Learning stochastic dynamical systems with neural networks mimicking the Euler-Maruyama scheme

Noura Dridi, Lucas Drumetz, Ronan Fablet
(Dept.Mathematical and Electrical Engineering).IMT Atlantique, UMR CNRS Lab-STICC, Brest, France
{noura.dridi, lucas.drumetz, ronan.fablet}@imt-atlantique.fr

***Abstract*—Stochastic differential equations (SDEs) are one of the most important representations of dynamical systems. They are notable for the ability to include a deterministic component of the system and a stochastic one to represent random unknown factors. However, this makes learning SDEs much more challenging than ordinary differential equations (ODEs). In this paper, we propose a data driven approach where parameters of the SDE are represented by a neural network with a built-in SDE integration scheme. The loss function is based on a maximum likelihood criterion, under order one Markov Gaussian assumptions. The algorithm is applied to the geometric brownian motion and a stochastic version of the Lorenz-63 model. The latter is particularly hard to handle due to the presence of a stochastic component that depends on the state. The algorithm performance is attested using different simulations results. Besides, comparisons are performed with the reference gradient matching method used for non linear drift estimation, and a neural networks-based method, that does not consider the stochastic term.**

***Index Terms*—Stochastic differential equations, Maximum likelihood, Neural networks, Lorenz-63 system**

I. Introduction

Understanding the space-time variations of geophysical systems is a challenging task in geosciences. Data assimilation approaches typically combine a physical model and remote sensing data or in situ observations. The state-time evolution is represented by a dynamical model, based on an approximate representation of the physics of the real system [1]. The observation model relates the observation to the true hidden state, with a random noise to include observation error and uncertainties. In this paper, we focus on the state model described by a dynamical model, assuming an identity observation operator. Dynamical systems are usually represented by Ordinary differential Equations (ODEs) or Stochastic Differential Equations (SDEs). The latter encompass a noise-driving process in addition to the deterministic component. The noise represents the uncertainty behind the model, i.e. processes not included in the model but present in the real system. They are widely used in domains such as finance [2], turbulence study [3], the motion of vehicles in a traffic [4], oceanography [5].

Identification of governing equations of a dynamical system from data can be performed using physical priors and/or machine learning approaches. A method including polynomial representations was proposed in [6]. In [7], the authors propose a sparse regression framework with linear and non linear terms. However, such a formulation cannot deal with SDEs or even observation noise [8]. With the availability of larger data sets, machine learning approaches become relevant, using for instance a neural network representation of the unknown operator. They include autoregressive models such as Recurrent Neural Networks (RNN) [9], LSTM [10], Resnet [11], or reservoir computing. The latter has been used successfully for short-term prediction and attractor reconstruction of chaotic dynamical systems from time series data [12], [13]. Besides, the approach provides promising results compared to Echo State Networks (ESNs) for high dimensional chaotic systems. However, good performance is achieved under ideal conditions, i.e. noiseless and regularly sampled with high frequency data. For dynamical systems represented by ODEs, Neural network based algorithms were proposed [9], [14], [15]. In [8], [16] the identification used noisy and partial observations, therefore both estimation of the hidden state and identification of governing equations are tackled. For SDEs, the identification comprises estimation of the drift and the diffusion. It can be performed using pre-defined parametric representations of the drift and the diffusion [17]. Non-parametric representation of the drift includes linear and non linear drift. The first can be carried using a variational smoothing algorithm [18] or a variational mean field approximation [19]. Non linear drifts are modelled using Gaussian processes [20]. For both parametric and non parametric drifts, the gradient matching approximation method is largely exploited [21], [22]. The drift is estimated to match the empirical gradients of data, while the diffusion relates to the residual of the approximation. On the other hand, in [23], the authors generalize the adjoint sensitivity approach to compute gradients of the solutions and combine with a stochastic variational inference scheme to train a latent SDE. Besides, the method is used to the Lorenz-63 system [24], with an additive diffusion term linearly dependent on the state. Multiple trajectories are required for learning.

