# DANSE: Data-driven Non-linear State Estimation of Model-free Process in Unsupervised Bayesian Setup

Anubhab Ghosh*, Antoine Honoré*, Saikat Chatterjee*
* Digital Futures, and School of Electrical Eng. and Comp. Science, KTH Royal Institute of Technology, Sweden
{anubhabg, honore, sach}@kth.se

***Abstract*—We propose DANSE – a data-driven non-linear state estimation method. DANSE provides a closed-form posterior of the state of a model-free process, given linear measurements of the state in a Bayesian setup, like the celebrated Kalman filter (KF). Non-linear dynamics of the state are captured by data-driven recurrent neural networks (RNNs). The training of DANSE combines maximum-likelihood and gradient-descent in an unsupervised framework, i.e. only measurement data and no process data are required. Using simulated linear and non-linear process models, we demonstrate that DANSE - without knowledge of the process model - provides competitive performance against model-based approaches such as KF, unscented KF (UKF), extended KF (EKF), and a hybrid approach such as KalmanNet.**

***Index Terms*—recurrent neural networks, neural networks, state estimation, deep learning**

## I. Introduction

*Short Background:* Kalman filter (KF) is a well-known Bayesian state estimation method, providing posterior of states given measurements [1]. KF has a linear measurement model, and it uses a linear state model of underlying process dynamics. KF has a-priori knowledge of the process model. KF has limitations when the process is non-linear (and complex). Non-linear complex processes are in abundance today.

*Purpose:* In this article, we consider a non-linear complex process that do not have a good tractable model. Therefore we have to deal with a model-free process. Further, we have no access to process data. The process data can not be collected easily. Therefore we can not create labelled data between a measurement and the corresponding process state. This is an unsupervised learning scenario for a state estimation method. The question is: Can we design a Bayesian unsupervised state estimation method for a complex model-free process using a data-driven approach? Yes, we can. We propose DANSE – *Da*ta-driven *N*onlinear *S*tate *E*stimation method.

*Application context:* DANSE can be used for many applications where there is no dearth of measurement / observation data, but it is difficult to collect the (hidden) state data for supervised learning or design of a tractable model of the non-linear process. For example, positioning and navigation of autonomous systems in hazardous conditions, say under-water / off-road conditions where GPS (global positioning system) and/or GNSS (global navigation satellite system) helps are not available to create labelled data [2].

*Aspects of DANSE:* Using a set of measurements as training data, the key aspects of the proposed DANSE are as follows.

1) DANSE can handle a model-free process.
2) It provides closed-form Bayesian posterior of state given a tractable prior parameterized by data-driven recurrent neural networks (RNNs).
3) Unsupervised learning of DANSE parameters can be realized using a standard Bayesian principle.
4) Causal system, and can be used for forecasting problems.
5) An interpretable system, comparable to white-box KF and its non-linear extensions, such as UKF and EKF.

### A. Literature Survey

There can be three broad approaches for state estimation: a model-based approach, a data-driven approach, or a combination of the two. The model-driven approach is traditional, and the famous example is KF [1], [3]. Since dynamical systems are often non-linear in nature, an extension to the Kalman filter, namely the extended Kalman filter (EKF), was proposed to account for the non-linear variations [4]. The beauty of the EKF scheme is that it maintains analytical tractability like KF using an approximately linear, time-varying state space model (SSM). Another non-linear extension of the KF is the Unscented Kalman filter (UKF) that seeks to approximate the unknown dynamics using a derivative-free approximation as opposed to the EKF [5]. Schemes based on sequential Monte Carlo (SMC) sampling such as the particle filters (PFs) are also capable of handling non-linear, non-Gaussian dynamics but are often computationally intensive [6]. A review of model-based approaches has been provided in [7].

A second approach is the data-driven one, where deep neural networks, mainly RNNs are used such as gated recurrent units (GRUs) [8] and long short-term memory networks (LSTMs) [9]. Approaches for state estimation using prediction error methods and RNNs were proposed in [10]. In general, a data-driven approach does not require explicit models, such as the Markovian relation between states. They often involve modeling the distributions of the underlying SSM using (deep) neural networks. Example schemes include the class of dynamical variational autoencoders [11], [12] where training is performed using approximate Bayesian inference called variational inference (VI). The use of VI and complex relational structures does not provide closed-form analytical expressions, owing to the black-box nature of purely data-driven methods. Specifically, it is not possible to obtain a closed-form Bayesian posterior of the state. The training is also computationally intensive.

