LEARNING STOCHASTIC DYNAMICS FROM DATA

Ziheng Guo
Dept. of Applied Math
Illinois Tech
Chicago, IL 60616
zguo22@hawk.iit.edu

Igor Cialenco
Dept. of Applied Math
Illinois Tech
Chicago, IL 60616
cialenco@iit.edu

Ming Zhong
Dept. of Applied Math
Illinois Tech
Chicago, IL 60616
mzhong3@iit.edu

ABSTRACT

We present a noise guided trajectory based system identification method for inferring the dynamical structure from observation generated by stochastic differential equations. Our method can handle various kinds of noise, including the case when the components of the noise are correlated. Our method can also learn both the noise level and drift term together from trajectory. We present various numerical tests for showcasing the superior performance of our learning algorithm.

1 INTRODUCTION

Stochastic Differential Equation (SDE) is a fundamental modeling tool in various science and engineering fields Evans (2013); Särkkä" & Solin (2019). Compared with traditional deterministic models which often fall short in capturing the stochasticity in the nature, the significance of SDE lies in the ability to model complex systems influenced by random perturbations. Hence, SDE provides insights into the behavior of such systems under uncertainty. By incorporating a random component, typically through a Brownian motion, SDE provides a more realistic and flexible framework for simulating and predicting the behavior of these complex and dynamic systems.

We consider the models following the SDE of the following form, $d\mathbf{x}_t = \boldsymbol{f}(\mathbf{x}_t)dt + d\mathbf{w}_t$, where the state vector $\mathbf{x}_t \in \mathbb{R}^d$, the drift term $\boldsymbol{f} : \mathbb{R}^d \to \mathbb{R}^d$, and the stochastic noise $\mathbf{w}_t$ is a Brownian motion with covariance matrix $\boldsymbol{D}(\boldsymbol{x})$ which also depends on the state. In the field of mathematical finance, there are several SDE models that are widely used, for example, Black-Scholes Model Black & Scholes (1973) in option pricing, Vasicek Model Vasicek (1977) for analyzing interest rate dynamic and Heston Model Heston (2015) for volatility studies. Apart from the field of finance, SDE has its application in physics Coffey et al. (2004); Ebeling et al. (2008) and biology Székely & Burrage (2014); Dingli & Pacheco (2011) where SDE is used to study evolution of particles subjected to random forces and modeling physical and biological systems.

The calibration of these models is crucial for their effective application in areas like derivative pricing and asset allocation in finance, as well as in the analysis of particle systems in quantum mechanics and the study of biological system behaviors in biology. This requires the use of diverse statistical and mathematical techniques to ensure the models' outputs align with empirical data, thereby enhancing their predictive and explanatory power. Since these SDE models have explicit function form of both drift and diffusion terms, the calibration or estimation of parameters is mostly done by minimizing the least square error between observation and model prediction Mrázek & Pospíšil (2017); Abu-Mostafa (2001). SDE has also been studied for more general cases where the drift term does not have an explicit form where it satisfies $\boldsymbol{f} = \boldsymbol{f}(\mathbf{x}_t, \theta)$ with θ being a vector a unknown parameters. These models can be estimated with the maximum likelihood estimator approach presented in Phillips (1972). The other approach is to obtain the likelihood function by Radon Nikodym derivative over the whole observation $\{\mathbf{x}_t\}_{t=0}^T$. Inspired by theorem 7.4 in Särkkä" & Solin (2019) and the Girsanov theorem, we derive a likelihood function that incorporates state-dependent correlated noise. By utilizing such likelihood function, we are able to capture the essential structure of $\boldsymbol{f}$ from data with complex noise structure.

The remainder of the paper is structured as follows, section 2 outlines the framework we use to learn the drift term and the noise level (if not state dependent), we demonstrates the effectiveness of our learning by testing it on various cases summarized in section 4 and additional examples presented

in Appendix B, we conclude our paper in section 5 with a few pointers for ongoing and future developments.

2 LEARNING FRAMEWORK

We consider the following SDE

$$d\mathbf{x}_t = \boldsymbol{f}(\mathbf{x}_t)dt + d\mathbf{w}_t, \quad \mathbf{x}_t, \mathbf{w}_t \in \mathbb{R}^d, \tag{1}$$

where $\boldsymbol{f} : \mathbb{R}^d \to \mathbb{R}^d$ is a drift term, and $\mathbf{w}$ represents the Brownian noise with a symmetric positive definite covariance matrix $\boldsymbol{D} = \boldsymbol{D}(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^{d\times d}$.

We consider the scenario when we are given continuous observation data in the form of $\{\mathbf{x}_t, d\mathbf{x}_t\}_{t\in[0,T]}$ for $\mathbf{x}_0 \sim \mu_0$. We will find the minimizer to the following loss function

$$\mathcal{E}_{\mathcal{H}}(\tilde{\boldsymbol{f}}) = \mathbb{E}_{\mathbf{x}_0\in\mu_0}\Big[\frac{1}{2T}\int_{t=0}^{T}\big(< \tilde{\boldsymbol{f}}(\mathbf{x}_t), \boldsymbol{D}^{-1}(\mathbf{x}_t)\tilde{\boldsymbol{f}}(\mathbf{x}_t) > \, dt - 2 < \tilde{\boldsymbol{f}}(\mathbf{x}_t), \boldsymbol{D}^{-1}(\mathbf{x}_t)d\mathbf{x}_t > \big)\Big], \tag{2}$$