In this work we propose to extend [9] to model stochastic dynamical systems. Although our motivation and applications are related to problems in ocean and atmospheric sciences, the method can be applied for more general issues. The method is used to learn the Geometric Brownian Motion (GBM) parameters. The GBM is a reduced dimension toy model, that includes a stochastic component depending on the state. A second application is to learn the dynamical operator of a stochastic version of the Lorenz 63 model. The deterministic Lorenz model introduced in [24] is a system of three coupled

This work was supported by the SAD initiative from the Bretagne region and the Carnot TSN institute.

ODEs including linear and non linear terms. The Lorenz-63 system is an appropriate simplified representation of the ocean-atmosphere interactions derived from the Navier-Stokes equations. The Stochastic Lorenz (SL) system presented in [25] describes the evolution of the deterministic Lorenz model when the local position of the state is uncertain. This uncertainty is represented by a multiplicative noise on the y and z components representing variables induced by small scale velocity fluctuations. x is the large scale variable, which is indirectly affected by the small scale uncertainty. The SL system is an important model to represent geophysical flows. It helps to efficiently explore the entire dynamical landscape of the flows, with a reduced order flow representation [25], [26]. We propose a deep learning based approach to identify the parameters of an SDE, in which both the drift and diffusion are represented using neural networks. A parametric formulation is considered, without prior assumptions on the parameters. They are estimated using a maximum likelihood criterion, to define the loss function of the learning algorithm. We maximize the probability of obtaining the observed time series with a Markovian model of order one on the time samples. The performance are attested on the GBM and the SL system. Compared with an algorithm based on a deterministic formulation, the simulations results illustrate, the relevance of considering an SDE formulation to deal with the SL system. The remainder of the paper is organized as follows: section II is dedicated to the neural networks for stochastic dynamical system, section III describes the learning algorithm. Simulations results are presented in section IV. We end the paper by a conclusion with the perspectives of this work.

II. Neural Networks for stochastic dynamical systems

In this section the Neural Network (NN) to represent and generate a process using a stochastic differential equation, is described. A SDE takes the general form of

$$d\boldsymbol{x} = \mathcal{F}(\boldsymbol{x}(t), t)dt + \mathcal{L}(\boldsymbol{x}(t), t)d\boldsymbol{\beta}_t \tag{1}$$

where $\mathcal{F}$ and $\mathcal{L}$ are linear/non linear functions of $\mathbf{x}$ and $d\boldsymbol{\beta}_t$ is a multivariate Brownian motion, $\boldsymbol{x}$ is a d-dimensional vector, $\mathcal{F} : \mathbf{R}^d \times \mathbf{R} \rightarrow \mathbf{R}^d$ is a function of $\boldsymbol{x}$ and the time t, $\mathcal{L}$ is a state dependent matrix in $\mathbf{R}^{d\times s}$, and $d\boldsymbol{\beta}_t$ a vector of dimension s. For one dimensional case, $d = s = 1$ and linear functions $\mathcal{F}$ and $\mathcal{L}$, an analytical solution of the SDE can be calculated such as with the Ornstein–Uhlenbeck process or the GBM. However, in general, an explicit solution cannot be obtained from Eq(1) and numerical approximation methods are used, such as the Euler-Maruyama (EM) integration scheme:

$$\boldsymbol{x}_{t+\tau} = \boldsymbol{x}_t + \tau\mathcal{F}(\boldsymbol{x}_t, t) + \mathcal{L}(\boldsymbol{x}_t, t) \triangle \beta_t \tag{2}$$

with τ is the time resolution. The EM method is the equivalent to Euler integration scheme for SDEs [27]. The EM can be used to simulate trajectories from SDEs and the result converges to the true solution in the limit $\tau \rightarrow 0$. When $\mathcal{L}$ equals to identity, the formulation given by Eq.(2), can be regarded as an autoregressive model of first order.

