AReS and MaRS - Adversarial and MMD-Minimizing Regression for SDEs

Gabriele Abbati* GABB@ROBOTS.OX.AC.UK
Department of Engineering Science
University of Oxford
Philippe Wenk* WENKPH@ETHZ.CH
Learning and Adaptive Systems Group
ETH Zürich and Max Planck ETH Center for Learning Systems
Stefan Bauer STEFAN.BAUER@TUEBINGEN.MPG.DE
Empirical Inference Group
Max Planck Institute for Intelligent Systems, Tübingen
Michael A Osborne MOSB@ROBOTS.OX.AC.UK
Department of Engineering Science
University of Oxford
Andreas Krause KRAUSEA@ETHZ.CH
Learning and Adaptive Systems Group
ETH Zürich
Bernhard Schölkopf BERNHARD.SCHOELKOPF@TUEBINGEN.MPG.DE
Empirical Inference Group
Max Planck Institute for Intelligent Systems, Tübingen

Abstract

Stochastic differential equations are an important modeling class in many disciplines. Consequently, there exist many methods relying on various discretization and numerical integration schemes. In this paper, we propose a novel, probabilistic model for estimating the drift and diffusion given noisy observations of the underlying stochastic system. Using state-of-the-art adversarial and moment matching inference techniques, we circumvent the use of the discretization schemes as seen in classical approaches. This yields significant improvements in parameter estimation accuracy and robustness given random initial guesses. On four commonly used benchmark systems, we demonstrate the performance of our algorithms compared to state-of-the-art solutions based on extended Kalman filtering and Gaussian processes.

1. INTRODUCTION

Modeling discretely observed time series is a challenging problem that arises in quantitative sciences and many fields of engineering. While it is possible to tackle such problems with difference equations or ordinary differential equations (ODEs), both approaches suffer from serious drawbacks. Difference equations are difficult to apply if the observation times are unevenly distributed. Furthermore, they do not generalize well across observation frequencies, while natural laws do not care about this artificial construct. ODEs can deal with these challenges, but fail to incorporate the inherent stochastic

∗. The first two authors contributed equally.

behavior present in many physical, chemical or biological systems. These effects can be captured by introducing stochasticity in the dynamics model, which brings to stochastic differential equations (SDEs). In this paper, we will use exclusively the Itô-form

$$\mathrm{d}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), \boldsymbol{\theta})\mathrm{d}t + \mathbf{g}(\mathbf{x}(t), \boldsymbol{\theta})\mathrm{d}\mathbf{w}(t), \tag{1}$$

where $\mathbf{x}(t)$ is the time-dependent vector of states we would like to model, $\boldsymbol{\theta}$ collects the parameters of the model, $\mathbf{f}$ is the drift term, $\mathbf{g}$ is the vector-valued diffusion function and $\mathbf{w}(t)$ is a Wiener-process of the same dimension as the state vector $\mathbf{x}$.

While SDEs can efficiently capture stochasticity and deal with unevenly spaced observation times or observation frequency, inference is rather challenging. Due to the stochasticity of $\mathbf{w}(t)$, the state vector $\mathbf{x}(t)$ is itself a random variable. Except for few special cases, it is not possible to find an analytic solution for the statistics of $\mathbf{x}(t)$ for general drift and diffusion terms. The problem is even more challenging if we were to condition on or state-estimate some discrete time observations $\mathbf{y}$ (filtering/smoothing) or infer some statistics for the parameters $\boldsymbol{\theta}$ (parameter inference). It is well known that the parameter inference problem is a difficult task, with most approaches either being very sensitive to initialization (Picchini, 2007), strongly dependent on the choice of hyperparameters like the spacing of the integration grid (Bhat et al., 2015) or using excessive amount of computational resources even for small scale systems and state-of-the-art implementation (Ryder et al., 2018).

The difficulty of the parameter estimation problem of estimating parameters of drift and diffusion under observational noise is readily exemplified by the fact that even major scientific programming environment providers like MATLAB are still lack an established toolbox for practical use. In this paper, we will take a step into a novel direction tackling this open and exciting research question.

1.1 Related Work

While it is impossible to cover all related research efforts, we would like to give a quick overview by mentioning some of the most relevant. For a more in-depth discussion, we recommend Tronarp and Särkkä (2018), who provide an excellent review of the current state-of-the-art smoothing schemes. Moreover, Sørensen (2004) and Nielsen et al. (2000) provide extensive explanations of the more traditional approaches.

Most classical methods rely on calculating $p(\mathbf{x}|\boldsymbol{\theta})$. Since $p(\mathbf{x}|\boldsymbol{\theta})$ is usually analytically intractable, approximation schemes are necessary. Elerian et al. (2001) and Eraker (2001) used the Euler-Maruyama discretization to approximate $p(\mathbf{x}|\boldsymbol{\theta})$ on a fixed, fine grid of artificial observation times later to be leveraged in a MCMC sampling scheme. Pieschner and Fuchs (2018) and van der Meulen et al. (2017) subsequently refined this approach with improved bridge constructs and incorporated partial observability. Ryder et al. (2018) recently followed up on this idea by combining discretization procedures with variational inference. Särkkä et al. (2015) investigate different approximation methods based on Kalman filtering, while Archambeau et al. (2007) and Vrettas et al. (2015) use a variational Gaussian Process-based approximation. Finally, it should also be mentioned that $p(\mathbf{x}|\boldsymbol{\theta})$ can be inferred by solving the Fokker-Planck-Kolmogorov equations using standard numerical methods for PDEs (Hurn and Lindsay, 1999; Aït-Sahalia, 2002).

Instead of approximating $p(\mathbf{x}, \boldsymbol{\theta})$ in a variational fashion, Gaussian Processes can as well be used to directly model $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ and $\mathbf{g}(\mathbf{x}, \boldsymbol{\theta})$, ignoring in this way any prior knowledge about their parametric form. This approach was investigated by Ruttor et al. (2013), whose linearization and discretization assumptions which were later relaxed by Yildiz et al. (2018). While we will show in

our experiments that these methods can be used for parameter estimation if the parametric form of drift and diffusion are known, it should be noted that parameter inference was not the original goal of their work.

1.2 Our Work

To the best of our knowledge, there are only very few works that try to circumvent calculating $p(\mathbf{x}|\boldsymbol{\theta})$ at all. Our approach is probably most closely related to the ideas presented by Riesinger et al. (2016). Our proposal relies on the Doss-Sussman transformation (Doss, 1977; Sussmann, 1978) to reduce the parameter inference problem to parameter inference in an ensemble of random ordinary differential equations (RODEs). These equations can then be solved path-wise using either standard numerical schemes or using the computationally efficient gradient matching scheme of Gorbach et al. (2017) as proposed by Bauer et al. (2017).

The path-wise method of Bauer et al. (2017) has natural parallelization properties, but there is still an inherent approximation error due to the Monte Carlo estimation of the expectation over the stochastic element in the RODE. Furthermore, their framework imposes severe linearity restrictions on the functional form of the drift $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$, while it is unable to estimate the diffusion matrix $\mathbf{g}(\mathbf{x}, \boldsymbol{\theta})$.

While we will keep their assumption of a constant diffusion matrix, i.e. $\mathbf{g}(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{G}$, our approach gets rid of the linearity assumptions on the drift $\mathbf{f}$. Furthermore, we substitute the Monte Carlo approximation by embedding the SDE into a fully statistical framework, allowing for efficient estimation of both $\mathbf{G}$ and $\boldsymbol{\theta}$ using state-of-the-art statistical inference methods.

Even though a constant diffusion assumption might seem restrictive at first, such SDE models are widely used, e.g. in chemical engineering (Karimi and McAuley, 2018), civil engineering (Jiménez et al., 2008), pharmacology (Donnet and Samson, 2013) and of course in signal processing, control and econometrics. While we believe that this approach could be extended approximately to systems with general diffusion matrices, we leave this for future work.

