Operator learning with Gaussian processes - Tutorial

Matthieu Darcy[1]

[1]California Institute of Technology

University of Bern - Institute of Mathematical Statistics and Actuarial Science
January, 2026

Table of Contents

Gaussian Processes: Definition

A *Gaussian process* (GP) is a distribution over functions

$$f : \mathcal{X} \to \mathbb{R}$$

Note: $\mathcal{X}$ *arbitrary* and output is $\mathbb{R}$.

For any finite set of inputs $X = \{x_1, \ldots, x_n\}$, the random vector $f(X) = \big(f(x_1), \ldots, f(x_n)\big)$ is jointly Gaussian:

$$\mathbf{f} \sim \mathcal{N}\big(m(X), K(X, X)\big).$$

Gaussian Processes: Definition

A *Gaussian process* (GP) is a distribution over functions

$$f : \mathcal{X} \to \mathbb{R}$$

Note: $\mathcal{X}$ *arbitrary* and output is $\mathbb{R}$.

For any finite set of inputs $X = \{x_1, \ldots, x_n\}$, the random vector $f(X) = \big(f(x_1), \ldots, f(x_n)\big)$ is jointly Gaussian:

$$\mathbf{f} \sim \mathcal{N}\big(m(X), K(X, X)\big).$$

A GP is requires:

- Mean function: $m(x) = \mathbb{E}[f(x)]$ (The prior mean will always be 0.)
- Covariance kernel: $k(x, x') = \mathrm{Cov}(f(x), f(x'))$ (Big question: which one?)

Gaussian processes for prediction

Let:

- Training inputs $X = \{x_1, \ldots, x_n\}$ with *noisy Gaussian* observations $\mathbf{y}$
- Test inputs $X_* = \{x_*^{(1)}, \ldots, x_*^{(m)}\}$

Conditional mean (deterministic/kernel prediction)

$$\mathbb{E}[f(X_*) \mid f(X) = \mathbf{y} + \sigma\varepsilon] = K(X_*, X)(K(X, X) + \sigma^2 I)^{-1}\mathbf{y}$$

Conditional covariance: (probabilistic prediction)

$$\mathrm{Cov}(f(X_*) \mid f(X) = \mathbf{y} + \sigma\varepsilon) = K(X_*, X_*) - K(X_*, X)(K(X, X) + \sigma^2 I)^{-1}K(X, X_*)$$

Table of Contents

The operator learning problem

The operator learning problem (abstract version)

Let $\{u_i, v_i\}_{i=1}^N$ be N elements of $\mathcal{U} \times \mathcal{V}$ such that

$$\mathcal{S}(u_i) = v_i, \quad \text{for } i = 1, \dots, N\,.$$

The data driven operator learning problem is summarized as :

Given the data $\{u_i, v_i\}_{i=1}^N$ approximate $\mathcal{S}$.

$\mathcal{U}$ is a space of functions $u : \Omega \to \mathbb{R}$

$\mathcal{V}$ is a space of functions $v : D \to \mathbb{R}$

Gaussian processes for Operator learning

We want to apply Gaussian process prediction to this problem. Two main obstacles:

1. How to deal with functional input?
2. How to deal with functional output?

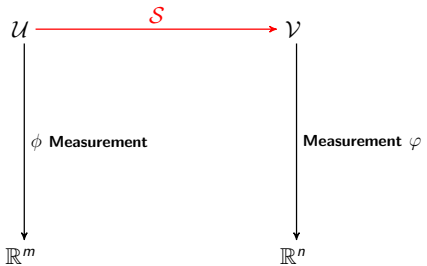
Note

1. In the definition of a Gaussian process $\mathcal{X}$ is arbitrary!
2. The output space is more problematic, cannot be just $\mathbb{R}$.

General framework

In practice we only have access to finite dimensional measurements of (u, v) given by $(\phi(u), \varphi(v))$

Encoder ϕ, φ



General framework

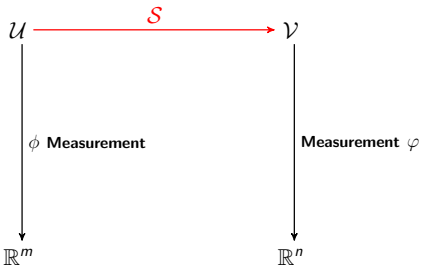
In practice we only have access to finite dimensional measurements of (u, v) given by $(\phi(u), \varphi(v))$

Encoder ϕ, φ

Common examples are:

$\phi : u \mapsto (u(x_1), \ldots, u(x_m))$ Pointwise values.

$\phi : u \mapsto (\langle u, e_1 \rangle, \ldots, \langle u, e_m \rangle)$ Projection in a basis (PCA/POD, Fourier...)



General framework

In practice we only have access to finite dimensional measurements of (u, v) given by $(\phi(u), \varphi(v))$

Encoder ϕ, φ

Common examples are:

$\phi : u \mapsto (u(x_1), \ldots, u(x_m))$ Pointwise values.

