

pDANSE: Particle-based Data-driven Nonlinear State Estimation from Nonlinear Measurements

Anubhab Ghosh ^{*}, Yonina C. Eldar [†], Saikat Chatterjee ^{*}

^{*} Digital Futures Centre and School of Elect. Engg. & Comp. Sc., KTH Royal Institute of Technology, Sweden

[†] Faculty of Mathematics and Computer Science, The Weizmann Institute of Science, Israel

^{*} anubhabg@kth.se [†] yonina.eldar@weizmann.ac.il ^{*} sach@kth.se

Abstract—We consider the problem of designing a data-driven nonlinear state estimation (DANSE) method that uses (noisy) nonlinear measurements of a process whose underlying state transition model (STM) is unknown. Such a process is referred to as a model-free process. A recurrent neural network (RNN) provides parameters of a Gaussian prior that characterize the state of the model-free process, using all previous measurements at a given time point. In the case of DANSE, the measurement system was linear, leading to a closed-form solution for the state posterior. However, the presence of a nonlinear measurement system renders a closed-form solution infeasible. Instead, the second-order statistics of the state posterior are computed using the nonlinear measurements observed at the time point. We address the nonlinear measurements using a reparameterization trick-based particle sampling approach, and estimate the second-order statistics of the state posterior. The proposed method is referred to as particle-based DANSE (pDANSE). The RNN of pDANSE uses sequential measurements efficiently and avoids the use of computationally intensive sequential Monte-Carlo (SMC) and/or ancestral sampling. We describe the semi-supervised learning method for pDANSE, which transitions to unsupervised learning in the absence of labeled data. Using a stochastic Lorenz-63 system as a benchmark process, we experimentally demonstrate the state estimation performance for four nonlinear measurement systems. We explore cubic nonlinearity and a camera-model nonlinearity where unsupervised learning is used; then we explore half-wave rectification nonlinearity and Cartesian-to-spherical nonlinearity where semi-supervised learning is used. The performance of state estimation is shown to be competitive vis-à-vis particle filters that have complete knowledge of the STM of the Lorenz-63 system.

Index Terms—Bayesian state estimation, particle filter, neural networks, unsupervised learning, semi-supervised learning.

I. INTRODUCTION

Bayesian state estimation (BSE) of a nonlinear dynamical process from noisy observations (measurements) is an active area of research interest with many applications [1], [2]. The relationship between the state and the measurement is typically characterized using a measurement system that can be linear or nonlinear. Given a measurement system, prime

examples of model-driven BSE methods are Kalman filter (KF), extended KF (EKF), unscented KF (UKF), cubature KF (CKF), and particle filter (PF). All of these approaches use a state-transition model (STM) to characterize the temporal dynamics of the process, as a-priori knowledge for BSE. An STM is typically Markovian. KF is theoretically optimal when both the measurement system and the STM are linear Gaussian. PF uses sequential Monte-Carlo (SMC) methods and is asymptotically optimal if the number of samples (or particles) is asymptotically high.

Here, we consider the design of a data-driven BSE method for a model-free process where there is no knowledge of the STM, and the measurement system is nonlinear. Data-driven methods typically use machine learning techniques, mainly neural networks, and train them on the available data. Recently, the DANSE (data-driven nonlinear state estimation) method was developed and demonstrated for BSE of several nonlinear processes where the measurement system is linear Gaussian [3]. Our main contribution in this article is to extend DANSE for nonlinear measurement systems.

DANSE uses a recurrent neural network (RNN) called the gated recurrent unit (GRU) [4]. It can also use other popular RNN architectures like long short-term memory networks (LSTM) [5]. The RNN provides parameters of a Gaussian prior characterizing the state of the model-free process, using all previous measurements at a discrete time point. The RNN uses sequential measurements over time efficiently, maintaining causality. DANSE uses a linear Gaussian measurement system, which allows for a closed-form Gaussian posterior of the state and prediction of the measurements. The closed-form prediction of the measurements given all the previous measurements helps in training the RNN directly by formulating an appropriate maximum-likelihood based optimization function.

For nonlinear measurements, the analytical tractability is limited and the extension of DANSE is challenging. We address the challenge using a Monte-Carlo (MC) approach - a reparameterization trick-based particle sampling technique. The approach helps to compute the second-order statistics of the posterior of the state, like a PF. Hence, the proposed method is referred to as particle DANSE (pDANSE). Our method enables computing a lower bound on the prediction of the measurements, in turn helping to train the RNN of pDANSE. The usage of RNN in pDANSE remains the same as that of DANSE, which means to provide parameters of a Gaussian prior. Unlike PF, pDANSE does not know the STM

This work has been submitted to the IEEE for future publication. Copyright may be transferred without notice, after which this version will no longer be available. The work is supported by a research grant from Digital Futures Centre. Project title: “DLL: Data-limited learning of complex systems”. Website: <https://www.digitalfutures.kth.se/>. The authors would also like to mention that most of the work was carried out when Y. C. Eldar was at The Weizmann Institute of Science, Israel. Recently, in August 2025, Y. C. Eldar is also affiliated with the Department of Electrical and Computer Engineering, Northeastern University, USA (Email: y.eldar@northeastern.edu).

and does not use computationally intensive SMC.

We now provide a brief review of relevant methods for tackling the BSE of a nonlinear dynamical process from noisy, nonlinear measurements. In the literature, there exists a variety of methods. They can be model-driven, data-driven, or hybrid. As mentioned earlier, model-driven methods assume knowledge of the underlying STM. Notable examples are UKF, CKF, and PF [6]–[10]. The UKF uses a collection of cleverly designed sigma points and corresponding weights to represent an approximate prior distribution of the state (using second-order moments). Using the unscented transform [6], the sigma points are propagated using the dynamical model and subsequently the measurement system to obtain the corresponding posterior estimates. The CKF follows a similar philosophy, but instead of the unscented transform, utilizes the spherical integration rule for designing sigma points and propagating them using the SSM [8], [11, Chap. 8]. The PF is sampling-based and can work with non-Gaussian dynamical systems, unlike the UKF and the CKF. It approximates the underlying prior and posterior distributions of the state using a weighted system of particles [10]. In the simplest version of the PF, known as the Bootstrap filter [9], the STM is used as the proposal distribution, from which the particles at the next time instant are drawn. Then, using the STM and the measurement system, scalar weights for each particle are computed recursively. Finally, to avoid particle degeneracy, resampling is also carried out by sampling particles again using the computed weights [12].

Data-driven methods use machine learning techniques and learn from available training data. The type of learning depends on the availability of data; a supervised learning approach is feasible when pairs of true state trajectories and corresponding noisy measurement trajectories are available as labelled training data, while an unsupervised learning approach uses only noisy measurement trajectories as unlabelled training data. Finally, when a limited amount of labeled data and a large amount of unlabeled data are available, semi-supervised techniques can be viable.

Notable examples of data-driven methods that do not use any STM are DANSE [3] and recurrent filters [13]. Both of them assume a linear measurement system and use an RNN to provide a prior for the underlying state. Another example is the differentiable particle filter (DPF) [14]–[17]. The DPF is the extension of PF, which allows an STM to be learned from data. DPF uses SMC, and a deep neural network (DNN) is used to parameterize the STM. Similarly, DNN-based STMs are used in dynamical variational autoencoder (DVAE) methods [18]–[21]. They use computationally intensive ancestral sampling and variational approximations for the posterior. The DVAE was designed for time-series modeling and can be modified for BSE. An implementation of a specific DVAE approach, known as the deep KF (DKF) [20] was demonstrated for BSE using linear Gaussian measurements and compared with other relevant methods in [3, Section III.]. There are also Gaussian process-based (GP-based) methods for learning the STM using the kernel trick and mainly explored in system identification literature [22]–[25]. The GP parameters are learned using ancestral sampling with particle Gibbs [26], [27].

Hybrid methods borrow ideas from both data-driven and model-driven approaches. A notable class of hybrid methods includes the recent KalmanNet and its variations [28]–[30]. KalmanNet utilizes full or partial knowledge of the STM and the measurement system, and learns the Kalman gain component using training data. The learning mechanism can be supervised [29] or unsupervised [28], and can handle nonlinear measurement systems.

In this article, our contribution builds on our prior work [31], on extending DANSE for handling nonlinear measurements. We have two major additions compared to our prior work. First, while the prior work explored semi-supervised learning as the principal learning framework for pDANSE, in this paper, we also explore the unsupervised learning of pDANSE. We show how the unsupervised learning formulation of pDANSE arises naturally as a sub-case of semi-supervised learning when the labelled data is unavailable. Secondly, we extend the experimental study of our prior work [31] and investigate three additional nonlinear measurement systems comprising a cubic nonlinearity, a high-dimensional, camera model nonlinearity, and a Cartesian-to-spherical nonlinearity. These are explored in addition to the half-wave rectification nonlinearity shown in the prior work [31]. The experiments are carried out using a stochastic Lorenz-63 system as a benchmark nonlinear dynamical process [32]. We demonstrate cubic nonlinearity and a camera-model nonlinearity where unsupervised learning is used; then we demonstrate half-wave rectification nonlinearity and Cartesian-to-spherical nonlinearity where semi-supervised learning is used. The performance of pDANSE is against the model-driven PF, which knows the underlying STM exactly at the time of inference.

The outline of the paper is as follows: In Section II, we begin with a mathematical description of the problem formulation and provide a brief background on DANSE. We then introduce our proposed approach pDANSE, explaining the associated inference and learning problems. Next, in Section III, we describe our experimental setup, where we illustrate the BSE performance of pDANSE for the stochastic Lorenz-63 system using the four different nonlinear measurement systems. Finally, in Section IV, we provide concluding remarks.