The contributions of our framework are the following:

- We derive a new statistical framework for diffusion and drift parameter estimation of SDEs using the Doss-Sussmann transformation and Gaussian Processes.
- We introduce a grid-free, computationally efficient and robust parameter inference scheme that combines a non-parametric Gaussian Process model with adversarial loss functions.
- We demonstrate that our method is able to estimate constant but non-diagonal diffusion terms of stochastic differential equations without any functional form assumption on the drift.
- We show that our method significantly outperforms the state-of-the-art algorithms for SDEs with multi-modal state posteriors, both in terms of diffusion and drift parameter estimation.
- We share and publish our code to facilitate future research.[1]

2. BACKGROUND

In this section, we formalize our problem and introduce the necessary notation and background drawn from Gaussian process-based gradient matching for ODEs.

1. Code available at https://github.com/gabb7/AReS-MaRS

2.1 Problem Setting

We consider SDEs of the form

$$\mathrm{d}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), \boldsymbol{\theta})\mathrm{d}t + \mathbf{G}\mathrm{d}\mathbf{w}(t), \tag{2}$$

where $\mathbf{x}(t) = [\mathbf{x}_1(t), \dots, \mathbf{x}_K(t)]^\top$ is the K-dimensional state vector at time t; $\mathrm{d}\mathbf{w}(t)$ are the increments of a standard Wiener process; $\mathbf{f}$ is an arbitrary, potentially highly nonlinear function whose parametric form is known, save for the unknown parameter vector $\boldsymbol{\theta}$; $\mathbf{G}$ is the unknown but constant diffusion matrix. Without loss of generality, we can assume $\mathbf{G}$ to be a lower-diagonal, positive semi-definite matrix.

The system is observed at N arbitrarily spaced time points $\mathbf{t} = [t_1, \dots, t_N]$, subjected to Gaussian observation noise:

$$\mathbf{y}(t_n) = \mathbf{x}(t_n) + \mathbf{e}(t_n) \quad \forall\, n = 1, \dots, N, \tag{3}$$

where we assume the noise variances to be state-dependent but time-independent, i.e.

$$p(\mathbf{e}(t_n)) = \prod_{k=1}^{K} \mathcal{N}\left(\mathbf{e}_k(t_n) \mid 0, \sigma_k\right), \tag{4}$$

for $n = 1, \dots, N$ and $k = 1, \dots, K$.

2.2 Deterministic ODE Case

In the context of Bayesian parameter inference for deterministic ordinary differential equations, Calderhead et al. (2009) identify numerical integration as the main culprit for bad computational performance. Thus, they propose to turn the parameter estimation procedure on its head: instead of calculating $p(\mathbf{y} \mid \boldsymbol{\theta})$ using numerical integration, they extract two probabilistic estimates for the derivatives, one using only the noisy observations $\mathbf{y}$ and one using the differential equations. The main challenge is then to combine these two distributions, such that more information about $\mathbf{y}$ can guide towards good parameter estimates $\boldsymbol{\theta}$. For this purpose, Calderhead et al. (2009) propose a product of experts heuristics that was accepted and reused until recently Wenk et al. (2018) showed that this heuristic leads to severe theoretical issues. They instead propose an alternative graphical model, forcing equality between the data based and the ODE based model save for a Gaussian distributed slack variable.

In this paper, we use another interpretation of gradient matching, which is aimed at finding parameters $\boldsymbol{\theta}$ such that the two distributions over $\dot{\mathbf{x}}$ match as closely as possible. Acknowledging the fact that standard methods like minimizing the KL divergence are not tractable, we use robust moment matching techniques while solving a much harder problem with $\mathbf{G} \neq \mathbf{0}$. However, it should be clear that our methodology could easily be applied to the special case of deterministic ODEs and thus provides an additional contribution towards parameter estimation for systems of ODEs.

2.3 Notation

Throughout this paper, bold, capital letters describe matrices. Values of a time-dependent quantities such as the state vector $\mathbf{x}(t)$ can be collected in the matrix $\mathbf{X} = [\mathbf{x}(t_1), \dots, \mathbf{x}(t_n)]$ of dimensions $K \times N$, where the k-th row collect the N single-state values at times $\mathbf{t} = [t_1, \dots, t_n]$ for the state k.

The matrix $\mathbf{X}$ can be vectorized by concatenating its rows and defining in this way the vector $\mathbf{x} = [\mathbf{x}_1, \ldots, \mathbf{x}_K]^\top$. This vector should not be confused with $\mathbf{x}(t)$, which is still a time-dependent vector of dimension K.

As we work with Gaussian processes, it is useful to standardize the state observations by subtracting the mean and dividing by the standard deviation, in a state-wise fashion. We define the vector of the data standard deviation $\boldsymbol{\sigma}_\mathbf{y} = [\sigma_{\mathbf{y}_1}, \ldots, \sigma_{\mathbf{y}_K}]$, and the matrix $\mathbf{S}$ as:

$$\mathbf{S} = \boldsymbol{\sigma}_\mathbf{y} \otimes \mathbf{I}_N \tag{5}$$

where $\otimes$ indicates the Kronecker product and $\mathbf{I}_N$ is the identity matrix of size $N \times N$. Similarly for the means, we can define the $N \times K$ vector $\boldsymbol{\mu}_\mathbf{y}$ that contains the K state-wise means of the observations, each repeated N times. Thus the standardize vector $\tilde{\mathbf{x}}$ can be defined as:

$$\tilde{\mathbf{x}} = \mathbf{S}^{-1}(\mathbf{x} - \boldsymbol{\mu}_\mathbf{y}). \tag{6}$$

For the sake of clarity, we omit the normalization in the following sections. It should be noted however that in a finite sample setting, standardization strongly improves the performance of GP regression. In both our implementation and all the experiments shown in section 4, we assume a GP prior on $\mathbf{z}$ that were standardized using the state-wise mean and standard deviation of the observations $\mathbf{y}$.

For coherence with the current Gaussian process-based gradient matching literature, we follow the notation introduced by Calderhead et al. (2009) and Wenk et al. (2018) wherever possible.

3. METHODS

In the deterministic case with $\mathbf{G} = \mathbf{0}$, the GP regression model can be directly applied to the states $\mathbf{x}$. However, if $\mathbf{G} \neq \mathbf{0}$, the derivatives of the states with respect to time t do no longer exist due to the contributions of the Wiener process. Thus, performing direct gradient matching on the states is not feasible.

3.1 Latent States Representation

We propose to tackle this problem by introducing a latent variable $\mathbf{z}$, defined via the linear coordinate transformation

$$\mathbf{z}(t) = \mathbf{x}(t) - \mathbf{o}(t), \tag{7}$$

where $\mathbf{o}(t)$ is the solution of the following SDE:

$$\mathrm{d}\mathbf{o}(t) = -\mathbf{o}(t) + \mathbf{G}\mathrm{d}\mathbf{w}(t). \tag{8}$$

Without loss of generality, we set $\mathbf{z}(0) = \mathbf{x}(0)$ and thus $\mathbf{o}(0) = 0$. While in principle the framework supports any initial condition as long as $\mathbf{z}(0) + \mathbf{o}(0) = \mathbf{x}(0)$, the reasons for this choice will become clear in section 3.2.3.

Using Itô's formula, we obtain the following SDE for $\mathbf{z}$

$$\mathrm{d}\mathbf{z}(t) = \{\mathbf{f}(\mathbf{z}(t) + \mathbf{o}(t), \boldsymbol{\theta}) + \mathbf{o}(t)\}\,\mathrm{d}t \tag{9}$$

This means that for a given realization of $\mathbf{o}(t)$, we obtain a differentiable latent state $\mathbf{z}(t)$. In principle, we could sample realizations of $\mathbf{o}(t)$ and solve the corresponding deterministic problems, which is

equivalent to approximately marginalizing over $\mathbf{o}(t)$. However, it is actually possible to treat this problem statistically, completely bypassing such marginalization. We do this by creating probabilistic generative models for observations and derivatives analytically. The equations are derived in this section, while the final models are shown in Figure 1.