for $\tilde{\boldsymbol{f}} \in \mathcal{H}$; the function space $\mathcal{H}$ is designed to be convex and compact and it is also data-driven. This loss functional is derived from the Girsanov theorem as well as inspiration from Theorem 7.4 from Särkkä" & Solin (2019). In the case of uncorrelated noise, i.e. $\boldsymbol{D}(\boldsymbol{x}) = \sigma^2(\boldsymbol{x})\mathbf{I}$, where $\mathbf{I}$ is the $d \times d$ identity matrix and $\sigma : \mathbb{R}^d \to \mathbb{R}^+$ is a scalar function depends on the state, representing the noise level, 2 can be simplified to

$$\mathcal{E}^{\text{Sim}}_{\mathcal{H}}(\tilde{\boldsymbol{f}}) = \mathbb{E}_{\mathbf{x}_0\in\mu_0}\Big[\frac{1}{2T}\big(\int_{t=0}^{T}\frac{||\tilde{\boldsymbol{f}}(\mathbf{x}_t)||^2}{\sigma^2(\mathbf{x}_t)}\,dt - 2\frac{< \tilde{\boldsymbol{f}}(\mathbf{x}_t), d\mathbf{x}_t >}{\sigma^2(\mathbf{x}_t)}\big)\Big]. \tag{3}$$

Moreover, we provide three different performance measures of our estimated drift. First, if we have access to original drift function $\boldsymbol{f}$, then we will use the following error to compute the difference between $\hat{\boldsymbol{f}}$ (our estimator) to $\boldsymbol{f}$ with the following norm

$$||\boldsymbol{f} - \hat{\boldsymbol{f}}||^2_{L^2(\rho)} = \frac{1}{T}\int_{\mathbf{x}\in\Omega}||\boldsymbol{f}(\mathbf{x}) - \hat{\boldsymbol{f}}(\mathbf{x})||^2_{\ell^2(\mathbb{R}^d)}\,d\rho(\mathbf{x}). \tag{4}$$

Here the weighted measure ρ is defined on Ω, where it defines the region of $\mathbf{x}$ explored due to the dynamics defined by equation 1; therefore ρ is given as follows

$$\rho(\mathbf{x}) = \mathbb{E}_{\mathbf{x}_0\sim\mu_0}\Big[\frac{1}{T}\int_{t=0}^{T}\delta_{\mathbf{x}_t}(\mathbf{x})\Big], \quad \text{where } \mathbf{x}_t \text{ evolves from } \mathbf{x}_0 \text{ by equation 1.} \tag{5}$$

The norm given by equation 4 is useful only from the theoretical perspective, i.e. showing convergence. In real life situation, $\boldsymbol{f}$ is most likely non-accessible. Hence we will look at a performance measure that compare the difference between $\mathbf{X}(\boldsymbol{f}, \mathbf{x}_0, T) = \{\mathbf{x}_t\}_{t\in[0,T]}$ (the observed trajectory evolves from $\mathbf{x}_0 \sim \mu_0$ with the unknown $\boldsymbol{f}$) and $\hat{\mathbf{X}}(\hat{\boldsymbol{f}}, \mathbf{x}_0, T) = \{\hat{\mathbf{x}}_t\}_{t\in[0,T]}$ (the estimated trajectory evolves from the same $\mathbf{x}_0$ with the learned $\hat{\boldsymbol{f}}$ and the same random noise as used by the original dynamics). Then, the difference between the two trajectories is measured as follows

$$||\mathbf{X} - \hat{\mathbf{X}}|| = \mathbb{E}_{\mathbf{x}_0\sim\mu_0}\Big[\frac{1}{T}\int_{t=0}^{T}||\mathbf{x}_t - \hat{\mathbf{x}}_t||^2_{\ell^2(\mathbb{R}^d)}\,dt\Big]. \tag{6}$$

However, comparing two sets of trajectories (even with the same initial condition) on the same random noise is not realistic. We compare the distribution of the trajectories over different initial conditions and all possible noise at some chosen time snapshots using the Wasserstein distance.

3 COMPARISON TO OTHERS

When the noise level becomes a constant, i.e. $\sigma(\mathbf{x}) = \sigma > 0$, we end up a much simpler loss

$$\mathcal{E}^{\text{Simpler}}_{\mathcal{H}}(\tilde{\boldsymbol{f}}) = \mathbb{E}_{\mathbf{x}_0\in\mu_0}\Big[\frac{1}{2T\sigma^2}\big(\int_{t=0}^{T}||\tilde{\boldsymbol{f}}(\mathbf{x}_t)||^2\,dt - 2 < \tilde{\boldsymbol{f}}(\mathbf{x}_t), d\mathbf{x}_t > \big)\Big],$$

which has been investigated in Lu et al. (2022). System identification of the drift term has been studied in many different scenarios, e.g. identification by enforcing sparsity such as SINDy Brunton et al. (2016), neural network based methods such as NeuralODE Chen et al. (2018) and PINN Raissi et al. (2019), regression based Cucker & Smale (2002), and high-dimensional reduction framework Lu et al. (2019). The uniqueness of our method is that we incorporate the covariance matrix into the learning and hence improving the estimation especially when the noise is correlated.

