

Multifidelity Gaussian process regression for solving nonlinear partial differential equations

Machine Learning + Simulation, Fall Workshop

O. Roustant, INSA Toulouse

Joint work with: F.-Z. El Boukkouri (INSA T.) & J. Garnier (X)

Slides contributors: F.-Z. El Boukkouri, I. Henderson (ISAE Supareo).

13 October 2026

Outline

1 Introduction

→ a quick picture of PDE solving with kernel methods

2 Learning functions with kernel methods

→ basics on Gaussian process regression and RKHS
and how to account for with linear (e.g., derivative) information

3 Solving (nonlinear) PDEs with kernel methods

→ assuming that a 'good' kernel is provided

4 Learning kernels within a multifidelity context

5 Perspectives

Why Gaussian processes?

GPs are a good complement to deep-learning methods, and widely used in machine learning/uncertainty quantification (UQ). The reasons are many:

- They work with scarce data (and can be adapted to handle huge datasets)
- They provide uncertainty quantification (confidence/credibility intervals)
- They are suitable for approximation problems (they extend splines)
- They are parameterized by two functions (trend, kernel)
 - ▶ highly flexible, they can be defined on general spaces
- They naturally handle linear constraints (e.g. derivatives)
- They are mathematically sounded
 - ▶ at the crossroad between three well-established theories: random processes, reproducing kernel Hilbert spaces (RKHS), positive definite functions.
- ...

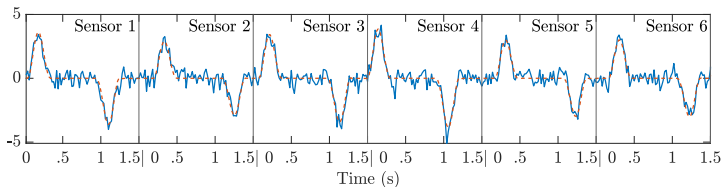
Some applications w.r.t. PDEs

In general, GP models are relevant for **surrogate modelling** of expensive computer codes → sensitivity analysis, UQ, ...

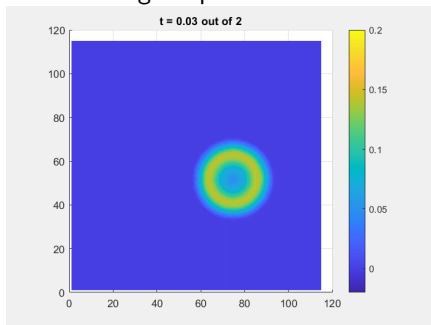
In the PDE setting, GPs are also relevant for **inverse problems** :

- Efficient **estimation of the solution of a PDE** given scattered pointwise measurements of the solution;
- **Estimation of the physical parameters of the PDE** by calibrating the hyperparameters of the kernel method;
- **Better uncertainty quantification** on the estimated sol. of a PDE ;

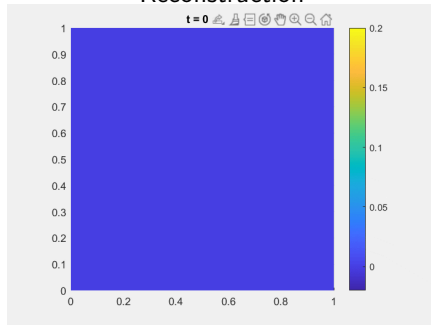
Ex 1 - Estimation of the solution of the wave equation



Original phenomenon



Reconstruction



Ex 2 - Better uncertainty quantification : 1D heat equation

Estimate the sol. of $\partial_t u = D \partial_{xx}^2 u$, $u(0, x) = u_0(x)$ where u_0 is unknown.

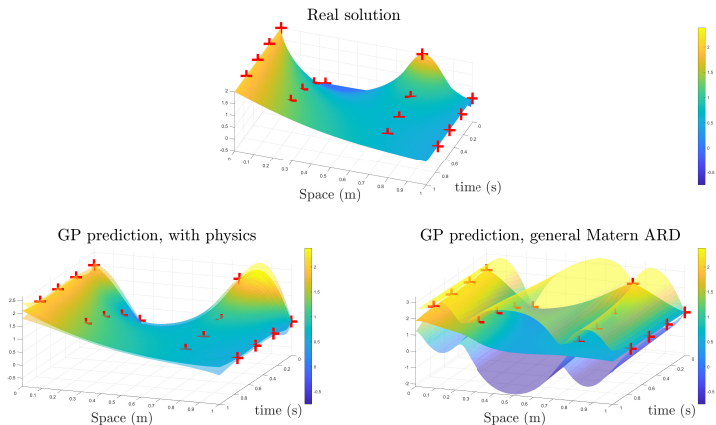


Figure: Red : measurement points. The prediction (and any realization) of the physics-informed GP verifies the equation **everywhere**.

Ex 3 - Solving the (nonlinear) Burgers' equation

Estimate the solution of $\partial_t u + \alpha(x)u\partial_x u - \nu(x)\partial_x^2 u = 0$ on $\Omega = (-1, 1) \times (0, 1]$, with $u(0, x) = -\sin(\pi x)$ and $u(t, -1) = u(t, 1) = 0$.

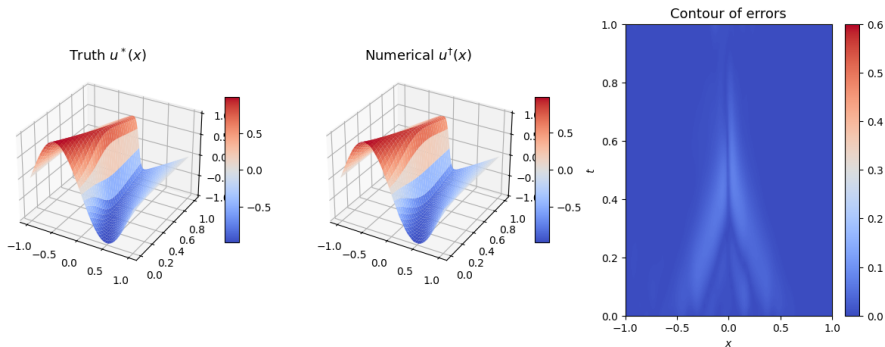


Figure: The estimated solution verifies the equation on a finite set of 'collocation' points (1000 in Ω , 201 in $\partial\Omega$) uniformly sampled. No measurement points have been used (but could be added to improve accuracy).