The third approach, known as the hybrid approach, seeks to use the best of both worlds. It incorporates both model-driven and data-driven approaches. A recent example is the KalmanNet method [13] that proposes an online, recursive, low-complexity, and data-efficient scheme based on KF architecture. KalmanNet involves modeling the Kalman gain using deep neural networks (DNNs), thus ensuring the structure of the model-based KF but incorporating some data-driven aspects. The architecture is however learned in a supervised learning approach using the true states and noisy observations. A modification to the above scheme is also proposed as an unsupervised KalmanNet in [14], which involves a learning scheme using only the noisy measurements. KalmanNet assumes knowledge of the underlying process model.

### B. Notations

We use bold font lowercase symbols to denote vectors, and regular lowercase font to denote scalars, e.g. $\mathbf{x}$ represents a vector while $x_j$ represents the $j^{th}$ component of $\mathbf{x}$. A sequence of vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t$ is compactly denoted by $\mathbf{x}_{1:t}$, where $t$ denotes discrete time index. Upper case symbols in bold font, like $\mathbf{H}$, represent matrices. The operator $(\cdot)^\top$ denotes the transpose. $\mathcal{N}(\cdot; \mathbf{m}, \mathbf{L})$ represents Gaussian distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{L}$.

## II. Proposed DANSE

### A. Bayesian Inference and Unsupervised Learning

For DANSE, we have an unsupervised Bayesian setup. Let, there be a dynamical signal $\mathbf{x}_{1:T}$ (with $\mathbf{x}_t \in \mathbb{R}^m$), representing a model-free process of length $T$. The process may be arbitrarily complex and we have no prior knowledge of the process. Neither do we know its statistical properties nor have direct access to the process data in the learning stage. We assume that we have access to the linear measurements $\mathbf{y}_t \in \mathbb{R}^n$ of the process, where

$$\mathbf{y}_t = \mathbf{H}_t \mathbf{x}_t + \mathbf{w}_t, \ t = 1, 2, \ldots, T. \tag{1}$$

Here $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_w)$ is a standard measurement noise with zero mean and covariance $\mathbf{C}_w \in \mathbb{R}^{n \times n}$, and $\mathbf{H}_t \in \mathbb{R}^{n \times m}$ denotes the known measurement system. Maintaining causality, the Bayesian inference tasks are mentioned below.

(T1) *State estimation problem:* The inference task is to estimate the posterior of the current state $\mathbf{x}_t$ given $\mathbf{y}_{1:t}$, denoted by $p(\mathbf{x}_t|\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t) \triangleq p(\mathbf{x}_t|\mathbf{y}_{1:t})$. In addition to estimate the posterior of the time series $\mathbf{x}_{1:t}$, denoted by $p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t|\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t) \triangleq p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$.

(T2) *Forecasting problem:* The inference task is to estimate $p(\mathbf{x}_{t+1}|\mathbf{y}_{1:t})$ and $p(\mathbf{y}_{t+1}|\mathbf{y}_{1:t})$.

To develop DANSE, we have to learning its parameters. We have a training dataset $\mathcal{D}$ comprised of $N$ time-series measurements as $\mathcal{D} = \{\mathbf{y}^{(i)}_{1:T^{(i)}}\}_{i=1}^{N}$. Here $\mathbf{y}^{(i)}_{1:T^{(i)}} = \mathbf{y}^{(i)}_1, \mathbf{y}^{(i)}_2, \ldots, \mathbf{y}^{(i)}_{T^{(i)}}$ is the $i^{th}$ time-series measurements of length $T^{(i)}$. We do not have access to state $\mathbf{x}_t$. Due to the absence of $\mathbf{x}_t$ in $\mathcal{D}$, the learning problem is unsupervised.

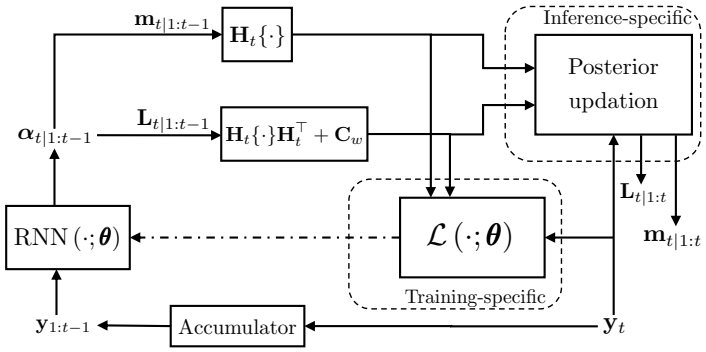


Fig. 1: DANSE architecture. The dash-dotted line represents the gradient flow during the training phase, solid lines indicate information flow during training/inference. Training/inference-specific blocks are shown within dashed borders. The 'Accumulator' block collects observations recursively.