4 EXAMPLES

In this section, we demonstrate the application of our trajectory-based method for estimating drift functions, showcasing a variety of examples. We explore drift functions ranging from polynomials in one and two dimensions to combinations of polynomials and trigonometric functions, as well as a deep learning approach in one dimension. The observations, serving as the input dataset for testing our method, are generated by the Euler-Maruyama scheme, utilizing the drift functions as we just mentioned. The basis space $\mathcal{H}$ is constructed employing either B-spline or piecewise polynomial methods for maximum degree p-max equals 2. For higher order dimensions where $d \geq 2$, each basis function is derived through a tensor grid product, utilizing one-dimensional basis defined by knots that segment the domain in each dimension.

The parameters for the following examples are listed in table 1. Due to space constraints, additional examples are provided in appendix B. The estimation results are evaluated using several different metrics. We record the noise terms, $d\mathbf{w}_t$, from the trajectory generation process and compare the trajectories produced by the estimated drift functions, $\hat{\boldsymbol{f}}$, under identical noise conditions. We examine trajectory-wise errors using equation 6 with relative trajectory error and plot both $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$ to calculate the relative L^2 error using 4, where ρ is derived by 5. Additionally, we assess the distribution-wise discrepancies between observed and estimated results, computing the Wasserstein distance at various time steps.

Table 1: Parameters Setup for Examples

Simulation Scheme		Euler-Maruyama	
T	1	$D\ (d=1)$	0.6
dt	0.001	$D\ (d=2)$	$\left(\begin{smallmatrix} 0.6 & 0 \\ 0 & 0.8 \end{smallmatrix}\right)$
$\mathbf{x}_0$	Uniform(0,10)	M	10000
p-max	2	Basis Type	B-Spline / PW-Polynomial

4.1 EXAMPLE: SINE/COSINE DRIFT

We initiate our numerical study with a one-dimensional ($d = 1$) drift function that incorporates both polynomial and trigonometric components, given by $\boldsymbol{f} = 2 + 0.08\mathbf{x} - 0.05\sin(\mathbf{x}) + 0.02\cos^2(\mathbf{x})$.

Figure 1 illustrates the comparison between the true drift function $\boldsymbol{f}$ and the estimated drift function $\hat{\boldsymbol{f}}$, alongside a comparison of trajectories. Notably, Figure 1(a) on the left includes a background region depicting the histogram of $\mathbf{x}_t$, which represents the distribution of observations over the domain of $\mathbf{x}$. This visualization reveals that in regions where $\mathbf{x}$ has a higher density of observations—indicated by higher histogram values—the estimation of $\hat{\boldsymbol{f}}$ tends to be more accurate. Conversely, in less dense regions of the dataset (two ends of the domain), the estimation accuracy of $\hat{\boldsymbol{f}}$ diminishes. Table 2 presents a detailed quantitative analysis of the estimation results, including the L^2 norm difference between $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$, as well as the trajectory error. Furthermore, the table compares the distributional distances between $\mathbf{x}_t$ and $\hat{\mathbf{x}}_t$ at selected time steps, with the Wasserstein distance results included.

4.2 EXAMPLE: DEEP NEURAL NETWORKS

We continue our numerical investigation with a one-dimensional ($d = 1$) drift function which is given by $\boldsymbol{f} = 0.08\mathbf{x}$. Figure 2 illustrates the comparison between the true drift function $\boldsymbol{f}$ and the estimated drift function $\hat{\boldsymbol{f}}$, alongside a comparison of trajectories. Recall the setup of figures are

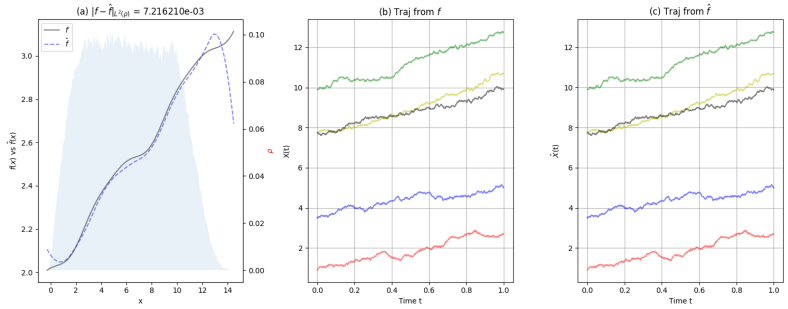


Figure 1: Left: Comparison of $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$. Middle: 5 trajectories generated by $\boldsymbol{f}$. Right: 5 trajectories generated by $\hat{\boldsymbol{f}}$ with same noise.

