Contributions

- We propose a novel method for simulator calibration, explicitly dealing with covariate shift.

- The proposed method is based on **Kernel ABC** (**A**pproximate **B**ayesian **C**omputation) and an **importance-weighted kernel.**

Our Contributions

- We propose a novel method for simulator calibration, explicitly dealing with covariate shift.

- The proposed method is based on **Kernel ABC** (**A**pproximate **B**ayesian **C**omputation) and an **importance-weighted kernel.**

- The entire framework is based on **Kernel Mean Embedding**.

Our Contributions

- We propose a novel method for simulator calibration, explicitly dealing with covariate shift.

- The proposed method is based on **Kernel ABC** (**A**pproximate **B**ayesian **C**omputation) and an **importance-weighted kernel.**

- The entire framework is based on **Kernel Mean Embedding**.

- Theoretical analysis is provided, with a novel result on conditional mean embedding.

Our Contributions

- We propose a novel method for simulator calibration, explicitly dealing with covariate shift.

- The proposed method is based on **Kernel ABC** (**A**pproximate **B**ayesian **C**omputation) and an **importance-weighted kernel.**

- The entire framework is based on **Kernel Mean Embedding**.

- Theoretical analysis is provided, with a novel result on conditional mean embedding.

- Experiments on **manufacturing process simulators**.

Outline

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric function on a set $\mathcal{X}$.

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric function on a set $\mathcal{X}$.

The function $k(x, x')$ is called a **positive definite kernel**, if

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x_i, x_j) \geq 0 \quad \text{holds}$$

$$\text{for all} \quad n \in \mathbb{N}, \quad c_1, \ldots, c_n \in \mathbb{R}, \quad x_1, \ldots, x_n \in \mathcal{X}.$$

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric function on a set $\mathcal{X}$.

The function $k(x, x')$ is called a **positive definite kernel**, if

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x_i, x_j) \geq 0 \quad \text{holds}$$

$$\text{for all} \quad n \in \mathbb{N}, \quad c_1, \dots, c_n \in \mathbb{R}, \quad x_1, \dots, x_n \in \mathcal{X}.$$

Examples of positive definite kernels on $\mathcal{X} = \mathbb{R}^d$:

$$\begin{array}{rlcl} \text{Gaussian} & k(x, x') & = & \exp(-\|x - x'\|^2/\gamma^2). \\ \text{Laplace (Matérn)} & k(x, x') & = & \exp(-\|x - x'\|/\gamma). \\ \text{Linear} & k(x, x') & = & \langle x, x' \rangle . \\ \text{Polynomial} & k(x, x') & = & (\langle x, x' \rangle + c)^m. \end{array}$$

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric function on a set $\mathcal{X}$.

The function $k(x, x')$ is called a **positive definite kernel**, if

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(x_i, x_j) \geq 0 \quad \text{holds}$$

for all $\quad n \in \mathbb{N}, \quad c_1, \ldots, c_n \in \mathbb{R}, \quad x_1, \ldots, x_n \in \mathcal{X}.$

Examples of positive definite kernels on $\mathcal{X} = \mathbb{R}^d$:

$$\begin{array}{rrcl} \text{Gaussian} & k(x, x') & = & \exp(-\|x - x'\|^2/\gamma^2). \\ \text{Laplace (Matérn)} & k(x, x') & = & \exp(-\|x - x'\|/\gamma). \\ \text{Linear} & k(x, x') & = & \langle x, x' \rangle . \\ \text{Polynomial} & k(x, x') & = & (\langle x, x' \rangle + c)^m. \end{array}$$

In this talk, I will simply call k a kernel.

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

For any kernel k, there is a **uniquely associated Hilbert space** $\mathcal{H}$ consisting of functions on $\mathcal{X}$ such that

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

For any kernel k, there is a **uniquely associated Hilbert space $\mathcal{H}$** consisting of functions on $\mathcal{X}$ such that

$$\text{(i)} \quad k(\cdot, x) \in \mathcal{H} \quad \text{for all } x \in \mathcal{X}$$

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

For any kernel k, there is a **uniquely associated Hilbert space $\mathcal{H}$** consisting of functions on $\mathcal{X}$ such that

$$\text{(i)} \quad k(\cdot, x) \in \mathcal{H} \quad \text{for all } x \in \mathcal{X}$$

where $k(\cdot, x)$ is the function of the first argument with x fixed:

$$x' \in \mathcal{X} \to k(x', x).$$

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

For any kernel k, there is a **uniquely associated Hilbert space** $\mathcal{H}$ consisting of functions on $\mathcal{X}$ such that

$$\text{(i)} \quad k(\cdot, x) \in \mathcal{H} \quad \text{for all } x \in \mathcal{X}$$

where $k(\cdot, x)$ is the function of the first argument with x fixed:

$$x' \in \mathcal{X} \to k(x', x).$$

$$\text{(ii)} \quad f(x) = \langle f, k(\cdot, x) \rangle_{\mathcal{H}} \quad \text{for all } f \in \mathcal{H} \text{ and } x \in \mathcal{X},$$

which is called the **reproducing property**.

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

For any kernel k, there is a **uniquely associated Hilbert space $\mathcal{H}$** consisting of functions on $\mathcal{X}$ such that

$$\text{(i)} \quad k(\cdot, x) \in \mathcal{H} \quad \text{for all } x \in \mathcal{X}$$

where $k(\cdot, x)$ is the function of the first argument with x fixed:

$$x' \in \mathcal{X} \to k(x', x).$$

$$\text{(ii)} \quad f(x) = \langle f, k(\cdot, x) \rangle_{\mathcal{H}} \quad \text{for all } f \in \mathcal{H} \text{ and } x \in \mathcal{X},$$

which is called the **reproducing property**.

- - $\mathcal{H}$ is called the **RKHS** of k.

Kernels and Reproducing Kernel Hilbert Spaces (RKHS)

For any kernel k, there is a **uniquely associated Hilbert space $\mathcal{H}$** consisting of functions on $\mathcal{X}$ such that

$$\text{(i)} \quad k(\cdot, x) \in \mathcal{H} \quad \text{for all } x \in \mathcal{X}$$

where $k(\cdot, x)$ is the function of the first argument with x fixed:

$$x' \in \mathcal{X} \to k(x', x).$$

$$\text{(ii)} \quad f(x) = \langle f, k(\cdot, x) \rangle_{\mathcal{H}} \quad \text{for all } f \in \mathcal{H} \text{ and } x \in \mathcal{X},$$

which is called the **reproducing property**.

- - $\mathcal{H}$ is called the **RKHS** of k.
- - $\mathcal{H}$ can be written as

$$\mathcal{H} = \overline{\text{span}\, \{k(\cdot, x) \mid x \in \mathcal{X}\}}$$

Kernel Mean Embeddings [Smola et al., 2007]

A framework for representing distributions in an RKHS.

Kernel Mean Embeddings [Smola et al., 2007]

A framework for representing distributions in an RKHS.

- - Let $\mathcal{P}$ be the set of **all probability distributions** on $\mathcal{X}$.

Kernel Mean Embeddings [Smola et al., 2007]

A framework for representing distributions in an RKHS.

- - Let $\mathcal{P}$ be the set of **all probability distributions** on $\mathcal{X}$.
- - Let k be a kernel on $\mathcal{X}$, and $\mathcal{H}$ be its RKHS.

Kernel Mean Embeddings [Smola et al., 2007]

A framework for representing distributions in an RKHS.

- - Let $\mathcal{P}$ be the set of **all probability distributions** on $\mathcal{X}$.
- - Let k be a kernel on $\mathcal{X}$, and $\mathcal{H}$ be its RKHS.

For each distribution $P \in \mathcal{P}$, define the **kernel mean**:

$$\mu_P := \int k(\cdot, x) dP(x) \in \mathcal{H}.$$

which is a **representation** of P in $\mathcal{H}$.

Kernel Mean Embeddings [Smola et al., 2007]

A framework for representing distributions in an RKHS.

- - Let $\mathcal{P}$ be the set of **all probability distributions** on $\mathcal{X}$.
- - Let k be a kernel on $\mathcal{X}$, and $\mathcal{H}$ be its RKHS.

For each distribution $P \in \mathcal{P}$, define the **kernel mean**:

$$\mu_P := \int k(\cdot, x) dP(x) \in \mathcal{H}.$$

which is a **representation** of P in $\mathcal{H}$.

A key concept: Characteristic kernels [Fukumizu et al., 2008].

- The kernel k is called characteristic, if for any $P, Q \in \mathcal{P}$,

$$\mu_P = \mu_Q \quad \text{if and only if} \quad P = Q.$$

Kernel Mean Embeddings [Smola et al., 2007]

A framework for representing distributions in an RKHS.

- - Let $\mathcal{P}$ be the set of **all probability distributions** on $\mathcal{X}$.
- - Let k be a kernel on $\mathcal{X}$, and $\mathcal{H}$ be its RKHS.

For each distribution $P \in \mathcal{P}$, define the **kernel mean**:

$$\mu_P := \int k(\cdot, x) dP(x) \in \mathcal{H}.$$

which is a **representation** of P in $\mathcal{H}$.

A key concept: Characteristic kernels [Fukumizu et al., 2008].