Comments

- Solving **linear PDEs with constant coef.** with GPs can be done using
 - ① a linear representation of the solution (relying e.g., on the Green function)
 - ② a well chosen prior (here kernel) for the initial condition (or forcing terms)→ **only tractable in particular cases** (e.g., the ones illustrated before)

Comments

- Solving **linear PDEs with constant coef.** with GPs can be done using
 - ① a linear representation of the solution (relying e.g., on the Green function)
 - ② a well chosen prior (here kernel) for the initial condition (or forcing terms)→ **only tractable in particular cases** (e.g., the ones illustrated before)
- In all **the other cases**, the collocation approach, based on the RKHS viewpoint, can be used
→ **we loose uncertainty quantification**

Comments

- Solving **linear PDEs with constant coef.** with GPs can be done using
 - ① a linear representation of the solution (relying e.g., on the Green function)
 - ② a well chosen prior (here kernel) for the initial condition (or forcing terms)→ **only tractable in particular cases** (e.g., the ones illustrated before)
- In all **the other cases**, the collocation approach, based on the RKHS viewpoint, can be used
→ **we loose uncertainty quantification**

A detailed presentation can be find on the Github deposit *Physics informed Gaussian process training*, with both theory and computer labs.

Outline

1 Introduction

→ a quick picture of PDE solving with kernel methods

2 Learning functions with kernel methods

→ basics on Gaussian process regression and RKHS
and how to account for with linear (e.g., derivative) information

3 Solving (nonlinear) PDEs with kernel methods

→ assuming that a 'good' kernel is provided

4 Learning kernels within a multifidelity context

5 Perspectives

Gaussian processes

Definition

A random process Y defined on $\mathbb{X}$ is a *Gaussian process* (GP) if for all locations $x_1, \dots, x_n \in \mathbb{X}$, the random vector $(Y(x_1), \dots, Y(x_n))$ is a Gaussian vector.

The law of such random vectors is fully characterized by two **functions**:

- 1 The **mean** $m(x) = \mathbb{E}(Y(x))$, representing a **trend**
- 2 The **kernel** $k(x, x') = \text{Cov}(Y(x), Y(x'))$, rep. the **spatial dependence**

We will denote $Y \sim GP(m, k)$.

Gaussian processes

Definition

A random process Y defined on $\mathbb{X}$ is a *Gaussian process* (GP) if for all locations $x_1, \dots, x_n \in \mathbb{X}$, the random vector $(Y(x_1), \dots, Y(x_n))$ is a Gaussian vector.

The law of such random vectors is fully characterized by two **functions**:

- 1 The **mean** $m(x) = \mathbb{E}(Y(x))$, representing a **trend**
- 2 The **kernel** $k(x, x') = \text{Cov}(Y(x), Y(x'))$, rep. the **spatial dependence**

We will denote $Y \sim GP(m, k)$.

Two key stability properties

- 1 a GP is stable by conditioning
- 2 a GP is stable by linear mapping

Interpolating/approximating functions with GPs: GP regression

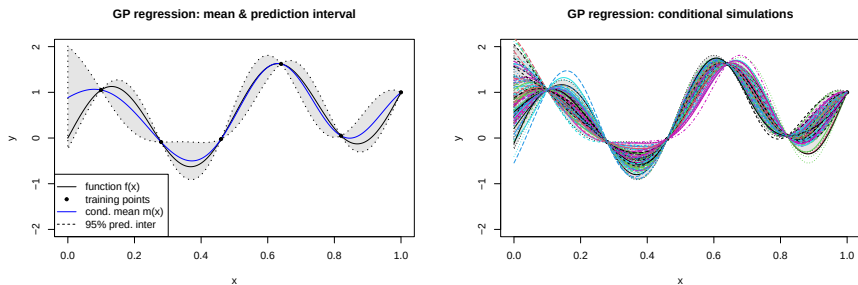


Figure: Illustration of GP regression, with the squared exponential kernel $k(s, t) = \exp(-25(s - t)^2)$.

GP regression with additional linear constraints

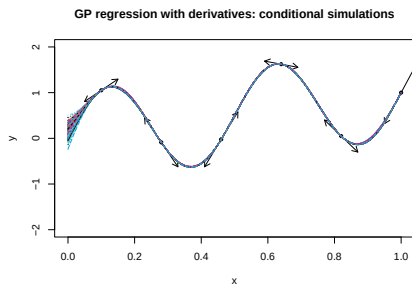
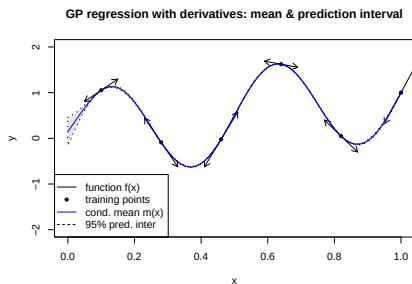


Figure: Illustration of GP regression knowing both function and derivative values, with the same kernel as in previous slide.

GP regression¹: technical details

Let f be a function we wish to interpolate at some data points (x_i, y_i) .

Here we view f as one realization of a GP Y with mean m and kernel k

Then, interpolating is equivalent to condition Y on $Y(x_i) = y_i$ ($i = 1, \dots, n$)

In a Gaussian setting, the conditional process is known explicitly:

$Y|\{Y(x_i) = y_i, i = 1, \dots, n\}$ is a GP with mean m_c and kernel k_c , with:

$$\begin{aligned} m_c(x) &= m(x) + k(x, X)k(X, X)^{-1}(y - m(X)) \\ k_c(x, x') &= k(x, x') - k(x, X)k(X, X)^{-1}k(X, x') \end{aligned}$$

Notations:

- $y = (y_1, \dots, y_n)^\top$
- $m(X) = (m(x_1), \dots, m(x_n))^\top$
- $k(x, X) = (k(x_1, x), \dots, k(x_n, x))$, $k(X, x) = k(x, X)^\top$
- $k(X, X) = (k(x_i, x_j))_{1 \leq i, j \leq n}$.

¹[Sacks et al., 1989, Rasmussen and Williams, 2006]

RKHS² (definition)

Reproducing kernel Hilbert spaces (RKHS) are Hilbert spaces of functions $f : \mathbb{X} \rightarrow \mathbb{R}$ such that for all $x \in \mathbb{X}$ the evaluation $f \mapsto f(x)$ is continuous.