Table 2: One-dimensional Drift Function Estimation Summary

Number of Basis	8	Wasserstein Distance	
Maximum Degree	2	$t = 0.25$	0.0291
Relative $L^2(\rho)$ Error	0.007935	$t = 0.50$	0.0319
Relative Trajectory Error	0.0020239 ± 0.002046	$t = 1.00$	0.0403

similar to the ones presented in previous section. The error for learning $\boldsymbol{f}$ turns out to be bigger,

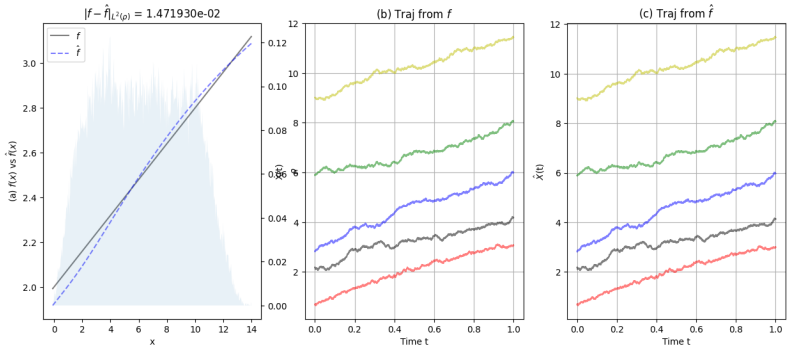


Figure 2: Left: Comparison of $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$. Middle: 5 trajectories generated by $\boldsymbol{f}$. Right: 5 trajectories generated by $\hat{\boldsymbol{f}}$ with same noise.

especially towards the two end points of the interval. However, the errors happen mostly during the two end points of the data interval, where the distribution of the data appears to be small, i.e. few data present in the learning. We are able to recover most the trajectory.

4.3 Example: 2D with Non-diagonal covariance matrix

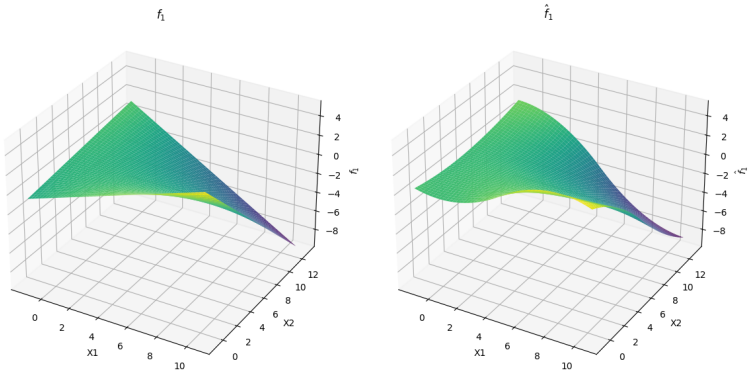
In this example, we incorporate a non-diagonal covariance matrix into a 2 dimensional SDE system. As specified in table 1, all parameters remain unchanged except for D and M. We change total trajectory observation number M to 1000 for faster calculation and

$$D = \begin{pmatrix} 0.6 & 0.2 \\ 0.2 & 0.8 \end{pmatrix}$$

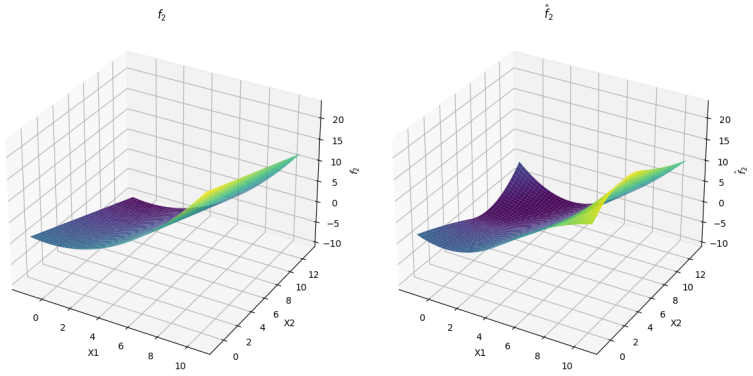
This change in D implies that the Brownian motions within the system are correlated. The drift function is defined using the notation, $\boldsymbol{f} = [f_1(\mathbf{x}) \quad f_2(\mathbf{x})]^\top$ and $\mathbf{x} = [x_1 \quad x_2]^\top$, where $\boldsymbol{f}_i : \mathbb{R}^2 \to \mathbb{R}$ and $\mathbf{x}_i \in \mathbb{R}$ for $i \in \{1, 2\}$. For polynomial drift function $\boldsymbol{f}$, we set $f_1 = 0.4\mathbf{x}_1 - 0.1\mathbf{x}_1\mathbf{x}_2$ and $f_2 = -0.8\mathbf{x}_2 + 0.2\mathbf{x}_1^2$. The estimation result can be found in figure 3a and 3b and table 3.

Table 3: Two-dimensional Correlated Noise Drift Function Estimation Summary

Relative $L^2(\rho)$ Error	0.042944	Wasserstein Distance	
Relative Trajectory Error	0.010166±0.01998	$t = 1.00$	0.2422



(a)



(b)

Figure 3: Two-dimensional Correlated Noise Comparison of $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$. (a) Surface of Dimension 1 (b) Surface of Dimension 2

5 Conclusion

We have introduced a novel methodology for learning the drift and diffusion components through a trajectory-based loss function guided by noise. This loss function is derived from the negative-log ratio of likelihood functions, quantifying the ratio of probabilities of observing two stochastic processes sharing the same initial starting point. Our approach accommodates various noise structures, provided that the covariance information is available. Even in cases where the noise structure is unknown and the covariance is a scalar, our method remains effective through the utilization of quadratic variation. However, ongoing research aims to explore scenarios where the noise exhibits state dependence and matrix structure. Evaluation on high-dimensional drift terms and complex noise structures is currently underway. Our loss function solely relies on observable data (trajectories) and accounts for potential correlated noise.