### B. DANSE System

The proposed approach seeks to utilize the tractability provided by model-based approaches such as KF and EKF while utilizing linear measurements for calculating certain model parameters. Principally, we model the unknown prior probability distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ as a Gaussian distribution parameterized using an RNN. At time instant $t$, an RNN recursively uses the input sequence $\mathbf{y}_{1:t-1}$ and provides the parameters of the Gaussian prior as the output, collectively denoted as $\boldsymbol{\alpha}_{t|1:t-1}$. An RNN has its own parameters $\boldsymbol{\theta}$, thus its output also depends on $\boldsymbol{\theta}$. To indicate this, we write $\boldsymbol{\alpha}_{t|1:t-1} \triangleq \boldsymbol{\alpha}_{t|1:t-1}(\boldsymbol{\theta})$. A block diagram of DANSE is shown in Fig. 1.

*1) Bayesian State Estimation (Solution of T1):* The task is to obtain current state posterior $p(\mathbf{x}_t|\mathbf{y}_1, \ldots, \mathbf{y}_{t-1}, \mathbf{y}_t) = p(\mathbf{x}_t|\mathbf{y}_{1:t})$, and the time-series posterior $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$. The prior $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ and the observation distribution $p(\mathbf{y}_t|\mathbf{x}_t)$ are shown below in (2).

$$\begin{gathered}\text{Prior}: p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) \triangleq \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})), \\ \text{such that } \{\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\} = \boldsymbol{\alpha}_{t|1:t-1}(\boldsymbol{\theta}), \\ \boldsymbol{\alpha}_{t|1:t-1}(\boldsymbol{\theta}) \triangleq \text{RNN}(\mathbf{y}_{1:t-1}; \boldsymbol{\theta}). \\ \text{Observation}: p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t; \mathbf{H}_t\mathbf{x}_t, \mathbf{C}_w).\end{gathered} \tag{2}$$

Here, $\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta})$ and $\mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})$ denote the mean and covariance matrix of the Gaussian prior distribution, respectively. $\mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})$ is modelled as a diagonal covariance matrix. Then, using 'completing the square' approach [15, Chap. 2], the posterior distribution of the current state $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is obtained in closed-form as

$$\begin{aligned}&p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t}(\boldsymbol{\theta})), \\ &\mathbf{m}_{t|1:t}(\boldsymbol{\theta}) = \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}) + \mathbf{K}_{t|1:t-1}\boldsymbol{\epsilon}_t, \\ &\mathbf{L}_{t|1:t}(\boldsymbol{\theta}) = \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}) - \mathbf{K}_{t|1:t-1}\mathbf{R}_{\boldsymbol{\epsilon}}\mathbf{K}^\top_{t|1:t-1},\end{aligned} \tag{3}$$

where the second equation in (3) is obtained using the Woodbury matrix identity/matrix inversion lemma, $\mathbf{K}_{t|1:t-1} \triangleq \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\mathbf{H}_t^\top \mathbf{R}_{\boldsymbol{\epsilon}}^{-1}$, $\mathbf{R}_{\boldsymbol{\epsilon}} \triangleq \mathbf{H}_t\mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\mathbf{H}_t^\top + \mathbf{C}_w$ and $\boldsymbol{\epsilon}_t \triangleq \mathbf{y}_t - \mathbf{H}_t\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta})$. We note in (3) a similarity with

the standard KF update equations [16, Chap. 5]. We can also estimate closed-form posterior of the time series $\mathbf{x}_{1:t}$ as

$$\begin{aligned} p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t}) &= p\left(\mathbf{x}_t|\mathbf{y}_{1:t}, \mathbf{x}_{1:t-1}\right) p\left(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t}\right) \\ &= p\left(\mathbf{x}_t|\mathbf{y}_{1:t}\right) p\left(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1}\right) \\ &= \textstyle\prod_{\tau=1}^{t} p\left(\mathbf{x}_\tau|\mathbf{y}_{1:\tau}\right) \\ &= \textstyle\prod_{\tau=1}^{t} \mathcal{N}(\mathbf{x}_\tau; \mathbf{m}_{\tau|1:\tau}(\boldsymbol{\theta}), \mathbf{L}_{\tau|1:\tau}(\boldsymbol{\theta})). \end{aligned} \tag{4}$$