Hereafter, unless otherwise specified, τ is set to 1 to simplify the notation, without any loss of generality. Considering a neural network architecture, the EM approximation Eq.(2) can be regarded as a recurrent network with one residual layer, and the SDE in Eq.(1), is reformulated as:

$$d\boldsymbol{x} = (A_1\boldsymbol{x}(t) + A_2\boldsymbol{x}(t) \times A_3\boldsymbol{x}(t))\, dt + B\boldsymbol{x}(t)d\beta_t \tag{3}$$

where A_i, for i in $1, 2, 3$ and B are scalar valued matrices representing respectively the operators $\mathcal{F}$ and $\mathcal{L}$. The architecture of the network is defined based on Eq(3), so that the formulation of the neural network includes the true parameterization of the model. This is a general architecture that is convenient for the GBM an the SL models. The first term corresponds to the linear component of the model represented by a fully connected layer, the second term includes bilinear elements of the model represented by an element-wise product operator to embed a second-order polynomial representation for operator $\mathcal{F}$. The whole representation for the first two components is a bilinear neural net architecture as in [9]. The third term refer to the stochastic part of the model with multiplicative noise, where the linear component is designed for $\mathcal{L}$. The network is represented with a fully connected layer multiplied by sampling layer that simulates the noise. This sampling layer is used only for the model to be able to generate new trajectories at test phase, but is not directly used in the training phase since the training is not stochastic.

III. Learning algorithm

The aim is to identify the unknown SDE, that is to estimate $\mathcal{F}$ and $\mathcal{L}$ from the data. Based on the NN representation, the identification can be stated as learning the parameters of a recurrent residual network. The deterministic dynamical system is represented as an ODE, and a learning algorithm using a residual neural network architecture was proposed in [9]. The latter approach is related to the Neural ODE scheme that has recently gained popularity [15], the authors propose an ODE solver using continuous-depth models. A theoretical guarantee of convergence was proved in [15] and [23], showing that residual networks are well suited to learn from data governed by both ODEs or SDEs.

In this paper, we propose to extend the architecture to learn the stochastic dynamical model from data. Given an initial condition, the EM method Eq(2) is used to generate the training trajectory. The goal is to identify the set of parameters $\Theta = \{\omega, \phi\}$ from the data, where ω and ϕ denote the parameters of $\mathcal{F}$ and $\mathcal{L}$, respectively. To relate with the NN notation in Eq.(3), $\omega = \{A_1, A_2, A_3\}$, and $\phi = \{B\}$. It is worth pointing here that the stochastic nature of the model complicates the reconstruction process. Indeed, even for known parameters we cannot reconstruct the original trajectory used for training. Therefore, unlike for deterministic systems [9], the short-term prediction error is not suitable as optimization criterion. From a probabilistic point of view,

and given that the state process is Markovian, the likelihood function is given by:

$$p_\Theta(\boldsymbol{x}_{0:T}) = p_\Theta(\boldsymbol{x}_0)\prod_{t=0}^{T} p_\Theta(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) \tag{4}$$

where T is the number of points in the trajectory. From Eq.(2), the distribution of $\boldsymbol{x}_{t+1}$ given $\boldsymbol{x}_t$ is obtained:

$$\boldsymbol{x}_{t+1}|\boldsymbol{x}_t \sim N(\boldsymbol{m}_t, \boldsymbol{\Sigma}_t) \tag{5}$$

where N refers to a multivariate Gaussian distribution with mean $\boldsymbol{m}_t = \boldsymbol{x}_t + \mathcal{F}(\boldsymbol{x}_t)$, and covariance matrix $\boldsymbol{\Sigma}_t = \mathcal{L}(\boldsymbol{x},t)\mathcal{L}^T(\boldsymbol{x},t)$. It is worth pointing here that only the the conditional distribution $\boldsymbol{x}_{t+1}|\boldsymbol{x}_t$ is Gaussian and not the whole process. Plugging Eq.(5) into Eq.(4), and considering $\mathcal{F}_\omega$ and $\mathcal{L}_\phi$, then the maximum likelihood estimation of the parameters is equivalent to the minimization:

$$\arg\min_{\omega,\phi} \sum_{t=0}^{T} \|\boldsymbol{x}_{t+1} - \boldsymbol{m}_t\|^2_{\boldsymbol{\Sigma}_t^{-1}} + \sum_{t=0}^{T} \log|\boldsymbol{\Sigma}_t| \tag{6}$$