Acknowledgments

ZG developed the algorithm, analyzed the data and implemented the software package. ZG develops the theory with IC and MZ. MZ designed the research. All authors wrote the manuscript. MZ is partially supported by NSF-2225507 and the startup fund provided by Illinois Tech.

REFERENCES

Y.S. Abu-Mostafa. Financial model calibration using consistency hints. *IEEE Transactions on Neural Networks*, 12(4):791–808, 2001. doi: 10.1109/72.935092.

Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973. ISSN 00223808, 1537534X. URL http://www.jstor.org/stable/1831029.

S. L. Brunton, J. L. Proctor, and N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *PNAS*, 113(15):3932 – 3937, 2016.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

William Coffey, Yuri Kalmykov, and John Waldron. *The Langevin Equation: With Applications to Stochastic Problems in Physics, Chemistry and Electrical Engineering*. 09 2004. ISBN 978-981-238-462-1. doi: 10.1142/9789812795090.

Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American mathematical society*, 39(1):1–49, 2002.

D. Dingli and J.M. Pacheco. Stochastic dynamics and the evolution of mutations in stem cells. *BMC Biol*, 9:41, 2011.

Werner Ebeling, Ewa Gudowska-Nowak, and Igor M. Sokolov. On Stochastic Dynamics in PHysics - Remarks on History and Terminology. *Acta Physica Polonica B*, 39(5):1003 – 1018, 2008.

Lawrence C. Evans. *An Introduction to Stochastic Differential Equations*. American Mathematical Society, 2013.

Steven L. Heston. A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options. *The Review of Financial Studies*, 6(2):327–343, 04 2015. ISSN 0893-9454. doi: 10.1093/rfs/6.2.327. URL https://doi.org/10.1093/rfs/6.2.327.

Fei Lu, Ming Zhong, Sui Tang, and Mauro Maggioni. Nonparametric inference of interaction laws in systems of agents from trajectory data. *Proceedings of the National Academy of Sciences*, 116 (29):14424–14433, 2019.

Fei Lu, Maruo Maggioni, and Sui Tang. Learning interaction kernels in stochastic systems of interacting particles from multiple trajectories. *Foundations of Computational Mathematics*, 22:1013 – 1067, 2022.

Milan Mrázek and Jan Pospíšil. Calibration and simulation of heston model. *Open Mathematics*, 15 (1):679–704, 2017. doi: doi:10.1515/math-2017-0058. URL https://doi.org/10.1515/math-2017-0058.

P. C. B. Phillips. The structural estimation of a stochastic differential equation system. *Econometrica*, 40(6):1021–1041, 1972. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1913853.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

Simo Särkkä" and Arno Solin. *Applied Stochastic Differential Equations*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2019.

Tamás Székely and Kevin Burrage. Stochastic simulation in systems biology. *Computational and Structural Biotechnology Journal*, 12(20):14–25, 2014. ISSN 2001-0370. doi: https://doi.org/10.1016/j.csbj.2014.10.003. URL https://www.sciencedirect.com/science/article/pii/S2001037014000403.

Oldrich Vasicek. An equilibrium characterization of the term structure. *Journal of Financial Economics*, 5(2):177–188, 1977. ISSN 0304-405X. doi: https://doi.org/10.1016/0304-405X(77)90016-2. URL https://www.sciencedirect.com/science/article/pii/0304405X77900162.

A IMPLEMENTATION

In this section, we will discuss in details how the algorithm is implemented for our learning framework presented in section 2. In real applications, continuous data is rarely obtainable, we will have to deal with discrete observation data, i.e. $\{\mathbf{x}_l^m\}_{l,m=1}^{L,M}$ with $\mathbf{x}_0^m$ being i.i.d sample from μ_0, where $\mathbf{x}_l^m = \mathbf{x}^{(m)}(t_l)$ and $0 = t_1 < \cdots < t_L = T$. We use a discretized version of 2,

$$\begin{aligned}\mathcal{E}_{L,M,\mathcal{H}}(\tilde{\boldsymbol{f}}) = \frac{1}{2TM}\sum_{l,m=1}^{L-1,M}\Big(&< \tilde{\boldsymbol{f}}(\mathbf{x}_l^m), \boldsymbol{D}^{-1}(\mathbf{x}_l^m)\tilde{\boldsymbol{f}}(\mathbf{x}_l^m) > (t_{l+1} - t_l) \\ &- 2 < \tilde{\boldsymbol{f}}(\mathbf{x}_l^m), \boldsymbol{D}^{-1}(\mathbf{x}_l^m)(\mathbf{x}_{l+1}^m - \mathbf{x}_l^m) > \Big),\end{aligned} \tag{7}$$

for $\tilde{\boldsymbol{f}} \in \mathcal{H}$. Moreover, we also assume that $\mathcal{H}$ is a finite-dimensional function space, i.e. $\dim(\mathcal{H}) = n < \infty$. Then for any $\tilde{\boldsymbol{f}} \in \mathcal{H}$, $\tilde{\boldsymbol{f}}(\boldsymbol{x}) = \sum_{i=1}^n \boldsymbol{a}_i\psi_i(\boldsymbol{x})$, where $\boldsymbol{a}_i \in \mathbb{R}^d$ is a constant vector coefficient and $\psi_i : \Omega \to \mathbb{R}$ is a basis of $\mathcal{H}$ and the domain Ω is constructed by finding out the $\min / \max$ of the components of $\mathbf{x}_t \in \mathbb{R}^d$ for $t \in [0, T]$. We consider two scenarios for constructing ψ_i, one is to use pre-determined basis such as piecewise polynomials, Clamped B-spline, Fourier basis, or a mixture of all of the aforementioned ones; the other is to use neural networks, where the basis functions are also trained from data. Next, we can put the basis representation of $\tilde{\boldsymbol{f}}$ back to equation 7, we obtain the following loss based on the coefficients