- The kernel k is called **characteristic**, if for any $P, Q \in \mathcal{P}$,

$$\mu_P = \mu_Q \quad \text{if and } \textbf{only if} \quad P = Q.$$

- In other words, k is characteristic if

$$\text{the mapping} \quad P \in \mathcal{P} \to \mu_P \in \mathcal{H} \quad \text{is } \textbf{injective}.$$

Kernel Mean Embeddings [Smola et al., 2007]

Intuitively, k being **characteristic** implies that $\mathcal{H}$ **is large enough.**

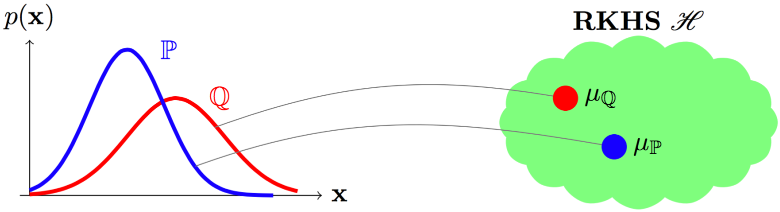


Figure 3: Injective embedding [Muandet et al., 2017, Figure 2.3]

Kernel Mean Embeddings [Smola et al., 2007]

Intuitively, k being characteristic implies that $\mathcal{H}$ **is large enough.**

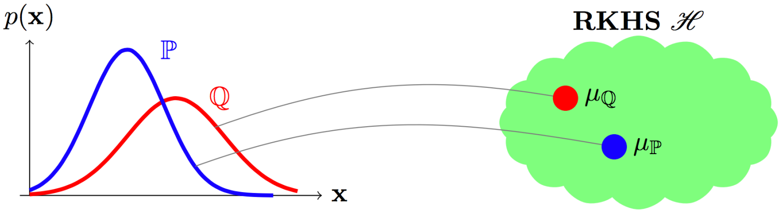


Figure 3: Injective embedding [Muandet et al., 2017, Figure 2.3]

Examples of characteristic kernels on $\mathcal{X} = \mathbb{R}^d$:

- Gaussian and Matérn kernels [Sriperumbudur et al., 2010].

Kernel Mean Embeddings [Smola et al., 2007]

Intuitively, k being **characteristic** implies that $\mathcal{H}$ **is large enough.**

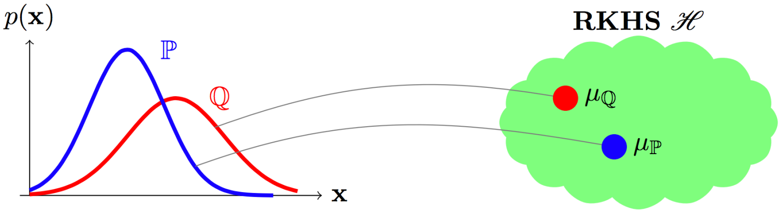


Figure 3: Injective embedding [Muandet et al., 2017, Figure 2.3]

Examples of characteristic kernels on $\mathcal{X} = \mathbb{R}^d$:
- Gaussian and Matérn kernels [Sriperumbudur et al., 2010].

Examples of **non**-characteristic kernels on $\mathcal{X} = \mathbb{R}^d$:
- Linear and polynomial kernels.

Outline

Vector-valued Function Defined with the Simulator

- We define a vector-valued function $r^n : \Theta \to \mathbb{R}^n$ by

$$r^n(\theta) := (r(X_1, \theta), \ldots, r(X_n, \theta))^\top \in \mathbb{R}^n, \quad \theta \in \Theta.$$

where $r(x, \theta)$ is the simulation model.

Vector-valued Function Defined with the Simulator

- We define a vector-valued function $r^n : \Theta \to \mathbb{R}^n$ by

$$r^n(\theta) := (r(X_1, \theta), \dots, r(X_n, \theta))^\top \in \mathbb{R}^n, \quad \theta \in \Theta.$$

where $r(x, \theta)$ is the simulation model.

- We assume that input points $X_1, \dots, X_n \in \mathcal{X}$ are **given and fixed** (throughout the talk).

Optimal Parameters and Predictions under Covariate Shift

- Define $\Theta^* \subset \Theta$ as the set of optimal parameters minimizing the weighted squares.

- i.e., for all $\theta^* \in \Theta^*$, we have

$$\sum_{i=1}^{n} \beta(X_i)(Y_i - r(X_i, \theta^*))^2 = \min_{\theta \in \mathrm{supp}(\pi)} \sum_{i=1}^{n} \beta(X_i)(Y_i - r(X_i, \theta))^2.$$

Optimal Parameters and Predictions under Covariate Shift

- Define $\Theta^* \subset \Theta$ as the set of optimal parameters minimizing the weighted squares.

- i.e., for all $\theta^* \in \Theta^*$, we have

$$\sum_{i=1}^{n} \beta(X_i)(Y_i - r(X_i, \theta^*))^2 = \min_{\theta \in \mathrm{supp}(\pi)} \sum_{i=1}^{n} \beta(X_i)(Y_i - r(X_i, \theta))^2.$$

- Θ^* may contain multiple (or even infinitely meany) elements.

Optimal Parameters and Predictions under Covariate Shift

- Define $\Theta^* \subset \Theta$ as the set of optimal parameters minimizing the weighted squares.

- i.e., for all $\theta^* \in \Theta^*$, we have

$$\sum_{i=1}^{n} \beta(X_i)(Y_i - r(X_i, \theta^*))^2 = \min_{\theta \in \mathrm{supp}(\pi)} \sum_{i=1}^{n} \beta(X_i)(Y_i - r(X_i, \theta))^2.$$

- Θ^* may contain multiple (or even infinitely meany) elements.

- We assume that the resulting simulator outputs are unique, i.e.,

$$r^* := r^n(\theta^*) = r^n(\tilde{\theta^*}), \quad \forall \theta^*, \tilde{\theta^*} \in \Theta^*.$$

- r^* is "optimal" predictions under covariate shift.

"Target" Posterior Distribution

- For the prior $\pi(\theta)$, define a random variable $\vartheta \sim \pi$.

"Target" Posterior Distribution

- For the prior $\pi(\theta)$, define a random variable $\vartheta \sim \pi$.

- Then $r^n(\vartheta) = (r(X_1, \vartheta), \ldots, r(X_n, \vartheta))^\top \in \mathbb{R}^n$ is also a random variable.

"Target" Posterior Distribution

- For the prior $\pi(\theta)$, define a random variable $\vartheta \sim \pi$.

- Then $r^n(\vartheta) = (r(X_1, \vartheta), \ldots, r(X_n, \vartheta))^\top \in \mathbb{R}^n$ is also a random variable.

- The distribution of $r^n(\vartheta)$ is the **push-forward measure** of π under the mapping $r^n : \Theta \to \mathbb{R}^n$, and denoted it by $r^n\pi$.

"Target" Posterior Distribution

- For the prior $\pi(\theta)$, define a random variable $\vartheta \sim \pi$.

- Then $r^n(\vartheta) = (r(X_1, \vartheta), \ldots, r(X_n, \vartheta))^\top \in \mathbb{R}^n$ is also a random variable.

- The distribution of $r^n(\vartheta)$ is the **push-forward measure** of π under the mapping $r^n : \Theta \to \mathbb{R}^n$, and denoted it by $r^n\pi$.

- The support of the push-forward measure $r^n\pi$ is given by

$$\operatorname{supp}(r^n\pi) = \{r^n(\theta) \mid \theta \in \operatorname{supp}(\pi)\}.$$

"Target" Poster Distribution

- For the joint random variables

$$(\vartheta, r^n(\vartheta)) \in \Theta \times \mathbb{R}^n.$$

consider the **conditioning** $r^n(\vartheta) = \boldsymbol{y} \in \mathrm{supp}(r^n \pi)$.

"Target" Poster Distribution

- For the joint random variables

$$(\vartheta, r^n(\vartheta)) \in \Theta \times \mathbb{R}^n.$$

consider the **conditioning** $r^n(\vartheta) = \boldsymbol{y} \in \mathrm{supp}(r^n \pi)$.

- Denote the resulting conditional distribution on Θ by

$$P_\pi(\theta|\boldsymbol{y}), \quad \boldsymbol{y} \in \mathrm{supp}(r^n \pi)$$

- This is the "posterior" distribution of ϑ, provided that $\boldsymbol{y} = r^n(\vartheta)$ is "observed".

"Target" Poster Distribution

- For the joint random variables

$$(\vartheta, r^n(\vartheta)) \in \Theta \times \mathbb{R}^n.$$

consider the **conditioning** $r^n(\vartheta) = \boldsymbol{y} \in \mathrm{supp}(r^n\pi)$.

- Denote the resulting conditional distribution on Θ by

$$P_\pi(\theta|\boldsymbol{y}), \quad \boldsymbol{y} \in \mathrm{supp}(r^n\pi)$$

- This is the "posterior" distribution of ϑ, provided that $\boldsymbol{y} = r^n(\vartheta)$ is "observed".

- Well-defined as a **disintegration** [Chang and Pollard, 1997].

"Target" Poster Distribution

- For the joint random variables

$$(\vartheta, r^n(\vartheta)) \in \Theta \times \mathbb{R}^n.$$

consider the **conditioning** $r^n(\vartheta) = \boldsymbol{y} \in \mathrm{supp}(r^n\pi)$.

- Denote the resulting conditional distribution on Θ by

$$P_\pi(\theta|\boldsymbol{y}), \quad \boldsymbol{y} \in \mathrm{supp}(r^n\pi)$$

- This is the "posterior" distribution of ϑ, provided that $\boldsymbol{y} = r^n(\vartheta)$ is "observed".

- Well-defined as a **disintegration** [Chang and Pollard, 1997].

- It will turn out that our approach aims at estimating

$$P_\pi(\theta|r^*)$$

the "posterior" distribution of ϑ, provided that $r^* = r^n(\vartheta)$ is "observed".

Outline

Overview of the Proposed Approach

- Let $k_\Theta(\theta, \theta')$ be a characteristic kernel on Θ, with $\mathcal{H}_\Theta$ its the RKHS.

Overview of the Proposed Approach

- Let $k_\Theta(\theta, \theta')$ be a characteristic kernel on Θ, with $\mathcal{H}_\Theta$ its the RKHS.

- We consider the embedding of the target posterior $P_\pi(\theta|r^*)$ in $\mathcal{H}_\Theta$:

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot, \theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta.$$

Overview of the Proposed Approach

- Let $k_\Theta(\theta, \theta')$ be a characteristic kernel on Θ, with $\mathcal{H}_\Theta$ its the RKHS.

- We consider the embedding of the target posterior $P_\pi(\theta|r^*)$ in $\mathcal{H}_\Theta$:

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot, \theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta.$$

- Our method consists of two parts:

1) Estimation of $\mu_{\Theta|r^*}$ from observed data $Y^n = (Y_1, \ldots, Y_n)^\top$, using **Kernel ABC** and an **importance-weighted kernel**.