Eq.(6), defines the loss function used to train the NN. Unlike the deterministic case, $\boldsymbol{\Sigma}_t$ can not be reduced to the identity matrix to obtain the short term prediction error as in [8].

IV. Results

Numerical experiments are presented to evaluate the performance of the proposed algorithm. Data are generated using the EM scheme Eq.(2), for fixed parameters and time steps. The proposed algorithm named Bi-NN-SDE is compared to:

- The deterministic algorithm proposed in [9], using residual Bilinear Neural Network architecture, named Bi-NN.
- A Gradient matching approximation predominantly used for non linear drift and diffusion estimation [17]. Drift and the diffusion are respectively estimated using an ensemble average for non stationary processes.

$$\hat{\mathcal{F}}(\boldsymbol{x}_t) = \frac{1}{N}\sum_{t_j\in\alpha}\{\boldsymbol{x}(t_j+1) - \boldsymbol{x}(t_j)\} \tag{7}$$

$$\hat{\mathcal{L}}(\boldsymbol{x}_t) = \frac{1}{N}\sum_{t_j\in\alpha}\{\boldsymbol{x}(t_j+1) - \boldsymbol{x}(t_j) - f(\boldsymbol{x}_{t_j},\boldsymbol{x}_j)\}^2$$

N is the number of the ensemble realizations used for the stochastic process, one realization is denoted by α.

First, the algorithm is applied to learn the parameters of an SDE that describes a GBM process $x(t)$ [28]:

$$dx = \mu x dt + \sigma x d\beta_t \tag{8}$$

Our goal is to estimate the drift $\mathcal{F} = \mu$ and the volatility $\mathcal{L} = \sigma$ then reconstruct the trajectories using the learned model from a given initial condition. For the GBM, the theoretical mean and variance are known:

$$\begin{aligned} E[x] &= x_0\exp(\mu t) \\ V(x) &= x_0^2\exp(2\mu t)[\exp(\sigma^2 t) - 1] \end{aligned} \tag{9}$$

These values are compared with their equivalent by replacing μ and σ by the learned values. Comparison includes also, empirical curves measured using 1000 trajectories generated respectively by the true model and the learned ones using the algorithm BiNN and BiNN-SDE. Below, we consider $\mu = 0.5$ and $\sigma = 1$, $\tau = 0.001$ and the trajectory length $T = 3000$. Fig 1 represents one representative run of the

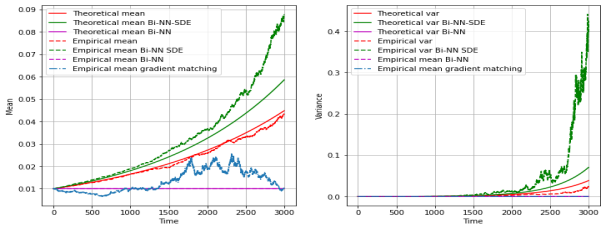

Fig. 1: *Theoretical and empirical mean and variance for the GBM process, with* $\mu = 0.5$, $\sigma = 1$

model, corresponding to a learned drift $\hat{\mu}$ and a diffusion $\hat{\sigma}$ close to the mean value obtained over 100 trainings using different trajectories table.(I). Curves with solid lines correspond to the theoretical mean and variance, while the one with dashed lines refer to the empirical parameters. The red color corresponds to the true model, while the green, purple and blue are assigned respectively to the BiNN-SDE, BiNN and the gradient matching. For the latter, since we don't have access to the parameters, there are no curves for the theoretical mean and variance. It is illustrated that, despite the stochastic nature of the model, the Bi-NN-SDE algorithm learns the mean and variance of the GBM, with lower error than the Bi-NN and the gradient matching. For $t < 2000$, performance of the proposed algorithm are slightly different from the theoretical one, and it is expected to increase for higher t. For the BiNN algorithm, the learned trajectory is deterministic, therefore theoretical and empirical parameters are almost the same.