$$\begin{aligned}\mathcal{E}_{L,M,\mathcal{H}}(\{\boldsymbol{a}_\eta\}_{i=1}^n) = \frac{1}{2TM}\sum_{l,m=1}^{L-1,M}\Big(\sum_{i=1}^n\sum_{j=1}^n &< \boldsymbol{a}_i\psi_i(\mathbf{x}_l^m), \boldsymbol{D}^{-1}(\mathbf{x}_l^m)\boldsymbol{a}_j\psi_j(\mathbf{x}_l^m) > (t_{l+1} - t_l) \\ &- 2\sum_{i=1}^n < \boldsymbol{a}_i\psi_i(\mathbf{x}_l^m), D^{-1}(\mathbf{x}_l^m)(\mathbf{x}_{l+1}^m - \mathbf{x}_l^m) > \Big),\end{aligned} \tag{8}$$

In the case of covariance matrix $\boldsymbol{D}$ being a diagonal matrix, i.e.

$$\boldsymbol{D}(\boldsymbol{x}) = \begin{bmatrix}\sigma_1^2(\boldsymbol{x}) & 0 & \cdots & 0 \\ 0 & \sigma_2^2(\boldsymbol{x}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_d^2(\boldsymbol{x})\end{bmatrix} \in \mathbb{R}^{d\times d}, \quad \sigma_i > 0, \quad i = 1, \cdots, d.$$

Then equation 8 can be re-written as

$$\begin{aligned}\mathcal{E}_{L,M,\mathcal{H}}(\{\boldsymbol{a}_\eta\}_{i=1}^n) = \frac{1}{2TM}\sum_{l,m=1}^{L-1,M}\Big(&\sum_{i=1}^n\sum_{j=1}^n\sum_{k=1}^d \frac{(\boldsymbol{a}_i)_k(\boldsymbol{a}_j)_k}{\sigma_k^2(\mathbf{x}_l^m)}\psi_i(\mathbf{x}_l^m)\psi_j(\mathbf{x}_l^m)(t_{l+1} - t_l) \\ &- 2\sum_{i=1}^n\sum_{k=1}^d \frac{(\boldsymbol{a}_i)_k(\mathbf{x}_{l+1}^m - \mathbf{x}_l^m)_k}{\sigma_k^2(\mathbf{x}_l^m)}\psi_i(\mathbf{x}_l^m)\Big),\end{aligned}$$

Here $(\boldsymbol{x})_k$ is the k^{th} component of any vector $\boldsymbol{x} \in \mathbb{R}^d$. We define $\boldsymbol{\alpha}_k = [(\boldsymbol{a}_1)_k \quad \cdots \quad (\boldsymbol{a}_n)_k]^\top \in \mathbb{R}^n$, and $A_k \in \mathbb{R}^{n\times n}$ as

$$A_k(i,j) = \frac{1}{2TM}\sum_{l,m=1}^{L-1,M}\Big(\sum_{i=1}^n\sum_{j=1}^n \frac{(\boldsymbol{a}_i)_k(\boldsymbol{a}_j)_k}{\sigma_k^2(\mathbf{x}_l^m)}\psi_i(\mathbf{x}_l^m)\psi_j(\mathbf{x}_l^m)(t_{l+1} - t_l),$$

and $\boldsymbol{b}_k \in \mathbb{R}^n$ as

$$\boldsymbol{b}_k(i) = \sum_{i=1}^n \frac{(\boldsymbol{a}_i)_k(\mathbf{x}_{l+1}^m - \mathbf{x}_l^m)_k}{\sigma_k^2(\mathbf{x}_l^m)}\psi_i(\mathbf{x}_l^m)\Big).$$

Then equation 8 can be re-written as

$$\mathcal{E}_{L,M,\mathcal{H}}(\{\boldsymbol{a}_\eta\}_{i=1}^n) = \sum_{k=1}^d(\boldsymbol{\alpha}_k^\top A_k\boldsymbol{\alpha}_k - 2\boldsymbol{\alpha}_k^\top\boldsymbol{b}_k).$$

Since each $\boldsymbol{\alpha}_k^\top A_k \boldsymbol{\alpha}_k - 2\boldsymbol{\alpha}_k^\top \boldsymbol{b}_k$ is decoupled from each other, we just need to solve simultaneously

$$A_k \hat{\boldsymbol{\alpha}}_k - \boldsymbol{b}_k = 0, \quad k = 1, \cdots, d.$$

Then we can obtain $\hat{\boldsymbol{f}}(\boldsymbol{x}) = \sum_{i=1}^n \hat{\boldsymbol{a}}_i \psi_k(\boldsymbol{x})$. However when $\boldsymbol{D}$ does not have a diagonal structure, we will have to resolve to gradient descent methods to minimize equation 8 in order to find the coefficients $\{\boldsymbol{a}_i\}_{i=1}^n$ for a total number of nd parameters. However, if a data-driven basis is desired, then we set $\mathcal{H}$ to be the space neural networks with the same depth, same number of neurons and same activation functions on the hidden layers. Furthermore, we find $\hat{\boldsymbol{f}}$ from minimizing equation 7 using any deep learning optimizer such as Stochastic Gradient Descent or Adam from well-known deep learning packages.