3.2 Generative Model for Observations

Let us define $\mathbf{e}(t)$ as the Gaussian observation error at time t. Using the matrix notation introduced in section 2.3, we can write

$$\mathbf{Y} = \mathbf{X} + \mathbf{E} = \mathbf{Z} + \mathbf{O} + \mathbf{E}, \tag{10}$$

where $\mathbf{Z}$ and $\mathbf{O}$ are the matrices corresponding to the lower-case variables introduced in the previous section. In contrast to standard GP regression, we have an additional additive noise term $\mathbf{O}$, which is the result of the stochastic process described by equation (8). As in standard GP regression, it is possible to recover a closed form Gaussian distribution for each term.

3.2.1 GP PRIOR

We assume a zero-mean Gaussian prior over the latent states $\mathbf{z}$, whose covariance matrix is given by a kernel function $k(x, y)$, in turn parameterized by the hyperparameter vector $\boldsymbol{\phi}$:

$$p(\mathbf{z} \mid \boldsymbol{\phi}) = \mathcal{N}\left(\mathbf{z} \mid \mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}}\right). \tag{11}$$

We treat all state dimensions as independent, meaning that we put independent GP priors with separate hyperparameters $\boldsymbol{\phi}_k$ on the time evolution of each state. Consequently, $\mathbf{C}_{\boldsymbol{\phi}}$ will be a block diagonal matrix with K blocks each of dimension $N \times N$. The blocks model the correlation over time introduced by the GP prior.

3.2.2 ERROR MODEL

In equation (4), we assume that our errors are i.i.d. Gaussians uncorrelated over time. The joint distribution of all errors is thus still a Gaussian distribution, whose covariance $\mathbf{T}$ has only diagonal elements given by the GP likelihood variances $\boldsymbol{\sigma} = \{\sigma_k^2\}_{k=1}^K$. More precisely:

$$\mathbf{T} = \boldsymbol{\sigma} \otimes \mathbf{I}_N. \tag{12}$$

and

$$p(\mathbf{e} \mid \boldsymbol{\sigma}) = \mathcal{N}\left(\mathbf{e} \mid \mathbf{0}, \mathbf{T}\right). \tag{13}$$

3.2.3 ORNSTEIN-UHLENBECK PROCESS

Through the coordinate transformation in equation (7), all stochasticity is captured by the stochastic process $\mathbf{o}(t)$ described by equation (8). Such mathematical construct has a closed-form, Gaussian solution and is called Ornstein-Uhlenbeck process. For the one-dimensional case with zero initial condition and unit diffusion

$$\mathrm{d}\hat{o}(t) = -\hat{o}(t) + \mathrm{d}w(t), \tag{14}$$

we get the following mean and covariance:

$$\mathbb{E}[\hat{o}(t)] = 0 \tag{15}$$

$$\mathrm{cov}[\hat{o}(t_i), \hat{o}(t_j)] = \frac{1}{2} e^{-|t_i - t_j|} - \frac{1}{2} e^{-(t_i + t_j)}. \tag{16}$$

Sampling $\hat{o}(t)$ at the N discrete sample points $\mathbf{t} = [t_1, \ldots, t_N]$ yields the vector $\hat{\mathbf{o}}(\mathbf{t}) = [\hat{o}(t_1), \ldots, \hat{o}(t_N)]$, which is Gaussian distributed:

$$p(\hat{\mathbf{o}}(\mathbf{t})) = \mathcal{N}(\hat{\mathbf{o}}(\mathbf{t}) \mid \mathbf{0}, \mathbf{\Omega}_{\text{one}}), \tag{17}$$

where $[\mathbf{\Omega}_{\text{one}}]_{ij} = \text{cov}[\hat{o}(t_i), \hat{o}(t_j)]$ as given by equation (16). In the case of a K-dimensional process with identity diffusion, i.e.

$$\mathrm{d}\hat{\mathbf{o}}(t) = -\hat{\mathbf{o}} + \mathbf{I}_K \mathrm{d}\mathbf{w}(t), \tag{18}$$

we can just treat each state dimension as an independent, one-dimensional OU process. Thus, after sampling $\hat{\mathbf{o}}(t)$ K times at the N time points in $\mathbf{t}$ and unrolling the resulting matrix as described in section 2.3, we get

$$p(\hat{\mathbf{o}}) = \mathcal{N}(\hat{\mathbf{o}} \mid \mathbf{0}, \mathbf{\Omega}), \tag{19}$$

where $\mathbf{\Omega}$ is a block diagonal matrix with one $\mathbf{\Omega}_{\text{one}}$ for each state dimension.

Using Itô's formula, we can show that the samples of the original Ornstein-Uhlenbeck process $\mathbf{o}$ at each time point can be obtained via the linear coordinate transformation

$$\mathbf{o}(t) = \mathbf{G}\hat{\mathbf{o}}(t). \tag{20}$$

Let $\mathbf{B}$ be defined as the matrix that performs this linear transformation for the unrolled vectors $\mathbf{o} = \mathbf{B}\hat{\mathbf{o}}$. We can then write the density of the original Ornstein-Uhlenbeck process as

$$p(\mathbf{o} \mid \mathbf{G}) = \mathcal{N}\left(\mathbf{o} \mid \mathbf{0}, \mathbf{B}\mathbf{\Omega}\mathbf{B}^\top\right). \tag{21}$$

3.2.4 MARGINALS OF THE OBSERVATIONS

Using equation (10), the marginal distribution of $\mathbf{y}$ can be computed as the sum of three independent Gaussian-distributed random variables with zero mean, described respectively by equations (11), (13) and (21). Thus, $\mathbf{y}$ is again Gaussian-distributed, according to

$$p(\tilde{\mathbf{y}} \mid \boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) = \mathcal{N}(\mathbf{y} \mid \mathbf{0}, \mathbf{\Sigma}), \tag{22}$$

where

$$\mathbf{\Sigma} = \mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T} + \mathbf{B}\mathbf{\Omega}\mathbf{B}^T. \tag{23}$$

It is important to note that due to the latent state representation, the diffusion matrix $\mathbf{G}$ is now a part of the hyperparameters of the observation model. It will later be inferred alongside the hyperparameters of our GP using maximum evidence (Rasmussen, 2004). Using a stationary kernel k, $\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T}$ captures the stationary part of $\mathbf{z}$ as in standard GP regression, while the parameters in $\mathbf{G}$ describe the non-stationary part due deriving from $\mathbf{\Omega}$. This ultimately leads to an identifiable problem.

3.3 Generative Model for Derivatives

Similarly to gradient matching approaches, we create two generative models for our derivatives, one based on the data and one based on the SDE model.

(a) SDE-based model (b) Data based model

Figure 1: Generative models for the two different ways to compute the derivatives of the latent states $\mathbf{z}$.

3.3.1 DATA-BASED MODEL

As shown e.g. in the appendix of Wenk et al. (2018), the prior defined in equation (11) automatically induces a GP prior on the conditional derivatives of $\mathbf{z}$. Defining

$$\mathbf{D} \coloneqq {}'\mathbf{C}_{\boldsymbol{\phi}}\mathbf{C}_{\boldsymbol{\phi}}^{-1}, \tag{24}$$

$$\mathbf{A} \coloneqq \mathbf{C}''_{\boldsymbol{\phi}} - {}'\mathbf{C}_{\boldsymbol{\phi}}\mathbf{C}_{\boldsymbol{\phi}}^{-1}\mathbf{C}'_{\boldsymbol{\phi}} \tag{25}$$

where

$$\left[{}'\mathbf{C}_{\boldsymbol{\phi}}\right]_{i,j} \coloneqq \frac{\partial}{\partial a} k_{\boldsymbol{\phi}}(a,b)\Big|_{a=t_i,b=t_j}, \tag{26}$$

$$\left[\mathbf{C}'_{\boldsymbol{\phi}}\right]_{i,j} \coloneqq \frac{\partial}{\partial b} k_{\boldsymbol{\phi}}(a,b)\Big|_{a=t_i,b=t_j}, \tag{27}$$

$$\left[\mathbf{C}''_{\boldsymbol{\phi}}\right]_{i,j} \coloneqq \frac{\partial^2}{\partial a \partial b} k_{\boldsymbol{\phi}}(a,b)\Big|_{a=t_i,b=t_j}. \tag{28}$$

we can write

$$p(\dot{\mathbf{z}} \mid \mathbf{z}, \boldsymbol{\phi}) = \mathcal{N}\left(\dot{\mathbf{z}} \mid \mathbf{D}\mathbf{z}, \mathbf{A}\right). \tag{29}$$