This automatically gives a kernel. Indeed, by Riesz theorem, for all $x \in \mathbb{X}$, there exists a unique $k_x \in \mathcal{H}$ s.t. for all $f \in \mathcal{H}$,

$$f(x) = \langle f, k_x \rangle_{\mathcal{H}} \quad (*)$$

Define $k : \mathbb{X} \times \mathbb{X} \rightarrow \mathbb{R}$ by $k(x, x') = k_x(x')$. Choosing $f = k_{x'}$ in $(*)$ gives

$$\langle k(x, \cdot), k(x', \cdot) \rangle_{\mathcal{H}} = k(x, x') \quad (**)$$

which shows in particular that k is a kernel.

$(*)$, $(**)$ are called **reproducing properties** and k the **reproducing kernel** of $\mathcal{H}$.

²[Aronszajn, 1950, Berlinet and Thomas-Agnan, 2004, Saitoh and Sawano, 2016]

RKHS (equivalence with psd functions)

We have just seen that if $\mathcal{H}$ is a RKHS, then it provides a kernel k .

Moore-Aronszajn theorem

Conversely, if k is a kernel, one can construct a unique RKHS $\mathcal{H}_k$ with kernel k . This RKHS is given by

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

with inner product defined on the $k(x, \cdot)$'s by $\langle k(x, \cdot), k(y, \cdot) \rangle := k(x, y)$, and extended to $\mathcal{H}_k$ by linearity and continuity.

RKHS (examples)

- Finite-dimensional Hilbert spaces (i.e. Euclidean spaces)
- Infinite dimensional Hilbert spaces **with minimal regularity**
 - ▶ If $\mathbb{X}$ is a bounded interval, $L^2(\mathbb{X})$ is NOT a RKHS
 - ▶ When $X = \mathbb{R}^d$, the Sobolev space $H^m(\mathbb{X})$ is a RKHS iff $m > d/2$
 $\rightarrow H^1(\mathbb{R}), H^2(\mathbb{R}), \dots$ are RKHS
 - ▶ Matérn kernels correspond to Sobolev spaces with specific norms
[Durrande et al., 2016]

Gaussian processes, splines and RKHS

Correspondence between interpolation spline and GP conditional mean

The interpolation spline is defined by the functional problem

$$(*) \quad \min_{h \in \mathcal{H}} \|h\|_{\mathcal{H}} \quad \text{s.t.} \quad h(x_i) = y_i, \quad i = 1, \dots, n$$

If $\mathcal{H}$ is the RKHS of kernel k , and if $k(X, X) = (k(x_i, x_j))_{1 \leq i, j \leq n}$ is invertible, $(*)$ has a unique solution in the finite dimensional space spanned by the $k(x_i, \cdot)$:

$$\begin{aligned} h^*(x) &= \mathbb{E}[Y(x) | Y(x_i) = y_i, i = 1, \dots, n] \\ &= k(x, X)k(X, X)^{-1}y \end{aligned}$$

where $Y \sim GP(0, k)$, and $\|h^*\|_{\mathcal{H}}^2 = y^\top k(X, X)^{-1}y$.

The first part (reduction to finite dimension) is known as *Representer theorem*, and comes from [Aronszajn, 1950]. The correspondence with GP can be found in [Kimeldorf and Wahba, 1970].

Extension to linear constraints

Solving interpolation problem with RKHS extends to linear constraints.

This is the functional viewpoint of GP regression. As, an example:

Representer theorem with derivatives

Under regularity assumptions on k , the minimization problem

$$(*) \quad \min_{h \in \mathcal{H}} \|h\|_{\mathcal{H}} \quad s.t. \quad h(x_i) = y_i, h'(x_i) = d_i \quad (i = 1, \dots, n)$$

can be solved in the finite-dim. space spanned by the $k(x_i, \cdot)$ and the $\partial_x k(x_i, \cdot)$

Furthermore, if $\begin{bmatrix} k(X, X) & \partial_y k(X, X) \\ \partial_y k(X, X)^\top & \partial_{xy}^2 k(X, X) \end{bmatrix} = \mathbb{Cov} \begin{pmatrix} Y(X) \\ Y'(X) \end{pmatrix}$ is invertible, then

(*) has a unique solution given by

$$\begin{aligned} h^*(x) &= \mathbb{E}(Y | \{Y(X) = y, Y'(X) = d\}) \\ &= \begin{bmatrix} k(x, X) & \partial_y k(x, X) \end{bmatrix} \begin{bmatrix} k(X, X) & \partial_y k(X, X) \\ \partial_y k(X, X)^\top & \partial_{xy}^2 k(X, X) \end{bmatrix}^{-1} \begin{bmatrix} y \\ d \end{bmatrix} \end{aligned}$$

Outline

1 Introduction

→ a quick picture of PDE solving with kernel methods

2 Learning functions with kernel methods

→ basics on Gaussian process regression and RKHS
and how to account for with linear (e.g., derivative) information

3 Solving (nonlinear) PDEs with kernel methods

→ assuming that a 'good' kernel is provided

4 Learning kernels within a multifidelity context

5 Perspectives

Solving non-linear PDEs with GPs

The idea is to use the functional view of GPs, i.e. RKHS.

The next slides are based on [[Chen et al., 2021](#)].

Solving non-linear PDEs with GPs

As an example, we focus on the PDE

$$\begin{aligned} -\Delta u(x) + (u(x))^3 &= f(x) & \forall x \in \Omega \\ u(x) &= g(x) & \forall x \in \partial\Omega \end{aligned}$$

where Ω is an open set of $\mathbb{R}^d$ and $f : \Omega \rightarrow \mathbb{R}$, $g : \partial\Omega \rightarrow \mathbb{R}$ are continuous functions.

The aim is to predict $u(x)$ knowing values of f on Ω and values of g on $\partial\Omega$.

Solving non-linear PDEs with GPs

Let $\mathcal{H}$ be a RKHS with kernel k .

Let us consider the minimization problem

$$\min_{u \in \mathcal{H}} \|u\|_{\mathcal{H}}^2 \quad \text{s.t.} \quad \begin{cases} -\Delta u(x_m) + (u(x_m))^3 = f(x_m) & m \in \mathcal{C}_{\Omega} = \{1, \dots, M_{\Omega}\} \\ u(x_m) = g(x_m) & m \in \mathcal{C}_{\partial\Omega} = \{M_{\Omega} + 1, \dots, M\} \end{cases}$$

where $x_1, \dots, x_M$ are fixed points (collocation points).