B Additional Examples

For $d = 1$, we also worked on polynomial drift function $\boldsymbol{f} = 2 + 0.08\mathbf{x} - 0.01\mathbf{x}^2$. The estimation results are depicted in 4 and detailed in Table 4.

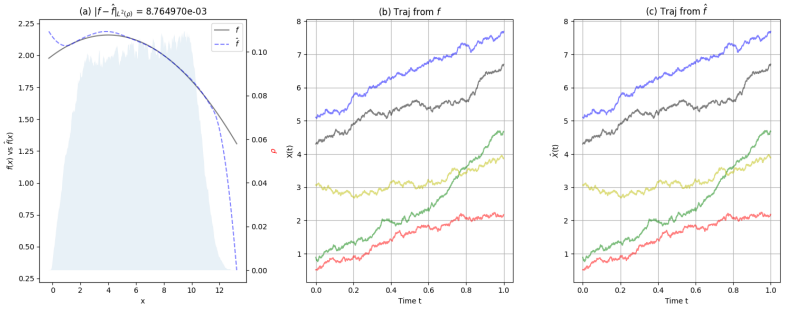


Figure 4: One-dimensional Polynomial Drift Comparison Summary

Table 4: One-dimensional Polynomial Drift Function Estimation Summary

Number of Basis	10	Wasserstein Distance	
Maximum Degree	2	$t = 0.25$	0.0153
Relative $L^2(\rho)$ Error	0.0087649	$t = 0.50$	0.0154
Relative Trajectory Error	$0.00199719 \pm 0.00682781$	$t = 1.00$	0.0278

For $d = 2$, we examine two types of drift function $\boldsymbol{f}$: polynomial and trigonometric. Denote

$$\boldsymbol{f} = \begin{pmatrix} \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \end{pmatrix} \quad \text{and} \quad \mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}$$

where $\boldsymbol{f}_i : \mathbb{R}^2 \to \mathbb{R}$ and $\mathbf{x}_i \in \mathbb{R}$ for $i \in \{1, 2\}$.

For polynomial drift function $\boldsymbol{f}$, we set

$$\boldsymbol{f}_1 = 0.4\mathbf{x}_1 - 0.1\mathbf{x}_1\mathbf{x}_2$$
$$\boldsymbol{f}_2 = -0.8\mathbf{x}_2 + 0.2\mathbf{x}_1^2.$$

Figure 5, Figure 6a, 6b and Table 5 shows evaluation of the polynomial drift function estimation result.

For trigonometric drift function $\boldsymbol{f}$, we set

$$\boldsymbol{f}_1 = 2\sin(0.2\mathbf{x}_1) + 1.5\cos(0.1\mathbf{x}_2)$$
$$\boldsymbol{f}_2 = 3\sin(0.3\mathbf{x}_1)\cos(0.1\mathbf{x}_2).$$

Figure 7, Figure 8a, 8b and Table 6 shows evaluation of the trigonometric drift function estimation result.

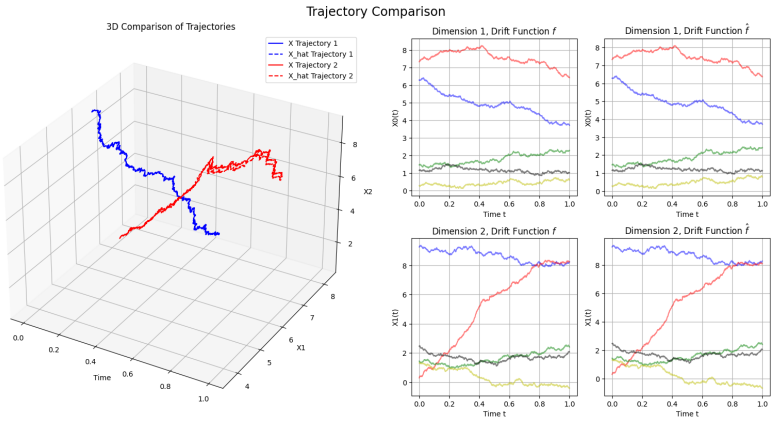
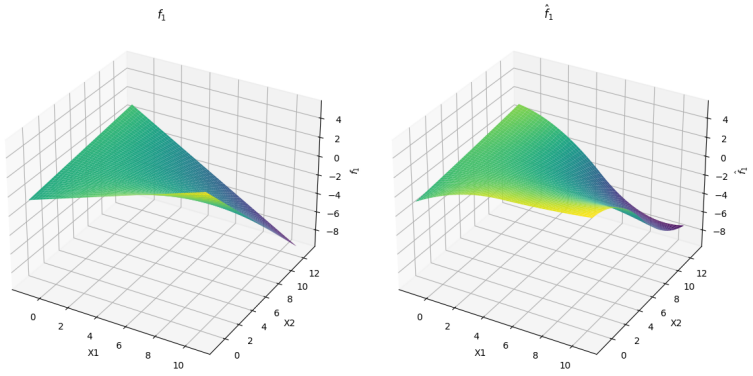
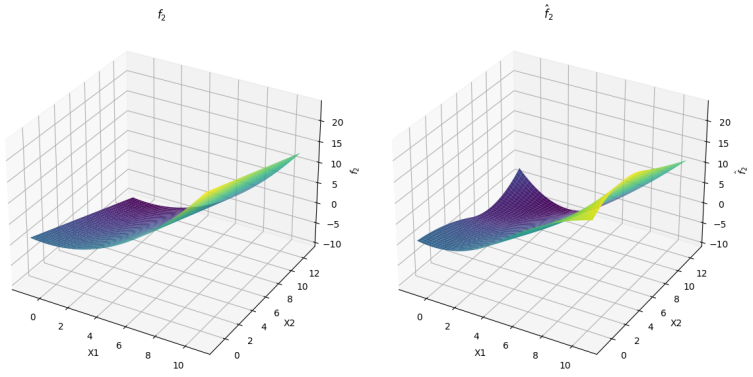