$\phi : u \mapsto (\langle u, e_1 \rangle, \ldots, \langle u, e_m \rangle)$ Projection in a basis (PCA/POD, Fourier...)

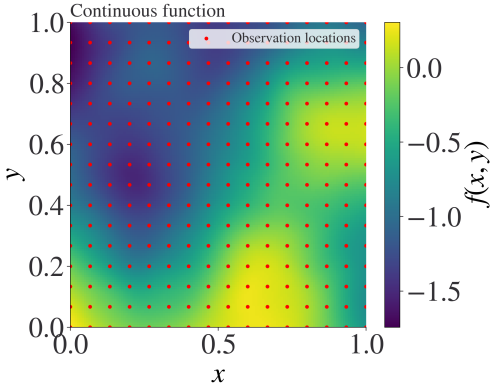


Figure: Continuum and observations

General framework

We have access to finite dimensional measurements of (u, v) given by $(\phi(u), \varphi(v))$:

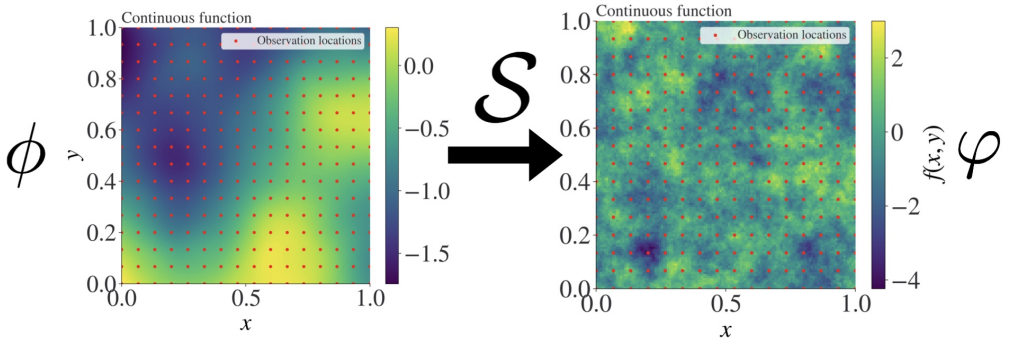
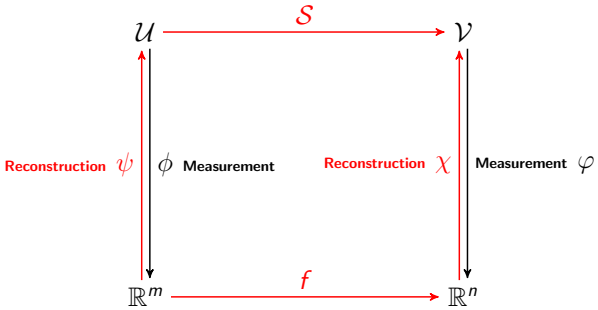


Figure: Continuum and observations

General framework

Encoder ϕ, φ

Decoder ψ, χ

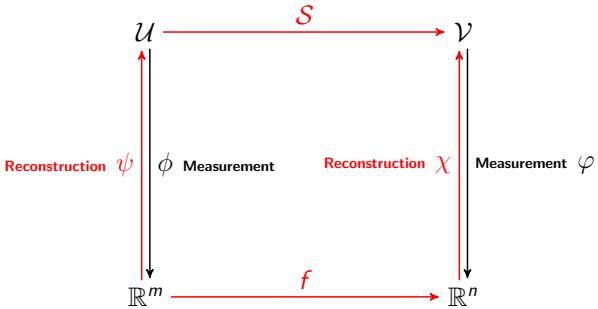


General framework

Encoder ϕ, φ

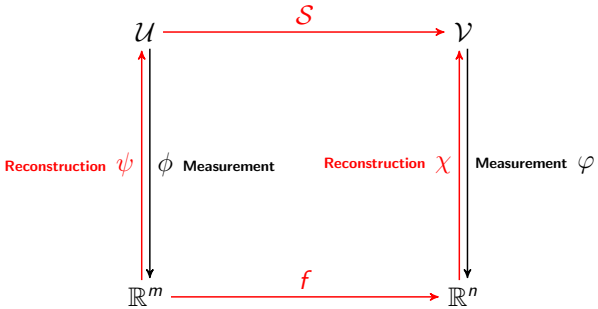
Decoder ψ, χ

Finite dimensional map $f := \varphi \circ \mathcal{S} \circ \psi$



General framework

Encoder	ϕ, φ
Decoder	ψ, χ
Finite dimensional map	$f := \varphi \circ \mathcal{S} \circ \psi$



Summary of our method

1. Define the reconstructions ψ and χ as optimal recovery maps.
2. Approximate the function f using a kernel method/GP (coordinate wise).