As the PDE is non-linear, we cannot use the representer theorem. The trick is to recast this problem as the bilevel optimization problem (denote $\mathcal{C} = \mathcal{C}_{\Omega} \cup \mathcal{C}_{\partial\Omega}$):

$$\min_{(z_m^{(1)}, z_m^{(2)})_m} \left(\min_{\substack{u \in \mathcal{H} \\ u(x_m) = z_m^{(1)}, m \in \mathcal{C} \\ \Delta u(x_m) = z_m^{(2)}, m \in \mathcal{C}_{\Omega}}} \|u\|_{\mathcal{H}}^2 \right) \quad \text{s.t.} \quad \begin{cases} -z_m^{(2)} + (z_m^{(1)})^3 = f(x_m) & m \in \mathcal{C}_{\Omega} \\ z_m^{(1)} = g(x_m) & m \in \mathcal{C}_{\partial\Omega} \end{cases}$$

Solving non-linear PDEs with GPs

Inner optimization

$$\min_{u \in \mathcal{H}} \|u\|_{\mathcal{H}}^2 \quad \text{s.t.} \quad \begin{cases} u(x_m) = z_m^{(1)} & m \in \mathcal{C} \\ \Delta u(x_m) = z_m^{(2)} & m \in \mathcal{C}_\Omega \end{cases}$$

where all $z_m^{(1)}, z_m^{(2)}$ are real numbers.

As Δ is a linear operator, we can use a version of the representer theorem. Let $Y \sim GP(0, k)$. The solution is unique and given by

$$u_z^*(x) = \mathbb{E} \left(Y(x) | \{Y(x_m) = z_m^{(1)}\}_{m \in \mathcal{C}}, \{\Delta Y(x_m) = z_m^{(2)}\}_{m \in \mathcal{C}_\Omega} \right)$$

and the value at minimum is quadratic w.r.t. $z = (z_1^{(1)}, \dots, z_M^{(1)}, z_1^{(2)}, \dots, z_{M_\Omega}^{(2)})$

$$\|u_z^*\|_{\mathcal{H}}^2 = z^\top K(X, X)^{-1} z$$

where $K(X, X) = \text{Cov}(Y(x_1), \dots, Y(x_M), \Delta Y(x_1), \dots, \Delta Y(x_{M_\Omega}))$.

Solving non-linear PDEs with GPs

Outer minimization problem

Step 2. Minimize $\|u_z^\star\|_{\mathcal{H}}^2 = z^\top K(X, X)^{-1} z$ w.r.t $(z_m^{(1)}, z_m^{(2)})_{m \in \mathcal{C}_\Omega} \in \mathbb{R}^{2|\mathcal{C}_\Omega|}$
subject to non-linear constraints inherited from the PDE

$$\min_{(z_m^{(1)}, z_m^{(2)})_{m \in \mathcal{C}_\Omega}} z^\top K(X, X)^{-1} z \quad \text{s.t.} \quad \begin{cases} -z_m^{(2)} + (z_m^{(1)})^3 = f(x_m) & m \in \mathcal{C}_\Omega \\ z_m^{(1)} = g(x_m) & m \in \mathcal{C}_{\partial\Omega} \end{cases}$$

This is a **finite-dimensional quadratic optimization problem**.

The authors prove the convergence of the solution of this problem to the solution u of the PDE when the number of collocation points M goes to infinity, *under the assumption that $u \in \mathcal{H}$ (and other technical ones)*
→ *requires to choose the kernel k such that $u \in \mathcal{H}$*

Solving non-linear PDEs with GPs

Notice that the constraints can often be removed by substitution.

$$\min_{(z_m^{(1)}, z_m^{(2)})_{m \in \mathcal{C}_\Omega}} z^\top K(X, X)^{-1} z \quad \text{s.t.} \quad \begin{cases} -z_m^{(2)} + (z_m^{(1)})^3 = f(x_m) & m \in \mathcal{C}_\Omega \\ z_m^{(1)} = g(x_m) & m \in \mathcal{C}_{\partial\Omega} \end{cases}$$

$\Downarrow$ rewrite $z_m^{(2)}$ in function of $z_m^{(1)}$

$$\min_{(z_m^{(1)})_{m \in \mathcal{C}_\Omega}} v^\top K(X, X)^{-1} v \quad \text{with} \quad v = \begin{bmatrix} (z_m^{(1)})_{m \in \mathcal{C}_\Omega} \\ (g(x_m))_{m \in \mathcal{C}_{\partial\Omega}} \\ ((z_m^{(1)})^3 - f(x_m))_{m \in \mathcal{C}_\Omega} \end{bmatrix}$$

The outer problem is an unconstrained non-linear optimization problem.

Example: the Burgers' equation

We consider the Burgers' equation with parameters $\nu = 0.02$ and $\alpha = 1$:

$$\begin{aligned}\partial_t u + \alpha u \partial_x u - \nu \partial_x^2 u &= 0, \quad (x, t) \in (-1, 1) \times (0, 1], \\ u(0, x) &= -\sin(\pi x), \quad u(t, -1) = u(t, 1) = 0.\end{aligned}$$

We consider an anisotropic Gaussian kernel with length scales ℓ_t and ℓ_x :

$$k((t, x), (t', x')) = \sigma^2 \exp\left(-\frac{(t - t')^2}{2\ell_t^2} - \frac{(x - x')^2}{2\ell_x^2}\right).$$

We choose at random 1000 collocation points in the domain, and 201 at the boundaries, from a uniform distribution.