Figure 5: Two-dimensional Polynomial Trajectory Comparison



(a)



(b)

Figure 6: Two-dimensional Polynomial Comparison of $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$. (a) Surface of Dimension 1 (b) Surface of Dimension 2

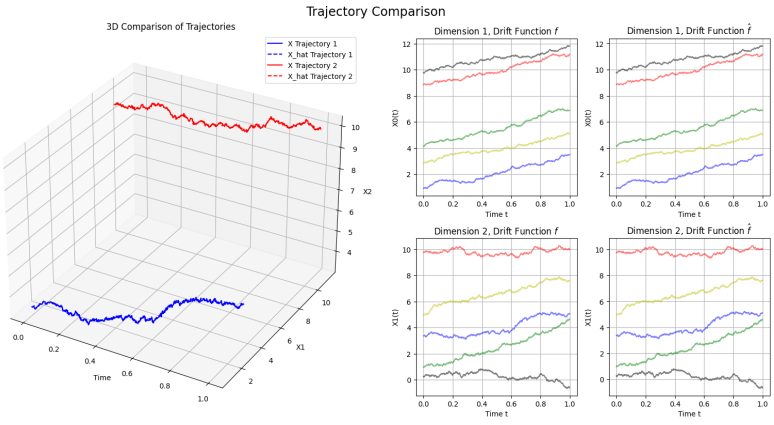
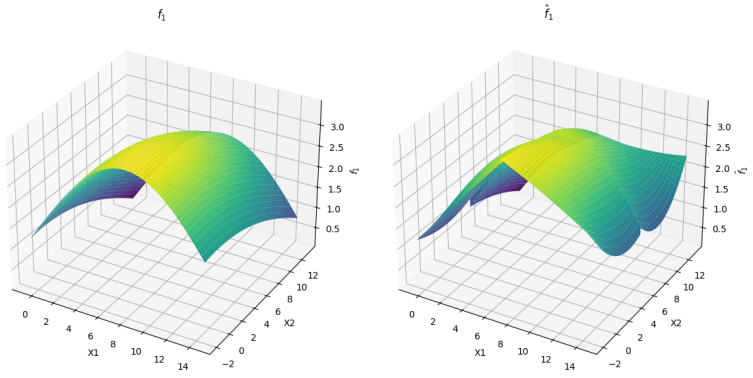
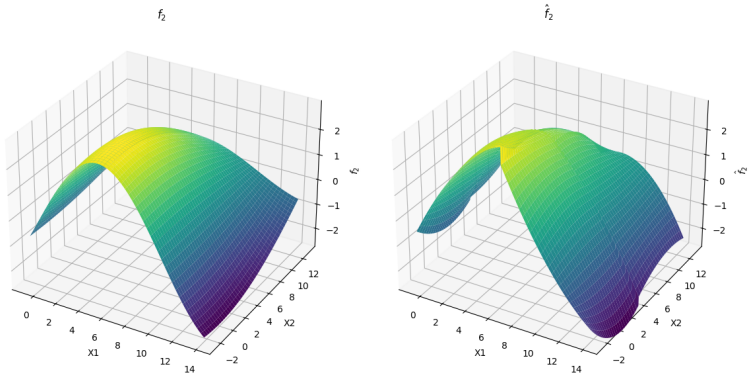


Figure 7: Two-dimensional Trigonometric Trajectory Comparison



(a)



(b)

Figure 8: Two-dimensional Trigonometric Comparison of $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$. (a) Surface of Dimension 1 (b) Surface of Dimension 2

Table 5: Two-dimensional Polynomial Drift Function Estimation Summary

Number of Basis	16	Wasserstein Distance	
Maximum Degree	2	$t = 0.25$	0.2338
Relative $L^2(\rho)$ Error	0.0449609	$t = 0.50$	0.2431
Relative Trajectory Error	0.0100373 ± 0.0230949	$t = 1.00$	0.2256

Table 6: Two-dimensional Trigonometric Drift Function Estimation Summary

Number of Basis	36	Wasserstein Distance	
Maximum Degree	2	$t = 0.25$	0.1011
Relative $L^2(\rho)$ Error	0.02734505	$t = 0.50$	0.1119
Relative Trajectory Error	0.0041613 ± 0.0079917	$t = 1.00$	0.1293