Overview of the Proposed Approach

- Let $k_\Theta(\theta, \theta')$ be a characteristic kernel on Θ, with $\mathcal{H}_\Theta$ its the RKHS.

- We consider the embedding of the target posterior $P_\pi(\theta|r^*)$ in $\mathcal{H}_\Theta$:

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot, \theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta.$$

- Our method consists of two parts:

1) Estimation of $\mu_{\Theta|r^*}$ from observed data $Y^n = (Y_1, \dots, Y_n)^\top$, using **Kernel ABC** and an **importance-weighted kernel**.

2) Sampling parameters from the estimate of $\mu_{\Theta|r^*}$, using **Kernel Herding**.

Importance-weighted Kernel on Data Space

- We define the "data space" as $\mathbb{R}^n$.

- Recall observed data is $Y^n = (Y_1, \ldots, Y_n) \in \mathbb{R}^n$.

Importance-weighted Kernel on Data Space

- We define the "data space" as $\mathbb{R}^n$.

- Recall observed data is $Y^n = (Y_1, \ldots, Y_n) \in \mathbb{R}^n$.

- We define a kernel $k_{\mathbb{R}^n}(Y_a^n, Y_b^n)$ for $Y_a^n, Y_b^n \in \mathbb{R}^n$ in the following way:

$$k_{\mathbb{R}^n}(Y_a^n, Y_b^n) = \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\beta(X_i)(Y_{ai} - Y_{bi})^2\right).$$

where ...

Importance-weighted Kernel on Data Space

- We define the "data space" as $\mathbb{R}^n$.

- Recall observed data is $Y^n = (Y_1, \ldots, Y_n) \in \mathbb{R}^n$.

- We define a kernel $k_{\mathbb{R}^n}(Y_a^n, Y_b^n)$ for $Y_a^n, Y_b^n \in \mathbb{R}^n$ in the following way:

$$k_{\mathbb{R}^n}(Y_a^n, Y_b^n) = \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\beta(X_i)(Y_{ai} - Y_{bi})^2\right).$$

where ...

- Importance weights $\beta(X_1), \ldots, \beta(X_n)$: assumed to be given (e.g., known by design, or estimated in advance).

Importance-weighted Kernel on Data Space

- We define the "data space" as $\mathbb{R}^n$.

- Recall observed data is $Y^n = (Y_1, \ldots, Y_n) \in \mathbb{R}^n$.

- We define a kernel $k_{\mathbb{R}^n}(Y_a^n, Y_b^n)$ for $Y_a^n, Y_b^n \in \mathbb{R}^n$ in the following way:

$$k_{\mathbb{R}^n}(Y_a^n, Y_b^n) = \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\beta(X_i)(Y_{ai} - Y_{bi})^2\right).$$

where ...

- Importance weights $\beta(X_1), \ldots, \beta(X_n)$: assumed to be given (e.g., known by design, or estimated in advance).

- Input points $X_1, \ldots, X_n$: assumed to be given and **fixed**.

Importance-weighted Kernel on Data Space

- We define the "data space" as $\mathbb{R}^n$.

- Recall observed data is $Y^n = (Y_1, \ldots, Y_n) \in \mathbb{R}^n$.

- We define a kernel $k_{\mathbb{R}^n}(Y_a^n, Y_b^n)$ for $Y_a^n, Y_b^n \in \mathbb{R}^n$ in the following way:

$$k_{\mathbb{R}^n}(Y_a^n, Y_b^n) = \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} \beta(X_i)(Y_{ai} - Y_{bi})^2 \right).$$

where ...

- Importance weights $\beta(X_1), \ldots, \beta(X_n)$: assumed to be given (e.g., known by design, or estimated in advance).

- Input points $X_1, \ldots, X_n$: assumed to be given and **fixed**.

- We assume $0 < \beta(X_1), \ldots, \beta(X_n) < \infty$.

Importance-weighted Kernel on Data Space

- We define the "data space" as $\mathbb{R}^n$.

- Recall observed data is $Y^n = (Y_1, \dots, Y_n) \in \mathbb{R}^n$.

- We define a kernel $k_{\mathbb{R}^n}(Y_a^n, Y_b^n)$ for $Y_a^n, Y_b^n \in \mathbb{R}^n$ in the following way:

$$k_{\mathbb{R}^n}(Y_a^n, Y_b^n) = \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\beta(X_i)(Y_{ai} - Y_{bi})^2\right).$$

where ...

- Importance weights $\beta(X_1), \dots, \beta(X_n)$: assumed to be given (e.g., known by design, or estimated in advance).

- Input points $X_1, \dots, X_n$: assumed to be given and **fixed**.

- We assume $0 < \beta(X_1), \dots, \beta(X_n) < \infty$.

- $\sigma^2 > 0$: constant.

Kernel ABC with the Importance-weighted Kernel

- To estimate the kernel mean of the target posterior $P_\pi(\theta|r^*)$,

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot, \theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta,$$

we use Kernel ABC (**A**pproximate **B**ayesian **C**omputation) [Nakagome et al., 2013].

Kernel ABC with the Importance-weighted Kernel

- To estimate the kernel mean of the target posterior $P_\pi(\theta|r^*)$,

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot, \theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta,$$

we use Kernel ABC (**A**pproximate **B**ayesian **C**omputation) [Nakagome et al., 2013].

- For this we use the importance-weighted kernel.

Kernel ABC with the Importance-weighted Kernel

Step 1: Simulate **parameter-data pairs**

$$(\bar{\theta}_1, \bar{Y}_1^n), \ldots, (\bar{\theta}_m, \bar{Y}_m^n) \in \Theta \times \mathbb{R}^n$$

as follows:

Kernel ABC with the Importance-weighted Kernel

Step 1: Simulate **parameter-data pairs**

$$(\bar{\theta}_1, \bar{Y}_1^n), \ldots, (\bar{\theta}_m, \bar{Y}_m^n) \in \Theta \times \mathbb{R}^n$$

as follows:

- Random sampling from the prior $\pi(\theta)$:

$$\bar{\theta}_1, \ldots, \bar{\theta}_m \sim \pi(\theta) \quad (i.i.d.)$$

Kernel ABC with the Importance-weighted Kernel

Step 1: Simulate **parameter-data pairs**

$$(\bar{\theta}_1, \bar{Y}_1^n), \ldots, (\bar{\theta}_m, \bar{Y}_m^n) \in \Theta \times \mathbb{R}^n$$

as follows:

- Random sampling from the prior $\pi(\theta)$:

$$\bar{\theta}_1, \ldots, \bar{\theta}_m \sim \pi(\theta) \quad (i.i.d.)$$

- Generate pseudo data $\bar{Y}_j^n$ by running the simulator $r(\cdot, \theta)$ with $\theta = \bar{\theta}_j$ for $j = 1, \ldots, m$:

$$\bar{Y}_j^n := r^n(\bar{\theta}_j) = \left(r(X_1, \bar{\theta}_j), \ldots, r(X_n, \bar{\theta}_j)\right)^\top \in \mathbb{R}^n,$$

Kernel ABC with the Importance-weighted Kernel

- Step 2: Regard $(k_\Theta(\cdot, \bar{\theta}_1), \bar{Y}_1^n), \ldots, (k_\Theta(\cdot, \bar{\theta}_m), \bar{Y}_m^n)$ as "training data" for **regression from $\mathbb{R}^n$ to $\mathcal{H}_\Theta$.**

Kernel ABC with the Importance-weighted Kernel

- Step 2: Regard $(k_\Theta(\cdot, \bar{\theta}_1), \bar{Y}_1^n), \ldots, (k_\Theta(\cdot, \bar{\theta}_m), \bar{Y}_m^n)$ as "training data" for **regression from $\mathbb{R}^n$ to $\mathcal{H}_\Theta$.**

- Perform **Kernel Ridge Regression** using observed data Y^n as an "input":

$$\hat{\mu}_{\Theta|r^*} := \sum_{j=1}^{m} w_j k_\Theta(\cdot, \bar{\theta}_j) \ \in \mathcal{H}_\Theta,$$

$$(w_1, ..., w_m)^\top := (G + m\varepsilon I_m)^{-1} \mathbf{k}_{\mathbb{R}^n}(Y^n) \in \mathbb{R}^m,$$

Kernel ABC with the Importance-weighted Kernel

- Step 2: Regard $(k_\Theta(\cdot, \bar{\theta}_1), \bar{Y}_1^n), \ldots, (k_\Theta(\cdot, \bar{\theta}_m), \bar{Y}_m^n)$ as "training data" for **regression from $\mathbb{R}^n$ to $\mathcal{H}_\Theta$.**

- Perform **Kernel Ridge Regression** using observed data Y^n as an "input":

$$\hat{\mu}_{\Theta|r^*} := \sum_{j=1}^{m} w_j k_\Theta(\cdot, \bar{\theta}_j) \ \in \mathcal{H}_\Theta,$$

$$(w_1, ..., w_m)^\top := (G + m\varepsilon I_m)^{-1} \mathbf{k}_{\mathbb{R}^n}(Y^n) \in \mathbb{R}^m,$$

where $\varepsilon > 0$ is a regularization constant and

$$\begin{aligned} \mathbf{k}_{\mathbb{R}^n}(Y^n) \quad &:= \quad (k_{\mathbb{R}^n}(\bar{Y}_1^n, Y^n), ..., k_{\mathbb{R}^n}(\bar{Y}_m^n, Y^n))^\top \in \mathbb{R}^m \\ G \quad &:= \quad (k_{\mathbb{R}^n}(\bar{Y}_j^n, \bar{Y}_{j'}^n))_{j,j'=1}^m \in \mathbb{R}^{m \times m}. \end{aligned}$$

Kernel ABC with the Importance-weighted Kernel

- Step 2: Regard $(k_\Theta(\cdot,\bar\theta_1),\bar Y_1^n),\ldots,(k_\Theta(\cdot,\bar\theta_m),\bar Y_m^n)$ as "training data" for **regression from $\mathbb{R}^n$ to $\mathcal{H}_\Theta$.**

- Perform **Kernel Ridge Regression** using observed data Y^n as an "input":

$$\hat\mu_{\Theta|r^*} := \sum_{j=1}^m w_j k_\Theta(\cdot,\bar\theta_j) \ \in \mathcal{H}_\Theta,$$

$$(w_1,\ldots,w_m)^\top := (G + m\varepsilon I_m)^{-1}\mathbf{k}_{\mathbb{R}^n}(Y^n) \in \mathbb{R}^m,$$

where $\varepsilon > 0$ is a regularization constant and

$$\begin{aligned} \mathbf{k}_{\mathbb{R}^n}(Y^n) &:= (k_{\mathbb{R}^n}(\bar Y_1^n, Y^n),\ldots,k_{\mathbb{R}^n}(\bar Y_m^n, Y^n))^\top \in \mathbb{R}^m \\ G &:= (k_{\mathbb{R}^n}(\bar Y_j^n, \bar Y_{j'}^n))_{j,j'=1}^m \in \mathbb{R}^{m\times m}. \end{aligned}$$

- Here $k_{\mathbb{R}^n}$ is the importance-weighted kernel:

$$k_{\mathbb{R}^n}(Y_a^n, Y_b^n) = \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^n \beta(X_i)(Y_{ai}^n - Y_{bi}^n)^2\right).$$

Posterior Sampling from the Estimate of Kernel ABC

- We generate parameters $\check{\theta}_1, \ldots, \check{\theta}_m \in \Theta$ from the posterior embedding $\hat{\mu}_{\Theta|r^*} \in \mathcal{H}_\Theta$.