3.3.2 SDE-BASED MODEL

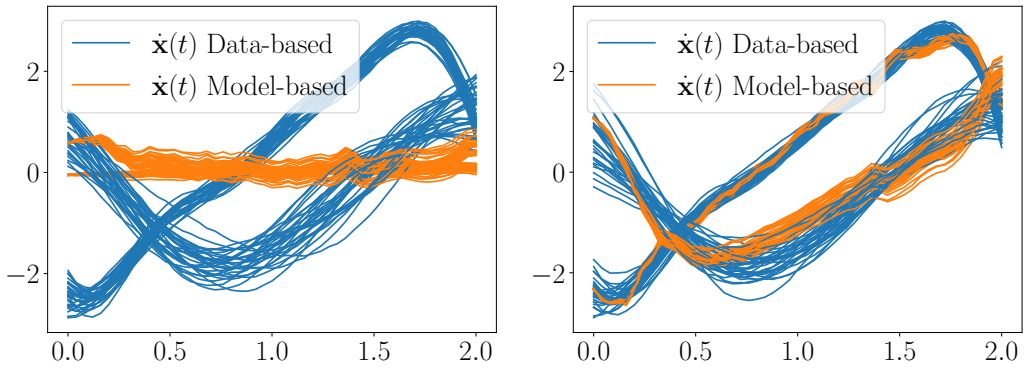
There also is a second way of obtaining an expression for the derivatives of $\mathbf{z}$, namely using equation (9):

$$p(\dot{\mathbf{z}} \mid \mathbf{o}, \mathbf{z}, \boldsymbol{\theta}) = \delta(\dot{\mathbf{z}} - \mathbf{f}(\mathbf{z} + \mathbf{o}, \boldsymbol{\theta}) - \mathbf{o}), \tag{30}$$

where δ represents the dirac delta.

3.4 Inference

Combined with the modeling paradigms introduced in the previous sections, this yields two generative models for the observations, as pictured in Figure 1. The graphical model in Figure 1a represents the derivatives we get via the generative process described by the SDEs via the nonlinear drift function $\mathbf{f}$. The model in Figure 1b represents the derivatives yielded by the generative process described by the



(a) Before Training (b) After Training

Figure 2: Comparing the sampled gradient from the graphical model in Figure 1a (Model-based) and the graphical model in Figure 1b (Data-based) before and after adversarial training on Lotka Volterra.

Gaussian process. Assuming a perfect GP fit and access to the true parameters $\boldsymbol{\theta}$, intuitively these two distributions should be equal. We thus want to find parameters $\boldsymbol{\theta}$ that minimizes the difference between these two distributions.

Compared to the deterministic ODE case, the graphical models in Figure 1 contain additional dependencies on the contribution of the Ornstein-Uhlenbeck process $\mathbf{o}$. Furthermore, the SDE-driven probability distribution of $\dot{\mathbf{z}}$ in Figure 1a depends on $\mathbf{z}$, $\mathbf{o}$, and $\boldsymbol{\theta}$ through a potentially highly nonlinear drift function $\mathbf{f}$. Thus, it is not possible to do analytical inference without making restrictive assumptions on the functional form of $\mathbf{f}$.

However, as shown in the appendix in section A.2, it is possible to derive computationally efficient ancestral sampling schemes for both models, as summarized in Algorithm 1. While this rules out classical approaches like analytically minimizing the KL divergence, we are now able to deploy likelihood-free algorithms that were designed for matching two probability densities based on samples.

Algorithm 1 Ancestral sampling for $\dot{\mathbf{z}}$

Input: $\mathbf{y}, \mathbf{f}(\mathbf{z}, \boldsymbol{\theta}), \mathbf{t}, \boldsymbol{\sigma}, \mathbf{G}$
Ancestral sampling the SDE model
Sample $\mathbf{o}_\mathrm{s}$ by drawing from $p(\mathbf{o} \mid \mathbf{G})$
Sample $\mathbf{z}_\mathrm{s}$ by drawing from $p(\mathbf{z} \mid \mathbf{y}, \mathbf{o}, \boldsymbol{\sigma})$ using $\mathbf{o}_\mathrm{s}$
Sample $\dot{\mathbf{z}}_\mathrm{s}$ by drawing from $p(\dot{\mathbf{z}} \mid \mathbf{o}, \mathbf{z}, \boldsymbol{\theta})$ using $\mathbf{o}_\mathrm{s}, \mathbf{z}_\mathrm{s}$
Ancestral sampling the Data model
Sample $\mathbf{z}_\mathrm{d}$ by drawing from $p(\mathbf{z} \mid \mathbf{y}, \mathbf{G}, \boldsymbol{\sigma})$
Sample $\dot{\mathbf{z}}_\mathrm{d}$ by drawing from $p(\dot{\mathbf{z}} \mid \mathbf{z}, \boldsymbol{\phi})$
Return: $\dot{\mathbf{z}}_\mathrm{s}, \dot{\mathbf{z}}_\mathrm{d}$

3.5 Adversarial Sample-based Inference

Arguably, among the most popular algorithms of this kind, one can find the generative adversarial network (GAN), where a parametric neural network is trained to match the unknown likelihood of our data (Goodfellow et al., 2014). The basic GAN setup consists of a fixed data set, a generator that tries to create realistic samples of said dataset and a discriminator that tries to tell the fake samples from the true ones. As recently shown by Yang et al. (2018), GANs have a huge potential for solving stochastic partial differential equations (SPDEs). Yang et al. (2018) assume a fixed data set consisting of observations (similar to our $\mathbf{y}$) and use an SPDE-inspired neural network as a generator for realistic observations. In the case of SDEs however, this would still involve a lot of numerical integration. Thus, we modify the GAN setup by leaving behind the idea of having a fixed data set. Instead of relying on bootstrapped samples of our observations, we sample the derivatives of our data based model shown in Figure 1b, assuming a sufficiently good model fit, thus representing the true derivatives of our latent variable $\mathbf{z}$. We then use the SDE-based model shown in Figure 1a as a generator. To avoid standard GAN problems such as training instability and to improve robustness, we choose to replace the discriminator with a critic, trained to estimate the Wasserstein distance between samples as proposed by Arjovsky et al. (2017). The resulting algorithm, summarized in Algorithm 2, can be interpreted as performing Adversarial Regression for SDEs and will thus be called AReS. In Figure 2, we show the derivatives sampled from the two models both before and after training for one example run of the Lotka Volterra system (cf. Section 4.4). While not perfect, the GAN is clearly able to push the SDE gradients towards the gradients of the observed data.

3.6 Maximum Mean Discrepancy

Even though GANs work well in practical settings, they need careful balancing between their generator and discriminator. Dziugaite et al. (2015) propose to solve this problem using Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) as a metric to substitute the discriminator. As proposed by Li et al. (2015), we choose a rational quadratic kernel to obtain a robust discriminator that can be deployed without fine-tuning on a variety of problems. The resulting algorithm, summarized in Algorithm 3, can be interpreted as performing Maximum mean discrepancy-minimizing Regression for SDEs and will thus be called MaRS.

4. EXPERIMENTS

4.1 Setups

To evaluate the empirical performance of our method, we conduct several experiments on simulated data, using four standard benchmark systems and comparing against the EKF-based approach of Särkkä et al. (2015) and the two GP-based approaches by Vrettas et al. (2015) and Yildiz et al. (2018).

The first system is a simple Ornstein-Uhlenbeck process shown in Figure 3a, given by the SDE

$$\mathrm{d}x(t) = \theta_0(\theta_1 - x(t))\mathrm{d}t + G\mathrm{d}w(t). \tag{31}$$

As mentioned in Section 3.2.3, this system has an analytical Gaussian process solution and serves thus more academic purposes. We simulate our dataset using $\boldsymbol{\theta} = [0.5, 1.0]$, $G = 0.5$ and $x(0) = 10$.