Example: the Burgers' equation

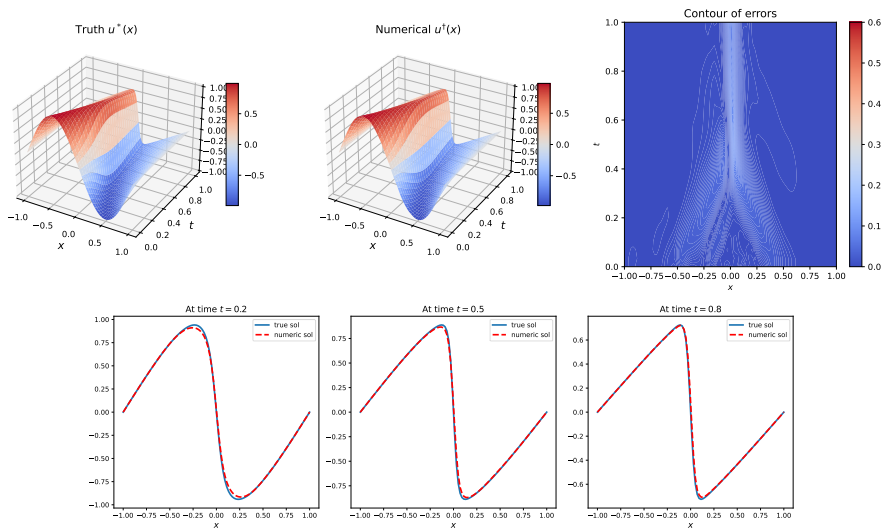


Figure: Estimated solution, with $\ell_t = 0.47, \ell_x = 0.07$. The space-time L^2 error is 0.016, and the L^∞ error is 0.1.

Burgers' equation: sensitivity to kernel hyperparameters

What's happening if we choose another kernel hyperparameters?
 Let us sample ℓ_t and ℓ_x from a uniform distribution $\sim \mathcal{U}([0.01, 1]^2)$

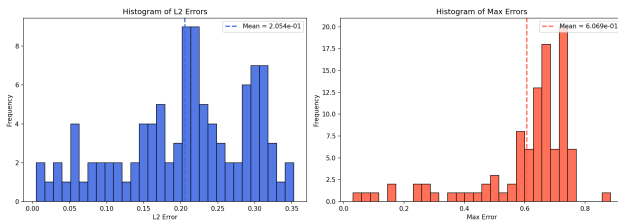


Figure: Histogram of errors when parameters are chosen at random.

The estimation accuracy is *highly sensitive to the kernel choice*.

Outline

1 Introduction

→ a quick picture of PDE solving with kernel methods

2 Learning functions with kernel methods

→ basics on Gaussian process regression and RKHS
and how to account for with linear (e.g., derivative) information

3 Solving (nonlinear) PDEs with kernel methods

→ assuming that a 'good' kernel is provided

4 Learning kernels within a multifidelity context

5 Perspectives

From physics-informed GP to multifidelity learning³

Key issue: the performance of kernel methods strongly depends on the choice of the RKHS $\mathcal{H}(k)$, or equivalently, on the choice of the kernel k .

³[Kennedy and O'Hagan, 2000, Yang et al., 2020, Le Gratiet, 2013]

From physics-informed GP to multifidelity learning³

Key issue: the performance of kernel methods strongly depends on the choice of the RKHS $\mathcal{H}(k)$, or equivalently, on the choice of the kernel k .

Solutions

- ❶ To add a *third* optimization loop to estimate kernel hyperparameters
→ see [Nelsen et al., 2025]
- ❷ To use a multifidelity setting to fit a kernel
 - ▶ Low-fidelity *stochastic* simulator: fast PDE solver, *uncertain* parameters
→ uncertainty allows *inferring a kernel from empirical spatial covariances*
 - ▶ High-fidelity *data*: true observations are available with sensors

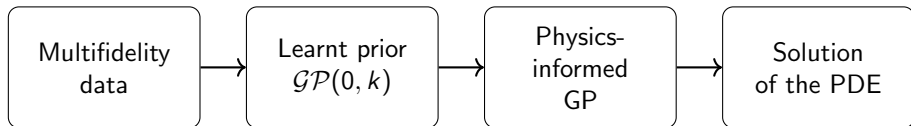
³[Kennedy and O'Hagan, 2000, Yang et al., 2020, Le Gratiet, 2013]

From physics-informed GP to multifidelity learning³

Key issue: the performance of kernel methods strongly depends on the choice of the RKHS $\mathcal{H}(k)$, or equivalently, on the choice of the kernel k .

Solutions

- ① To **add a third optimization loop** to estimate kernel hyperparameters
→ see [Nelsen et al., 2025]
- ② To **use a multifidelity setting** to fit a kernel
 - ▶ Low-fidelity **stochastic** simulator: fast PDE solver, **uncertain** parameters
→ uncertainty allows **inferring a kernel from empirical spatial covariances**
 - ▶ High-fidelity **data**: true observations are available with sensors



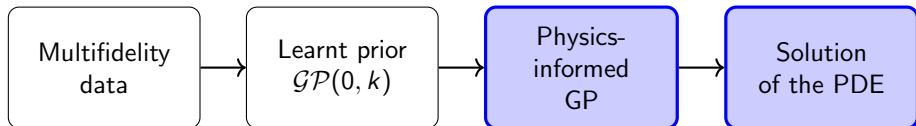
³[Kennedy and O'Hagan, 2000, Yang et al., 2020, Le Gratiet, 2013]

From physics-informed GP to multifidelity learning³

Key issue: the performance of kernel methods strongly depends on the choice of the RKHS $\mathcal{H}(k)$, or equivalently, on the choice of the kernel k .

Solutions

- ❶ To *add a third optimization loop* to estimate kernel hyperparameters
→ see [Nelsen et al., 2025]
- ❷ To *use a multifidelity setting* to fit a kernel
 - ▶ Low-fidelity *stochastic* simulator: fast PDE solver, *uncertain* parameters
→ uncertainty allows *inferring a kernel from empirical spatial covariances*
 - ▶ High-fidelity *data*: true observations are available with sensors



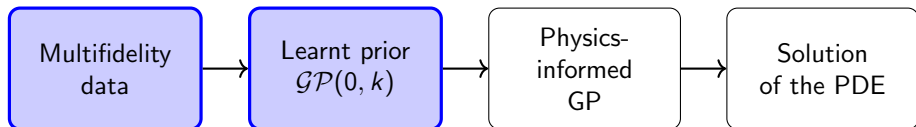
³[Kennedy and O'Hagan, 2000, Yang et al., 2020, Le Gratiet, 2013]

From physics-informed GP to multifidelity learning³

Key issue: the performance of kernel methods strongly depends on the choice of the RKHS $\mathcal{H}(k)$, or equivalently, on the choice of the kernel k .