Posterior Sampling from the Estimate of Kernel ABC

- We generate parameters $\check{\theta}_1, \ldots, \check{\theta}_m \in \Theta$ from the posterior embedding $\hat{\mu}_{\Theta|r^*} \in \mathcal{H}_\Theta$.

- To this end, we apply Kernel Herding [Chen et al., 2010], a **deterministic, sequential** sampling method:

$$
\begin{aligned}
\check{\theta}_1 &:= \arg\max_{\theta \in \Theta} \hat{\mu}_{\Theta|r^*}(\theta) \\
\check{\theta}_t &:= \arg\max_{\theta \in \Theta} \hat{\mu}_{\Theta|r^*}(\theta) - \frac{1}{t} \sum_{j=1}^{t-1} k_\Theta(\theta, \check{\theta}_j) \quad (t = 2, \ldots, m).
\end{aligned}
$$

Posterior Sampling from the Estimate of Kernel ABC

- We generate parameters $\check{\theta}_1, \ldots, \check{\theta}_m \in \Theta$ from the posterior embedding $\hat{\mu}_{\Theta|r^*} \in \mathcal{H}_\Theta$.

- To this end, we apply Kernel Herding [Chen et al., 2010], a **deterministic, sequential** sampling method:

$$\begin{aligned} \check{\theta}_1 &:= \arg\max_{\theta \in \Theta} \hat{\mu}_{\Theta|r^*}(\theta) \\ \check{\theta}_t &:= \arg\max_{\theta \in \Theta} \hat{\mu}_{\Theta|r^*}(\theta) - \frac{1}{t}\sum_{j=1}^{t-1} k_\Theta(\theta, \check{\theta}_j) \quad (t = 2, \ldots, m). \end{aligned}$$

- Convergence guarantee under a mild condition [Bach et al., 2012].

$$\left\| \hat{\mu}_{\Theta|r^*} - \frac{1}{t}\sum_{j=1}^{t} k_\Theta(\cdot, \check{\theta}_j) \right\|_{\mathcal{H}_\Theta} = O(t^{-1/2}) \quad (t \to \infty).$$

Posterior Sampling from the Estimate of Kernel ABC

- We generate parameters $\check{\theta}_1, \ldots, \check{\theta}_m \in \Theta$ from the posterior embedding $\hat{\mu}_{\Theta|r^*} \in \mathcal{H}_\Theta$.

- To this end, we apply Kernel Herding [Chen et al., 2010], a **deterministic, sequential** sampling method:

$$
\begin{aligned}
\check{\theta}_1 &:= \arg\max_{\theta \in \Theta} \hat{\mu}_{\Theta|r^*}(\theta) \\
\check{\theta}_t &:= \arg\max_{\theta \in \Theta} \hat{\mu}_{\Theta|r^*}(\theta) - \frac{1}{t}\sum_{j=1}^{t-1} k_\Theta(\theta, \check{\theta}_j) \quad (t = 2, \ldots, m).
\end{aligned}
$$

- Convergence guarantee under a mild condition [Bach et al., 2012].

$$
\left\| \hat{\mu}_{\Theta|r^*} - \frac{1}{t}\sum_{j=1}^{t} k_\Theta(\cdot, \check{\theta}_j) \right\|_{\mathcal{H}_\Theta} = O(t^{-1/2}) \quad (t \to \infty).
$$

- Kernel Herding is a version of Quasi Monte Carlo [Dick et al., 2013].

Predictions with the Posterior-sampled Parameters

- For any test input $x \in \mathcal{X}$, we define the predictive output distribution as

$$P_\pi(y|x, r^*) = \int \delta\left(y = r(x, \theta)\right) dP_\pi(\theta|r^*)$$

where $\delta(\cdot)$ is the Dirac distribution at 0.

Predictions with the Posterior-sampled Parameters

- For any test input $x \in \mathcal{X}$, we define the predictive output distribution as

$$P_\pi(y|x, r^*) = \int \delta\left(y = r(x, \theta)\right) dP_\pi(\theta|r^*)$$

where $\delta(\cdot)$ is the Dirac distribution at 0.

We approximate $P_\pi(y|x, r^*)$ in the following way:

- For each sampled parameter $\check{\theta}_j$, run the simulator $r(x, \check{\theta}_j)$, and obtain a set of predictive outputs

$$r(x, \check{\theta}_1), \dots, r(x, \check{\theta}_m) \in \mathbb{R}.$$

Predictions with the Posterior-sampled Parameters

- For any test input $x \in \mathcal{X}$, we define the predictive output distribution as

$$P_\pi(y|x, r^*) = \int \delta\left(y = r(x, \theta)\right) dP_\pi(\theta|r^*)$$

where $\delta(\cdot)$ is the Dirac distribution at 0.

We approximate $P_\pi(y|x, r^*)$ in the following way:

- For each sampled parameter $\check{\theta}_j$, run the simulator $r(x, \check{\theta}_j)$, and obtain a set of predictive outputs

$$r(x, \check{\theta}_1), \dots, r(x, \check{\theta}_m) \in \mathbb{R}.$$

- Then approximate $P_\pi(y|x, r^*)$ as an empirical distribution

$$\hat{P}_\pi(y|x, r^*) := \frac{1}{m} \sum_{j=1}^{m} \delta(y - r(x, \check{\theta}_j)).$$

Why Should It Work?

We present a theoretical justification after describing experimental results.

Outline

Common Setting for All Experiments (Evaluation Metric)

-Training data $(X_1, Y_1), \ldots, (X_n, Y_n)$ are generated as

$$X_1, \ldots, X_n \sim q_0 \quad \text{(i.i.d.)}$$
$$Y_i = R(X_i) + \varepsilon_i, \quad i = 1, \ldots, n,$$

where $\varepsilon_i \sim N(0, \sigma^2_{\mathrm{noise}})$ are independent Gaussian noises.

Common Setting for All Experiments (Evaluation Metric)

-Training data $(X_1, Y_1), \ldots, (X_n, Y_n)$ are generated as

$$X_1, \ldots, X_n \sim q_0 \quad \text{(i.i.d.)}$$
$$Y_i = R(X_i) + \varepsilon_i, \quad i = 1, \ldots, n,$$

where $\varepsilon_i \sim N(0, \sigma^2_{\mathrm{noise}})$ are independent Gaussian noises.

- We evaluate the quality of the sampled parameters $\check{\theta}_1, \ldots, \check{\theta}_m$, in **predictions at test input locations**

$$\tilde{X}_1, \ldots, \tilde{X}_n \sim q_1 \quad \text{(i.i.d.)}.$$

Common Setting for All Experiments (Evaluation Metric)

-Training data $(X_1, Y_1), \ldots, (X_n, Y_n)$ are generated as

$$X_1, \ldots, X_n \sim q_0 \quad \text{(i.i.d.)}$$
$$Y_i = R(X_i) + \varepsilon_i, \quad i = 1, \ldots, n,$$

where $\varepsilon_i \sim N(0, \sigma^2_{\mathrm{noise}})$ are independent Gaussian noises.

- We evaluate the quality of the sampled parameters $\check{\theta}_1, \ldots, \check{\theta}_m$, in **predictions at test input locations**

$$\tilde{X}_1, \ldots, \tilde{X}_n \sim q_1 \quad \text{(i.i.d.)}.$$

- To this end, we compute Root Mean Square Errors (RMSE) defined as

$$\sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(R(\tilde{X}_i) - \frac{1}{m} \sum_{j=1}^{m} r(\tilde{X}_i, \check{\theta}_j) \right)^2}.$$

Common Setting for All Experiments (Proposed Method)

We use the following kernels for all the experiments:

$$
\begin{aligned}
k_{\mathbb{R}^n}(Y_a^n, Y_b^n) &= \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\beta(X_i)(Y_{ai}-Y_{bi})^2\right), \\
k_\Theta(\theta,\theta') &= \exp(-\|\theta-\theta'\|^2/2\sigma_\Theta^2).
\end{aligned}
$$

Common Setting for All Experiments (Proposed Method)

We use the following kernels for all the experiments:

$$
\begin{aligned}
k_{\mathbb{R}^n}(Y_a^n, Y_b^n) &= \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\beta(X_i)(Y_{ai}-Y_{bi})^2\right), \\
k_\Theta(\theta,\theta') &= \exp(-\|\theta-\theta'\|^2/2\sigma_\Theta^2).
\end{aligned}
$$

- Importance weights $\beta(X_i) = q_1(X_i)/q_0(X_i)$ are assumed to be known.

- Constants $\sigma^2, \sigma_\Theta^2 > 0$ are determined by the median heuristic using the simulated pairs $(\bar{\theta}_j, \bar{Y}_j^n)_{j=1}^m$.

