AIAA SciTech Forum
6-10 January 2020, Orlando, FL
AIAA Scitech 2020 Forum

10.2514/6.2020-1869

On Neural Network Training from Noisy Data using a Novel Filtering Framework

Vedang Deshpande*, Niladri Das†, Vaishnav Tadiparthi‡and Raktim Bhattacharya§
Department of Aerospace Engineering, Texas A&M University, College Station, TX 77843

We discuss a novel method to train a neural network from noisy data, using Optimal Transport based filtering. We show a comparative study of this methodology with three other filters: the Extended Kalman filter, the Ensemble Kalman filter, and the Unscented Kalman filter, that can also be used for the purpose of training a neural network. We empirically establish that Optimal Transport based filter performs better than the other three filters with respect to root mean square error measure, for non-Gaussian noise in the output. We demonstrate the efficacy of utilizing the Optimal Transport based filtering for neural network training in the context of predicting Mackey-Glass chaotic time series data.

Introduction

In aerospace engineering, neural networks (NNs) are being extensively employed both in academia and in industry in optimally designing structural components [1], enhancing one-way fluid-structure interaction (FSI) analysis during unnamed aerial vehicle (UAV) mechanical design [2], airfoil inverse design [3], as well as improving general aircraft design by exploiting a database of airfoil geometry and aerodynamic performance [4]. A NN is a set of algorithms modeled loosely on the human brain, designed expressly for the purpose of recognizing patterns in data. This data in a broader context, can be of any form. It can be images, text, speech, or even as plainly numerical as that derived from climate observations or user clicks on social media. Along with research related to different facets of aerospace engineering, NNs have been widely studied and quite successfully applied in multiple fields of science and engineering due to their ability to deal with highly nonlinear systems.

A NN comprises of numerous layers of interconnected nodes, connected using vertices. The nodes are function blocks that transform an input signal to that node, whereas vertices have the dual utility of signal transmission to specific nodes while scaling the signals using preassigned weights (model weights). Using a given dataset, NN training seeks to create an accurate mapping of inputs to outputs parameterized by the model weights. Given that the error between desired output and NN predicted output, the error manifold is likely to be non-convex and may contain local minima or saddle points (zero gradient regions), optimizing for the model weights remains a hard problem. Furthermore, the curse of dimensionality limits the efficacy of most algorithms known today. Stochastic gradient descent is one of the most popular algorithms being used in neural network training [5]. The backpropagation algorithm, introduced in 1986 [6] is based on stochastic gradient descent and is extensively used for NN training. Although it works reasonably well for problems of small size, the learning speed could be very slow for large problems. Convergence is also poor for large NNs [7]. However, modifications inspired by nonlinear programming techniques have been proposed and proven to improve speed [8].

The effectiveness of back-propagation algorithm for NNs for modeling a system, is attenuated in presence of noise in the output. In reality, most observations taken today are contaminated by sensor noise, thereby directly limiting the performance of any chosen techniques in identifying or modeling a system, specially using a NN. Moreover, real time system modeling or identification demands training NNs from sequential data streams. This requires a NN training scheme that addresses both of these issues simultaneously under the same framework. Thus, estimating NN weights based on filtering paradigms [9] become useful when working with systems influenced unduly by noise and based on sequential data streams. The Kalman Filter can handle additive noise to the output in linear systems whereas its extensions such as the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF) perform favorably with nonlinear dynamical systems [10]. It has even been shown that backpropagation itself is a degenerate form of EKF [11].

*Graduate Student and AIAA Student Member
†Graduate Student and AIAA Student Member
‡Graduate Student and AIAA Student Member
§Associate Professor and AIAA Associate Fellow

In the context of NN, where the problem is associated with calculating unknown fixed weights, a filter is used as a *parameter estimator* [12], where the parameters are the unknown weights.

This paper investigates a novel filtering technique in the context of training a NN in presence of non-Gaussian noise in the output. The paper provides three distinct contributions. First, we present a new methodology based on Optimal Transport (OT) based filtering, for neural network training which has not been explored before. Secondly, we provide a comparative study of our proposed technique with three other filter based training techniques: EKF, UKF, and Ensemble Kalman filter (EnKF). In addition, we provide experimental results of this new technique for the problem of predicting Mackey-Glass chaotic time series data.