On the other hand to evaluate the variability of the estimator, we present the mean and variance of the learned parameters using 100 different learned models table.(I), confirming the superiority of the BiNN-SDE algorithm. Note that with the Bi-NN a deterministic formulation is considered, therefore the drift σ is not estimated. The gradient matching algorithm, provides an approximation of $\mathcal{F}(x_t)$ and $\mathcal{L}(x_t)$, thus an explicit form of μ and σ cannot be calculated.

Second, the algorithm is applied to the SL system, that

Algorithm	Bi-NN-μ	Bi-NN-SDE-μ	Bi-NN-SDE-σ
Mean	0.188	0.535	0.998
Variance	0.083	0.201	10^{-4}

TABLE I: *Mean and variance of the learned drift (second and third columns) and the learned diffusion (fourth column).The true values equal to* $\mu = 0.5$ *and* $\sigma = 1$

is a stochastic version of Lorenz-63 model. The latter is a dynamical system largely used to represent ocean-atmosphere interactions: it presents chaotic patterns, i.e. a small perturbation in the initial condition leads to very different trajectories in the long run. Moreover, the Lorenz-63 is a nice toy model to learn dynamics from data : it is a reduced order, easy to visualise model including non linear interactions. The stochastic version of the Lorenz system, introduced in [25] is applied

to represent large-scale geophysical flows, with uncertainty behind the local position of the state. We propose to model the system as an SDE and thus the identification includes the estimation of the drift and the diffusion terms. Identification of such a system is difficult due to the multivariate dimension, non linear relations between state components, the chaoticity of the system in addition to the stochastic component. Besides, it is worth pointing that the diffusion term depends on the state, that is different from a simple deterministic Lorenz with additive noise. The model is given as follows:

$$\frac{dx}{dt} = \sigma y - \left(\sigma + \frac{2}{\gamma}\right) x \tag{10}$$

$$dy = \left((\rho - z)\, x - \left(1 + \frac{2}{\gamma}\right) y\right) dt + \frac{\rho - z}{\sqrt{\gamma}} d\beta_t \tag{11}$$

$$dz = \left(xy - \left(\beta + \frac{4}{\gamma}\right) z\right) dt + \frac{y}{\sqrt{\gamma}} d\beta_t \tag{12}$$

$d\beta_t$ is a Brownian motion, and γ characterizes the noise level: the higher γ, the lower the noise level. σ, ρ and β are the system parameters, also present in the deterministic version of the Lorenz 63. The model is composed of an ODE on the velocity Eq.(10), with two SDEs associated to temperature fluctuations Eq.(11) and Eq.(12).

Note that for low noise level $\gamma \to \infty$, we retrieve the deterministic Lorenz equations [24], and that the γ influences the x variable, though it is not directly perturbed by noise. The SL model can be considered as stochastic differential equations that take the general form of Eq.(1). Note the difference with the GBM, the dimension (for the SL it equals to 3), that implies higher number of parameters to learn: 27 for $\mathcal{F}$ and 9 for $\mathcal{L}$. The evaluation is performed on estimation of the parameters using the RMSE, by matching the obtained parametrization in Eq.(3) with the true one (including the coefficients that should be zero). The ability of the algorithm to reconstruct the attractor of the SL system is also illustrated. The data are generated using the EM method from the true SL model, for one trajectory, a fixed number of time steps and an integration step, and different noise levels.

