One shot learning of Stochastic Differential Equations with Gaussian Processes

Matthieu Darcy [1] Boumediene Hamzi [1,2] Giulia Livieri [3] Houman Owhadi [1] Peyman Tavallali [4]

[1]California Institute of Technology [2]Johns Hopkins University [3]Scuola Normale Superiore [4]Jet Propulsion Lab, NASA

Main takeway

The drift f and the diffusion σ of stochastic differential equations of the form

$$dX_t = f(X_t)dt + \sigma(X_t)dW_t,$$

can be learned from a **single sample trajectory** using **Gaussian Processes and kernels learned from the data** [1].

Discretization

We have access to observations $X := (X_n)_{n=1}^N$ separated by time-steps Δt_n. We use the Euler-Maruyama dsciretization:

$$X_{n+1} = X_n + f(X_n)\Delta t_n + \sigma(X_n)\sqrt{\Delta t_n}\xi_n + \varepsilon_n$$

$\xi_n \overset{d}{\sim} \mathcal{N}(0,1)$ is noise **inherent to the dynamical system**,

$\varepsilon_n \overset{d}{\sim} \mathcal{N}(0,\lambda)$ is noise coming from the **discretization error**.

Defining $Y_n := X_{n+1} - X_n$, our model is restated as

$$Y_n = f(X_n)\Delta t_n + \sigma(X_n)\sqrt{\Delta t_n}\xi_n + \varepsilon_n\,.$$

Gaussian Process Prior

We assume that f and σ are distributed according to **independent** Gaussian processes:

$$f \overset{d}{\sim} \mathcal{GP}(\mathbf{0}, \boldsymbol{K})$$
$$\sigma \overset{d}{\sim} \mathcal{GP}(\mathbf{0}, \boldsymbol{G}).$$

The kernel functions $\boldsymbol{K}, \boldsymbol{G}$ are parameterized by the hyper-parameters $\boldsymbol{\theta}$. We first recover the values of f and σ at the data points:

$$\bar{f}_n := f(X_n)$$
$$\bar{\sigma}_n := \sigma(X_n)$$

using **Maximum A Posteriori estimation** and Bayes' rule

$$\bar{f}^*, \bar{\sigma}^* := \underset{\bar{f},\bar{\sigma}}{\arg\min}\, \mathcal{L}(\bar{f},\bar{\sigma}) := \underset{\bar{f},\bar{\sigma}}{\arg\min}\, p(Y|\bar{f},\bar{\sigma}X)p(\bar{f}|X)p(\bar{\sigma}|X).$$

Representer theorem

For any given $\bar{\sigma}$, the minimizer in $\bar{f}$ of $\mathcal{L}(\bar{f},\bar{\sigma})$ is

$$\bar{f}^*(\bar{\sigma}) := \underset{\bar{f}}{\arg\min}\, \mathcal{L}(\bar{f},\bar{\sigma}) = K(X,X)\Lambda\Big(\Lambda K(X,X)\Lambda + \Sigma + \lambda I\Big)^{-1}Y.$$

Using the representer theorem, we minimize in $\bar{\sigma}$ the loss:

$$\mathcal{L}(\bar{f}^*(\bar{\sigma}),\bar{\sigma}).$$

Numerical example

We generate 500 points for training and 500 points for testing of the stochastic differential equation

$$dX_t = \sin(2k\pi X_t)dt + b\cos(2k\pi X_t)dW_t \quad \text{Trigonometric process.}$$

We use the Matérn Kernel with smoothness parameter $\nu = \frac{5}{2}$

$$K_{\text{Matern}}(x,y) = \sigma^2\Big(1 + \frac{\sqrt{5}||x-y||}{l} + \frac{5||x-y||^2}{3l^2}\Big)\exp\Big(-\frac{\sqrt{5}||x-y||}{l}\Big).$$

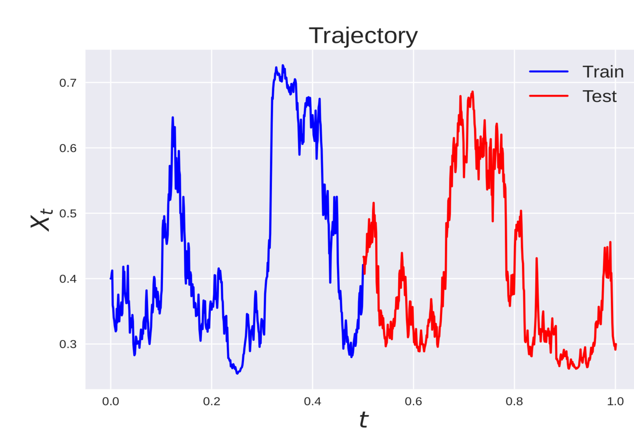

Figure 1. A sample trajectory of the process.