*2) Unsupervised learning of DANSE:* Now we address how to learn the RNN parameters $\boldsymbol{\theta}$ in DANSE using $\mathcal{D} = \{\mathbf{y}_{1:T^{(i)}}^{(i)}\}_{i=1}^{N}$. We first express $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ as follows -

$$\begin{aligned} &p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) \\ &= \int_{\mathbf{x}_t} p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{y}_{1:t-1})\, d\mathbf{x}_t \\ &= \int_{\mathbf{x}_t} \mathcal{N}(\mathbf{y}_t; \mathbf{H}_t\mathbf{x}_t, \mathbf{C}_w)\, \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}))\, d\mathbf{x}_t \\ &= \mathcal{N}(\mathbf{y}_t; \mathbf{H}_t\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{C}_w + \mathbf{H}_t\mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\mathbf{H}_t^\top) \\ &\triangleq p(\mathbf{y}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta}). \end{aligned} \tag{5}$$

Here we introduced the notation $p(\mathbf{y}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})$ to show the dependency of $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ on $\boldsymbol{\theta}$. Now, using the chain rule of probability, we can write

$$p\left(\mathbf{y}_{1:T^{(i)}}^{(i)}\right) = \prod_{t=1}^{T^{(i)}} p\left(\mathbf{y}_t^{(i)}|\mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta}\right). \tag{6}$$

Therefore, using the dataset $\mathcal{D}$, the maximum-likelihood based unsupervised learning problem is

$$\begin{aligned} \boldsymbol{\theta}^\star &= \arg\max_{\boldsymbol{\theta}} \log \textstyle\prod_{i=1}^{N} \prod_{t=1}^{T^{(i)}} p\left(\mathbf{y}_t^{(i)}|\mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta}\right) \\ &= \arg\max_{\boldsymbol{\theta}} \textstyle\sum_{i=1}^{N} \sum_{t=1}^{T^{(i)}} \log p\left(\mathbf{y}_t^{(i)}|\mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta}\right) \\ &= \arg\min_{\boldsymbol{\theta}} \textstyle\sum_{i=1}^{N} \mathcal{L}\left(\mathbf{y}_{1:T^{(i)}}^{(i)}; \boldsymbol{\theta}\right), \end{aligned} \tag{7}$$

with $p\left(\mathbf{y}_t^{(i)}|\mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta}\right)$ defined in (5) and $\mathcal{L}\left(\mathbf{y}_{1:T^{(i)}}^{(i)}; \boldsymbol{\theta}\right) \triangleq -\sum_{t=1}^{T^{(i)}} \log p\left(\mathbf{y}_t^{(i)}|\mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta}\right)$. The above optimization problem can be further formulated as a Bayesian learning problem if we have a suitable prior of $\boldsymbol{\theta}$ as $p(\boldsymbol{\theta})$; e.g. a Gaussian prior.

The optimization in (7) requires gradient-descent to learn the parameters $\boldsymbol{\theta}$. The learning problem is a suitable combination of Bayesian learning (or maximum-likelihood learning) and data-driven learning (of RNNs). Note we need not use variational inference (VI) to optimize (7). VI is an approximate Bayesian learning method. Instead, we endeavor direct optimization without approximations.

*3) Bayesian Forecasting (Solution of T2):* Note that (2) and (5) provide the solution of the forecasting problem T2.

*4) On RNNs and further scopes:* We can use prominent RNNs, such as GRUs [8] and LSTMs [9]. Here we used GRUs owing to their simplicity and popularity [17]. Specifically, for modeling the mean vector and diagonal covariance matrix of the parameterized Gaussian prior in (2), we transformed the latent state of the GRU using feed-forward networks. We used rectified linear unit (ReLU) activations to model the non-negative variances. Note that RNNs allow causality. If we let go causality, then we could use attention-based methods such as transformers to produce a sequence of $\{\mathbf{m}_t, \mathbf{L}_t\}_{t=1}^{T}$ given the whole sequence $\mathbf{y}_{1:T}$ [18].