Algorithm 2 AReS

Input: Observations $\mathbf{y}$ at times $\mathbf{t}$, a model $\mathbf{f}$, learning rate α, number of total iterations N_{it}, the clipping parameter c, the batch size M, the number of iterations of the critic per generator iteration n_{critic}.
Train the Gaussian Process on the data to recover the hyperparameters $\boldsymbol{\phi}$, $\boldsymbol{\sigma}$ and the diffusion $\mathbf{G}$
Initialize the initial critic parameters $\boldsymbol{\omega}_0$ and the initial SDE parameters $\boldsymbol{\theta}_0$
for $n_{\text{it}} = 1, \dots, N_{\text{it}}$ **do**
    for $n_{\text{c}} = 1, \dots, n_{\text{critic}}$ **do**
        Sample $\dot{\mathbf{z}}_{\text{s}} \sim p_{\text{s}}(\dot{\mathbf{z}})$ and $\dot{\mathbf{z}}_{\text{d}} \sim p_{\text{d}}(\dot{\mathbf{z}})$ as described in algorithm 1
        $g_{\boldsymbol{\omega}} \leftarrow \nabla_{\boldsymbol{\omega}} \left[\frac{1}{M} \sum_{i=1}^{M} f_{\boldsymbol{\omega}}(\dot{\mathbf{z}}_{\text{d}}^{(i)}) - \frac{1}{M} \sum_{i=1}^{M} f_{\boldsymbol{\omega}}(\dot{\mathbf{z}}_{\text{s}}^{(i)}) \right]$
        $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + \alpha \cdot \text{Adam}(\boldsymbol{\omega}, g_{\boldsymbol{\omega}})$
        $\boldsymbol{\omega} \leftarrow \text{clip}(\boldsymbol{\omega}, -c, c)$
    end for
    $g_{\boldsymbol{\theta}} \leftarrow -\nabla_{\boldsymbol{\theta}} \frac{1}{M} \sum_{i=1}^{M} f_{\boldsymbol{\omega}}(\dot{\mathbf{z}}_{\text{s}}^{(i)})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \cdot \text{Adam}(\boldsymbol{\theta}, g_{\boldsymbol{\theta}})$
end for

Algorithm 3 MaRS

Input: Observations $\mathbf{y}$ at times $\mathbf{t}$, SDE model $\mathbf{f}$, learning rate α, number of iterations N_{it}, batch size M
Train the Gaussian Process on the data to recover the hyperparameters $\boldsymbol{\phi}$, $\boldsymbol{\sigma}$ and the diffusion $\mathbf{G}$
Initialize the initial SDE parameters $\boldsymbol{\theta}_0$
for $n_{\text{it}} = 1, \dots, N_{\text{it}}$ **do**
    Sample $\dot{\mathbf{z}}_{\text{s}} \sim p_{\text{s}}(\dot{\mathbf{z}})$ and $\dot{\mathbf{z}}_{\text{d}} \sim p_{\text{d}}(\dot{\mathbf{z}})$ as described in algorithm 1. Each batch contains M elements
    $g_{\boldsymbol{\theta}} \leftarrow \nabla_{\boldsymbol{\theta}} MMD_u^2 [\dot{\mathbf{z}}_{\text{s}}, \dot{\mathbf{z}}_{\text{d}}]$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \cdot \text{Adam}(\boldsymbol{\theta}, g_{\boldsymbol{\theta}})$
end for

The second system is the Lorenz63 model given by the SDEs

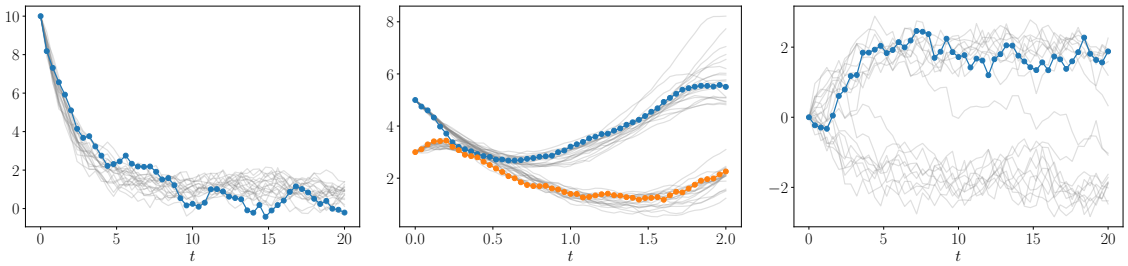
$$\begin{aligned} dx_1(t) &= \theta_1(x_2(t) - x_1(t))dt &&+ \sigma_1 dw_1(t) \\ dx_2(t) &= (\theta_2 x_1(t) - x_2(t) - x_1(t)x_3(t))dt &&+ \sigma_2 dw_2(t) \\ dx_3(t) &= (x_1(t)x_2(t) - \theta_3 x_3(t))dt &&+ \sigma_3 dw_3(t). \end{aligned}$$

In both systems, the drift function is linear in one state or one parameter conditioned on all the others (cf. Gorbach et al., 2017). Furthermore, there is no coupling across state dimensions in the diffusion matrix. This leads to two more interesting test cases.

To investigate the algorithm's capability to deal with off-diagonal terms in the diffusion, we introduce the two dimensional Lotka-Volterra system shown in Figure 3b, given by the SDEs

$$\mathrm{d}\mathbf{x}(t) = \begin{bmatrix} \theta_1 x_1(t) - \theta_2 x_1(t) x_2(t) \\ -\theta_3 x_2(t) + \theta_4 x_1(t) x_2(t) \end{bmatrix} \mathrm{d}t + \mathbf{G}\mathrm{d}\mathbf{w}(t), \tag{32}$$

where $\mathbf{G}$ is, without loss of generality, assumed to be a lower triangular matrix. The true vector parameter is $\boldsymbol{\theta} = [2, 1, 4, 1]$ and the system is simulated starting from $\mathbf{x}(0) = [3, 5]$. Since its original



(a) Ornstein Uhlenbeck (b) Lotka Volterra (c) Double Well

Figure 3: Sample trajectories for three different benchmark systems. While Ornstein Uhlenbeck and Lotka Volterra are rather tame, the Double Well system clearly exhibits a bifurcation effect.

introduction by Lotka (1932), the Lotka Volterra system has been widely used to model population dynamics in biology. The system is observed at 50 equidistant points in the interval $t = [0,\ 20]$. As it turns out, this problem is significantly challenging for all algorithms, even without the presence of observation noise.

To investigate the effect of strong non-linearities in the drift, we introduce the Ginzburg-Landau double-well potential shown in Figure 3c, defined by the SDE

$$\mathrm{d}x(t) = \theta_0 x(\theta_1 - x^2)\mathrm{d}t + G\mathrm{d}w(t). \tag{33}$$

Using $\boldsymbol{\theta} = [0.1, 4]$, $G = 0.5$ and $x(0) = 0$, this system exhibits an interesting bifurcation effect. While there are two stable equilibria at $x = \pm 2$, it is completely up to noise in which one the system will end up. This is thus the perfect system to test how well an algorithm can deal with multi-modal SDEs. The system is observed at 50 equidistant points in the interval $t = [0,\ 20]$, subjected to observational noise with $\sigma = 0.2$.

Lastly, some implementation details are constant throughout each experiment: the critic in the adversarial parameter estimation is a 2-layer fully connected neural network, with respectively 256 and 128 nodes. Every batch, for both MMD and adversarial training contains 256 elements. While Ornstein-Uhlenbeck and the double-well potential were modeled with a sigmoid kernel, for the Lotka Volterra we used a common RBF (we point at Rasmussen (2004) for more information about kernels and GPs).

4.2 Evaluation

For all systems, the parameters $\boldsymbol{\theta}$ turn out to be identifiable. Thus, the parameter value is a good indicator of how well an algorithm is able to infer the drift function. However, since the different dimensions of $d\mathbf{w}(t)$ are independent, there are multiple diffusion matrices $\mathbf{G}$ that create the same trajectories. We thus directly compare the variance of the increments, i.e. the components of $\mathbf{H} \coloneqq \mathbf{G}^T\mathbf{G}$.