Solutions

- ① To *add a third optimization loop* to estimate kernel hyperparameters
→ see [Nelsen et al., 2025]
- ② To *use a multifidelity setting* to fit a kernel
 - ▶ Low-fidelity *stochastic* simulator: fast PDE solver, *uncertain* parameters
→ uncertainty allows *inferring a kernel from empirical spatial covariances*
 - ▶ High-fidelity *data*: true observations are available with sensors



³[Kennedy and O'Hagan, 2000, Yang et al., 2020, Le Gratiet, 2013]

Solving the PDE, for a given kernel, in a multifidelity setting

Using the high-fidelity data, we enrich the kernel method by incorporating two types of constraints:

- **Physical constraints:** PDE and boundary conditions,
- **Interpolation constraints:** high-fidelity observations.

We solve, then, the following optimization problem :

$$u^* = \arg \min_{u \in \mathcal{H}(k)} \|u\|_{\mathcal{H}(k)} \quad \text{s.t.} \quad \begin{cases} \mathcal{P}(u)(x_m) = f(x_m), & 1 \leq m \leq M_\Omega, \\ \mathcal{B}(u)(x_m) = g(x_m), & M_\Omega + 1 \leq m \leq M, \\ u(x_{H,i}) = (y_H)_i, & i = 1, \dots, N_H. \end{cases}$$

When k is given, we can estimate u^* using a similar bi-optimization problem as the one presented in the previous section.

Infering a kernel with the ‘cokriging’ multifidelity model

- A **stochastic LF simulator** giving N_L realizations $(\mathbf{y}_L^i)_{1 \leq i \leq N_L}$ on a grid X_L .
 $\rightarrow \mathbf{y}_L$ is drawn from $Y_L \sim \mathcal{GP}(\mu_L, k_L)$
- A set of **HF data** $\mathbf{y}_H$ on a grid $X_H \subseteq X_L$.
 $\rightarrow \mathbf{y}_H$ is drawn from $Y_H \sim \mathcal{GP}(\mu_H, k_H)$.

Following the autoregressive model of [Kennedy and O'Hagan, 2000], we assume

$$Y_H = \rho Y_L + Y_d$$

where Y_L and Y_d are independent Gaussian processes.

This gives $k := k_H = \rho^2 k_L + k_d$. Now, we show how to infer k_L, k_d and ρ .

Kernel, kernel... but what about the mean?

μ_H \ k_H	k'_H	$\rho^2 k_L + k_d$	Comments on μ_H
0	Not a multifidelity setting	<i>MF ker-only</i>	
$\hat{\mu}_H$	<i>MF mean-only</i>	<i>MF mean-ker</i>	$\hat{\mu}_H :=$ GP prediction of $\rho\mu_L + \mu_d$ given observations.
Comments on k_H	$k'_H :=$ selected in a given family of kernels.	$k_L :=$ smooth kernel. $k_d :=$ anisotropic Gaussian kernel.	

Table: Summary of the three methodologies *MF ker-only*, *MF mean-only*, *MF mean-ker*, according to the choice of mean and kernel.

Kernel, kernel... but what about the mean?

μ_H \ k_H	k'_H	$\rho^2 k_L + k_d$	Comments on μ_H
0	Not a multifidelity setting	<i>MF ker-only</i>	
$\hat{\mu}_H$	<i>MF mean-only</i>	<i>MF mean-ker</i>	$\hat{\mu}_H :=$ GP prediction of $\rho\mu_L + \mu_d$ given observations.
Comments on k_H	$k'_H :=$ selected in a given family of kernels.	$k_L :=$ smooth kernel. $k_d :=$ anisotropic Gaussian kernel.	

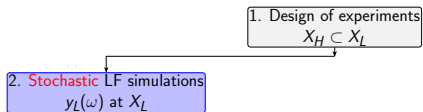
Table: Summary of the three methodologies *MF ker-only*, *MF mean-only*, *MF mean-ker*, according to the choice of mean and kernel.