## III. Experiments and Results

The architectures for the RNN and the feed-forward networks were found experimentally by cross-validation. For the RNN, we used a 2 layer GRU model with 30 hidden nodes. The feed-forward networks were shallow nets with a single hidden layer having 32 hidden nodes. We implemented DANSE in Python and PyTorch [19] and trained using GPU support [1] The training algorithm was mini-batch gradient descent with a batch size of 64. The optimizer was Adam [20] with an adaptive learning rate set at a starting value $10^{-2}$ and decreased step-wise by a factor of 0.9 every fixed number of epochs. The maximum number of training epochs was 2000, and early stopping was used as regularization. We evaluate the schemes in terms of the ability to estimate state sequences from measurement data. The performances are reported in terms of average normalized-mean-squared-error (NMSE) between estimated state sequences $\{\hat{\mathbf{x}}_t\}$ and true state sequences $\{\mathbf{x}_t\}$, where for the $i^{th}$ state sequence

$$\text{NMSE}^{(i)} = 10\log_{10} \frac{\sum_t \|\mathbf{x}_t^{(i)} - \hat{\mathbf{x}}_t^{(i)}\|_2^2}{\sum_t \|\mathbf{x}_t^{(i)}\|_2^2}. \tag{8}$$

The estimated state $\hat{\mathbf{x}}_t$ for DANSE corresponds to the posterior mean in (3). For comparability, our experimental setup is inspired from [13], [14]. We experiment with measurement and state sequences generated from two different SSMs with additive white Gaussian noise. In the measurement model (1), the measurement noise level is varied.

### A. Baselines

The performances of our method are compared with baseline methods that estimate the state sequence from the measurement data. We used classical, model-based approaches such as KF, EKF, UKF and data-driven hybrid approaches such as the offline-trained unsupervised KalmanNet model [14]. We implemented KF, EKF, and UKF using PyTorch and FilterPy [21]. KalmanNet assumes knowledge of the state dynamics; KF, EKF, and UKF also assume perfect knowledge of the state dynamics and process noise. These are strong practical constraints, which are not required by our method.

### B. Linear SSM

We first perform a proof of principle experiment with a linear SSM as follows:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{F}\mathbf{x}_{t-1} + \mathbf{e}_t, \\ \mathbf{y}_t &= \mathbf{H}\mathbf{x}_t + \mathbf{w}_t, \end{aligned} \tag{9}$$

where $t = 1, 2, \ldots, T$, $\mathbf{F} \in \mathbb{R}^{m\times m}$, $\mathbf{H} \in \mathbb{R}^{n\times m}$, $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_e)$ is the white Gaussian process noise variable with $\mathbf{C}_e = \sigma_e^2\mathbf{I}_m$ and $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_w)$ is the white Gaussian measurement noise with $\mathbf{C}_w = \sigma_w^2\mathbf{I}_n$ ($\mathbf{I}_n$ denotes the $n \times n$ identity matrix). The matrices $\mathbf{F}, \mathbf{H}$ were time-invariant and constructed similar to the setup in [13], [14].

We simulated (9) using $m = 2$, $n = 2$ and generated our training dataset. We compared our approach with the

[1]The code is available at https://github.com/anubhabghosh/danse.

standard KF and an offline-trained unsupervised KalmanNet. We trained DANSE using dataset parameters $N = 500, T = 500$. KalmanNet was trained on shorter trajectories of length $T = 80$ as described in [14], due to convergence issues during training for longer trajectories. The quantity $1/\sigma_w^2$ was varied from between $-10$ and 30 dB, where $1/\sigma_w^2$ is proportional to the signal-to-noise ratio (SNR). $\sigma_e$ is controlled by a ratio parameter $\nu = \sigma_e^2/\sigma_w^2$. All the methods were evaluated on the same test set. The results are shown in Table I.

TABLE I: NMSE (in dB) for $2 \times 2$ linear SSM, averaged over $N_{\text{test}} = 100$ trajectories with $T_{\text{test}} = 1000$, $\nu = 0$ dB.

| $1/\sigma_w^2$ [dB] | -10 | 0 | 10 | 20 | 30 |
|---|---|---|---|---|---|
| KF | $-49.64$ $\pm 4.56$ | $-50.12$ $\pm 3.87$ | $-50.75$ $\pm 3.52$ | $-50.31$ $\pm 3.67$ | $-49.75$ $\pm 4.17$ |
| DANSE | $-46.42$ $\pm 4.97$ | $-49.96$ $\pm 3.77$ | $-50.45$ $\pm 3.51$ | $-50.28$ $\pm 3.69$ | $-49.74$ $\pm 4.16$ |
| KalmanNet [14] | $-47.54$ $\pm 5.76$ | $-49.96$ $\pm 3.97$ | $-50.64$ $\pm 3.59$ | $-50.20$ $\pm 3.74$ | $-49.55$ $\pm 4.29$ |