Common Setting for All Experiments (Baseline)

- For comparison, we used the Metropolis-Hastings (MH) algorithm for sampling from the posterior

$$\Pr\left(\theta \mid |(X_i, Y_i)_{i=1}^n\right) \propto \pi(\theta) \prod_{i=1}^{n} \Pr(Y_i | X_i, \theta).$$

Common Setting for All Experiments (Baseline)

- For comparison, we used the Metropolis-Hastings (MH) algorithm for sampling from the posterior

$$\Pr(\theta \mid |(X_i, Y_i)_{i=1}^n) \propto \pi(\theta) \prod_{i=1}^n \Pr(Y_i | X_i, \theta).$$

where

$$\Pr(Y_i | X_i, \theta) \propto \exp\left(-\frac{1}{2{\sigma_{\text{noise}}}^2} \sum_{i=1}^n \beta(X_i)(Y_i - r(X_i, \theta))^2\right).$$

Common Setting for All Experiments (Baseline)

- For comparison, we used the Metropolis-Hastings (MH) algorithm for sampling from the posterior

$$\mathrm{Pr}\left(\theta \mid |(X_i, Y_i)_{i=1}^{n}\right) \propto \pi(\theta) \prod_{i=1}^{n} \mathrm{Pr}(Y_i|X_i, \theta).$$

where

$$\mathrm{Pr}(Y_i|X_i, \theta) \propto \exp\left(-\frac{1}{2{\sigma_{\mathrm{noise}}}^2} \sum_{i=1}^{n} \beta(X_i)(Y_i - r(X_i, \theta))^2\right).$$

- For MH, we assume that the **perfect knowledge** of the likelihood $\mathrm{Pr}(Y_i|X_i)$ is available, including its noise distribution

$$\varepsilon_i \sim N(0, \sigma^2_{\mathrm{noise}}).$$

- This is an unfair advantage over the proposed method.

Simple Synthetic Experiment: Setting

- The input space is $\mathcal{X} = \mathbb{R}$.

- True (unknown) regression function: **Third order polynomial**

$$R(x) = -x + x^3.$$

Simple Synthetic Experiment: Setting

- The input space is $\mathcal{X} = \mathbb{R}$.

- True (unknown) regression function: **Third order polynomial**

$$R(x) = -x + x^3.$$

- Simulation model: **Linear function** (model misspecification!)

$$r(x, \theta) = \theta_0 + \theta_1 x. \quad \left(\theta = (\theta_1, \theta_2)^\top \in \Theta = \mathbb{R}^2.\right)$$

- For demonstration, we treat this model as intractable (i.e., only generating output $y = r(x, \theta)$ is possible.)

Simple Synthetic Experiment: Setting

- The input space is $\mathcal{X} = \mathbb{R}$.

- True (unknown) regression function: **Third order polynomial**

$$R(x) = -x + x^3.$$

- Simulation model: **Linear function** (model misspecification!)

$$r(x, \theta) = \theta_0 + \theta_1 x. \quad \left(\theta = (\theta_1, \theta_2)^\top \in \Theta = \mathbb{R}^2.\right)$$

- For demonstration, we treat this model as intractable (i.e., only generating output $y = r(x, \theta)$ is possible.)

- Input distributions for training and test

$$q_0(x) = N(0.5, 0.5), \quad q_1(x) = N(0, 0.3).$$

Simple Synthetic Experiment: Setting

- The input space is $\mathcal{X} = \mathbb{R}$.

- True (unknown) regression function: **Third order polynomial**

$$R(x) = -x + x^3.$$

- Simulation model: **Linear function** (model misspecification!)

$$r(x, \theta) = \theta_0 + \theta_1 x. \quad \left(\theta = (\theta_1, \theta_2)^\top \in \Theta = \mathbb{R}^2.\right)$$

- For demonstration, we treat this model as intractable (i.e., only generating output $y = r(x, \theta)$ is possible.)

- Input distributions for training and test

$$q_0(x) = N(0.5, 0.5), \quad q_1(x) = N(0, 0.3).$$

- Prior on the parameter space: $\pi(\theta) = N(\mathbf{0}, 5I_2)$.

- Training data $(X_i, Y_i)_{i=1}^n$ with $n = 100$.

Simple Synthetic Experiment: Setting

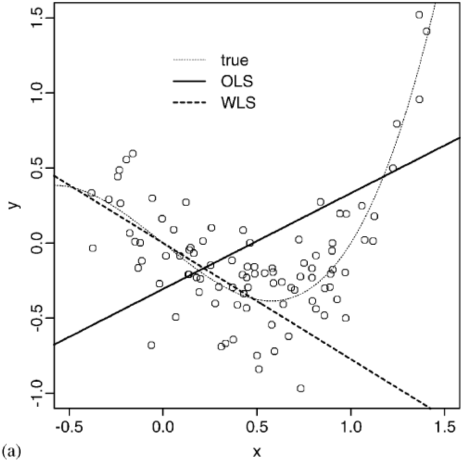


Figure 4: Fig.1(a) from [Shimodaira, 2000].

Simple Synthetic Experiments: Setting

- For the proposed method, we set the regularization constant to be $\varepsilon = 1.0$.

- We set the proposal distribution of MH to be Gaussian, whose variance is tuned so as to make the acceptance ratios to be about 20%, 40%, and 60%.

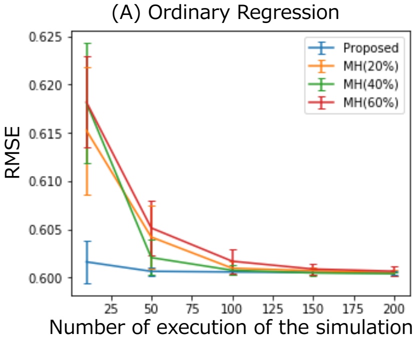
Simple Synthetic Experiments: Results

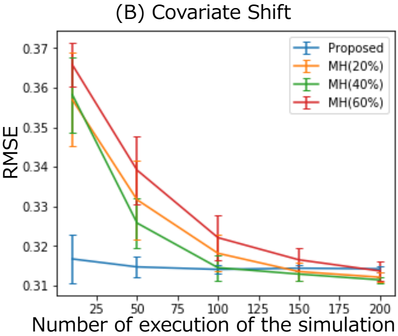
- (A) Ordinary regression ($q_1 = q_0$); (B) Covariate shift ($q_1 \neq q_0$).

- Horizontal axis: the number of simulations m.

- The proposed method performs better for smaller m.

- Promising result, since often simulations are computationally expensive.





Manufacturing Process Simulator (WITNESS Software)

- Computer simulator for a factory assembling certain products.

Manufacturing Process Simulator (WITNESS Software)

- Computer simulator for a factory assembling certain products.

- ASSEMBLY machine assembles three items (TOPS, BOTTOMS and SCREWS) into one product.

Manufacturing Process Simulator (WITNESS Software)

- Computer simulator for a factory assembling certain products.

- ASSEMBLY machine assembles three items (TOPS, BOTTOMS and SCREWS) into one product.

- INSPECTION machine inspects 4 such products at one time.

Manufacturing Process Simulator (WITNESS Software)

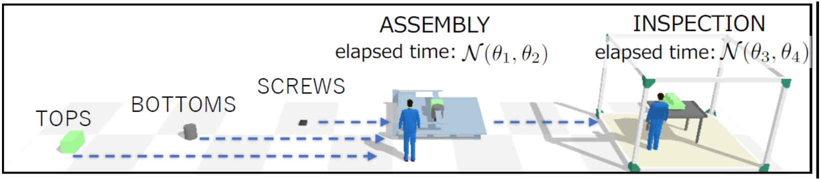
- Computer simulator for a factory assembling certain products.

- ASSEMBLY machine assembles three items (TOPS, BOTTOMS and SCREWS) into one product.

- INSPECTION machine inspects 4 such products at one time.

- There are 4 parameters $\theta = (\theta_1, \theta_2, \theta_3, \theta_4)^\top \in \mathbb{R}^n$ to be specified.

- θ_1 is the mean time spent in ASSEMBLY, and θ_3 in INSPECTION.



Manufacturing Process Simulator: Setting

- Input $x \in \mathcal{X} = (0, \infty)$: the number of products to be produced in one day.

Manufacturing Process Simulator: Setting

- Input $x \in \mathcal{X} = (0, \infty)$: the number of products to be produced in one day.

- Output $r(x, \theta) \in (0, \infty)$: the total time spent on producing all the products.

Manufacturing Process Simulator: Setting

- Input $x \in \mathcal{X} = (0, \infty)$: the number of products to be produced in one day.

- Output $r(x, \theta) \in (0, \infty)$: the total time spent on producing all the products.

- The data generating process $y(x) = R(x) + \varepsilon$ is defined as

$$R(x) = \begin{cases} r(x, (2, 0.5, 5, 1)^\top) & \text{if } x < 110 \\ r(x, (3.5, 0.5, 7, 1)^\top) & \text{if } x \geq 110. \end{cases}$$

- When $x > 110$, the production efficiency decreases because of overload of the workers.

Manufacturing Process Simulator: Setting

- Input $x \in \mathcal{X} = (0, \infty)$: the number of products to be produced in one day.

- Output $r(x, \theta) \in (0, \infty)$: the total time spent on producing all the products.

- The data generating process $y(x) = R(x) + \varepsilon$ is defined as

$$R(x) = \begin{cases} r(x, (2, 0.5, 5, 1)^\top) & \text{if } x < 110 \\ r(x, (3.5, 0.5, 7, 1)^\top) & \text{if } x \geq 110. \end{cases}$$

- When $x > 110$, the production efficiency decreases because of overload of the workers.

- Input distributions for training $q_0(x) = N(100, 10)$ and test $q_1(x) = N(120, 10)$.

Manufacturing Process Simulator: Setting

- Input $x \in \mathcal{X} = (0, \infty)$: the number of products to be produced in one day.

- Output $r(x, \theta) \in (0, \infty)$: the total time spent on producing all the products.