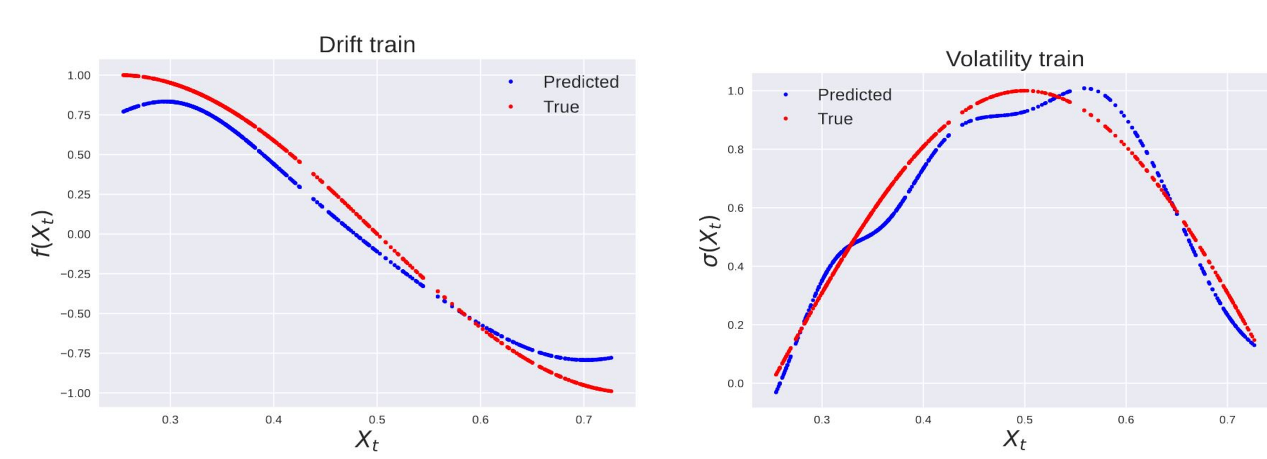

Figure 2. Recovery of the drift f (left) and volatility σ (right).

Learning kernels from data: method

Learning the hyper-parameters $\boldsymbol{\theta}$ of the kernel functions $\boldsymbol{K}$ and $\boldsymbol{G}$ drastically improves the recovery of the functions f and σ. We use a **randomized cross-validation** approach to **learn the kernels from data**.

- **Cross validation:** optimize the model on a subset $\mathcal{D}_\Pi$ of the data and measure the performance on a withheld subset $\mathcal{D}_{\Pi^c}$.
- **Randomized:** as proposed in [3], sample subsets $(\mathcal{D}_\Pi, \mathcal{D}_{\Pi^c})$ randomly and use a noisy loss $\mathcal{L}_{\text{CV}}$.

We use of the likelihood of the withheld data $\mathcal{D}_{\Pi^c}$ as the cross validation loss.

$$\mathcal{L}_{\text{CV}}(\boldsymbol{\theta}; \bar{f}^*, \bar{\sigma}^*, \mathcal{D}_{\Pi^c}) = p(Y_\Pi|\bar{f}^*, \bar{\sigma}^*, X_\Pi)$$

This noisy loss is optimized with a Bayesian Optimization algorithm.

Learning kernels from data: numerical results

Learning the hyper-parameters $\boldsymbol{\theta}$ improves both the recovery of f and σ at the training data points and the prediction of future values

$$dX_t = \mu X_t dt + b\exp(-X_t^2)dW_t \quad \text{Exponential decay volatility.}$$