*Interpretation of results*: In Table I, we find that across noise levels, the standard KF performs similarly, at around $-50$ dB. KF is optimal for the linear SSM and the performances are matched by DANSE and KalmanNet at $1/\sigma_w^2 \geq 0$ dB. Both KalmanNet and DANSE achieve similar performances at $1/\sigma_w^2 = -10$ dB, and slightly under-perform compared to standard KF. We note also that all the standard deviation of the NMSE performances of all the models increases at $1/\sigma_w^2 = -10$ dB. Also for $1/\sigma_w^2 \geq 0$, DANSE performs comparably to KalmanNet, despite it not knowing the exact process model, i.e. matrix $\mathbf{F}$ and process noise covariance $\mathbf{C}_e$. This is of primary importance since in practice, these quantities are seldom exactly known.

### C. Lorenz attractor SSM

We also experiment with a non-linear Lorenz attractor SSM [22]. The SSM is specified here with a discrete-time, time-varying dynamics similar to [13]:

$$\mathbf{x}_t = \mathbf{F}_t(\mathbf{x}_{t-1})\mathbf{x}_{t-1} + \mathbf{e}_t, \tag{10}$$

where $\mathbf{x}_t \in \mathbb{R}^3$, $\mathbf{e}_t \in \mathbb{R}^3$ is similar to the linear SSM in (9), i.e. $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_e)$ with $\mathbf{C}_e = \sigma_e^2 \mathbf{I}_3$, and

$$\mathbf{F}_t(\mathbf{x}_{t-1}) = \exp\left(\begin{bmatrix} -10 & 10 & 0 \\ 28 & -1 & -x_{t-1,1} \\ 0 & x_{t-1,1} & -\frac{8}{3} \end{bmatrix} \Delta\right), \tag{11}$$

with the step-size $\Delta = 0.02$ seconds. In our simulations, we used a finite-Taylor series approximation of $5^{th}$ order for (11). We assumed $\mathbf{H}_t = \mathbf{I}_3$ in the measurement model (1). The measurement noise was $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_w)$ with $\mathbf{C}_w = \sigma_w^2 \mathbf{I}_3$. We trained DANSE using the simulated dataset parameters $N = 500$, $T = 1000$, while for KalmanNet we used $N = 500, T = 100$ [14]. The value of the ratio parameter was kept at $-20$ dB. A 3D plot of the true state trajectory of length $T = 2000$ at $\nu = -20$ dB is depicted in Fig. 2.

Since the process dynamics are non-linear, we compared our method with UKF, EKF, and the offline-trained unsupervised KalmanNet. The quantity $1/\sigma_w^2$ was varied from $-25$ to 20 dB. All the methods were evaluated on the same test set. The results are shown in Table II.

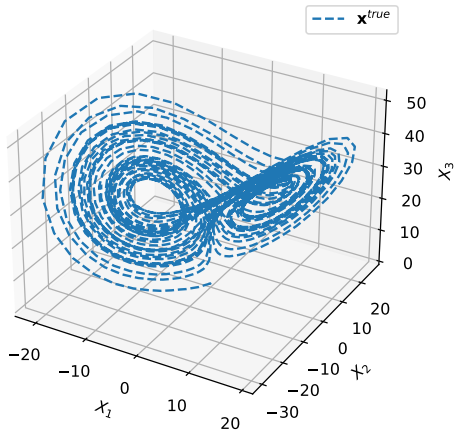


Fig. 2: True state trajectory of the Lorenz attractor SSM

TABLE II: NMSE (in dB) for Lorenz attractor SSM, averaged over $N_{\text{test}} = 100$ trajectories with $T_{\text{test}} = 2000$, $\nu = -20$ dB.

| $1/\sigma_w^2$ [dB] | -25 | -20 | -10 | 0 | 10 | 20 |
|---|---|---|---|---|---|---|
| UKF | $-9.95$ $\pm 0.27$ | $-13.73$ $\pm 0.34$ | $-19.52$ $\pm 0.31$ | $-20.97$ $\pm 0.18$ | $-21.20$ $\pm 0.17$ | $-21.19$ $\pm 0.13$ |
| EKF | $-9.53$ $\pm 0.35$ | $-13.69$ $\pm 0.37$ | $-19.59$ $\pm 0.32$ | $-21.01$ $\pm 0.18$ | $-21.23$ $\pm 0.17$ | $-21.22$ $\pm 0.13$ |
| DANSE | $-9.16$ $\pm 0.30$ | $-13.00$ $\pm 0.38$ | $-19.12$ $\pm 0.32$ | $-20.99$ $\pm 0.18$ | $-21.19$ $\pm 0.17$ | $-21.21$ $\pm 0.13$ |
| KalmanNet [14] | $-8.74$ $\pm 0.31$ | $-12.49$ $\pm 0.38$ | $-18.78$ $\pm 0.49$ | $-20.18$ $\pm 1.85$ | $-21.00$ $\pm 0.91$ | 21.14 $\pm 0.13$ |