To account for statistical fluctuations, we use 100 independent realizations of the SDE systems and compare the mean and standard deviation of $\boldsymbol{\theta}$ and $\mathbf{H}$. AReS and MaRS are compared against the Gaussian process-based VGPA by Vrettas et al. (2015) and NPSDE by Yildiz et al. (2018) as well as the classic Kalman filter-based ESGF recommended by Särkkä et al. (2015).

4.3 Locally Linear Systems

As mentioned in Section 4.1, the functional form of the drift functions of both the Ornstein Uhlenbeck process and the Lorenz 63 system satisfy a local linearity assumption, while their diffusion is kept diagonal. They thus serve as excellent benchmark systems for a wide variety of parameter inference algorithms. The empirical results are shown in Table 1 for the Ornstein-Uhlenbeck. Unfortunately, VGPA turns out to be rather unstable if both diffusion and parameters are unknown, even though it takes on average roughly 58 hours to converge. We have thus provided it with the true $\mathbf{G}$ and show only its empirical parameter estimates. Since both ARES and MARS are using Equation (22) to determine $\mathbf{G}$, they share the same values. Due to space restrictions, we moved the results for Lorenz 63 to Table 4 in the appendix. As demonstrated by this systems, ARES and MARS are both able to deal with locally linear systems, occasionally outperforming their competitors, especially in their estimates of the diffusion.

4.4 Non-Diagonal Diffusion

To investigate the effect of off-diagonal entries in $\mathbf{G}$, we use the Lotka Volterra dynamics. For the diffusion estimates, ARES and MARS share again the same values. Since NPSDE was unable to model non-diagonal diffusions, we provide it with the true $\mathbf{G}$ and only compare parameter estimates. As VGPA was already struggling in the lower dimensional cases, we omitted it from this comparison due to limited computational resources. The results are shown in Table 2. ARES and MARS are clearly outperforming their competition in terms of diffusion estimation, while ESGF is the only algorithm able to keep up in terms of drift parameter estimation.

4.5 Dealing with Multi-Modality

As a final challenge, we investigate the Ginzburg-Landau double well system. While it is one dimensional, its state distribution is multi-modal even if all parameters are known. As shown in Table 3, this is a huge challenge for all classical approaches. While NPSDE probably does not have enough data points for its non-parametric proxy for the drift function, the time-dependent Gaussianity assumptions in both VGPA and ESGF are problematic in this case. In our gradient matching framework, no such assumption is made. Thus, both ARES and MARS are able to deal with the multimodality of the problem.

5. CONCLUSION

Parameter and diffusion estimation in stochastic systems arises in quantitative sciences and many fields of engineering. Current estimation techniques based on Kalman filtering or Gaussian processes try to approximate the state distribution conditioned on the parameters and then iteratively optimize the data likelihood. In this work, we propose to turn this procedure on its head by leveraging key ideas from gradient matching algorithms designed for deterministic ODEs. By introducing a novel noise model for Gaussian process regression using the Doss-Sussmann transformation, we are able to reliably estimate the diffusion process and the drift parameters. Our algorithm is able to keep up and occasionally outperform the state-of-the-art on the simpler benchmark systems, while it is also accurately estimating parameters for systems that exhibit multi-modal state densities, a case where traditional methods fail. While our approach is currently restricted to systems with a constant diffusion matrix $\mathbf{G}$, it would be interesting to see how it generalizes to other systems by using

Table 1: Inferred parameters over 100 independent realizations of the Ornstein-Uhlenbeck dynamics. For every algorithm, we show the median $\pm$ one standard deviation. The ground truth is shown in the left most column.

	NPSDE	VGPA	ESGF	AReS	MaRS
$\theta_0 = 0.5$	0.41 ± 0.11	0.53 ± 0.08	0.49 ± 0.07	$\mathbf{0.50 \pm 0.21}$	0.46 ± 0.06
$\theta_1 = 1$	0.71 ± 1.34	0.96 ± 0.31	0.96 ± 0.24	1.06 ± 0.93	$\mathbf{0.99 \pm 0.25}$
$H = 0.25$	0.00 ± 0.01	/	0.19 ± 0.06	$\mathbf{0.24 \pm 0.09}$	

Table 2: Inferred parameters over 100 independent realizations of the Lotka Volterra dynamics. For every algorithm, we show the median $\pm$ one standard deviation. The ground truth is shown in the left most column.

	NPSDE	ESGF	AReS	MaRS
$\theta_0 = 2$	1.58 ± 0.71	2.04 ± 0.09	2.36 ± 0.18	$\mathbf{2.00 \pm 0.09}$
$\theta_1 = 1$	0.74 ± 0.31	1.02 ± 0.05	1.18 ± 0.9	$\mathbf{1.00 \pm 0.04}$
$\theta_2 = 4$	2.26 ± 1.51	3.87 ± 0.59	$\mathbf{3.97 \pm 0.63}$	3.70 ± 0.51
$\theta_3 = 1$	0.49 ± 0.35	0.96 ± 0.14	$\mathbf{0.98 \pm 0.18}$	0.91 ± 0.14
$\mathbf{H}_{1,1} = 0.05$	/	0.01 ± 0.03	$\mathbf{0.03 \pm 0.004}$	
$\mathbf{H}_{1,2} = 0.03$	/	0.01 ± 0.01	$\mathbf{0.02 \pm 0.01}$	
$\mathbf{H}_{2,1} = 0.03$	/	0.01 ± 0.01	$\mathbf{0.02 \pm 0.01}$	
$\mathbf{H}_{2,2} = 0.09$	/	0.03 ± 0.02	$\mathbf{0.09 \pm 0.03}$	

Table 3: Inferred parameters over 100 independent realizations of the Ginzburg-Landau Double-Well dynamics. For every algorithm, we show the median $\pm$ one standard deviation. The ground truth is shown in the left most column.

	NPSDE	VGPA	ESGF	AReS	MaRS
$\theta_0 = 0.1$	0.09 ± 7.00	0.05 ± 0.04	0.01 ± 0.03	0.09 ± 0.04	$\mathbf{0.10 \pm 0.05}$
$\theta_1 = 4$	3.36 ± 248.82	1.11 ± 0.66	0.11 ± 0.16	3.68 ± 1.34	$\mathbf{3.85 \pm 1.10^2}$
$H = 0.25$	0.00 ± 0.02	/	0.20 ± 0.05	$\mathbf{0.21 \pm 0.09}$	

different or approximate bridge constructs. Unfortunately, this is outside of the scope of this work. However, we hope that our publicly available code will facilitate future research in that direction.

Acknowledgments

This research was supported by the Max Planck ETH Center for Learning Systems. GA acknowledges funding from Google DeepMind and University of Oxford.

References

Yacine Aït-Sahalia. Maximum likelihood estimation of discretely sampled diffusions: a closed-form approximation approach. *Econometrica*, 70(1):223–262, 2002.

Cedric Archambeau, Dan Cornford, Manfred Opper, and John Shawe-Taylor. Gaussian process approximations of stochastic differential equations. *Journal of machine learning research*, 1:1–16, 2007.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

Stefan Bauer, Nico S Gorbach, Djordje Miladinovic, and Joachim M Buhmann. Efficient and flexible inference for stochastic systems. In *Advances in Neural Information Processing Systems*, pages 6988–6998, 2017.

Harish S Bhat, RWMA Madushani, and Shagun Rawat. Parameter inference for stochastic differential equations with density tracking by quadrature. In *International Workshop on Simulation*, pages 99–113. Springer, 2015.

Ben Calderhead, Mark Girolami, and Neil D Lawrence. Accelerating bayesian inference over nonlinear differential equations with gaussian processes. In *Advances in neural information processing systems*, pages 217–224, 2009.

Sophie Donnet and Adeline Samson. A review on estimation of stochastic differential equations for pharmacokinetic/pharmacodynamic models. *Advanced Drug Delivery Reviews*, 65(7):929–939, 2013.