- The data generating process $y(x) = R(x) + \varepsilon$ is defined as

$$R(x) = \begin{cases} r(x, (2, 0.5, 5, 1)^\top) & \text{if } x < 110 \\ r(x, (3.5, 0.5, 7, 1)^\top) & \text{if } x \geq 110. \end{cases}$$

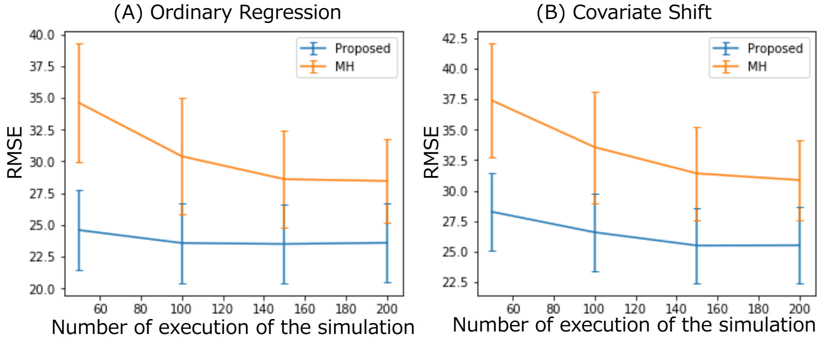
- When $x > 110$, the production efficiency decreases because of overload of the workers.

- Input distributions for training $q_0(x) = N(100, 10)$ and test $q_1(x) = N(120, 10)$.

- Prior $\pi(\theta)$ is uniform over $\Theta := [0, 5] \times [0, 2] \times [0, 10] \times [0, 2] \subset \mathbb{R}^4$.

Manufacturing Process Simulator: Results

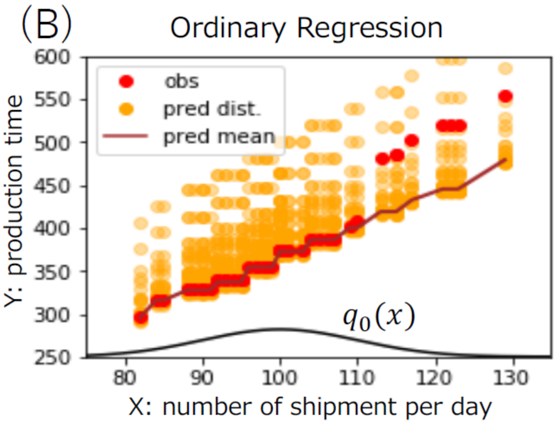
- The size of training data $(X_i, Y_i)_{i=1}^n$ is $n = 50$.
- (A) Ordinary regression ($q_1 = q_0$); (B) Covariate shift ($q_1 \neq q_0$).
- Horizontal axis: the number of simulations m.



Manufacturing Process Simulator: Results

- Results of our method *without* covariate shift adaptation ($q_1 = q_0$).

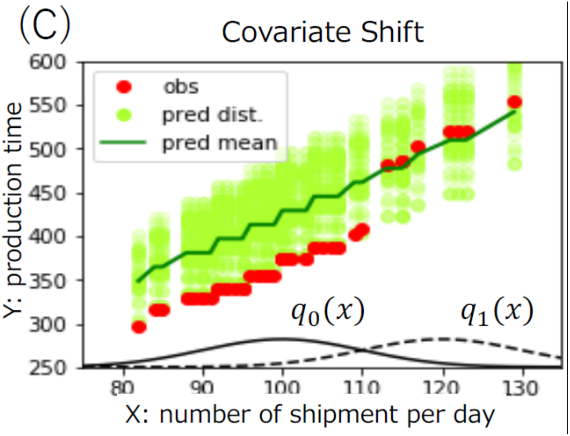
- Training data (red points), generated predictive outputs (orange) and their means (brown curve).



Manufacturing Process Simulator: Results

- Results of our method **with** covariate shift adaptation ($q_1 \neq q_0$).

- Training data (red points), generated predictive outputs (light green) and their means (green curve).

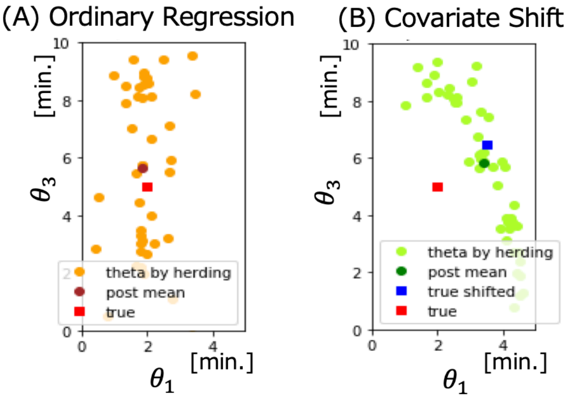


Manufacturing Process Simulator: Sensitivity Analysis

- Parameters generated from the posterior with the proposed method.

- (A) Ordinary regression ($q_1 = q_0$); (B) Covariate shift ($q_1 \neq q_0$).

- θ_1 (mean time for ASSEMBLY) is more sensitive than θ_3 (mean time for INSPECTION) for the outputs (total processing time).



Outline

Goal of the Theoretical Analysis

- We show that the estimate $\hat{\mu}_{\Theta|r^*}$ obtained from Kernel ABC and the importance-weighted kernel is an estimator of

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot, \theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta,$$

where $P_\pi(\theta|r^*)$ is the "target" posterior under prior $\pi(\theta)$.

Goal of the Theoretical Analysis

- We show that the estimate $\hat{\mu}_{\Theta|r^*}$ obtained from Kernel ABC and the importance-weighted kernel is an estimator of

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot,\theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta,$$

where $P_\pi(\theta|r^*)$ is the "target" posterior under prior $\pi(\theta)$.

- Recall that $r^* \in \mathbb{R}^n$ are "(unique) optimal predictions" defined by

$$r^* := r^n(\theta^*) = \left(r(X_1,\theta^*),\dots,r(X_n,\theta^*)\right)^\top, \quad \theta^* \in \Theta^*.$$

Goal of the Theoretical Analysis

- We show that the estimate $\hat{\mu}_{\Theta|r^*}$ obtained from Kernel ABC and the importance-weighted kernel is an estimator of

$$\mu_{\Theta|r^*} := \int k_\Theta(\cdot, \theta) dP_\pi(\theta|r^*) \in \mathcal{H}_\Theta,$$

where $P_\pi(\theta|r^*)$ is the "target" posterior under prior $\pi(\theta)$.

- Recall that $r^* \in \mathbb{R}^n$ are "(unique) optimal predictions" defined by

$$r^* := r^n(\theta^*) = \left(r(X_1, \theta^*), \ldots, r(X_n, \theta^*)\right)^\top, \quad \theta^* \in \Theta^*.$$

where Θ^* is the set of "optimal parameters" such that for all $\theta^* \in \Theta^*$:

$$\sum_{i=1}^n \beta(X_i)(Y_i - r(X_i, \theta^*))^2 = \min_{\theta \in \mathrm{supp}(\pi)} \sum_{i=1}^n \beta(X_i)(Y_i - r(X_i, \theta))^2.$$

Preliminaries: Covariance Operators

- Define joint random variables $(\vartheta, \boldsymbol{y}) \in \Theta \times \mathbb{R}^n$ by

$$\vartheta \sim \pi, \quad \boldsymbol{y} := r^n(\vartheta).$$

Preliminaries: Covariance Operators

- Define joint random variables $(\vartheta, \boldsymbol{y}) \in \Theta \times \mathbb{R}^n$ by

$$\vartheta \sim \pi, \quad \boldsymbol{y} := r^n(\vartheta).$$

- Covariance operators $C_{\vartheta \boldsymbol{y}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_\Theta$ and $C_{\boldsymbol{y}\boldsymbol{y}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_{\mathbb{R}^n}$ are then defined as

$$\begin{aligned} C_{\vartheta \boldsymbol{y}} f &:= \mathbb{E}[k_\Theta(\cdot, \vartheta) f(\boldsymbol{y})] \in \mathcal{H}_\Theta \quad (f \in \mathcal{H}_{\mathbb{R}^n}). \\ C_{\boldsymbol{y}\boldsymbol{y}} f &:= \mathbb{E}[k_{\mathbb{R}^n}(\cdot, \boldsymbol{y}) f(\boldsymbol{y})] \in \mathcal{H}_{\mathbb{R}^n} \quad (f \in \mathcal{H}_{\mathbb{R}^n}). \end{aligned}$$

Preliminaries: Covariance Operators

- Define joint random variables $(\vartheta, \boldsymbol{y}) \in \Theta \times \mathbb{R}^n$ by

$$\vartheta \sim \pi, \quad \boldsymbol{y} := r^n(\vartheta).$$

- Covariance operators $C_{\vartheta \boldsymbol{y}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_\Theta$ and $C_{\boldsymbol{y}\boldsymbol{y}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_{\mathbb{R}^n}$ are then defined as

$$\begin{aligned} C_{\vartheta \boldsymbol{y}} f &:= \mathbb{E}[k_\Theta(\cdot, \vartheta) f(\boldsymbol{y})] \in \mathcal{H}_\Theta \quad (f \in \mathcal{H}_{\mathbb{R}^n}). \\ C_{\boldsymbol{y}\boldsymbol{y}} f &:= \mathbb{E}[k_{\mathbb{R}^n}(\cdot, \boldsymbol{y}) f(\boldsymbol{y})] \in \mathcal{H}_{\mathbb{R}^n} \quad (f \in \mathcal{H}_{\mathbb{R}^n}). \end{aligned}$$

- $C_{\vartheta \boldsymbol{y}}$ encodes the joint distribution of $(\vartheta, \boldsymbol{y})$.

- $C_{\boldsymbol{y}\boldsymbol{y}}$ encodes the marginal distribution of $\boldsymbol{y}$.

Preliminaries: Empirical Covariance Operators

- Parameter-data pairs generated in Kernel ABC

$$(\bar{\theta}_j, \bar{Y}_j^n)_{j=1}^m = (\bar{\theta}_j, r^n(\bar{\theta}_j))_{j=1}^m \subset \Theta \times \mathbb{R}^n$$

are i.i.d. copies of $(\vartheta, \boldsymbol{y})$.