For consistency, we use similar notations like the work [3], [31]. Lower case $\mathbf{x}$ represents a vector while x_j represents the j 'th component of $\mathbf{x}$. Upper case, bold symbols like $\mathbf{H}$, represent matrices. A T -length sequence of vectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T$ is compactly denoted as $\mathbf{x}_{1:T} \triangleq \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t, \dots, \mathbf{x}_T$ and $\mathbf{x}_{1:T}^{(i)}$ denotes the i 'th sequence; $\mathbf{x}_t^{(i)}$ is the t 'th vector of the sequence $\mathbf{x}_{1:T}^{(i)}$. With a slight abuse of notation, we use $\mathbf{x}_t^{(l)}$ as the l 'th sample drawn from an appropriate distribution. The operator $(\cdot)^\top$ denotes the transpose, $\mathcal{N}(\cdot; \mathbf{m}, \mathbf{L})$ represents the probability density function (pdf) of a Gaussian distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{L}$ and $\mathbb{E}_p[\cdot]$ and $\text{tr}(\cdot)$ denotes the expectation operator w.r.t. the pdf p and the trace operator respectively. The natural logarithm (with respect to base e) is referred to as $\log$. $|\mathcal{D}|$ denotes the cardinality of the set $\mathcal{D}$. A set of N natural indices is denoted by $[N] \triangleq \{1, 2, \dots, N\}$. The expression $\exp(\mathbf{A})$ is the matrix exponential of $\mathbf{A}$.

II. PARTICLE-BASED DANSE (pDANSE)

In this section, we explain the BSE problem once again mathematically and provide a brief background on DANSE, which caters to the linear measurement system. Then, we explain the inference and learning problem for the proposed method pDANSE. Large parts of the text in Section II-C and Section II-D have similarities to those in [33, section II], [31]. We restate most of the derivations in this paper for completeness.

A. Problem formulation

Let $\mathbf{x}_t \in \mathbb{R}^m$ be the state of a dynamical process, where $t \in \mathbb{Z}_{\geq 0}$ denotes a discrete time index. The state is not directly observed. Instead, the state is observed through measurements $\mathbf{y}_t \in \mathbb{R}^n$, as follows

$$\mathbf{y}_t = \mathbf{h}(\mathbf{x}_t) + \mathbf{w}_t, \quad (1)$$

for $t = 1, \dots, T$. In (1), $\mathbf{h} : \mathbb{R}^m \rightarrow \mathbb{R}^n$ models the measurement relationship and $\mathbf{w}_t \in \mathbb{R}^n$ denotes the measurement noise with known distribution $p(\mathbf{w}_t) = \mathcal{N}(\mathbf{w}_t; \mathbf{0}, \mathbf{C}_w)$. The measurement system can be alternatively represented by the likelihood distribution $p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t), \mathbf{C}_w)$. Throughout this article, we assume that the measurement system (1) is known, i.e., we know the function $\mathbf{h}$ and the Gaussian noise covariance $\mathbf{C}_w$. The Bayesian state estimation (BSE) task is to find an estimate of $\mathbf{x}_t$ using the sequence of past and present measurements $\mathbf{y}_{1:t} \triangleq \mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_t$. That means, estimating the posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t})$, $\forall t$, or estimating statistical moments of $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ when computation of $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is infeasible. This is a filtering problem. Note that we maintain causality, and the BSE can be used for many applications, for example, tracking of the dynamical process from noisy measurements.

For BSE, model-driven methods typically use a state-transition model (STM) of the dynamical process. The STM is used as a-priori knowledge. An STM characterizes the temporal dynamics of the process. A widely used STM is Markovian with additive process noise [23], [25]

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}) + \mathbf{e}_t. \quad (2)$$

Here $\mathbf{f} : \mathbb{R}^m \rightarrow \mathbb{R}^m$ denotes the relationship between the current state $\mathbf{x}_t$ and the previous state $\mathbf{x}_{t-1}$, and $\mathbf{e}_t \in \mathbb{R}^m$ is the process noise at time t . In relevant literature, the measurement system (1) and the STM (2) are together called the state space model (SSM).

In this article, the dynamical process is assumed to be model-free. We do not know the STM of the process, i.e., neither the function $\mathbf{f}$ nor the distribution of $\mathbf{e}_t$. Without an STM, our task is to address the BSE in a data-driven manner. Here, in this article, particle-based DANSE (pDANSE) is developed for nonlinear measurements (1). Before delving into details regarding pDANSE, we first provide a brief background on DANSE, highlighting its key features and mentioning its shortcomings regarding dealing with nonlinear measurements.

B. Background: DANSE

The existing DANSE addresses the BSE problem where the measurements are linear Gaussian. This means that in (1), we have $\mathbf{h}(\mathbf{x}_t) = \mathbf{H}\mathbf{x}_t$ in the case of DANSE. The core of the DANSE method is based on parameterizing a prior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ as a Gaussian distribution with parameters from an RNN [3]. Owing to the causal nature of the state estimation task, the RNN operates in unidirectional mode, using measurements $\mathbf{y}_{1:t-1}$ and providing the Gaussian prior parameters for $\mathbf{x}_t$. The prior distribution is then as follows

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta}) &= \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})), \\ \text{s.t. } \{\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\} &\triangleq \text{RNN}(\mathbf{y}_{1:t-1}; \boldsymbol{\theta}). \end{aligned} \quad (3)$$

Here, $\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}) \in \mathbb{R}^m$ and $\mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}) \in \mathbb{R}^{m \times m}$ denote the mean and covariance matrix of the Gaussian prior distribution, respectively. The covariance matrix is modelled as a diagonal for convenience. After characterising the prior, we can compute the posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t}; \boldsymbol{\theta})$ in closed-form Gaussian distribution using the linear measurement setup. Recall that this means that in (1), we have $p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t; \mathbf{H}\mathbf{x}_t, \mathbf{C}_w)$ in the case of DANSE. The posterior is obtained by using the ‘completing the square’ approach [34, Chap. 2] as

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{y}_{1:t}; \boldsymbol{\theta}) &= \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})} \\ &= \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})}{\int p(\mathbf{y}_t|\mathbf{x}'_t)p(\mathbf{x}'_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})d\mathbf{x}'_t} \\ &= \frac{\mathcal{N}(\mathbf{y}_t; \mathbf{H}\mathbf{x}_t, \mathbf{C}_w)\mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}))}{\int \mathcal{N}(\mathbf{y}_t; \mathbf{H}\mathbf{x}'_t, \mathbf{C}_w)\mathcal{N}(\mathbf{x}'_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}))d\mathbf{x}'_t} \\ &= \frac{\mathcal{N}(\mathbf{y}_t; \mathbf{H}\mathbf{x}_t, \mathbf{C}_w)\mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}))}{\mathcal{N}(\mathbf{y}_t; \mathbf{H}\mathbf{m}_{t|1:t-1}, \mathbf{H}\mathbf{L}_{t|1:t-1}\mathbf{H}^\top + \mathbf{C}_w)} \\ &= \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t}(\boldsymbol{\theta})), \end{aligned} \quad (4)$$

where the posterior mean and covariance matrix at time t are

$$\begin{aligned} \mathbf{m}_{t|1:t}(\boldsymbol{\theta}) &= \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}) + \mathbf{K}_{t|1:t-1}(\boldsymbol{\theta})\boldsymbol{\varepsilon}_t(\boldsymbol{\theta}), \\ \mathbf{L}_{t|1:t}(\boldsymbol{\theta}) &= \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}) - \mathbf{K}_{t|1:t-1}(\boldsymbol{\theta})\mathbf{R}_{\varepsilon,t}(\boldsymbol{\theta})\mathbf{K}_{t|1:t-1}^\top(\boldsymbol{\theta}), \end{aligned} \quad (5)$$

with

$$\begin{aligned} \mathbf{K}_{t|1:t-1}(\boldsymbol{\theta}) &\triangleq \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\mathbf{H}^\top\mathbf{R}_{\varepsilon,t}^{-1}(\boldsymbol{\theta}), \\ \mathbf{R}_{\varepsilon,t}(\boldsymbol{\theta}) &\triangleq \mathbf{H}\mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\mathbf{H}^\top + \mathbf{C}_w, \text{ and} \\ \boldsymbol{\varepsilon}_t(\boldsymbol{\theta}) &\triangleq \mathbf{y}_t - \mathbf{H}\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}). \end{aligned} \quad (6)$$

The parameters of DANSE $\boldsymbol{\theta}$ are learned by maximizing the joint (log) likelihood of the measurements $p(\mathbf{y}_{1:T})$, where by the product rule $p(\mathbf{y}_{1:T}; \boldsymbol{\theta}) = \prod_{t=1}^T p(\mathbf{y}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})$ [3]. The conditional distribution $p(\mathbf{y}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})$ can be found in closed form as

$$\begin{aligned} p(\mathbf{y}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta}) &= \int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1}; \boldsymbol{\theta})d\mathbf{x}_t \\ &= \mathcal{N}(\mathbf{y}_t; \mathbf{H}\mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{C}_w + \mathbf{H}\mathbf{L}_{t|1:t-1}(\boldsymbol{\theta})\mathbf{H}^\top). \end{aligned} \quad (7)$$

In the above, we used (3), and the sum-rule for finding Gaussian marginals as in [34, Chap. 2]. Now, we can formulate a maximum-likelihood based optimization problem for learning

the parameters θ using noisy measurements, as shown in [3, Section II-C]. The overall optimization problem is non-convex and is solved using gradient descent.

Despite these encouraging features, we note that one of the key aspects of DANSE – the closed form of the posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)$ fails to hold in the case of a nonlinear measurement system like (1), since $p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)$ becomes non-Gaussian. Neither does the marginal distribution $p(\mathbf{y}_t|\mathbf{y}_{1:t-1};\theta)$ remain Gaussian either, prohibiting a direct optimization of the RNN parameters θ . In the next subsection, we explain how we address these shortcomings of DANSE.