We simulate $N_s = 100$ trajectories of the SL model, with $\tau = 10^{-3}$, and a trajectory length of $T = 10000$, $\gamma = 50$ (low noise level) and $\gamma = 10$ (high noise level). The models are trained once on each individual trajectory.

Algorithm	Bi-NN-$\mathcal{F}$		Bi-NN-SDE-$\mathcal{F}$		Bi-NN-$\mathcal{L}$	
Noise level γ	50	10	50	10	50	10
Mean RMSE	0.322	0.601	0.306	0.522	0.123	0.359
Var RMSE	0.019	0.065	0.017	0.0447	0.002	0.012

TABLE II: *Mean and variance of the RMSE*

Table (II) shows the mean and variance of the RMSE for the estimation the drift and the diffusion matrices. For $\gamma = 50$, the RMSE provided by the Bi-NN-SDE is lower than the Bi-NN, but all models perform relatively well, due to the small noise level. Furthermore, for strong noise $\gamma = 10$, using the Bi-NN-SDE algorithm clearly improves the estimation performance. The difference between RMSE is multiplied by five ($0.016 \to 0.08$). The above results present the global RMSE. However, the parameters of the model encompass a large set of zero coefficients. Therefore, it is convenient to give the estimation error for the non-null coefficients, and to check that the zero coefficients are close to zero. The gain

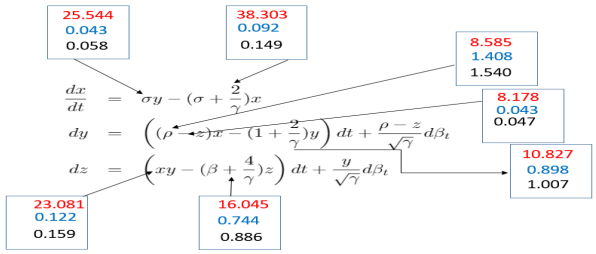

Fig. 2: *Mean of the RMSE per coefficient, black numbers refer to the Bi-NN, blue to Bi-NN-SDE and red for the gain rate.*

rate is computed by the difference between the RMSE of both algorithms normalized by the RMSE of the Bi-NN. As shown in Fig(2), the Bi-NN-SDE outperforms the Bi-NN one for all the model coefficients, the gain rate obtained is higher than 8% and can attain 38%. Regarding the null coefficients, the individual RMSE is $\leq 10^{-2}$ except for 3 coefficients (from 20 null ones) where it is of order of 10^{-1}.

Another possible parameterization involves setting to zeros the null coefficients of the diffusion term to reduce the number of parameters to learn and therefore improve the performance. This is physically acceptable, since the first equation of SL representing the large scale variable is deterministic. Due to limited space, results are not presented for this formulation.

Comparison of the obtained attractors

We propose to further illustrate the algorithm performance with visual comparison of the attractors generated by the true model and the one using the different learning algorithms. A more quantitative comparison is challenging due to the stochasticity: for instance extending Lyapunov exponents (which quantify the chaotic nature of attractors) to stochastic systems is not straightforward. We observe in Fig(3) that

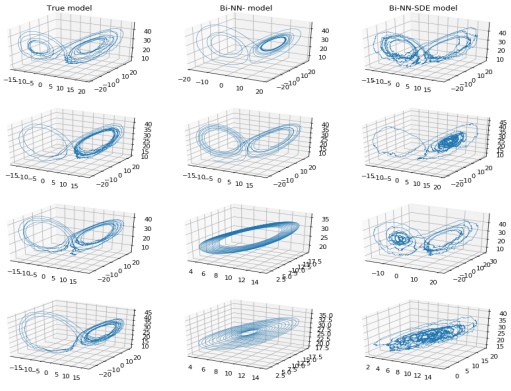

Fig. 3: *Different results for the SL trajectories generated using the true model and the learned ones with Bi-NN and Bi-NN-SDE, respectively*