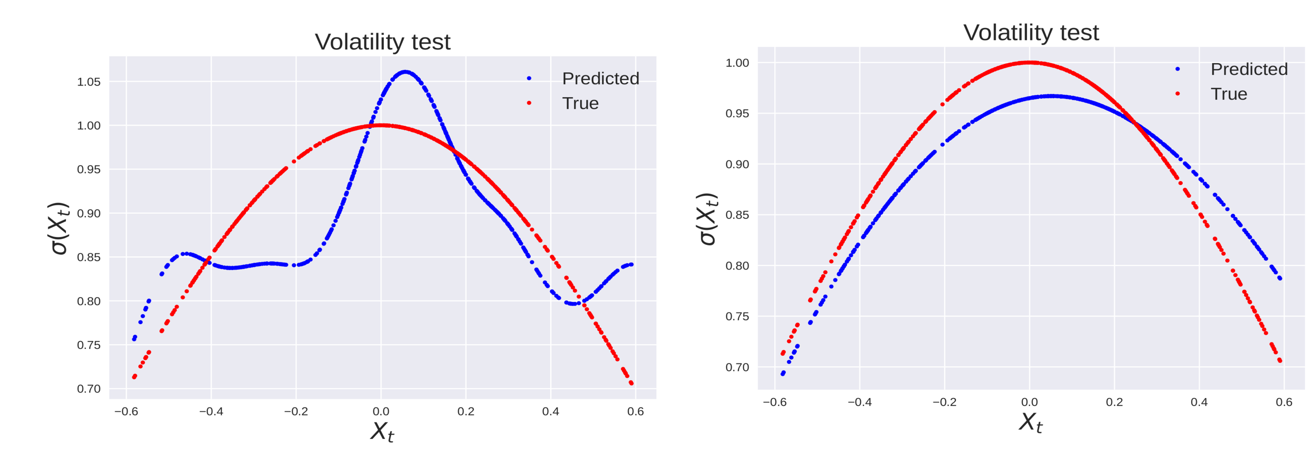

Figure 3. Prediction of the volatility: non-learned kernel versus learned kernel.

$$dX_t = \mu X_t dt + \sigma X_t dW_t \quad \text{Geometric Brownian motion.}$$

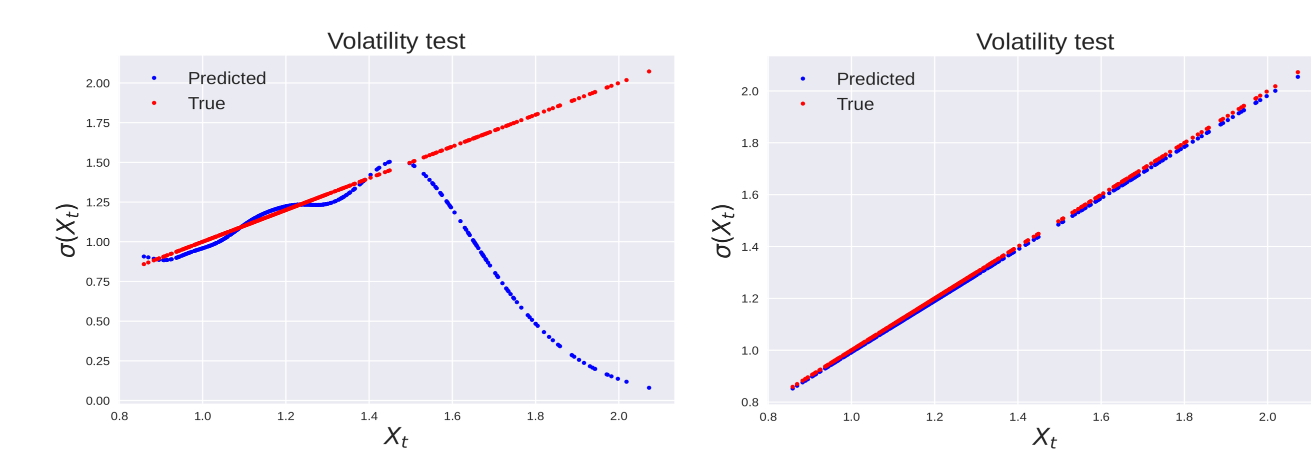

Figure 4. Prediction of the volatility: non-learned kernel versus learned kernel.

Going further: computational graph completion

This model can be cast as **computational graph** [2] representing dependencies between variables and functions. **Completing the graph** using the data allows to **recover the unknown functions**.

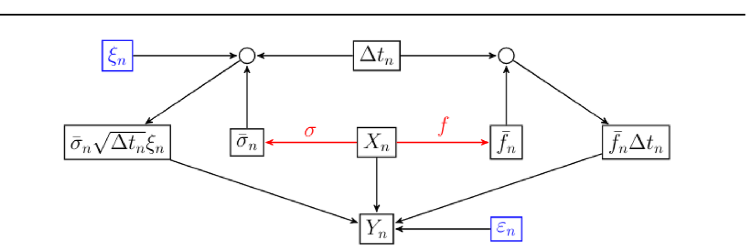

Figure 5. The computational graph

Computational Graph completion offers a general framework to recover unknown variables and functions, beyond the problem considered here.

References

[1] Matthieu Darcy, Boumediene Hamzi, Giulia Livieri, Houman Owhadi, and Peyman Tavallali. One-shot learning of stochastic differential equations with computational graph completion, 02 2022.

[2] Houman Owhadi. Computational graph completion, 2021.

[3] Houman Owhadi and Gene Ryan Yoo. Kernel flows: From learning kernels from data into the abyss. *Journal of Computational Physics*, 389:22–47, 2019.

 mdarcy@caltech.edu