C. Inference problem (Filtering): pDANSE

Let us now focus once again on the BSE problem with nonlinear measurements described in Section II-A. The inference problem (filtering) comprises finding the posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ at time t . Note that we assume that we do not know the underlying STM. Like DANSE [3], we assume that the prior distribution of $\mathbf{x}_t$ given $\mathbf{y}_{1:t-1}$ in pDANSE is also Gaussian (like in (3)). The parameters of the Gaussian prior, that is, the mean and covariance, are given by an RNN with learnable parameters θ as follows -

$$\text{Prior : } p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\theta) = \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\theta), \mathbf{L}_{t|1:t-1}(\theta)), \\ \{\mathbf{m}_{t|1:t-1}(\theta), \mathbf{L}_{t|1:t-1}(\theta)\} = \text{RNN}(\mathbf{y}_{1:t-1};\theta), \quad (8)$$

where $\mathbf{m}_{t|1:t-1}(\theta) \in \mathbb{R}^m$ and $\mathbf{L}_{t|1:t-1}(\theta) \in \mathbb{R}^{m \times m}$ are obtained using an RNN having the structure shown in [33, section II-A]. We use a diagonal covariance matrix $\mathbf{L}_{t|1:t-1}(\theta)$. Then, using the measurement system (1), we seek to compute the posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)$. However, since (1) is nonlinear, the likelihood distribution is $p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t), \mathbf{C}_w)$, resulting in $p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)$ being intractable. One can note this by observing that

$$p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta) \\ = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\theta)}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1};\theta)} \\ = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\theta)}{\int p(\mathbf{y}_t|\mathbf{x}'_t)p(\mathbf{x}'_t|\mathbf{y}_{1:t-1};\theta)d\mathbf{x}'_t} \quad (9) \\ = \frac{\mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t), \mathbf{C}_w)\mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\theta), \mathbf{L}_{t|1:t-1}(\theta))}{\int \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}'_t), \mathbf{C}_w)\mathcal{N}(\mathbf{x}'_t; \mathbf{m}_{t|1:t-1}(\theta), \mathbf{L}_{t|1:t-1}(\theta))d\mathbf{x}'_t}.$$

From (9), it can be seen that the nonlinear function $\mathbf{h}(\cdot)$ renders the posterior distribution non-Gaussian. It hinders the closed-form computation of the integral in the denominator and an exact posterior distribution as in (4) and (6).

Instead of the closed-form posterior, we compute moments of the posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)$, namely the first and second-order statistical moments, using Monte-Carlo (MC) approximations. Let $\phi: \mathbb{R}^m \rightarrow \mathcal{M}$ denote the moment function that maps the state $\mathbf{x}_t$ to the space of moments $\mathcal{M}$. Note that in the case of the mean vector, $\mathcal{M} = \mathbb{R}^m$, while for the covariance

matrix, $\mathcal{M} = \mathbb{R}^{m \times m}$. The moments can be determined by solving the following expectation

$$\mathbb{E}_{p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)}[\phi(\mathbf{x}_t)] = \int \phi(\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)d\mathbf{x}_t \\ = \int \phi(\mathbf{x}_t)\frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\theta)}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1};\theta)}d\mathbf{x}_t \\ = \int \phi(\mathbf{x}_t)\frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\theta)}{\int p(\mathbf{y}_t|\mathbf{x}'_t)p(\mathbf{x}'_t|\mathbf{y}_{1:t-1};\theta)d\mathbf{x}'_t}d\mathbf{x}_t \\ \approx \sum_{l=1}^L \phi(\mathbf{x}_t^{(l)}(\theta))w_t^{(l)}(\theta), \quad (10)$$

where we have particles sampled from the prior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\theta) = \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\theta), \mathbf{L}_{t|1:t-1}(\theta))$ as follows

$$\mathbf{x}_t^{(l)}(\theta) \sim \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\theta), \mathbf{L}_{t|1:t-1}(\theta)), \\ = \mathbf{m}_{t|1:t-1}(\theta) + \mathbf{L}_{t|1:t-1}^{1/2}(\theta)\epsilon^{(l)}, \quad (11)$$

where $\epsilon^{(l)} \sim \mathcal{N}(\epsilon; \mathbf{0}, \mathbf{I})$, for $l = 1, \dots, L$. L denotes the number of MC samples, and the superscript l is explicitly used as the sampling index. The samples are drawn in (11) using the reparameterization trick [35]. The posterior weights $w_t^{(l)}(\theta)$ for each particle in (10) are

$$w_t^{(l)}(\theta) = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(l)}(\theta))}{\sum_{l'=1}^L p(\mathbf{y}_t|\mathbf{x}_t^{(l')}(\theta))} \\ = \frac{\mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t^{(l)}(\theta)), \mathbf{C}_w)}{\sum_{l'=1}^L \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t^{(l')}(\theta)), \mathbf{C}_w)}, \quad (12)$$

where $\sum_{l=1}^L w_t^{(l)}(\theta) = 1, \forall t$ and using the Gaussian measurement system in (1). In practice, (12) is computed in the log-domain (for numerical stability), with the denominator computed using the well-known log-sum-exp trick [36]

$$\log w_t^{(l)}(\theta) \\ = \log \left(\frac{p(\mathbf{y}_t|\mathbf{x}_t^{(l)}(\theta))}{\sum_{l'=1}^L p(\mathbf{y}_t|\mathbf{x}_t^{(l')}(\theta))} \right) \\ = \log \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t^{(l)}(\theta)), \mathbf{C}_w) \\ - \log \sum_{l'=1}^L \exp \log \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t^{(l')}(\theta)), \mathbf{C}_w), \\ = \log \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t^{(l)}(\theta)), \mathbf{C}_w) \\ - \text{LSE} \left(\left\{ \log \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t^{(l')}(\theta)), \mathbf{C}_w) \right\}_{l'=1}^L \right). \quad (13)$$

Here $\text{LSE}(\{q^{(l)}\}_{l=1}^L) = \log \sum_{l=1}^L \exp q^{(l)}$ denotes the log-sum-exp (LSE) function, for a collection of L real-valued scalars $\{q^{(l)}\}_{l=1}^L$. Using (10), (12), and (13), we can obtain posterior mean and covariance estimates. Specifically, if we set $\phi(\mathbf{x}_t) = \mathbf{x}_t$ in (10), we obtain a posterior mean estimate $\mathbf{m}_{t|1:t}(\theta)$ as follows

$$\mathbf{m}_{t|1:t}(\theta) = \mathbb{E}_{p(\mathbf{x}_t|\mathbf{y}_{1:t};\theta)}[\mathbf{x}_t] \approx \sum_{l=1}^L \mathbf{x}_t^{(l)}(\theta)w_t^{(l)}(\theta). \quad (14)$$

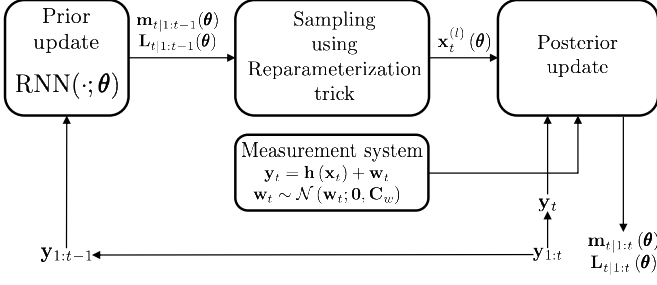


Fig. 1: Schematic of pDANSE at inference time.

Similarly, we get a posterior covariance estimate $\mathbf{L}_{t|1:t}(\boldsymbol{\theta})$ using $\boldsymbol{\phi}(\mathbf{x}_t) = \boldsymbol{\varepsilon}_t \boldsymbol{\varepsilon}_t^\top$, $\boldsymbol{\varepsilon}_t \triangleq \mathbf{x}_t - \mathbf{m}_{t|1:t}(\boldsymbol{\theta})$ in (10)

$$\begin{aligned} \mathbf{L}_{t|1:t}(\boldsymbol{\theta}) &= \mathbb{E}_{p(\mathbf{x}_t|\mathbf{y}_{1:t};\boldsymbol{\theta})} [\boldsymbol{\varepsilon}_t \boldsymbol{\varepsilon}_t^\top] \\ &\approx \sum_{l=1}^L \boldsymbol{\varepsilon}_t^{(l)}(\boldsymbol{\theta}) \left(\boldsymbol{\varepsilon}_t^{(l)}(\boldsymbol{\theta}) \right)^\top w_t^{(l)}(\boldsymbol{\theta}). \end{aligned} \quad (15)$$

In (15), we have $\boldsymbol{\varepsilon}_t^{(l)}(\boldsymbol{\theta}) = \mathbf{x}_t^{(l)}(\boldsymbol{\theta}) - \mathbf{m}_{t|1:t}(\boldsymbol{\theta})$. Also in (14) and (15), $\mathbf{x}_t^{(l)}(\boldsymbol{\theta})$ are sampled from $\mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}))$ for $l = 1, \dots, L$ and the corresponding weights $w_t^{(l)}(\boldsymbol{\theta})$ are computed in log-domain according to (13) $\forall t$. A complete schematic of the operation of pDANSE at inference time is shown in Fig. 1.

D. Learning problem: pDANSE

The learning problem concerns optimizing the parameters $\boldsymbol{\theta}$ of the RNN used in pDANSE. Here, we will show that pDANSE can be trained in an unsupervised manner, and, if required, alternatively in a semi-supervised manner. For the ease of illustration, we begin with the description of the semi-supervised learning setup and later show how one can obtain a purely unsupervised learning setup as a sub-case in the absence of labelled training data. For a linear $\mathbf{h}$ and compressed measurements, we have previously shown the utility of semi-supervised learning in our prior work on Bayesian state estimation using compressed, linear measurements [33].