Step 1: Optimal recovery

The reconstruction operators are defined as optimal recovery maps

$$\psi(\phi(u)) := \operatorname*{arg\,min}_{w \in \mathcal{U}} \|w\|_Q \quad \text{s.t.} \quad \phi(w) = \phi(u),$$
$$\chi(\varphi(v)) := \operatorname*{arg\,min}_{w \in \mathcal{V}} \|w\|_K \quad \text{s.t.} \quad \varphi(w) = \varphi(v),$$

Step 1: Optimal recovery

The reconstruction operators are defined as optimal recovery maps

$$\psi(\phi(u)) := \operatorname*{arg\,min}_{w \in \mathcal{U}} \|w\|_Q \quad \text{s.t.} \quad \phi(w) = \phi(u),$$
$$\chi(\varphi(v)) := \operatorname*{arg\,min}_{w \in \mathcal{V}} \|w\|_K \quad \text{s.t.} \quad \varphi(w) = \varphi(v),$$

Optimal recovery maps can be expressed by solving a linear system

$$\psi(\phi(u))(x) = Q(x, X)Q(X, X)^{-1}\phi(u) \quad \text{and} \quad \chi(\varphi(v))(y) = K(y, Y)K(Y, Y)^{-1}\varphi(v).$$

Kernel prediction with kernel K/Q!

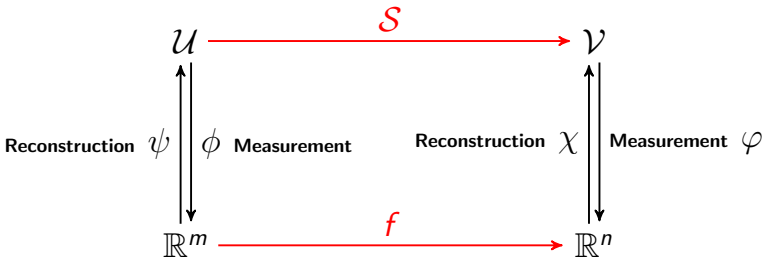
Step 2: Recovery of f

Once the reconstruction operators ψ and χ are defined, our best strategy is to reconstruct f in the diagram:

$$f^\dagger \approx f := \varphi \circ \mathcal{S} \circ \psi$$

and to approximate the operator $\mathcal{S}$ with the operator

$$\mathcal{S}^\dagger := \chi \circ f^\dagger \circ \phi\,.$$



A simple Gaussian process method for $f^\dagger$

Given a kernel S, we approximate $f : \mathbb{R}^m \to \mathbb{R}^n$ via optimal recovery **independently component wise**:

$$f_j^\dagger := \underset{h \in \mathcal{H}_S}{\arg\min} \|h\|_S \quad \text{s.t.} \qquad h(\phi(u_i)) = (\varphi(v_i))_j \qquad \text{for} \quad i = 1, \ldots, N.$$

A simple Gaussian process method for $f^\dagger$

Given a kernel S, we approximate $f : \mathbb{R}^m \to \mathbb{R}^n$ via optimal recovery **independently component wise**:

$$f_j^\dagger := \operatorname*{arg\,min}_{h \in \mathcal{H}_S} \|h\|_S \quad \text{s.t.} \qquad h(\phi(u_i)) = (\varphi(v_i))_j \qquad \text{for} \quad i = 1, \ldots, N.$$

This also has a closed-form solution given a linear solve:

$$f_j^\dagger(\boldsymbol{u}) = S(\boldsymbol{u}, U) S(U, U)^{-1} \boldsymbol{v}_j.$$

where $U_i := \phi(u_i)$ and $V_i := \varphi(v_i)$.
Component wise kernel prediction with kernel S!

Summary

How to deal with our two obstacles:

1. Functional input $\rightarrow$ Kernel defined on $\phi(u)$
2. Functional output $\rightarrow$ ouput is a single component of $\varphi(v)$
3. Reconstruction operators bridge the gap.

- There is a way to define a single Gaussian process/kernel that links all 3 steps [Batlle, D, Hosseini, and Owhadi, Sec. 2.1-2.3]
- There is prior work on operator/matrix valued kernels and interesting theoretical research directions [Kadri, Duflos, Preux, Canu, Rakotomamonjy, and Audiffren; Micchelli and Pontil]

Table of Contents

Burger's equation

$$\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} = \nu\frac{\partial^2 v}{\partial x^2}$$
$$v(0, x) = v_0(x)$$

Aim to learn

$$\mathcal{S} : u = v_0(\cdot) \mapsto v(1, \cdot)$$

References I

- Batlle, Pau et al. [2024]. "Kernel methods are competitive for operator learning". In: *Journal of Computational Physics* 496.
- Kadri, Hachem et al. [2016]. "Operator-valued Kernels for Learning from Functional Response Data". In: *Journal of Machine Learning Research* 17.20, pp. 1–54. URL: http://jmlr.org/papers/v17/11-315.html.
- Micchelli, Charles A. and Massimiliano Pontil [Jan. 2005]. "On Learning Vector-Valued Functions". In: *Neural Computation* 17.1, pp. 177–204.