Multifidelity kernel learning

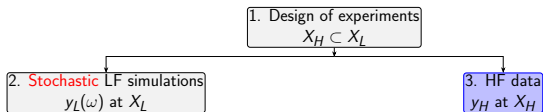
1. Design of experiments

$$X_H \subset X_L$$

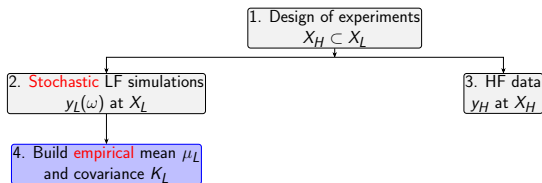
Multifidelity kernel learning



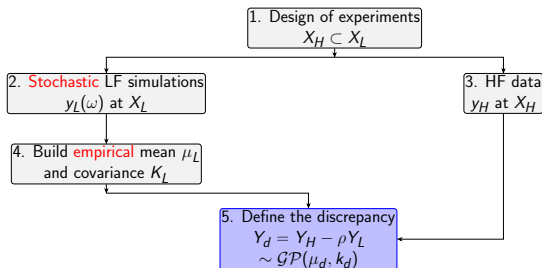
Multifidelity kernel learning



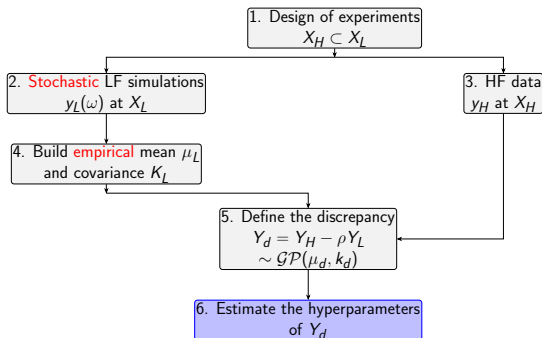
Multifidelity kernel learning



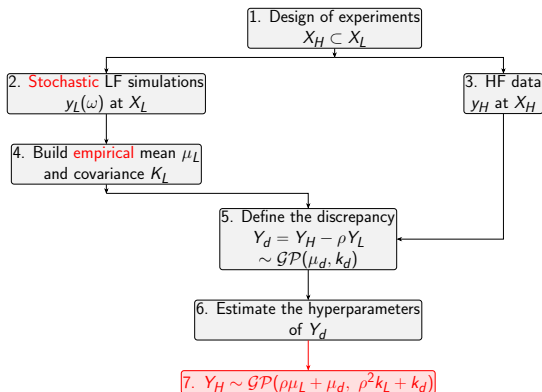
Multifidelity kernel learning



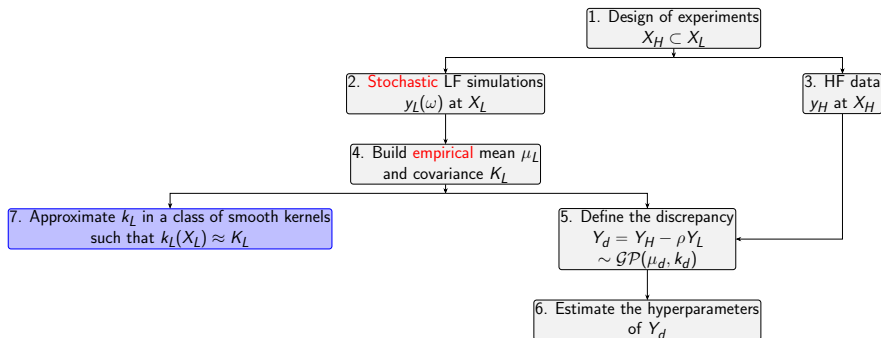
Multifidelity kernel learning



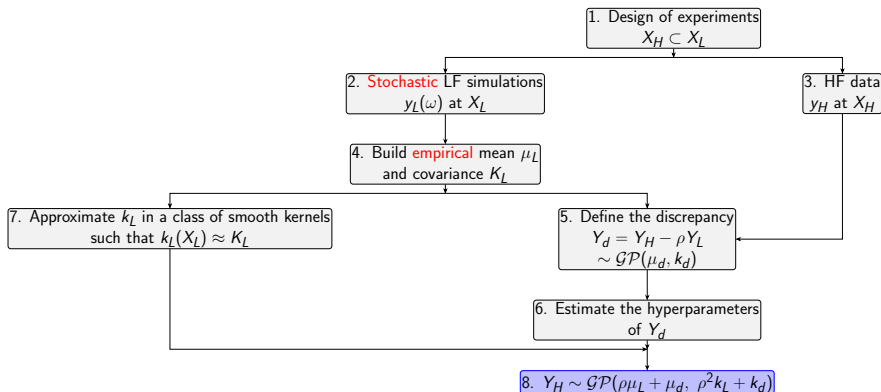
Multifidelity kernel learning



Multifidelity kernel learning



Multifidelity kernel learning



Application: Burgers' equation

Reconsider now the Burgers' equation :

$$\begin{aligned}\partial_t u + \alpha u \partial_x u - \nu \partial_x^2 u &= 0, & (x, t) \in (-1, 1) \times (0, 1], \\ u(0, x) &= -\sin(\pi x), & u(t, -1) = u(t, 1) = 0,\end{aligned}$$

with $\nu = 0.02$, $\alpha = 1$.

We now assume that the **parameters ν and α are not perfectly known**, and that **a few data are available**. This is a **multifidelity setting**:

- **High-fidelity data**: here obtained as the solution on a coarse grid (with the true values of ν, α), in practice obtained with sensors,

$$X_H \subset (-1, 1) \times (0, 1], \quad |X_H| = 10 \times 10$$

- **Low-fidelity simulator**: stochastic realizations on a finer grid

$$X_L \subset (-1, 1) \times (0, 1], \quad |X_L| = 10 \times 20$$

with $\alpha \sim \mathcal{U}(0.8, 1.1)$ and $\nu \sim \mathcal{U}(0.015, 0.03)$

Multifidelity Gaussian process construction

Using the autoregressive model:

$$Y_H(x) = \rho Y_L(x) + Y_d(x).$$

- $Y_d \sim \mathcal{GP}(\mu_d, k_d)$, with μ_d constant and k_d an anisotropic Gaussian kernel.
- $Y_L \sim \mathcal{GP}(\mu_L, k_L)$, where μ_L and k_L are estimated empirically from the low-fidelity realizations on X_L .

⁴[Paciorek and Schervish, 2006, Gibbs, 1998]

Multifidelity Gaussian process construction

Using the autoregressive model:

$$Y_H(x) = \rho Y_L(x) + Y_d(x).$$

- $Y_d \sim \mathcal{GP}(\mu_d, k_d)$, with μ_d constant and k_d an anisotropic Gaussian kernel.
- $Y_L \sim \mathcal{GP}(\mu_L, k_L)$, where μ_L and k_L are estimated empirically from the low-fidelity realizations on X_L .

We then approximate the empirical covariance k_L by fitting a Gibbs kernel, i.e. by minimizing its distance to the class of Gibbs kernels⁴, which yields

$$k_L = k_{\text{Gibbs}}.$$

$$k_{\text{Gibbs}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \prod_{j=1}^2 \sqrt{\frac{2 \ell_j(\mathbf{x}) \ell_j(\mathbf{x}')}{\ell_j(\mathbf{x})^2 + \ell_j(\mathbf{x}')^2}} \exp\left(-\sum_{j=1}^2 \frac{(x_j - x'_j)^2}{\ell_j(\mathbf{x})^2 + \ell_j(\mathbf{x}')^2}\right)$$

with

$$\ell_1(\mathbf{x}) = \alpha_1 + \beta_1 x_1, \quad \ell_2(\mathbf{x}) = \alpha_2 + \beta_2 x_2, \quad \text{for all } \mathbf{x} = (x_1, x_2) \in \mathbb{R}^2.$$

⁴[Paciorek and Schervish, 2006, Gibbs, 1998]

PDE solution

The PDE is then solved in the Hilbert space $\mathcal{H}(k)$ associated with

$$k = \rho^2 k_L + k_d.$$

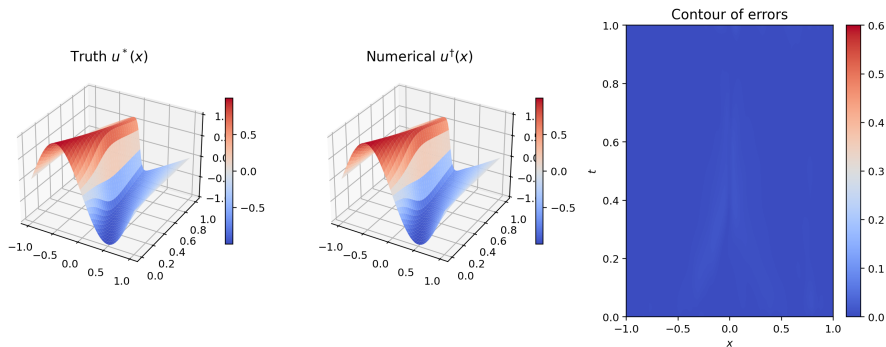


Figure: L_2 error = $(4.20 \pm 0.77) \times 10^{-3}$; max error = $(3.09 \pm 0.97) \times 10^{-2}$