Preliminaries: Empirical Covariance Operators

- Parameter-data pairs generated in Kernel ABC

$$(\bar{\theta}_j, \bar{Y}_j^n)_{j=1}^m = (\bar{\theta}_j, r^n(\bar{\theta}_j))_{j=1}^m \subset \Theta \times \mathbb{R}^n$$

are i.i.d. copies of $(\vartheta, \boldsymbol{y})$.

- Thus empirical covariance operators $\hat{C}_{\vartheta \boldsymbol{y}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_\Theta$ and $\hat{C}_{\boldsymbol{yy}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_{\mathbb{R}^n}$ are defined as

$$\hat{C}_{\vartheta \boldsymbol{y}} f := \frac{1}{m} \sum_{j=1}^m k_\Theta(\cdot, \bar{\theta}_j) f(\bar{Y}_j^n) \quad (f \in \mathcal{H}_{\mathbb{R}^n}).$$

$$\hat{C}_{\boldsymbol{yy}} f := \frac{1}{m} \sum_{j=1}^m k_{\mathbb{R}^n}(\cdot, \bar{Y}_j^n) f(\bar{Y}_j^n) \quad (f \in \mathcal{H}_{\mathbb{R}^n}).$$

Preliminaries: Empirical Covariance Operators

- Parameter-data pairs generated in Kernel ABC

$$(\bar{\theta}_j, \bar{Y}_j^n)_{j=1}^m = (\bar{\theta}_j, r^n(\bar{\theta}_j))_{j=1}^m \subset \Theta \times \mathbb{R}^n$$

are i.i.d. copies of $(\vartheta, \boldsymbol{y})$.

- Thus empirical covariance operators $\hat{C}_{\vartheta \boldsymbol{y}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_\Theta$ and $\hat{C}_{\boldsymbol{yy}} : \mathcal{H}_{\mathbb{R}^n} \to \mathcal{H}_{\mathbb{R}^n}$ are defined as

$$\hat{C}_{\vartheta \boldsymbol{y}} f := \frac{1}{m} \sum_{j=1}^m k_\Theta(\cdot, \bar{\theta}_j) f(\bar{Y}_j^n) \quad (f \in \mathcal{H}_{\mathbb{R}^n}).$$

$$\hat{C}_{\boldsymbol{yy}} f := \frac{1}{m} \sum_{j=1}^m k_{\mathbb{R}^n}(\cdot, \bar{Y}_j^n) f(\bar{Y}_j^n) \quad (f \in \mathcal{H}_{\mathbb{R}^n}).$$

- $\hat{C}_{\vartheta \boldsymbol{y}}$ is an empirical approximation of $C_{\vartheta \boldsymbol{y}}$.

- $\hat{C}_{\boldsymbol{yy}}$ is an empirical approximation of $C_{\boldsymbol{yy}}$.

Kernel ABC via Empirical Covariance Operators

- Using the empirical covariance operators, the estimator $\hat{\mu}_{\Theta|r^*}$ can be written as

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

Kernel ABC via Empirical Covariance Operators

- Using the empirical covariance operators, the estimator $\hat{\mu}_{\Theta|r^*}$ can be written as

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

- This is how the estimator was originally proposed [Song et al., 2009, Nakagome et al., 2013].

- The estimator is known as conditional mean embedding, and has been studied extensively [Grünewälder et al., 2012, Fukumizu, 2015, Singh et al., 2019].

Kernel ABC via Empirical Covariance Operators

- However, existing theoretical results are **not** directly applicable to our case.

Kernel ABC via Empirical Covariance Operators

- However, existing theoretical results are **not** directly applicable to our case.

- This is because observed data $Y^n = (Y_1, \ldots, Y_n)^\top$ may not lie in the support of the distribution of $\boldsymbol{y} = r^n(\vartheta)$:

$$Y^n \notin \{(r(X_1, \theta), \ldots, r(X_n, \theta)^\top \mid \theta \in \mathrm{supp}(\pi)\},$$

Kernel ABC via Empirical Covariance Operators

- However, existing theoretical results are **not** directly applicable to our case.

- This is because observed data $Y^n = (Y_1, \ldots, Y_n)^\top$ may not lie in the support of the distribution of $\boldsymbol{y} = r^n(\vartheta)$:

$$Y^n \notin \{(r(X_1, \theta), \ldots, r(X_n, \theta)^\top \mid \theta \in \mathrm{supp}(\pi)\},$$

since that the true regression function $R(x)$ may not belong to the model class $\{r(x, \theta) \mid \theta \in \Theta\}$ (**model misspecificfation!**)

Kernel ABC via Empirical Covariance Operators

- However, existing theoretical results are **not** directly applicable to our case.

- This is because observed data $Y^n = (Y_1, \ldots, Y_n)^\top$ may not lie in the support of the distribution of $\boldsymbol{y} = r^n(\vartheta)$:

$$Y^n \notin \{(r(X_1, \theta), \ldots, r(X_n, \theta)^\top \mid \theta \in \mathrm{supp}(\pi)\},$$

since that the true regression function $R(x)$ may not belong to the model class $\{r(x, \theta) \mid \theta \in \Theta\}$ (**model misspecificfation!**)

- Existing theoretical results on conditional mean embeddings do not cover this situation.

- We provide a novel theoretical analysis of conditional mean embedding in this regard.

Subspace Spanned by the Simulator

- Consider a subset in $\mathbb{R}^n$ given by the simulator $r(x, \theta)$ and prior $\pi(\theta)$:

$$\begin{aligned}\mathrm{supp}(r^n\pi) &= \{r^n(\theta) \mid \theta \in \mathrm{supp}(\pi)\} \\ &= \{(r(X_1, \theta), \ldots, r(X_n, \theta)) \mid \theta \in \mathrm{supp}(\pi)\} \subset \mathbb{R}^n,\end{aligned}$$

which is the support of $\boldsymbol{y} = r^n(\vartheta)$.

Subspace Spanned by the Simulator

- Consider a subset in $\mathbb{R}^n$ given by the simulator $r(x, \theta)$ and prior $\pi(\theta)$:

$$\begin{aligned} \mathrm{supp}(r^n \pi) &= \{ r^n(\theta) \mid \theta \in \mathrm{supp}(\pi) \} \\ &= \{ (r(X_1, \theta), \ldots, r(X_n, \theta)) \mid \theta \in \mathrm{supp}(\pi) \} \subset \mathbb{R}^n, \end{aligned}$$

which is the support of $\boldsymbol{y} = r^n(\vartheta)$.

- Then define a Hilbert subspace of the RKHS $\mathcal{H}_{\mathbb{R}^n}$ by

$$\mathcal{H}_{\boldsymbol{y}} := \overline{\mathrm{span} \left\{ k_{\mathbb{R}^n}(\cdot, \tilde{Y}^n) \mid \tilde{Y}^n \in \mathrm{supp}(r^n \pi) \right\}} \subset \mathcal{H}_{\mathbb{R}^n},$$

Subspace Spanned by the Simulator

- Consider a subset in $\mathbb{R}^n$ given by the simulator $r(x, \theta)$ and prior $\pi(\theta)$:

$$\begin{aligned} \operatorname{supp}(r^n \pi) &= \{ r^n(\theta) \mid \theta \in \operatorname{supp}(\pi) \} \\ &= \{ (r(X_1, \theta), \ldots, r(X_n, \theta)) \mid \theta \in \operatorname{supp}(\pi) \} \subset \mathbb{R}^n, \end{aligned}$$

which is the support of $\boldsymbol{y} = r^n(\vartheta)$.

- Then define a Hilbert subspace of the RKHS $\mathcal{H}_{\mathbb{R}^n}$ by

$$\mathcal{H}_{\boldsymbol{y}} := \overline{\operatorname{span}\left\{ k_{\mathbb{R}^n}(\cdot, \tilde{Y}^n) \mid \tilde{Y}^n \in \operatorname{supp}(r^n \pi) \right\}} \subset \mathcal{H}_{\mathbb{R}^n},$$

- We will call this the "subspace spanned by the simulator $r(x, \theta)$ (and the prior $\pi(\theta)$)."

Orthogonal Projection onto the Subspace

- For the observed data Y^n, consider the orthogonal projection of the feature vector $k_{\mathbb{R}^n}(\cdot, Y^n) \in \mathcal{H}_{\mathbb{R}^n}$ onto $\mathcal{H}_{\boldsymbol{y}}$:

$$h^* := \arg\min_{h \in \mathcal{H}_{\boldsymbol{y}}} \|h - k_{\mathbb{R}^n}(\cdot, Y^n)\|_{\mathcal{H}_{\mathbb{R}^n}}.$$

Orthogonal Projection onto the Subspace

- For the observed data Y^n, consider the orthogonal projection of the feature vector $k_{\mathbb{R}^n}(\cdot, Y^n) \in \mathcal{H}_{\mathbb{R}^n}$ onto $\mathcal{H}_{\boldsymbol{y}}$:

$$h^* := \arg\min_{h \in \mathcal{H}_{\boldsymbol{y}}} \|h - k_{\mathbb{R}^n}(\cdot, Y^n)\|_{\mathcal{H}_{\mathbb{R}^n}}.$$

- Then $k_{\mathbb{R}^n}(\cdot, Y^n)$ can be written as

$$k_{\mathbb{R}^n}(\cdot, Y^n) = h^* + h_\perp,$$

where $h_\perp \in \mathcal{H}_{\mathbb{R}^n}$ is orthogonal to $\mathcal{H}_{\boldsymbol{y}}$.

Population Conditional Mean Embedding via Projection

- Note that the Kernel ABC estimator

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

is an approximation of the corresponding population expression

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

Population Conditional Mean Embedding via Projection

- Note that the Kernel ABC estimator

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

is an approximation of the corresponding population expression

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

Lemma (Population Expression via Projection)

- *Assume k_Θ is bounded and continuous, and $0 < \beta(X_i) < \infty$ for $i = 1, \ldots, n$.*

- *Let h^* be the orthogonal projection of $k_{\mathbb{R}^n}(\cdot, Y^n)$ onto the subspace $\mathcal{H}_{\boldsymbol{y}}$ spanned by the simulator $r(x, \theta)$.*

Population Conditional Mean Embedding via Projection

- Note that the Kernel ABC estimator

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{y}\boldsymbol{y}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

is an approximation of the corresponding population expression

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{y}\boldsymbol{y}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n).$$

Lemma (Population Expression via Projection)

- Assume k_Θ is bounded and continuous, and $0 < \beta(X_i) < \infty$ for $i = 1, \ldots, n$.