In a semi-supervised learning scenario, one typically has access to a limited amount of labelled training data and a significantly larger amount of unlabelled training data [33, Section III-A]. Let there be two disjoint sets of indices $\mathcal{I}_s$ and $\mathcal{I}_u$ for supervised and unsupervised training datasets, respectively. The training dataset required for semi-supervised learning is denoted by $\mathcal{D}_{\text{semi}}$. $\mathcal{D}_{\text{semi}}$ consists of two parts – a labelled dataset $\mathcal{D}_s$ having cardinality $|\mathcal{I}_s| \triangleq N_s$ and an unlabelled dataset $\mathcal{D}_u$ having cardinality $|\mathcal{I}_u| \triangleq N_u$, s.t. $N_s \ll N_u \leq N = N_s + N_u$ and $\mathcal{I}_s \cup \mathcal{I}_u = \{1, 2, \dots, N\}$. Specifically, $\mathcal{D}_s = \left\{ \left(\mathbf{x}_{1:T}^{(j)}, \mathbf{y}_{1:T}^{(j)} \right) \right\}_{j \in \mathcal{I}_s}$ consists of pairwise state and noisy measurement trajectories, having the same length T . Note that the sequences could have varying lengths. Here we use the same length across sequences for notational clarity and later ease of implementation. Analogously, $\mathcal{D}_u = \left\{ \mathbf{y}_{1:T}^{(i)} \right\}_{i \in \mathcal{I}_u}$ consists of noisy measurement trajectories. Note that the superscripts i and j are explicitly used as sequence

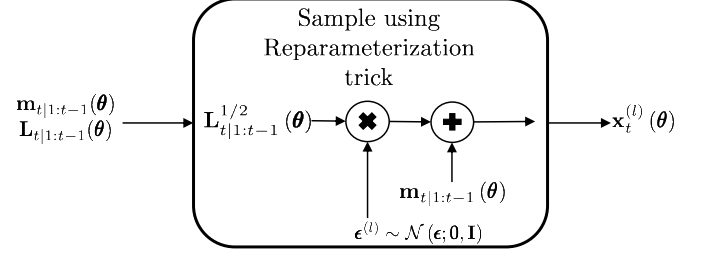


Fig. 2: Sampling from the Gaussian prior in (8) using the reparameterization trick, to obtain samples $\{\mathbf{x}_t^{(l)}(\boldsymbol{\theta})\}_{l=1}^L$.

indices and not to be confused with the superscript l used as sampling index. We can learn the parameters $\boldsymbol{\theta}$ by solving the following maximum-likelihood-based optimization problem:

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} \log p(\mathcal{D}_{\text{semi}}; \boldsymbol{\theta}) \\ &= \arg \max_{\boldsymbol{\theta}} \log p(\mathcal{D}_s, \mathcal{D}_u; \boldsymbol{\theta}) \\ &= \arg \max_{\boldsymbol{\theta}} \log p \left(\left\{ \left(\mathbf{x}_{1:T}^{(i)} | \mathbf{y}_{1:T}^{(j)} \right) \right\}_{j \in \mathcal{I}_s}; \boldsymbol{\theta} \right) \\ &\quad + \log p \left(\left\{ \mathbf{y}_{1:T}^{(i)} \right\}_{i \in \mathcal{I}_s \cup \mathcal{I}_u}; \boldsymbol{\theta} \right). \end{aligned} \quad (16)$$

From (16), the loss function consists of two terms - one constituted using only $\mathcal{D}_s$ and the other using measurement trajectories from $\mathcal{D}_s \cup \mathcal{D}_u$. We now focus on further simplification of the above loss function in the last line of (16). To start with, dropping the superscript i , we notice that the logarithm of the joint marginal distribution $p(\mathbf{y}_{1:t}; \boldsymbol{\theta})$ can be simplified according to the product rule, yielding

$$\begin{aligned} \log p(\mathbf{y}_{1:T}; \boldsymbol{\theta}) &= \log \prod_{t=1}^T p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta}) \\ &= \sum_{t=1}^T \log p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta}). \end{aligned} \quad (17)$$

Thus, we need to obtain the logarithm of the factorized marginal distribution $p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})$. However, the nonlinear measurement function $\mathbf{h}$ prohibits a closed-form expression of $\log p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})$. This is because

$$\begin{aligned} \log p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta}) &= \log \int p(\mathbf{y}_t | \mathbf{x}_t) p(\mathbf{x}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta}) d\mathbf{x}_t \\ &= \log \mathbb{E}_{p(\mathbf{x}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})} [\mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t), \mathbf{C}_w)] \end{aligned} \quad (18)$$

has analytically intractable expectation of the likelihood $\mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t), \mathbf{C}_w)$ with respect to the prior distribution $p(\mathbf{x}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})$. Instead of directly optimizing $\log p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})$, we optimize the following lower bound

$$\begin{aligned} \log p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta}) &= \log \mathbb{E}_{p(\mathbf{x}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})} [\mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t), \mathbf{C}_w)] \\ &\geq \mathbb{E}_{p(\mathbf{x}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})} [\log \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t), \mathbf{C}_w)] \\ &\approx \frac{1}{L} \sum_{l=1}^L \log \mathcal{N}(\mathbf{y}_t; \mathbf{h}(\mathbf{x}_t^{(l)}(\boldsymbol{\theta})), \mathbf{C}_w). \end{aligned} \quad (19)$$

The inequality in the second line of (19) is obtained using Jensen's inequality and the resulting expectation is approximated in the third line by an MC approximation with samples $\mathbf{x}_t^{(l)}(\boldsymbol{\theta})$ sampled from the Gaussian prior $p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_t; \mathbf{m}_{t|1:t-1}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}(\boldsymbol{\theta}))$ for $l = 1, \dots, L$. The parameters of the Gaussian prior are obtained using $\text{RNN}(\cdot; \boldsymbol{\theta})$ as per (8). Crucially, to enable gradient computation in the sampling step of (19), we use the reparameterization trick for multivariate Gaussian [35] as in (11). This is also schematically shown in Fig. 2.

Next, we consider the joint posterior distribution term in (16). Following the standard approach of mean-field approximation [34, Chap. 10] and maintaining causality [31], [33, Section II-C], we assume the following decomposition of the joint posterior distribution

$$\begin{aligned} \log p(\mathbf{x}_{1:T}|\mathbf{y}_{1:T};\boldsymbol{\theta}) &= \log \prod_{t=1}^T p(\mathbf{x}_t|\mathbf{y}_{1:T};\boldsymbol{\theta}) \\ &= \log \prod_{t=1}^T p(\mathbf{x}_t|\mathbf{y}_{1:t};\boldsymbol{\theta}) \\ &= \sum_{t=1}^T \log p(\mathbf{x}_t|\mathbf{y}_{1:t};\boldsymbol{\theta}). \end{aligned} \quad (20)$$

We can employ Bayes' rule on the component term of the RHS of (20) as follows

$$\begin{aligned} \log p(\mathbf{x}_t|\mathbf{y}_{1:t};\boldsymbol{\theta}) &= \log \left(\frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\boldsymbol{\theta})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1};\boldsymbol{\theta})} \right) \\ &= \log p(\mathbf{y}_t|\mathbf{x}_t) + \log p(\mathbf{x}_t|\mathbf{y}_{1:t-1};\boldsymbol{\theta}) \\ &\quad - \log p(\mathbf{y}_t|\mathbf{y}_{1:t-1};\boldsymbol{\theta}). \end{aligned} \quad (21)$$

Using (20), (21), (17) and (19), we simplify the optimization problem in (16) to

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} \log p \left(\left\{ \left(\mathbf{x}_{1:T}^{(j)} | \mathbf{y}_{1:T}^{(j)} \right) \right\}_{j \in \mathcal{I}_s}; \boldsymbol{\theta} \right) \\ &\quad + \log p \left(\left\{ \left(\mathbf{y}_{1:T}^{(i)} \right) \right\}_{i \in \mathcal{I}_u \cup \mathcal{I}_s}; \boldsymbol{\theta} \right) \\ &= \arg \max_{\boldsymbol{\theta}} \sum_{j \in \mathcal{I}_s} \sum_{t=1}^T \left\{ \log p \left(\mathbf{y}_t^{(j)} | \mathbf{x}_t^{(j)} \right) \right. \\ &\quad \left. + \log p \left(\mathbf{x}_t^{(j)} | \mathbf{y}_{1:t-1}^{(j)}; \boldsymbol{\theta} \right) - \log p \left(\mathbf{y}_t^{(j)} | \mathbf{y}_{1:t-1}^{(j)}; \boldsymbol{\theta} \right) \right\} \\ &\quad + \sum_{i \in \mathcal{I}_s \cup \mathcal{I}_u} \sum_{t=1}^T \log p \left(\mathbf{y}_t^{(i)} | \mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta} \right) \\ &= \arg \max_{\boldsymbol{\theta}} \sum_{j \in \mathcal{I}_s} \sum_{t=1}^T \left\{ \log p \left(\mathbf{y}_t^{(j)} | \mathbf{x}_t^{(j)} \right) \right. \\ &\quad \left. + \log p \left(\mathbf{x}_t^{(j)} | \mathbf{y}_{1:t-1}^{(j)}; \boldsymbol{\theta} \right) \right\} + \sum_{i \in \mathcal{I}_u} \sum_{t=1}^T \log p \left(\mathbf{y}_t^{(i)} | \mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta} \right) \\ &\equiv \arg \min_{\boldsymbol{\theta}} \left\{ \mathcal{L}_s(\mathcal{D}_s; \boldsymbol{\theta}) + \mathcal{L}_u(\mathcal{D}_u; \boldsymbol{\theta}) \right\}. \end{aligned} \quad (22)$$