Sensitivity analysis: role of the constraints

We consider the following optimization problem:

$$u^* = \arg \min_{u \in \mathcal{H}(k)} \|u\|_{\mathcal{H}(k)} \quad \text{s.t.} \quad \begin{cases} \mathcal{P}(u)(x_m) = f(x_m), & 1 \leq m \leq M_\Omega, \\ \mathcal{B}(u)(x_m) = g(x_m), & M_\Omega + 1 \leq m \leq M, \\ u(x_{H,i}) = (y_H)_i, & i = 1, \dots, N_H. \end{cases}$$

Sensitivity analysis: role of the constraints

We consider the following optimization problem:

$$u^* = \arg \min_{u \in \mathcal{H}(k)} \|u\|_{\mathcal{H}(k)} \quad \text{s.t.} \quad \begin{cases} \mathcal{P}(u)(x_m) = f(x_m), & 1 \leq m \leq M_\Omega, \\ \mathcal{B}(u)(x_m) = g(x_m), & M_\Omega + 1 \leq m \leq M, \\ u(x_{H,i}) = (y_H)_i, & i = 1, \dots, N_H. \end{cases}$$

Two types of constraints are involved:

- **Physical constraints:** PDE and boundary conditions.
- **Interpolation constraints:** high-fidelity observations.

Sensitivity analysis: role of the constraints

We consider the following optimization problem:

$$u^* = \arg \min_{u \in \mathcal{H}(k)} \|u\|_{\mathcal{H}(k)} \quad \text{s.t.} \quad \begin{cases} \mathcal{P}(u)(x_m) = f(x_m), & 1 \leq m \leq M_\Omega, \\ \mathcal{B}(u)(x_m) = g(x_m), & M_\Omega + 1 \leq m \leq M, \\ u(x_{H,i}) = (y_H)_i, & i = 1, \dots, N_H. \end{cases}$$

Two types of constraints are involved:

- **Physical constraints:** PDE and boundary conditions.
- **Interpolation constraints:** high-fidelity observations.

Question: What is the contribution of each type of constraint?

Sensitivity analysis: role of the constraints

Comparison of constraint configurations

N.B. We use here the 'good' kernel obtained with the multifidelity setting.

Physical	Interpolation	L_2 error	L_∞ error
✓	✗	$(3.09 \pm 2.00) \times 10^{-2}$	$(1.94 \pm 1.29) \times 10^{-1}$
✗	✓	5.0×10^{-2}	2.8×10^{-1}
✓	✓	$(4.20 \pm 0.77) \times 10^{-3}$	$(3.09 \pm 0.97) \times 10^{-2}$

Conclusion:

- **Physical constraints** → enforce physical consistency.
- **Interpolation constraints** → fit the high-fidelity data.
- **Combining both yields the most accurate solution.**

Outline

1 Introduction

→ *a quick picture of PDE solving with kernel methods*

2 Learning functions with kernel methods

→ *basics on Gaussian process regression and RKHS
and how to account for with linear (e.g., derivative) information*

3 Solving (nonlinear) PDEs with kernel methods

→ *assuming that a 'good' kernel is provided*

4 Learning kernels within a multifidelity context

5 Perspectives

Next challenges

- To extend the framework to a **system of PDEs**
 - *requires to use **vector-valued** kernel methods*
 - *work in progress on Shallow Water equation and other wave PDEs*
- To learn the **function-valued map**: *initial condition* → *solution*
 - **operator learning**
- To add uncertainty quantification, as for (some) linear PDEs
 - see [**Chen et al., 2025**] for a first step towards this direction

References



Aronszajn, N. (1950).

Theory of reproducing kernels.

[Transactions of the AMS](#), 68:307–404.



Berlinet, A. and Thomas-Agnan, C. (2004).

[Reproducing kernel Hilbert spaces in probability and statistics](#).

Kluwer Academic Publishers.



Chen, Y., Hosseini, B., Owhadi, H., and Stuart, A. M. (2021).

Solving and learning nonlinear PDEs with Gaussian processes.

[Journal of Computational Physics](#), 447:110668.



Chen, Y., Hosseini, B., Owhadi, H., and Stuart, A. M. (2025).

Gaussian measures conditioned on nonlinear observations: consistency, map estimators, and simulation.

[Statistics and Computing](#), 35(1):10.



Durrande, N., Hensman, J., Rattray, M., and Lawrence, N. (2016).

Detecting periodicities with Gaussian processes.

[PeerJ Computer Science](#) 2:e50.



Gibbs, M. N. (1998).

[Bayesian Gaussian processes for regression and classification](#).

PhD thesis, University of Cambridge, UK.



Kennedy, M. C. and O'Hagan, A. (2000).

Predicting the output from a complex computer code when fast approximations are available.

[Biometrika](#), 87(1):1–13.



Kimeldorf, G. S. and Wahba, G. (1970).

A correspondence between bayesian estimation on stochastic processes and smoothing by splines.

[The Annals of Mathematical Statistics](#), 41(2):495–502.



Le Gratiet, L. (2013).

Multi-fidelity Gaussian process regression for computer experiments.

PhD thesis, Université Paris-Diderot.



Nelsen, N. H., Owhadi, H., Stuart, A. M., Yang, X., and Zou, Z. (2025).

Bilevel optimization for learning hyperparameters: Application to solving PDEs and inverse problems with Gaussian processes.

[arXiv preprint arXiv:2510.05568](#).



Paciorek, C. J. and Schervish, M. J. (2006).

Spatial modelling using a new class of nonstationary covariance functions.

[Environmetrics: The official journal of the International Environmetrics Society](#), 17(5):483–506.



Rasmussen, C. E. and Williams, C. K. I. (2006).

Gaussian processes for machine learning.

The MIT press.



Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989).

Design and analysis of computer experiments.

Statistical Science, 4(4):409–435.



Saitoh, S. and Sawano, Y. (2016).

Theory of reproducing kernels and applications, volume 44.

Springer.



Yang, X., Zhu, X., and Li, J. (2020).

When bifidelity meets cokriging: An efficient physics-informed multifidelity method.

SIAM Journal on Scientific Computing, 42(1):A220–A249.