among the $N_s = 100$ experiments with $\gamma = 10$, in 90% of

cases (results of the first and third row) the sequence generated by the learned model Bi-NN-SDE captures the topology of the true model (the two side of the butterfly shape of the attractor), while the Bi-NN only achieves that in 60% of the cases (results showed in the first row), and is stuck only in one side in other cases. Another major difference is that the Bi-NN algorithm fails to detect the stochastic fluctuations since the algorithm consider the representation of the system as an ODE. Regarding the gradient matching method, given the estimators $\hat{\mathcal{F}}$ and $\hat{\mathcal{L}}$ Eq(7), different attractors are generated for different trajectories used to perform estimation. Some examples are given in Fig(4), as observed, while the learned system is indeed

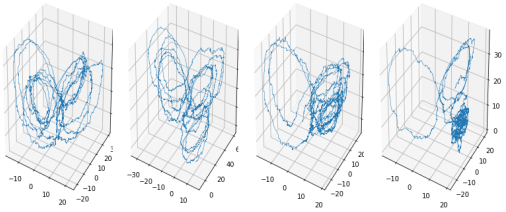

Fig. 4: *SL trajectories generated using the gradient matching.*

stochastic, the gradient matching method fails to preserve the topology of the SL model. In summary Bi-NN-SDE, achieves both recovery of the SL attractor and the stochastic fluctuations in most of the cases, even for significant noise levels.

V. Conclusion

An algorithm to learn stochastic dynamical systems is proposed. The system is modelled as an SDE, the drift and diffusion matrices are parameterized using NN. The algorithm mimicks the EM integration scheme, thus coming with an embedded SDE integration scheme. The networks are trained using a maximum likelihood on the posterior one-step ahead density of the samples. Preliminary results on the GBM process and a challenging SL model illustrate the relevance of considering an SDE formulation and good estimation performance in term of the RMSE. We are currently working on improving the quantitative performance of the algorithm, especially for the diffusion term, and extending the algorithm to include stochastic training as in variational autoencoders.

References

[1] A. Carrassi, M. Bocquet, L. Bertino, and G. Evensen, "Data assimilation in the geosciences: An overview of methods, issues, and perspectives," *WIREs Climate Change*, vol. 9, no. 5, 2018.

[2] C. A. Braumann, *Introduction to Stochastic Differential Equations with Applications to Modelling in Biology and Finance*, Wiley, 2019.

[3] R. Friedrich and J. Peinke, "Statistical properties of a turbulent cascade," *Physica D: Nonlinear Phenomena*, vol. 102, pp. 147–155, 03 1997.

[4] P. Verma, H. Yang, P. Kachroo, and S. Agarwal, "Modeling and estimation of the vehicle-miles traveled tax rate using stochastic differential equations," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 6, pp. 818–828, 2016.

[5] Brillinger David R., Preisler. Haiganoush K., Ager. Alan A., Kie. John G., and Stewart. Brent S., "Employing stochastic differential equations to model wildlife motion," *Bull Braz Math Soc*, vol. 33, 2002.

[6] J. Paduart, Lieve Lauwers, Jan Swevers, Kris Smolders, Johan Schoukens, and Rik Pintelon, "Identification of nonlinear systems using polynomial nonlinear state space models," *Automatica*, vol. 46, pp. 647–656, April 2010.

[7] S.L. Brunton, J.L. Proctor, L. Joshua L., and J.N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[8] D. Nguyen, S. Ouala L. Drumetz, and R. Fablet, "Variational deep learning for the identification and reconstruction of chaotic and stochastic dynamical systems from noisy and partial observations," *arXiv*, 2020.

[9] R. Fablet, S. Ouala, and C. Herzet, "Bilinear residual neural network for the identification and forecasting of geophysical dynamics," *European Signal Processing Conference (EUSIPCO)*, pp. 1477–1481, 2018.

[10] K. Yeo and I. Melnyk, "Deep learning algorithm for data-driven simulation of noisy dynamical system," *Journal of Computational Physics*, vol. 376, 2019.