We note from the last line in (22) that the optimization involves a combination of a supervised and an unsupervised loss. The

supervised loss term is

$$\begin{aligned} \mathcal{L}_s(\mathcal{D}_s; \boldsymbol{\theta}) &= - \sum_{j \in \mathcal{I}_s} \sum_{t=1}^T \left\{ \log p \left(\mathbf{y}_t^{(j)} | \mathbf{x}_t^{(j)} \right) \right. \\ &\quad \left. + \log p \left(\mathbf{x}_t^{(j)} | \mathbf{y}_{1:t-1}^{(j)}; \boldsymbol{\theta} \right) \right\} \\ &= - \sum_{j \in \mathcal{I}_s} \sum_{t=1}^T \left\{ \log \mathcal{N} \left(\mathbf{y}_t^{(j)}; \mathbf{h} \left(\mathbf{x}_t^{(j)} \right), \mathbf{C}_w \right) \right. \\ &\quad \left. + \log \mathcal{N} \left(\mathbf{x}_t^{(j)}; \mathbf{m}_{t|1:t-1}^{(j)}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}^{(j)}(\boldsymbol{\theta}) \right) \right\}, \end{aligned} \quad (23)$$

where $\{\mathbf{m}_{t|1:t-1}^{(j)}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}^{(j)}(\boldsymbol{\theta})\}$ is obtained using (8) with $\mathbf{y}_{1:t-1}^{(j)}$ as input (s.t. $j \in \mathcal{I}_s$). The unsupervised loss term using the lower bound (19) is

$$\begin{aligned} \mathcal{L}_u(\mathcal{D}_u; \boldsymbol{\theta}) &= - \frac{1}{L} \sum_{i \in \mathcal{I}_u} \sum_{t=1}^T \sum_{l=1}^L \log \mathcal{N} \left(\mathbf{y}_t^{(i)}; \mathbf{h} \left(\mathbf{x}_t^{(i,l)}(\boldsymbol{\theta}) \right), \mathbf{C}_w \right). \end{aligned} \quad (24)$$

Here $\mathbf{x}_t^{(i,l)}(\boldsymbol{\theta})$ is sampled for $l = 1, \dots, L$ using (11), with $\{\mathbf{m}_{t|1:t-1}^{(i)}(\boldsymbol{\theta}), \mathbf{L}_{t|1:t-1}^{(i)}(\boldsymbol{\theta})\}$ obtained using (8) with $\mathbf{y}_{1:t-1}^{(i)}$ as input (s.t. $i \in \mathcal{I}_u$). We remark that the effect of the supervised loss term (23) on the optimization problem (22) depends on the number of samples available in $\mathcal{D}_s$, i.e. $|\mathcal{I}_s|$. Typically $|\mathcal{I}_s| \ll N$ in a semi-supervised learning scenario. In the absence of any supervised data, i.e. $|\mathcal{I}_s| = 0$, we obtain an unsupervised learning scenario from (22) as a subcase, as follows:

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg \max_{\boldsymbol{\theta}} \log p(\mathcal{D}_{\text{semi}}; \boldsymbol{\theta}) \Big|_{|\mathcal{I}_s|=0} \\ &= \arg \max_{\boldsymbol{\theta}} \log p(\mathcal{D}_u; \boldsymbol{\theta}) \\ &= \arg \max_{\boldsymbol{\theta}} \log p \left(\left\{ \mathbf{y}_{1:T}^{(i)} \right\}_{i \in \mathcal{I}_s \cup \mathcal{I}_u}; \boldsymbol{\theta} \right) \\ &= \arg \max_{\boldsymbol{\theta}} \sum_{i \in \mathcal{I}_u} \sum_{t=1}^T \log p \left(\mathbf{y}_t^{(i)} | \mathbf{y}_{1:t-1}^{(i)}; \boldsymbol{\theta} \right) \\ &\equiv \arg \min_{\boldsymbol{\theta}} \mathcal{L}_u(\mathcal{D}_u; \boldsymbol{\theta}), \end{aligned} \quad (25)$$

where $\mathcal{L}_u(\mathcal{D}_u; \boldsymbol{\theta})$ is defined in (24). Note that in the case of DANSE, we would be able to directly optimize $\log p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})$ as it is possible to obtain it in closed-form as shown in (7). However, in the presence of a nonlinear measurement function $\mathbf{h}$, we resort to maximizing a lower bound of $\log p(\mathbf{y}_t | \mathbf{y}_{1:t-1}; \boldsymbol{\theta})$ given in (19), as shown in (25).

III. EXPERIMENTS AND RESULTS

In this section, we show the BSE performance of the proposed pDANSE against the PF. pDANSE does not use any knowledge of STM, and PF uses the full knowledge of STM. Also, PF is asymptotically optimal and hence used as a benchmark. We investigate four nonlinear measurement systems. They are: cubic nonlinearity, camera model nonlinearity [37], half-wave rectification nonlinearity, and Cartesian-to-spherical nonlinearity [38]. For all our experiments, we use a stochastic Lorenz-63 system [32] as a benchmark process, and it has

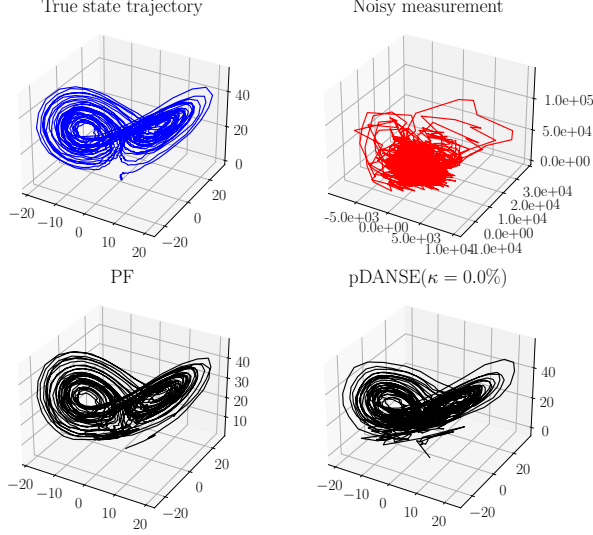


Fig. 3: Visual illustration of the BSE performance for cubic nonlinearity in an unsupervised learning setup at SMNR = 20 dB. Top-left - A true state trajectory for the stochastic Lorenz-63 system from $\mathcal{D}_{\text{test}}$. Top-right - The corresponding noisy measurement trajectory. Bottom-left - The estimated trajectory of PF. Bottom-right - The estimated trajectory of pDANSE.

been extensively used in the literature [3], [28]–[30], [39]. The Lorenz-63 STM is

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{F}_t(\mathbf{x}_t)\mathbf{x}_t + \mathbf{e}_t \in \mathbb{R}^3, \\ \text{s.t. } \mathbf{F}_t(\mathbf{x}_t) &= \exp\left(\begin{bmatrix} -10 & 10 & 0 \\ 28 & -1 & -x_{t,1} \\ 0 & x_{t,1} & -\frac{8}{3} \end{bmatrix} \Delta\right), \end{aligned} \quad (26)$$

where the process noise $\mathbf{e}_t \sim \mathcal{N}(\mathbf{e}_t; \mathbf{0}, \mathbf{C}_e)$ with $\mathbf{C}_e = \sigma_e^2 \mathbf{I}_3$, the step-size $\Delta = 0.02$ seconds, and σ_e^2 was set corresponding to -10 dB. In our simulations, we use 5th order Taylor series approximation for $\mathbf{F}_t(\mathbf{x}_t)$ shown in (26) similar to [3], [29], [39]. Note that $\dim(\mathbf{x}_t) \triangleq m = 3$ for all our experiments.

For ease of illustration, we define $\kappa \triangleq N_s/N$ to indicate the % of labelled data usage in training of pDANSE. Note that $0 \leq \kappa \leq 1$; $\kappa = 0$ (or 0%) refers to unsupervised learning and $\kappa = 1$ (or 100%) refers to a fully supervised learning framework. We train pDANSE on $\mathcal{D}_{\text{semi}}$ having a specified κ , $N = N_s + N_u = 1000$ trajectories, with each trajectory having the same length $T = 200$. After training, we test pDANSE and the PF on a labelled dataset $\mathcal{D}_{\text{test}} = \left\{ \left(\mathbf{x}_{1:T_{\text{test}}}^{(j)}, \mathbf{y}_{1:T_{\text{test}}}^{(j)} \right) \right\}_{j=1}^{N_{\text{test}}}$ having $N_{\text{test}} = 100$ trajectories each of length $T_{\text{test}} = 2000$, similar to [3]. For our experiments, we generate noisy test data where we use $\mathbf{C}_w = \sigma_w^2 \mathbf{I}_3$; here σ_w^2 is chosen using the average signal-to-measurement noise ratio (SMNR) in dB scale [3, section III-C]

SMNR

$$= \frac{1}{N_{\text{test}}} \sum_{j=1}^{N_{\text{test}}} 10 \log_{10} \sum_{t=1}^{T_{\text{test}}} \frac{\mathbb{E} \left[\left\| \mathbf{h}(\mathbf{x}_t^{(j)}) - \mathbb{E}[\mathbf{h}(\mathbf{x}_t^{(j)})] \right\|_2^2 \right]}{n\sigma_w^2}.$$

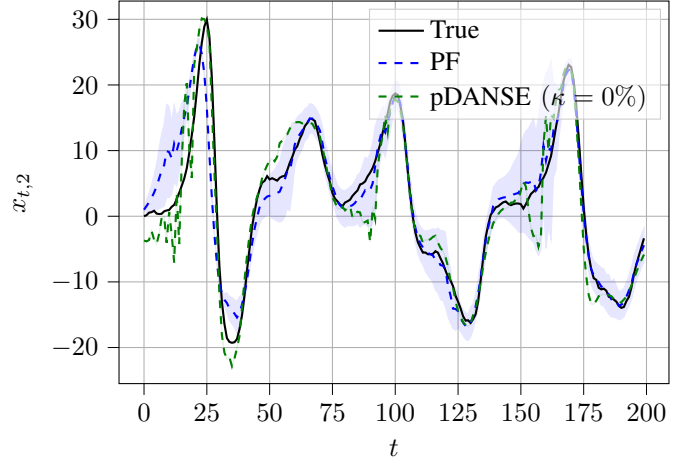


Fig. 4: Demonstrating the qualitative performance of pDANSE ($\kappa = 0\%$) for the BSE problem using noisy, element-wise cubic measurements of the Lorenz-63 process, at SMNR = 20 dB. A chosen realization of a true state trajectory from $\mathcal{D}_{\text{test}}$ is shown in black. The estimated state trajectories are the corresponding posterior mean estimates using PF (shown in blue) and pDANSE (shown in green). The shaded region indicates a $\pm 3\sigma$ confidence for the posterior mean estimate.