*Interpretation of results*: Table II shows that across the values of $1/\sigma_w^2$, DANSE shows only marginally different NMSE compared to both EKF and UKF. Notably, DANSE matches their performances without knowing the exact process model a-priori. Also, DANSE exhibits slightly better performances than KalmanNet at low values $1/\sigma_w^2 < 0$ dB. We further perform experiments with imperfect knowledge of the process model.

*1) EKF and UKF with incorrect process noise information:* To further show the relevance of our method, we performed an experiment where incorrect process noise statistics are provided to the methods. The evaluation data were generated with a parameter $\nu^\star = -20$ dB, $N_{\text{test}} = 100$ and $T_{\text{test}} = 2000$. We then provided inaccurate values of $\nu$ to EKF and UKF. Specifically, we experimented with an under-estimated process noise level obtained using the ratio $\nu = 0.5\nu^\star$ and an over-estimated process noise level using $\nu = 1.5\nu^\star$. The relative increases in average NMSE performances are shown in Table III.

*Interpretation of results*: Table III shows that, as expected, the performances of DANSE remain stable for varying $\nu$, the slight variations in performances are due to the randomness in the data generation. At values $1/\sigma_w^2 = -25$ and $-20$ dB, the mean

TABLE III: Relative increase (%) in average NMSE (in dB) for Lorenz attractor SSM with incorrect noise statistics compared to perfect knowledge (from Table II).

| $1/\sigma_w^2$ [dB] | $\nu$ | -25 | -20 | -10 | 0 | 10 | 20 |
|---|---|---|---|---|---|---|---|
| UKF | $0.5\nu^\star$ | $+16.62$ | $+14.90$ | $+6.36$ | $+2.63$ | $+2.22$ | $+0.92$ |
| | $1.5\nu^\star$ | $+15.70$ | $+9.62$ | $+2.96$ | $+0.31$ | $+0.05$ | $-0.09$ |
| EKF | $0.5\nu^\star$ | $+27.00$ | $+22.18$ | $+3.40$ | $+0.28$ | $+0.13$ | $+0.09$ |
| | $1.5\nu^\star$ | $+15.44$ | $+10.74$ | $+3.66$ | $+0.54$ | $+0.20$ | $+0.04$ |
| DANSE | $0.5\nu^\star$ | $-0.52$ | $-0.17$ | $+0.04$ | $-0.09$ | $+0.13$ | $+0.07$ |
| | $1.5\nu^\star$ | $-0.75$ | $-0.17$ | $+0.05$ | $+0.04$ | $+0.14$ | $+0.03$ |
| KalmanNet [14] | $0.5\nu^\star$ | $-0.21$ | $-0.20$ | $-0.24$ | $+3.53$ | $-0.10$ | $+0.08$ |
| | $1.5\nu^\star$ | $-0.66$ | $+0.09$ | $-0.05$ | $+2.46$ | $+0.12$ | $+0.05$ |

NMSE performances of EKF and UKF clearly degrade. When $\nu = 0.5\nu^\star$, the performances of EKF degrade by $27.00\%$ and $22.18\%$ respectively, and the performances of UKF degrade slightly less, by $16.62\%$ and $14.90\%$ respectively. Overestimating the noise level by providing $\nu = 1.5\nu^\star$, seems to limit the degradation in performances of EKF to $10.7\%$ and UKF to $9.6\%$ at $1/\sigma_w^2 = -20$ dB. Furthermore, the degradation of DANSE is quite comparable to KalmanNet for all $1/\sigma_w^2$, where KalmanNet knows the process dynamics and is also unaffected by changes in process noise statistics. This clearly shows the relevance of DANSE.

*2) Example state trajectory estimates:* We provide an example result of state trajectory estimation at SNR level $1/\sigma_w^2 = -10.0$ dB, with process noise using $\nu^\star = -20$ dB and incorrect information provided to EKF and UKF with $\nu = 0.5\nu^\star$. In Fig. 3, we show the estimation of the $3^{rd}$ state component for EKF, UKF, DANSE and KalmanNet. The state trajectory estimates of UKF and EKF are very close, except at the beginning where UKF is less accurate. DANSE and KalmanNet generally show similar trajectory estimation as EKF and UKF, with more accuracy at local optima of the state trajectory.