- Let h^* be the orthogonal projection of $k_{\mathbb{R}^n}(\cdot, Y^n)$ onto the subspace $\mathcal{H}_{\boldsymbol{y}}$ spanned by the simulator $r(x, \theta)$.

- Then we have

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{y}\boldsymbol{y}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n) = C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{y}\boldsymbol{y}} + \varepsilon I)^{-1} h^*.$$

Identifiability Assumption

- We make the following identifiability assumption for further analysis.

Identifiability Assumption

- We make the following identifiability assumption for further analysis.

Assumption (Identifiability)

- Let h^* be the orthogonal projection of $k_{\mathbb{R}^n}(\cdot, Y^n)$ onto the subspace $\mathcal{H}_{\mathbf{y}}$ spanned by the simulator $r(x, \theta)$.

Identifiability Assumption

- We make the following identifiability assumption for further analysis.

Assumption (Identifiability)

- Let h^* be the orthogonal projection of $k_{\mathbb{R}^n}(\cdot, Y^n)$ onto the subspace $\mathcal{H}_{\mathbf{y}}$ spanned by the simulator $r(x, \theta)$.

- We assume there exists

$$\tilde{Y}^n \in \mathrm{supp}(r^n \pi) = \left\{ (r(X_1, \theta), \ldots, r(X_n, \theta))^\top \mid \theta \in \mathrm{supp}(\pi) \right\}$$

such that

$$k_{\mathbb{R}^n}(\cdot, \tilde{Y}^n) = h^*.$$

Conditional Mean Embedding under Misspecification

- Recall that $r^* \in \mathbb{R}^n$ are "optimal predictions" defined by

$$r^* := r^n(\theta^*) = \left(r(X_1, \theta^*), \ldots, r(X_n, \theta^*)\right)^\top, \quad \theta^* \in \Theta^*.$$

where Θ^* is the set of "optimal parameters" minimizing the weighted squared loss.

Conditional Mean Embedding under Misspecification

- Recall that $r^* \in \mathbb{R}^n$ are "optimal predictions" defined by

$$r^* := r^n(\theta^*) = (r(X_1, \theta^*), \ldots, r(X_n, \theta^*))^\top, \quad \theta^* \in \Theta^*.$$

where Θ^* is the set of "optimal parameters" minimizing the weighted squared loss.

Theorem (Population Expression via Optimal Predictions)

- *Assume k_Θ is bounded and continuous, and $0 < \beta(X_i) < \infty$ for $i = 1, \ldots, n$.*

- *Suppose the Identifiability Assumption holds.*

Conditional Mean Embedding under Misspecification

- Recall that $r^* \in \mathbb{R}^n$ are "optimal predictions" defined by

$$r^* := r^n(\theta^*) = (r(X_1, \theta^*), \ldots, r(X_n, \theta^*))^\top, \quad \theta^* \in \Theta^*.$$

where Θ^* is the set of "optimal parameters" minimizing the weighted squared loss.

Theorem (Population Expression via Optimal Predictions)

- *Assume k_Θ is bounded and continuous, and $0 < \beta(X_i) < \infty$ for $i = 1, \ldots, n$.*

- *Suppose the Identifiability Assumption holds.*

- *Then we have*

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n) = C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, r^*).$$

Convergence Result

- As $m \to \infty$ (num. of simulations), the Kernel ABC estimator

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n)$$

converges to the population version

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n),$$

Convergence Result

- As $m \to \infty$ (num. of simulations), the Kernel ABC estimator

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n)$$

converges to the population version

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n),$$

which is equal to

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, r^*).$$

Convergence Result

- As $m \to \infty$ (num. of simulations), the Kernel ABC estimator

$$\hat{\mu}_{\Theta|r^*} = \hat{C}_{\vartheta \boldsymbol{y}}(\hat{C}_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n)$$

converges to the population version

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, Y^n),$$

which is equal to

$$C_{\vartheta \boldsymbol{y}}(C_{\boldsymbol{yy}} + \varepsilon I)^{-1} k_{\mathbb{R}^n}(\cdot, r^*).$$

On the other hand, as $\varepsilon \to 0$, this expression converges to

$$\mu_{\Theta|r^*} := \int k_{\Theta}(\cdot, \theta) dP_{\pi}(\theta | r^*),$$

where $P_{\pi}(\theta|r^*)$ is the "target" posterior under prior $\pi(\theta)$.

Convergence Result

Theorem

- *Suppose the assumptions made so far to hold.*

Convergence Result

Theorem

- *Suppose the assumptions made so far to hold.*

- *Assume that the eigenvalues* $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ *of* $C_{\boldsymbol{yy}}$ *satisfy*

$$\lambda_i \leq \beta i^{-b}, \quad \forall i \in \mathbb{N}$$

for some constants $\beta > 0$ *and* $b > 1$.

Convergence Result

Theorem

- *Suppose the assumptions made so far to hold.*

- *Assume that the eigenvalues* $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ *of* $C_{\boldsymbol{yy}}$ *satisfy*

$$\lambda_i \leq \beta i^{-b}, \quad \forall i \in \mathbb{N}$$

for some constants $\beta > 0$ *and* $b > 1$*.*

- *For any fixed* $C > 0$*, set the regularization constant of of* $\hat{\mu}_{\Theta|r^*}$ *as* $\varepsilon := \varepsilon_m := Cm^{-\frac{b}{1+4b}}$*.*

Convergence Result

Theorem

- *Suppose the assumptions made so far to hold.*
- *Assume that the eigenvalues* $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$ *of* $C_{\boldsymbol{yy}}$ *satisfy*

$$\lambda_i \leq \beta i^{-b}, \quad \forall i \in \mathbb{N}$$

for some constants $\beta > 0$ *and* $b > 1$.

- *For any fixed* $C > 0$, *set the regularization constant of of* $\hat{\mu}_{\Theta|r^*}$ *as* $\varepsilon := \varepsilon_m := Cm^{-\frac{b}{1+4b}}$.
- *Then, under an additional technical condition, we have*

$$\left\|\hat{\mu}_{\Theta|r^*} - \mu_{\Theta|r^*}\right\|_{\mathcal{H}_\Theta} = O_p\left(m^{-\frac{b}{1+4b}}\right) \ (m \to \infty).$$

Outline

Conclusions

- Covariate shift is ubiquitous in applications of computer simulation.

Conclusions

- Covariate shift is ubiquitous in applications of computer simulation.

- We proposed a kernel-based method for simulator calibration under this setting.

Conclusions

- Covariate shift is ubiquitous in applications of computer simulation.

- We proposed a kernel-based method for simulator calibration under this setting.

- We also contribute to the kernel literature, by providing a novel theoretical analysis of conditional mean embedding.

Conclusions

- Covariate shift is ubiquitous in applications of computer simulation.

- We proposed a kernel-based method for simulator calibration under this setting.

- We also contribute to the kernel literature, by providing a novel theoretical analysis of conditional mean embedding.

- Future work includes a formal analysis of the identifiability assumption.

Collaborators

- Keiichi Kisamori (NEC/AIST, Japan)
- Keisuke Yamazaki (AIST, Japan)

Bach, F., Lacoste-Julien, S., and Obozinski, G. (2012).
On the equivalence between herding and conditional gradient algorithms.
In *Proceedings of the 29th International Conference on Machine Learning (ICML2012)*, pages 1359–1366.

Chang, J. T. and Pollard, D. (1997).
Conditioning as disintegration.
Statistica Neerlandica, 51:287–317.

Chen, Y., Welling, M., and Smola, A. (2010).
Supersamples from kernel-herding.
In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI 2010)*, pages 109–116.

Dick, J., Kuo, F. Y., and Sloan, I. H. (2013).
High dimensional numerical integration - the Quasi-Monte Carlo way.
Acta Numerica, 22(133-288).

Fukumizu, K. (2015).

Nonparametric Bayesian Inference with Kernel Mean Embedding.

In *Modern Methodology and Applications in Spatial-Temporal Modeling*, Springer Briefs in Statistics. Springer, Tokyo.

Fukumizu, K., Gretton, A., Sun, X., and Schölkopf, B. (2008).
Kernel measures of conditional dependence.
In *Advances in Neural Information Processing Systems 20*, pages 489–496.

Grünewälder, S., Lever, G., Baldassarre, L., Patterson, S., Gretton, A., and Pontil, M. (2012).
Conditional mean embeddings as regressors.
In *Proceedings of the 29th International Conference on Machine Learning (ICML2012)*, pages 1823–1830.

Muandet, K., Fukumizu, K., Sriperumbudur, B. K., and Schölkopf, B. (2017).
Kernel mean embedding of distributions : A review and beyond.
Foundations and Trends in Machine Learning, 10(1–2):1–141.

Nakagome, S., Fukumizu, K., and Mano, S. (2013).
Kernel approximate Bayesian computation in population genetic inferences.
Statistical Applications in Genetics and Molecular Biology, 12(6):667–678.

Shimodaira, H. (2000).
Improving predictive inference under covariate shift by weighting the log-likelihood function.
Journal of Statistical Planning and Inference, 90(2):227–244.

Singh, R., Sahani, M., and Gretton, A. (2019).
Kernel instrumental variable regression.
In *NeurIPS*.

Smola, A., Gretton, A., Song, L., and Schölkopf, B. (2007).
A Hilbert space embedding for distributions.
In *Proceedings of the International Conference on Algorithmic Learning Theory*, volume 4754, pages 13–31. Springer.

Song, L., Huang, J., Smola, A., and Fukumizu, K. (2009).

Hilbert space embeddings of conditional distributions with applications to dynamical systems.
In *Proceedings of the 26th International Conference on Machine Learning (ICML2009)*, pages 961–968.

Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., and Lanckriet, G. R. (2010).
Hilbert space embeddings and metrics on probability measures.
Jounal of Machine Learning Research, 11:1517–1561.