Halim Doss. Liens entre équations différentielles stochastiques et ordinaires. 1977.

Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015.

Ola Elerian, Siddhartha Chib, and Neil Shephard. Likelihood inference for discretely observed nonlinear diffusions. *Econometrica*, 69(4):959–993, 2001.

Bjørn Eraker. Mcmc analysis of diffusion models with application to finance. *Journal of Business & Economic Statistics*, 19(2):177–191, 2001.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Nico S Gorbach, Stefan Bauer, and Joachim M Buhmann. Scalable variational inference for dynamical systems. In *Advances in Neural Information Processing Systems*, pages 4806–4815, 2017.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

AS Hurn and KA Lindsay. Estimating the parameters of stochastic differential equations. *Mathematics and computers in simulation*, 48(4-6):373–384, 1999.

MJ Jiménez, Henrik Madsen, JJ Bloem, and Bernd Dammann. Estimation of non-linear continuous time models for the heat exchange dynamics of building integrated photovoltaic modules. *Energy and Buildings*, 40(2):157–167, 2008.

Hadiseh Karimi and Kimberley B McAuley. Bayesian objective functions for estimating parameters in nonlinear stochastic differential equation models with limited data. *Industrial & Engineering Chemistry Research*, 57(27):8946–8961, 2018.

Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727, 2015.

Alfred J Lotka. The growth of mixed populations: two species competing for a common food supply. *Journal of the Washington Academy of Sciences*, 22(16/17):461–469, 1932.

Jan Nygaard Nielsen, Henrik Madsen, and Peter C Young. Parameter estimation in stochastic differential equations: an overview. *Annual Reviews in Control*, 24:83–94, 2000.

Umberto Picchini. Sde toolbox: Simulation and estimation of stochastic differential equations with matlab. 2007.

Susanne Pieschner and Christiane Fuchs. Bayesian inference for diffusion processes: Using higher-order approximations for transition densities. *arXiv preprint arXiv:1806.02429*, 2018.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. Springer, 2004.

Christoph Riesinger, Tobias Neckel, and Florian Rupp. Solving random ordinary differential equations on gpu clusters using multiple levels of parallelism. *SIAM Journal on Scientific Computing*, 38(4):C372–C402, 2016.

Andreas Ruttor, Philipp Batz, and Manfred Opper. Approximate gaussian process inference for the drift function in stochastic differential equations. In *Advances in Neural Information Processing Systems*, pages 2040–2048, 2013.

Thomas Ryder, Andrew Golightly, A Stephen McGough, and Dennis Prangle. Black-box variational inference for stochastic differential equations. *International Conference on Machine Learning*, 2018.

Simo Särkkä, Jouni Hartikainen, Isambi Sailon Mbalawata, and Heikki Haario. Posterior inference on parameters of stochastic differential equations via non-linear gaussian filtering and adaptive mcmc. *Statistics and Computing*, 25(2):427–437, 2015.

Helle Sørensen. Parametric inference for diffusion processes observed at discrete points in time: a survey. *International Statistical Review*, 72(3):337–354, 2004.

Héctor J Sussmann. On the gap between deterministic and stochastic ordinary differential equations. *The Annals of Probability*, pages 19–41, 1978.

Filip Tronarp and Simo Särkkä. Iterative statistical linear regression for gaussian smoothing in continuous-time non-linear stochastic dynamic systems. *arXiv preprint arXiv:1805.11258*, 2018.

Frank van der Meulen, Moritz Schauer, et al. Bayesian estimation of discretely observed multi-dimensional diffusion processes using guided proposals. *Electronic Journal of Statistics*, 11(1): 2358–2396, 2017.

Michail D Vrettas, Manfred Opper, and Dan Cornford. Variational mean-field algorithm for efficient inference in large systems of stochastic differential equations. *Physical Review E*, 91(1):012148, 2015.

Philippe Wenk, Alkis Gotovos, Stefan Bauer, Nico Gorbach, Andreas Krause, and Joachim M Buhmann. Fast gaussian process based gradient matching for parameter identification in systems of nonlinear odes. *arXiv preprint arXiv:1804.04378*, 2018.

Liu Yang, Dongkun Zhang, and George Em Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations. *arXiv preprint arXiv:1811.02033*, 2018.

Cagatay Yildiz, Markus Heinonen, Jukka Intosalmi, Henrik Mannerström, and Harri Lähdesmäki. Learning stochastic differential equations with gaussian processes without gradient matching. *arXiv preprint arXiv:1807.05748*, 2018.

Appendix A.

A.1 Parameter Estimation Lorenz 63

	NPSDE	ESGF	ARES	MARS
$\theta_0 = 10$	1.28 ± 2.32	$\mathbf{9.97 \pm 0.33}$	7.24 ± 1.08	9.82 ± 0.56
$\theta_1 = 28$	20.69 ± 5.73	$\mathbf{28.00 \pm 0.17}$	28.16 ± 1.08	27.96 ± 0.21
$\theta_0 = 2.667$	1.86 ± 1.08	$\mathbf{2.65 \pm 0.06}$	2.55 ± 0.10	2.64 ± 0.07
$G = \sqrt{10}$	6.51 ± 1.31	$\mathbf{3.03 \pm 0.2}$	3.54 ± 2.45	

Table 4: Median and standard deviation of the 65 best runs of each algorithm. As ESGF crashed in roughly one third of all experiments, we compare only the best 65 runs, where a crash is treated as a complete failure. While this provides somehow a fair comparison, it should be noted that this significantly overestimates the performance of all algorithms.

A.2 Densities for Ancestral Sampling of the SDE Based Model

Given the graphical model in Figure 1a, it is straight forward to calculate the densities used in the ancestral sampling scheme in Algorithm 1. After marginalizing out $\dot{\mathbf{z}}$, the joint density described by the graphical model can be written as

$$p(\mathbf{o}, \mathbf{z}, \mathbf{y}|\boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) = p(\mathbf{o}|\mathbf{G})p(\mathbf{z}|\boldsymbol{\phi})p(\mathbf{y}|\mathbf{z}, \mathbf{o}, \boldsymbol{\sigma}) \tag{34}$$

Inserting the densities given by Equations (10), (11), (13) and (21) yields

$$p(\mathbf{o}, \mathbf{z}, \mathbf{y}|\boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) = \mathcal{N}(\mathbf{o}|\mathbf{0}, \mathbf{B}\boldsymbol{\Omega}\mathbf{B}^T)\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}})\mathcal{N}(\mathbf{y}|\mathbf{z} + \mathbf{o}, \mathbf{T}). \tag{35}$$

Using variable substitution to simplify notation, we write

$$p(\mathbf{o}, \mathbf{z}, \mathbf{y}|\boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) = \mathcal{N}(\mathbf{o}|\mathbf{0}, \tilde{\boldsymbol{\Omega}})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}})\mathcal{N}(\mathbf{y}|\mathbf{z} + \mathbf{o}, \mathbf{T}). \tag{36}$$