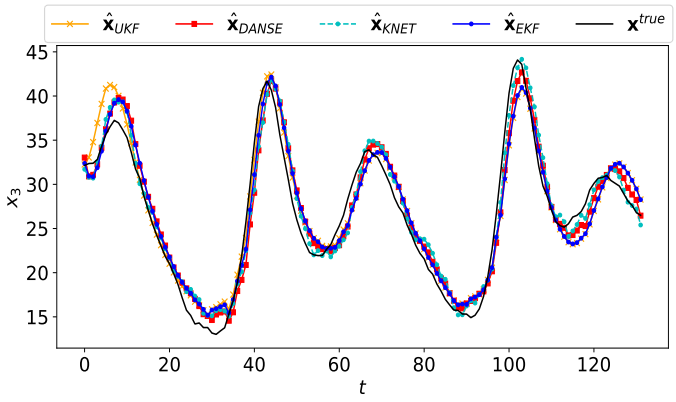


Fig. 3: Estimation of state $x_3$ of the Lorenz attractor model

## IV. Conclusion

We propose DANSE, a non-linear state estimation method, and evaluate it for two different synthetic SSMs and at varying measurement SNR. Our method matches the performances of the KF, EKF, and UKF, with only slight underperformance at low SNR. We also find a slight improvement in performance compared to KalmanNet for a nonlinear SSM. Notably, DANSE performs state estimation without knowledge of the process dynamics or the process noise, which is practically advantageous. In the case of imprecise process noise information, we have shown that DANSE outperforms both EKF and UKF.

## References


[1] R.E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME, D*, vol. 82, pp. 35–44, 1960.

[2] H. Guo, D. Cao, H. Chen, C. Lv, H. Wang, and S. Yang, "Vehicle dynamic state estimation: State of the art schemes and perspectives," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 2, pp. 418–431, 2018.

[3] R.E. Kalman, "New results in linear filtering and prediction theory," *J. Basic Eng.*, vol. 83, pp. 95–108, 1961.

[4] M. Gruber, "An approach to target tracking," Tech. Rep., MIT Lexington Lincoln Lab, 1967.

[5] S.J. Julier and J.K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[6] A. Doucet, N. De Freitas, N. J. Gordon, et al., *Sequential Monte Carlo methods in practice*, vol. 1, Springer, 2001.

[7] S.C. Patwardhan, S. Narasimhan, P. Jagadeesan, B. Gopaluni, and S.L. Shah, "Nonlinear bayesian state estimation: A review of recent developments," *Control Engineering Practice*, vol. 20, no. 10, pp. 933–953, 2012.

[8] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proc. of $8^{th}$ workshop, SSST-8*, 2014, pp. 103–111.

[9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] N. Yadaiah and G. Sowmya, "Neural network based state estimation of dynamical systems," in *The 2006 IEEE international joint conference on neural network proceedings*. IEEE, 2006, pp. 1042–1049.

[11] L. Girin, S. Leglaive, X. Bie, J. Diard, T. Hueber, and X. Alameda-Pineda, "Dynamical variational autoencoders: A comprehensive review," *Foundations and Trends in Machine Learning*, vol. 15, no. 1-2, pp. 1–175, 2021.

[12] M. Fraccaro, S. Kamronn, U. Paquet, and O. Winther, "A disentangled recognition and nonlinear dynamics model for unsupervised learning," *Advances in NeurIPS*, vol. 30, 2017.

[13] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R.J.G. Van Sloun, and Y.C. Eldar, "KalmanNet: Neural network aided Kalman filtering for partially known dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[14] G. Revach, N. Shlezinger, T. Locher, X. Ni, R.J.G. van Sloun, and Y.C. Eldar, "Unsupervised learned Kalman filtering," in *2022 30th European Signal Processing Conference (EUSIPCO)*. IEEE, 2022, pp. 1571–1575.

[15] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 4, Springer, 2006.

[16] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*, John Wiley & Sons, 2006.

[17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NeurIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in NeurIPS*, vol. 30, 2017.

[19] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," *Advances in NeurIPS*, vol. 32, 2019.

[20] D.P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, 2015.

[21] Labbe R, "FilterPy - Kalman and Bayesian filters in Python," URL: https://filterpy.readthedocs.io/en/latest/, 2018.

[22] E.N. Lorenz, "Deterministic nonperiodic flow," *Journal of atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.