A low value of SMNR indicates a high strength of the measurement noise. Similarly, noisy measurements of the training dataset are generated.

The BSE performance evaluation metric is the normalized mean squared error (NMSE) in dB scale, defined as [3, section III]

$$\text{NMSE} = \frac{1}{N_{\text{test}}} \sum_{j=1}^{N_{\text{test}}} 10 \log_{10} \left(\frac{\sum_{t=1}^{T_{\text{test}}} \|\mathbf{x}_t^{(j)} - \hat{\mathbf{x}}_t^{(j)}\|_2^2}{\sum_{t=1}^{T_{\text{test}}} \|\mathbf{x}_t^{(j)}\|_2^2} \right).$$

In the above equation, $\hat{\mathbf{x}}_t$ is set as the posterior mean estimate $\mathbf{m}_{t|1:t}(\boldsymbol{\theta})$, shown in (14).

The RNN that we used in the implementation is a GRU, with 2 hidden layers and 80 hidden units in each layer [4]. The output of the GRU was further mapped using 1-layer, feed-forward networks with 128 hidden units to obtain the mean and covariance in (8) similar to [33]. The specific GRU architecture was decided by grid search on a held-out validation dataset. We perform the simulations using Python and PyTorch [40]¹. pDANSE was trained with a single GPU support using mini-batch gradient descent with a batch size of 128. The optimizer was Adam with an adaptive learning rate starting at 10^{-4} and reduced by 20% after a fixed number of epochs [41].

A. Experiments in unsupervised learning setups

We begin our experiments with two nonlinearities in unsupervised learning setups where $\kappa = 0\%$. The nonlinearities are: an element-wise, cubic nonlinearity and a camera model nonlinearity inspired by [37].

¹The implementation code will be made available upon request.

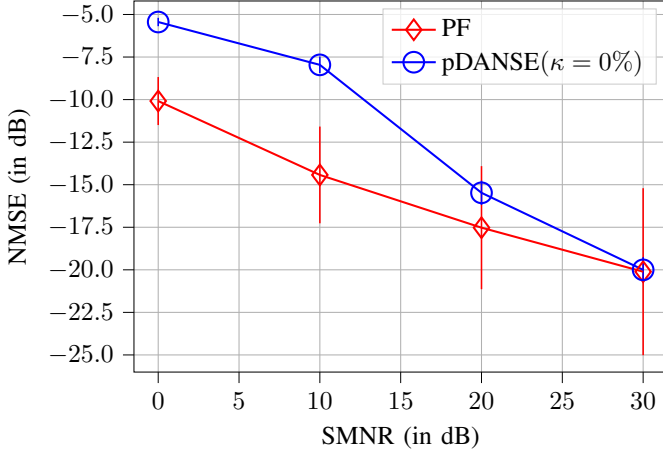


Fig. 5: NMSE (in dB) on $\mathcal{D}_{\text{test}}$ vs. SMNR (in dB), demonstrating the performance of pDANSE ($\kappa = 0\%$) for the BSE task using noisy, cubic measurements of the Lorenz-63 process. The nonlinear function $\mathbf{h}$ is defined in (27). pDANSE was trained using $N = 1000$, $T = 200$.

1) *Cubic nonlinearity*: Following (1), the elements of $\mathbf{h}(\mathbf{x}_t)$ are as follows:

$$h_i(\mathbf{x}_t) = x_{t,i}^3, \quad i = 1, 2, 3. \quad (27)$$

Subsequently, we added Gaussian measurement noise $\mathbf{w}_t \in \mathbb{R}^3$, s.t. $\mathbf{w}_t \sim \mathcal{N}(\mathbf{w}_t; \mathbf{0}, \sigma_w^2 \mathbf{I}_3)$ to create noisy measurements $\mathbf{y}_t \in \mathbb{R}^3$. For pDANSE, we use $L = 10$ MC samples for training and testing. We compare our approach with PF having 300 particles. A higher number of particles was used for the PF to serve as a strong benchmark. In Fig. 3, we provide a visual illustration of the performance of pDANSE, and compare against that of the PF, at SMNR = 20 dB. We observe that pDANSE provides good performance despite being trained in an unsupervised manner. In Fig. 4, we show a time plot of a randomly selected true state trajectory and corresponding estimates using PF and pDANSE, along with $\pm 1\sigma$ confidence regions. Finally, we show NMSE-versus-SMNR performance comparison vis-a-vis PF in Fig. 5. The gap between pDANSE and PF is higher at the low range of SMNR. Note that PF is an asymptotically optimal estimator.

2) *Camera model nonlinearity*: We now investigate the use of camera nonlinearity that provides high-dimensional measurements. Our inspiration for this investigation stems from the experimental setup described in [14], [37], where camera nonlinearity was used. The 3-dimensional state vector of the Lorenz-63 attractor is transformed using a Gaussian point spread function into a high-dimensional image, thereby mimicking a camera acquisition model. In our case, we transformed a 3-dimensional $\mathbf{x}_t$ to an 8×8 image, or equivalently 64-dimensional measurements. Note that one can also choose larger image sizes, e.g. 28×28 . In this case, we opt for an image size of 8×8 for two reasons. The main reason is that it corresponds to a low-resolution image, representing a cheap and noisy camera model. That makes the estimation task challenging. The second reason is that processing high-resolution images requires a high computational resource,

and hence, we avoid that. The j^{th} noise-free measurement component is constructed as follows:

$$h_j(\mathbf{x}_t; \boldsymbol{\chi}^{(j)}) = 10 \exp \left(-\frac{1}{2x_{t,3}} \left\| \begin{bmatrix} \chi_1^{(j)} - x_{t,1} \\ \chi_2^{(j)} - x_{t,2} \end{bmatrix} \right\|_2^2 \right). \quad (28)$$

Using (28), for $j = 1, \dots, 64$, one can generate a complete noise-free measurement vector $\mathbf{h}(\mathbf{x}_t) \in \mathbb{R}^{64}$. In (28), the 3-dimensional state $\mathbf{x}_t \in \mathbb{R}^3$ is mapped using a 2-dimensional reference point $\boldsymbol{\chi}^{(j)} = [\chi_1^{(j)} \quad \chi_2^{(j)}]^\top$ s.t. $\chi_1^{(j)} \in [-30, 30]$, $\chi_2^{(j)} \in [-40, 40]$. Both the ranges $[-30, 30]$ and $[-40, 40]$ are divided into 8 equally spaced grid points. This creates a 2-dimensional image grid of size 8×8 . The measurement function $\mathbf{h}$ thus maps $\mathbb{R}^3 \rightarrow \mathbb{R}^{64}$. Subsequently, we added Gaussian measurement noise $\mathbf{w}_t \in \mathbb{R}^{64}$, s.t. $\mathbf{w}_t \sim \mathcal{N}(\mathbf{w}_t; \mathbf{0}, \sigma_w^2 \mathbf{I}_{64})$ to create noisy measurements $\mathbf{y}_t \in \mathbb{R}^{64}$ following (1). Overall, a measurement sequence is a sequence of images, like a video.

In Fig. 6, we provide a visual illustration of the performance of pDANSE at SMNR = 0 dB. pDANSE uses $L = 10$ MC samples during training and testing. Note that, in this case, we have measurements that are low-resolution images with a considerable strength of noise (signal and noise powers are the same), and still unsupervised learning provides a reasonable performance. We also did NMSE-versus-SMNR performance comparison with PF, and that has a similar trend to Fig. 3. We skip to show the NMSE-versus-SMNR plots due to brevity. Upon repeated experiments, we find that, in some instances, the unknown state can be estimated by pDANSE only up to a constant phase factor.

B. Experiments in semi-supervised learning setups

In this section, we now show the BSE performance of pDANSE for two different nonlinearities in the semi-supervised learning setup. Here we vary κ and examine the performance trends. Here, the nonlinearities are: an element-wise, half-wave rectification nonlinearity and a Cartesian-to-spherical nonlinearity [38]. For training and testing of pDANSE, we use $L = 10$ MC samples as previously. For the PF, we experimentally found that 100 particles were sufficient to provide a reasonably good benchmark comparison.

1) *Half-wave rectification nonlinearity*: Here, we present the BSE performance of the proposed pDANSE method, using noisy, half-wave rectified measurements. Following (1), the elements of the measurement function $\mathbf{h}(\mathbf{x}_t)$ in this case are defined as follows:

$$h_i(\mathbf{x}_t) = \max(0, x_{t,i}), \quad i = 1, 2, 3. \quad (29)$$

We show the quantitative BSE performance of pDANSE at different κ values versus the model-driven PF in Fig. 8. Similar experiments have also been shown in our prior work [31, Section IV.]. We observe that the unsupervised version of pDANSE ($\kappa = 0\%$) performs poorly compared to the model-driven PF on the BSE task. Our ansatz is that this occurs due to the loss of information about the unknown state caused by the rectification nonlinearity (29), rendering unsupervised learning challenging. We overcome this challenge by using a limited amount of labelled data ($\kappa > 0\%$), and we observe that