This equation is now subsequently modified by observing that the product of two Gaussian densities in the same random variable is again a Gaussian density:

$$\begin{aligned}
p(\mathbf{o}, \mathbf{z}, \mathbf{y}|\boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) &= \mathcal{N}(\mathbf{o}|\mathbf{0}, \tilde{\boldsymbol{\Omega}})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}})\mathcal{N}(\mathbf{y}|\mathbf{z} + \mathbf{o}, \mathbf{T}) && (37)\\
&= \mathcal{N}(\mathbf{o}|\mathbf{0}, \tilde{\boldsymbol{\Omega}})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}})\mathcal{N}(\mathbf{z}|\mathbf{y} - \mathbf{o}, \mathbf{T}) && (38)\\
&= \mathcal{N}(\mathbf{o}|\mathbf{0}, \tilde{\boldsymbol{\Omega}})\mathcal{N}(\mathbf{y} - \mathbf{o}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})\mathcal{N}(\mathbf{z}|\mathbf{m}_z, \mathbf{C}_z) && (39)\\
&= \mathcal{N}(\mathbf{o}|\mathbf{0}, \tilde{\boldsymbol{\Omega}})\mathcal{N}(\mathbf{o}|\mathbf{y}, \mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})\mathcal{N}(\mathbf{z}|\mathbf{m}_z, \mathbf{C}_z) && (40)\\
&= \mathcal{N}(\mathbf{y}|\mathbf{0}, \tilde{\boldsymbol{\Omega}} + \mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})\mathcal{N}(\mathbf{o}|\mathbf{m}_o, \mathbf{C}_o)\mathcal{N}(\mathbf{z}|\mathbf{m}_z, \mathbf{C}_z) && (41)
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{m}_z &= \mathbf{C}_z(\mathbf{T}^{-1}(\mathbf{y} - \mathbf{o})) && (42)\\
\mathbf{C}_z &= (\mathbf{C}_{\boldsymbol{\phi}}^{-1} + \mathbf{T}^{-1})^{-1} && (43)\\
\mathbf{m}_o &= \mathbf{C}_o(\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1}\mathbf{y} && (44)\\
\mathbf{C}_o &= (\tilde{\boldsymbol{\Omega}}^{-1} + (\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1})^{-1} && (45)
\end{aligned}$$

This formula can be further modified using the Woodbury identity, i.e.

$$\begin{aligned}\mathbf{C}_z &= (\mathbf{C}_{\boldsymbol{\phi}}^{-1} + \mathbf{T}^{-1})^{-1} && (46)\\ &= \mathbf{C}_{\boldsymbol{\phi}} - \mathbf{C}_{\boldsymbol{\phi}}(\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1}\mathbf{C}_{\boldsymbol{\phi}} && (47)\\ &= \mathbf{C}_{\boldsymbol{\phi}}(\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1}\mathbf{T} && (48)\end{aligned}$$

which leads to

$$\mathbf{m}_z = \mathbf{C}_{\boldsymbol{\phi}}(\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1}(\mathbf{y} - \mathbf{o}) \tag{49}$$

and

$$\begin{aligned}\mathbf{C}_o &= (\tilde{\boldsymbol{\Omega}}^{-1} + (\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1})^{-1} && (50)\\ &= \tilde{\boldsymbol{\Omega}} - \tilde{\boldsymbol{\Omega}}(\tilde{\boldsymbol{\Omega}} + \mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1}\tilde{\boldsymbol{\Omega}} && (51)\\ &= \tilde{\boldsymbol{\Omega}}(\tilde{\boldsymbol{\Omega}} + \mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1}(\mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T}) && (52)\end{aligned}$$

which leads to

$$\mathbf{m}_o = \tilde{\boldsymbol{\Omega}}(\tilde{\boldsymbol{\Omega}} + \mathbf{C}_{\boldsymbol{\phi}} + \mathbf{T})^{-1}\mathbf{y} \tag{53}$$

Since we observe $\tilde{y}$, we are interested in calculating the conditional distribution

$$p(\mathbf{o}, \mathbf{z}|\mathbf{y}, \boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) = \frac{p(\mathbf{o}, \mathbf{z}, \mathbf{y}|\boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma})}{p(\mathbf{y}|\boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma})} \tag{54}$$

Conveniently enough, the marginal density of $\mathbf{y}$ is already factorized out in Equation (41) (compare Equation (22)). Thus, we have

$$p(\mathbf{o}, \mathbf{z}|\mathbf{y}, \boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) = \mathcal{N}(\mathbf{o}|\mathbf{m}_o, \mathbf{C}_o)\mathcal{N}(\mathbf{z}|\mathbf{m}_z, \mathbf{C}_z) \tag{55}$$

As $\mathcal{N}(\mathbf{o}|\mathbf{m}_o, \mathbf{C}_o)$ is independent of $\mathbf{z}$, we can employ ancestral sampling by first obtaining a sample of $\mathbf{o}$ by sampling $\mathcal{N}(\mathbf{o}|\mathbf{m}_o, \mathbf{C}_o)$ and then using that sample to obtain a sample of $\mathbf{z}$ by sampling $\mathcal{N}(\mathbf{z}|\mathbf{m}_z, \mathbf{C}_z)$.

A.3 Calculating the GP Posterior for Data Based Ancestral Sampling

Given the graphical model in Figure 1b, it is straight forward to calculate the densities used in the ancestral sampling scheme in Algorithm 1. After marginalizing out $\dot{\mathbf{z}}$ and using the variable substitutions introduced in Equation (36), the joint density described by the graphical model can be written as

$$\begin{aligned}p(\mathbf{o}, \mathbf{z}, \mathbf{y}|\boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) &= p(\mathbf{o}|\mathbf{G})p(\mathbf{y}|\boldsymbol{\sigma}, \mathbf{o}, \mathbf{z})p(\mathbf{z}|\boldsymbol{\phi}) && (56)\\ &= \mathcal{N}(\mathbf{o}|\mathbf{0}, \tilde{\boldsymbol{\Omega}})\mathcal{N}(\mathbf{y}|\mathbf{z} + \mathbf{o}, \mathbf{T})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}}) && (57)\\ &= \mathcal{N}(\mathbf{o}|\mathbf{0}, \tilde{\boldsymbol{\Omega}})\mathcal{N}(\mathbf{o}|\mathbf{y} - \mathbf{z}, \mathbf{T})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}}) && (58)\\ &= \mathcal{N}(\mathbf{o}|\mathbf{m}, \mathbf{C})\mathcal{N}(\mathbf{y} - \mathbf{z}|\mathbf{0}, \tilde{\boldsymbol{\Omega}} + \mathbf{T})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}}) && (59)\\ &= \mathcal{N}(\mathbf{o}|\mathbf{m}, \mathbf{C})\mathcal{N}(\mathbf{z}|\mathbf{y}, \tilde{\boldsymbol{\Omega}} + \mathbf{T})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{C}_{\boldsymbol{\phi}}) && (60)\\ &= \mathcal{N}(\mathbf{o}|\mathbf{m}, \mathbf{C})\mathcal{N}(\mathbf{y}|\mathbf{0}, \tilde{\boldsymbol{\Omega}} + \mathbf{T} + \mathbf{C}_{\boldsymbol{\phi}})\mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z) && (61)\end{aligned}$$

where

$$\boldsymbol{\mu}_z = \boldsymbol{\Sigma}_z(\tilde{\boldsymbol{\Omega}} + \mathbf{T})^{-1}\mathbf{y} \tag{62}$$

$$\boldsymbol{\Sigma}_z = ((\tilde{\boldsymbol{\Omega}} + \mathbf{T})^{-1} + \mathbf{C}_{\boldsymbol{\phi}}^{-1})^{-1} \tag{63}$$

$$= \mathbf{C}_{\boldsymbol{\phi}} - \mathbf{C}_{\boldsymbol{\phi}}(\tilde{\boldsymbol{\Omega}} + \mathbf{T} + \mathbf{C}_{\boldsymbol{\phi}})^{-1}\mathbf{C}_{\boldsymbol{\phi}} \tag{64}$$

$$= (\tilde{\boldsymbol{\Omega}} + \mathbf{T})(\tilde{\boldsymbol{\Omega}} + \mathbf{T} + \mathbf{C}_{\boldsymbol{\phi}})^{-1}\mathbf{C}_{\boldsymbol{\phi}} \tag{65}$$

$$= \mathbf{C}_{\boldsymbol{\phi}}(\tilde{\boldsymbol{\Omega}} + \mathbf{T} + \mathbf{C}_{\boldsymbol{\phi}})^{-1}(\tilde{\boldsymbol{\Omega}} + \mathbf{T}) \tag{66}$$

and $\mathbf{m}$, $\boldsymbol{\Sigma}$ is independent of $\mathbf{z}$.

After marginalizing out $\mathbf{o}$ and then dividing by the marginal of $\mathbf{y}$, we get the conditional distribution

$$p(\mathbf{z}|\mathbf{y}, \boldsymbol{\phi}, \mathbf{G}, \boldsymbol{\sigma}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z) \tag{67}$$