[11] T. Qin, K. Wu, and D. Xiu, "Data driven governing equations approximation using deep neural networks," *arXiv:1811.05537 [cs, math,stat]*, Nov 2018.

[12] J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, "Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 12, 2017.

[13] Lu. Zhixin, B.R. Hunt, and E. Ott, "Attractor reconstruction by machine learning," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 18, no. 6, 2018.

[14] S. Ouala, D. Nguyen, C. Herzet, L. Drumetz, B. Chapron, A. Pascual, F. Collard, and L. Gaultier, "Learning ocean dynamical priors from noisy data using assimilation-derived neural nets," *IEEE International Geoscience and Remote Sensing Symposium, Yokohama, Japan*, pp. 9451–9454, 2019.

[15] R.T.Q. Chen, Y. Rubanova, J. Bettencourt, and D.K. Duvenaud, "Neural ordinary differential equations," *Advances in Neural Information Processing Systems 31*, pp. 6571–6583, 2018.

[16] M. Bocquet, J. Brajard, A. Carrassi, and L. Bertino, "Bayesian inference of chaotic dynamics by merging data assimilation, machine learning and expectation-maximization," *Foundations of Data Science*, vol. 2, no. 1, 2020, arXiv: 2001.06270.

[17] R. Friedrich, J. Peinkeb, M. Sahimic, and R. Tabar, "Approaching complexity by stochastic methods: From biological systems to turbulence," *Phys. reports*, vol. 506, 2011.

[18] C. Archambeau, D. Cornford, M. Opper, and J. Shawe-Taylor, "Gaussian process approximations of stochastic differential equations," *Gaussian Processes in Practice*, vol. 1, pp. 1–16, 2007.

[19] M. Vrettas, M. Opper, and D. Cornford, "A variational mean field algorithm for efficient inference in large systems of stochastic differential equations," *Physical Review E*, vol. 91, no. 1, 2015.

[20] C. Yildiz, M. Heinonen, J. Intosalmi, H. Mannerstrom, and H. Lahdesmaki, "Learning stochastic differential equations with gaussian processes without gradient matching," in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2018, pp. 1–6.

[21] L.Dony, H. Fei, and M. Stumpf, "Parametric and non-parametric gradient matching for network inference: A comparison," *BMC Bioinformatics*, vol. 20, 01 2019.

[22] M. Niu, S. Rogers, M. Filippone, and D. Husmeier, "Fast parameter inference in nonlinear dynamical systems using iterative gradient matching," *International Conference on Machine Learning*, Jun 2016.

[23] X. Li, T.K Leonard Wong, R. T. Q. Chen, and D. Duvenaud, "Scalable gradients for stochastic differential equations," *Proceedings of Machine Learning Research*, vol. 108, Aug 2020.

[24] Edward N. Lorenz, "Deterministic Nonperiodic Flow," *Journal of the Atmospheric Sciences*, vol. 20, no. 2, pp. 130–141, 03 1963.

[25] Chapron. B, P. Derian, E. Memin, and V.Resseguier, "Large-scale flows under location uncertainty: a consistent stochastic framework," *Quarterly Journal Of The Royal Meteorological Society*, vol. 144, no. 710, pp. 3251–260, 2018.

[26] Y. Yang and E. Mémin, "Estimation of physical parameters under location uncertainty using an ensemble-expectation-maximization algorithm," *Quarterly Journal of the Royal Meteorological Society, Wiley*, vol. 145, no. 719, 2019.

[27] Kloeden. P.E. and Platen E., *Numerical Solution of Stochastic Differential Equations*, Springer, Berlin. ISBN 3-540-54062-8, 1992.

[28] Ross. Sheldon M., *Introduction to Probability Models(11th ed.). "Variations on Brownian Motion"*, Amsterdam: Elsevier. pp. 612–14. ISBN 978-0-12-407948-9., 2014.