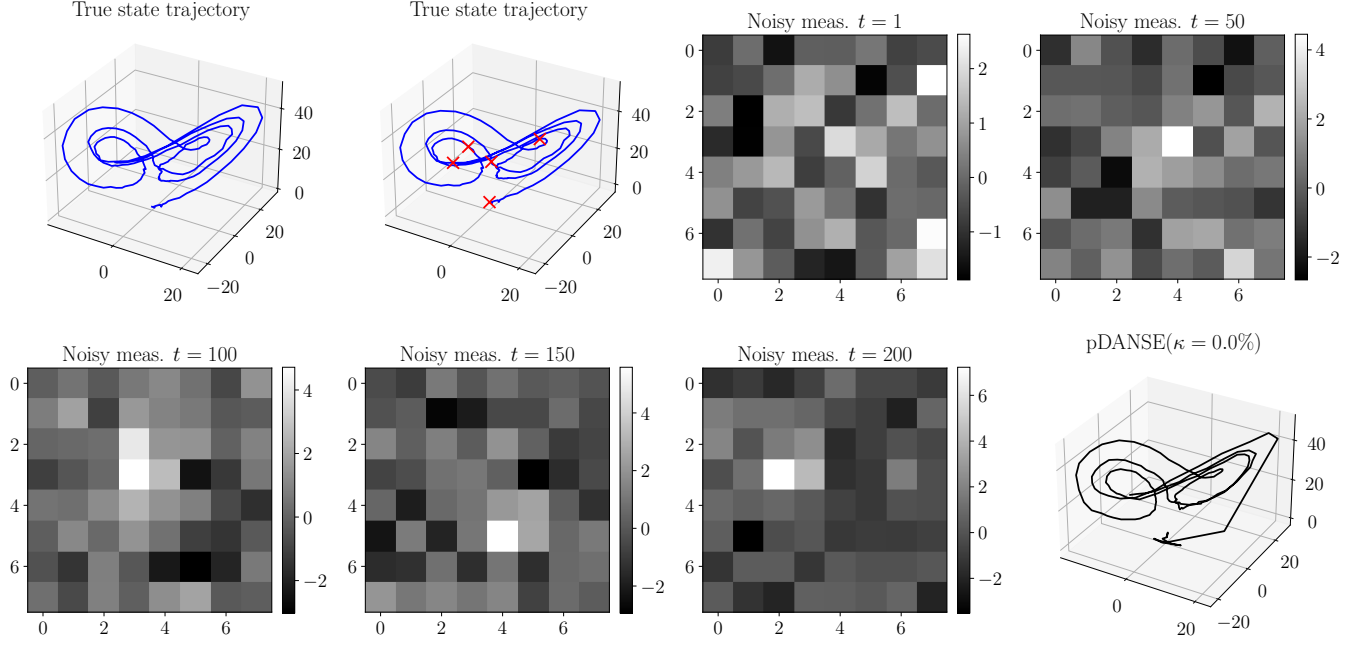


Fig. 6: Visual illustration of the BSE performance for camera model nonlinearity (8×8 image) in an unsupervised learning setup, at SMNR = 0 dB. A true state trajectory from $\mathcal{D}_{\text{test}}$ is shown in blue (first row, first plot). The snapshots of the noisy measurements at five different time instants ($t = 1, 50, 100, 150, 200$) marked by the red crosses in the true state trajectory (first row, second plot) are shown as images (first row, third image to second row, third image). The estimated state trajectory using pDANSE is shown in black (second row, last plot). A shorter test set trajectory ($T_{\text{test}} = 200$) is shown for clarity.

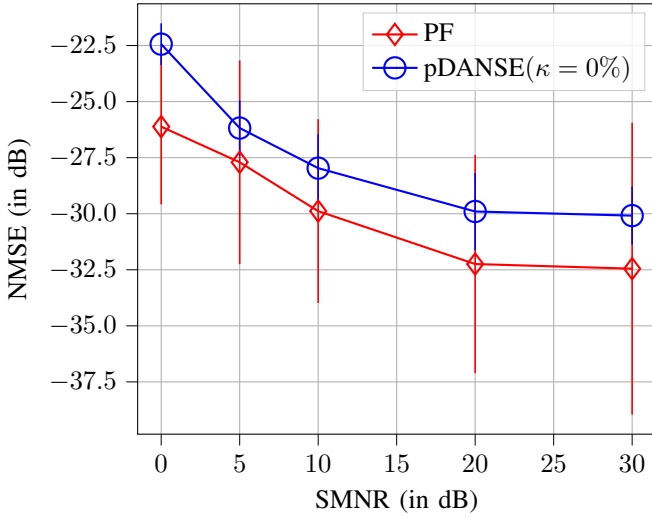


Fig. 7: NMSE (in dB) on $\mathcal{D}_{\text{test}}$ vs. SMNR (in dB), demonstrating the performance of pDANSE ($\kappa = 0\%$) against the PF for the BSE task using noisy, high-dimensional measurements of the Lorenz-63 process. The nonlinear function is defined in (28). The PF is run with 100 particles at the time of inference. pDANSE was trained using a training dataset consisting of $N = 1000$ trajectories of length $T = 200$ each.

the semi-supervised versions of pDANSE ($\kappa > 0\%$) perform quite close to that of the PF. Furthermore, we observe in Fig. 8 that there is significant improvement with just $\kappa = 1\%$, and

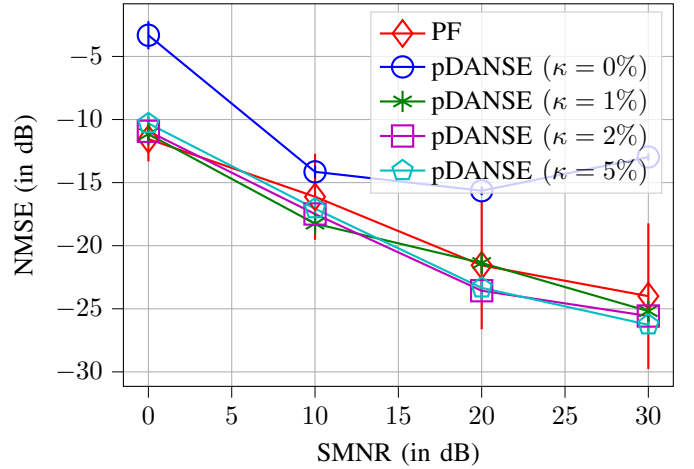


Fig. 8: NMSE (in dB) on $\mathcal{D}_{\text{test}}$ vs. SMNR (in dB), demonstrating the performance of pDANSE ($\kappa \geq 0\%$) vis-à-vis the PF for the BSE task using noisy, half-wave rectified measurements of the Lorenz-63 process. The nonlinear function is defined in (29). While pDANSE ($\kappa = 0\%$) underperforms, pDANSE ($\kappa > 0\%$) perform quite satisfactorily compared to the PF.

there is no drastic improvement by increasing $\kappa = 5\%$. This corroborates the efficiency of semi-supervised learning, where the usage of a limited amount of labeled data is helpful.

2) *Cartesian-to-spherical nonlinearity*: Next, we consider the Cartesian-to-spherical coordinate transformation function

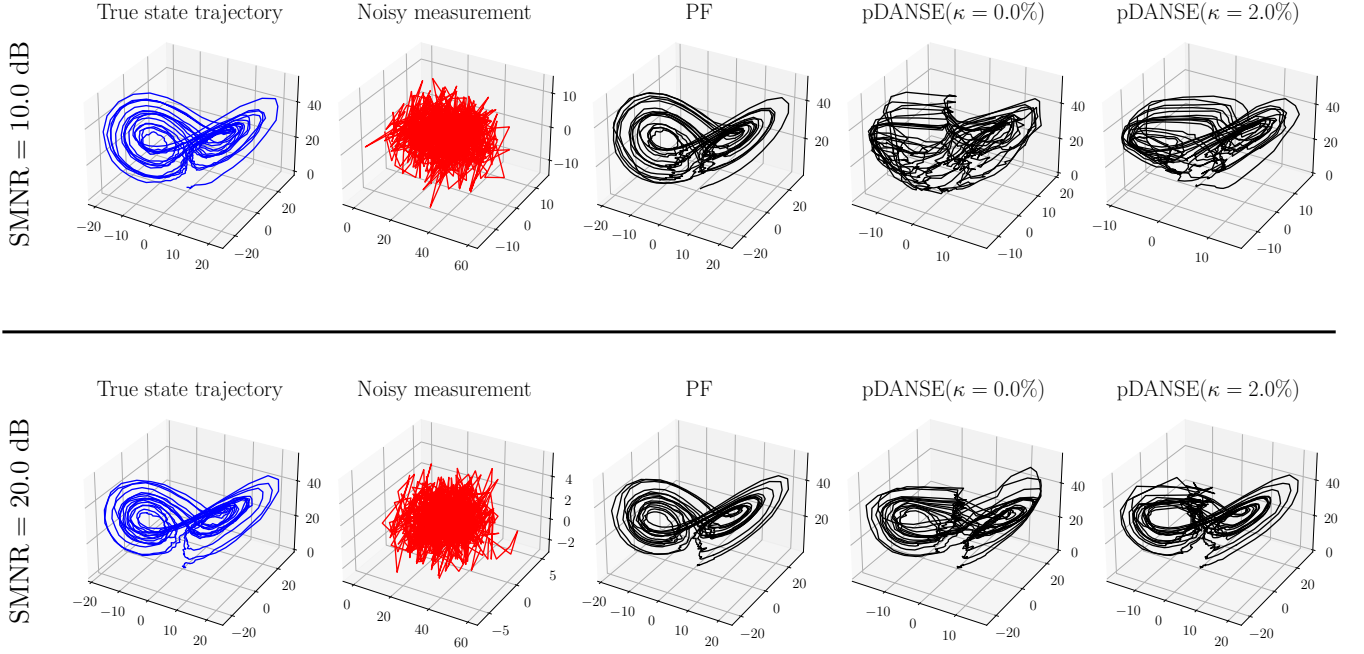


Fig. 9: Visual illustration of the performance of pDANSE for different κ for the BSE problem using noisy, Cartesian-to-spherical measurements of the Lorenz-63 process, at SMNR = 10 dB and SMNR = 20 dB. A shorter trajectory length $T_{\text{test}} = 1000$ is shown for clarity and ease of understanding. Far left - An example of a true state trajectory is shown in blue. Middle left - The corresponding noisy measurement (shown in red). Middle - The estimated state trajectory (shown in black) is the posterior mean estimates using the PF. Middle right and far right - The estimated state trajectories (shown in black) are the corresponding posterior mean estimates using PF and pDANSE at $\kappa = 0\%$ and $\kappa = 2\%$.

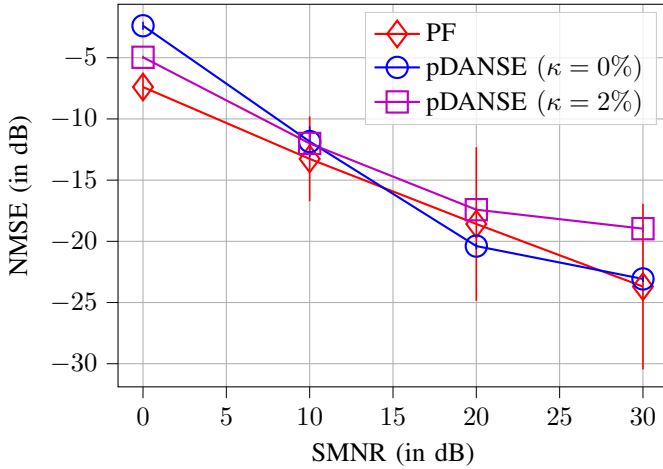


Fig. 10: NMSE (in dB) on $\mathcal{D}_{\text{test}}$ vs. SMNR (in dB), demonstrating the performance of pDANSE ($\kappa = 0\%$) and pDANSE ($\kappa = 2\%$) for the BSE task using noisy, Cartesian-to-spherical measurements of the Lorenz-63 process. The nonlinear function is defined in (30). The comparison is against the model-driven PF.

ment function $\mathbf{h}(\mathbf{x}_t)$ in this case are defined as follows:

$$\mathbf{h}(\mathbf{x}_t) = \begin{bmatrix} h_1(\mathbf{x}_t) \\ h_2(\mathbf{x}_t) \\ h_3(\mathbf{x}_t) \end{bmatrix} = \begin{bmatrix} \sqrt{\sum_{i=1}^3 x_{t,i}^2} \\ \tan^{-1} \left(\frac{x_{t,2}}{x_{t,1}} \right) \\ \tan^{-1} \left(\frac{x_{t,3}}{\sqrt{x_{t,1}^2 + x_{t,2}^2}} \right) \end{bmatrix}. \quad (30)$$

The measurement function in (30) is of significance in target tracking, as in the case of a dish radar [38].

This form of the nonlinearity in (30) renders the BSE task quite challenging. The nonlinearity has phase wrapping components in the second and third elements in (30). We visually illustrate the BSE performance of PF and pDANSE in Fig. 9 at two SMNR values at 10 dB and 20 dB. The BSE task appears quite challenging for both the semi-supervised and unsupervised versions, since they are uninformed about the STM and the phase wrapping components. We observe that PF, which knows the underlying STM, performs quite well. Also, pDANSE ($\kappa = 2\%$) performs better compared to pDANSE ($\kappa = 0\%$). We observe that the effect of the phase wrapping in the state estimate is noticeably reduced using semi-supervised learning compared to unsupervised learning.

Next, we show NMSE versus SMNR plots for pDANSE against PF in Fig. 10. Both the unsupervised pDANSE ($\kappa = 0\%$) and the semi-supervised pDANSE ($\kappa = 2\%$) are compared against the model-driven PF. pDANSE provides similar performance to that of the PF.

as a nonlinearity. Following (1), the elements of the measure-

IV. CONCLUSION

This work presents particle DANSE (pDANSE) – a method that utilizes an RNN-based Gaussian prior, which, together with a known, nonlinear measurement system, can handle the task of Bayesian state estimation for a nonlinear dynamical process without requiring knowledge of the state transition model. We show that pDANSE can be trained in an unsupervised manner and, if necessary, also in a semi-supervised manner. Owing to the presence of a nonlinearity in the measurement system, pDANSE can only provide second-order statistics of the posterior distribution at the time of inference. Using the benchmark stochastic Lorenz-63 system, we compare pDANSE against the model-driven PF, which is asymptotically optimal. Using four nonlinear measurement systems, we show that pDANSE offers competitive Bayesian state estimation performance compared to that of the PF.

REFERENCES

- [1] K. Dehghanpour, Z. Wang, J. Wang, Y. Yuan, and F. Bu, “A survey on state estimation techniques and challenges in smart distribution systems,” *IEEE Transactions on Smart Grid*, vol. 10, no. 2, pp. 2312–2322, 2019.
- [2] N. Shlezinger, G. Revach, A. Ghosh, S. Chatterjee, S. Tang, T. Imbiriba, J. Dunik, O. Straka, P. Closas, and Y. C. Eldar, “Artificial Intelligence-Aided Kalman Filters: AI-Augmented Designs for Kalman-Type Algorithms,” *IEEE Signal Processing Magazine*, pp. 2–26, 2025.
- [3] A. Ghosh, A. Honoré, and S. Chatterjee, “DANSE: Data-Driven Non-Linear State Estimation of Model-Free Process in Unsupervised Learning Setup,” *IEEE Transactions on Signal Processing*, vol. 72, pp. 1824–1838, 2024.
- [4] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” in *Proc. of 8th workshop, SSST-8*, 2014, pp. 103–111.
- [5] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [6] S.J. Julier and J.K. Uhlmann, “Unscented filtering and nonlinear estimation,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.
- [7] E.A. Wan and R. Van Der Merwe, “The unscented Kalman filter for nonlinear estimation,” in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. IEEE, 2000, pp. 153–158.
- [8] I. Arasaratnam and S. Haykin, “Cubature Kalman filters,” *IEEE Transactions on Automatic Control*, vol. 54, no. 6, pp. 1254–1269, 2009.
- [9] N. J. Gordon, D. J. Salmond, and A. FM. Smith, “Novel approach to nonlinear/non-gaussian bayesian state estimation,” in *IEE proceedings F (radar and signal processing)*. IET, 1993, vol. 140, pp. 107–113.
- [10] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking,” *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.
- [11] S. Särkkä and L. Svensson, *Bayesian filtering and smoothing*, vol. 17, Cambridge university press, 2023.
- [12] A. Doucet, N. De Freitas, N. J. Gordon, et al., *Sequential Monte Carlo methods in practice*, vol. 1, Springer, 2001.
- [13] D. Ruhe and P. Forré, “Self-supervised inference in state-space models,” in *ICLR*, 2021.
- [14] W. L. van Nierop, N. Shlezinger, and R. J. van Sloun, “Deep variational sequential monte carlo for high-dimensional observations,” in *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025, pp. 1–5.
- [15] A. Corenflos, J. Thornton, G. Deligiannidis, and A. Doucet, “Differentiable particle filtering via entropy-regularized optimal transport,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 2100–2111.
- [16] X. Chen and Y. Li, “An overview of differentiable particle filters for data-adaptive sequential bayesian inference,” 2023.
- [17] R. Jonschkowski, D. Rastogi, and O. Brock, “Differentiable particle filters: End-to-end learning with algorithmic priors,” *Robotics: Science and Systems XIV*, 2018.
- [18] 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.
- [19] R.G. Krishnan, U. Shalit, and D. Sontag, “Deep Kalman filters,” *arXiv preprint arXiv:1511.05121*, 2015.
- [20] R. Krishnan, U. Shalit, and D. Sontag, “Structured inference networks for nonlinear state space models,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, vol. 31.
- [21] 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.
- [22] J. Ko and D. Fox, “GP-BayesFilters: Bayesian filtering using Gaussian process prediction and observation models,” *Autonomous Robots*, vol. 27, pp. 75–90, 2009.
- [23] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen, “Bayesian inference and learning in gaussian process state-space models with particle mcmc,” *Advances in neural information processing systems*, vol. 26, 2013.
- [24] R. Frigola, Y. Chen, and C. E. Rasmussen, “Variational gaussian process state-space models,” *Advances in neural information processing systems*, vol. 27, 2014.
- [25] A. Svensson, A. Solin, S. Särkkä, and T. B. Schön, “Computationally efficient bayesian learning of gaussian process state space models,” in *Artificial Intelligence and Statistics*. PMLR, 2016, pp. 213–221.
- [26] F. Lindsten, T. Schön, and M. Jordan, “Ancestor sampling for particle gibbs,” *Advances in Neural Information Processing Systems*, vol. 25, 2012.
- [27] F. Lindsten, M. I. Jordan, and T. B. Schön, “Particle gibbs with ancestor sampling,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2145–2184, 2014.
- [28] 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.
- [29] 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.
- [30] X. Ni, G. Revach, and N. Shlezinger, “Adaptive kalmannet: Data-driven kalman filter with fast adaptation,” in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 5970–5974.
- [31] A. Ghosh, Y. C. Eldar, and S. Chatterjee, “Particle-based data-driven nonlinear state estimation of model-free process from nonlinear measurements,” in *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2025, pp. 1–5.
- [32] E.N. Lorenz, “Deterministic nonperiodic flow,” *Journal of atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.
- [33] A. Ghosh, Y. C. Eldar, and S. Chatterjee, “Semi-Supervised Model-Free Bayesian State Estimation from Compressed Measurements,” *arXiv preprint arXiv:2407.07368*, 2025.
- [34] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*, vol. 4, Springer, 2006.
- [35] D. P. Kingma and M. Welling, “Auto-Encoding Variational Bayes,” in *2nd International Conference on Learning Representations, ICLR*, 2014.
- [36] P. Blanchard, D. J. Higham, and N. J. Higham, “Accurately computing the log-sum-exp and softmax functions,” *IMA Journal of Numerical Analysis*, vol. 41, no. 4, pp. 2311–2330, 2021.
- [37] I. Buchnik, G. Revach, D. Steger, R. J. Van Sloun, T. Rottengberg, and N. Shlezinger, “Latent-kalmannet: Learned kalman filtering for tracking from high-dimensional signals,” *IEEE Transactions on Signal Processing*, vol. 72, pp. 352–367, 2023.
- [38] X. R. Li and V. P. Jilkov, “Survey of maneuvering target tracking: Iii. measurement models,” in *Signal and Data Processing of Small Targets 2001*. SPIE, 2001, vol. 4473, pp. 423–446.
- [39] V. Garcia Satorras, Z. Akata, and M. Welling, “Combining generative and discriminative models for hybrid inference,” *Advances in NeurIPS*, vol. 32, 2019.
- [40] A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” *Advances in NeurIPS*, vol. 32, 2019.
- [41] D.P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations (ICLR)*, 2015.