The paper is organized as follows. First, we introduce the problem of NN training in the language of parameter estimation using filters. In the succeeding section, we review the filters: EKF [13], UKF [12], and EnKF [14], along with the Optimal Transport based filtering, which is the focus of this work. We then discuss the data set on which the NN training will be performed. This section will also address filter parameter selection, which is extremely important in this work. Finally, we will present the results comparing the four filters for NN training.

Problem Formulation

Preliminaries on neural network configuration

The concept of artificial neural network was laid forward for the first time in the 1940s [15]. Since then, different models of neural networks [16] have been developed by various researchers. Among them, the feed-forward neural network has three distinct layers: the first layer is the input layer, the middle layer which often contains several layer in itself is known as the hidden layer, followed by the third layer which is the output layer. In fig.1 [17] the lowest level is the input layer which consists of 5 input nodes. The next two layers above it are the two hidden layers, followed by the topmost layers which is the output layers. Here output layer consists of single output node. Each of the vertex

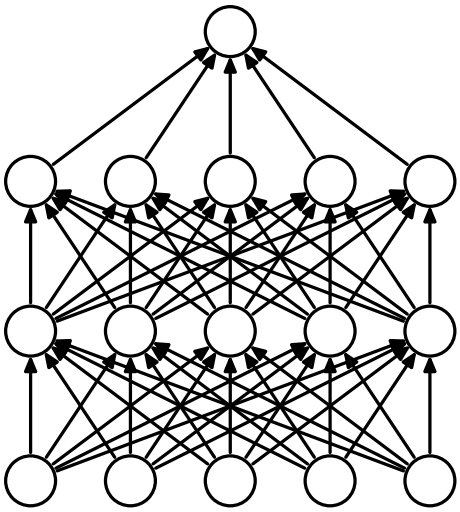

Fig. 1 Generic neural network configuration

connecting one node to another represents weights in that path. Weights between any two consecutive layers can be written as a matrix. We represent weights between layers k and l as $\boldsymbol{W}_{kl}$, where $[w_{ij}]_{kl}$ denotes the weight between i^{th} node of layer k to the j^{th} node of layer l. Each of the layers except the input layer has bias associated with each of the nodes represented as $\boldsymbol{B}_l$ for layer l. Each of the nodes except those in the input layer performs the following calculation:

$$[z_j]_l = g\Big(\sum_{p=1}^{n_k}[z_p]_k[w_{pj}]_{kl} + [b_j]_l\Big)$$

where $[z_j]_l$ is the j^{th} element of the vector $\boldsymbol{z}_l$ that represents the output of l^{th} layer and $[b_j]_l$ denotes the bias associated with the j^{th} node of l^{th} layer. The function $g(.)$ associated with each of the nodes is taken to be the tanh(.) or the hyperbolic tangent function, which is defined as:

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Filtering Model for Training a Neural Network on a Time-series Data:

We have a time-series input-output $(\boldsymbol{x}_k, \boldsymbol{y}_k)$ data. We assume a nonlinear mapping between them which we are modeling using a NN. The core problem thus involves determining a nonlinear mapping:

$$\boldsymbol{y}_k = \boldsymbol{G}(\boldsymbol{x}_k, \boldsymbol{w}) \tag{1}$$

where $\boldsymbol{x}_k$ is the input, $\boldsymbol{y}_k$ is the output, and the nonlinear map $\boldsymbol{G}$ is parameterized by the vector $\boldsymbol{w}$. The map $\boldsymbol{G}$ is modeled using a NN. Learning corresponds to estimating the parameters $\boldsymbol{w}$. Typically, a training set is provided with sample pairs consisting of known input and desired outputs, $\{\boldsymbol{x}_k, \boldsymbol{d}_k\}$. The error of the NN is defined as

$$\boldsymbol{e}_k = \boldsymbol{d}_k - \boldsymbol{G}(\boldsymbol{x}_k, \boldsymbol{w}).$$

The goal involves solving for the parameters $\boldsymbol{w}$ in order to minimize this expected squared error over a given data set of inputs and corresponding outputs. In the absence of measurement noise in the output, back propagation algorithm is typically used to train neural networks. We assume that the actual measurement is corrupted by a white Gaussian noise with known covariance. We aim to use Bayesian filtering techniques to estimate the parameters by writing a new state-space representation:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \boldsymbol{u}_k \tag{2}$$

$$\boldsymbol{y}_k = \boldsymbol{G}(\boldsymbol{x}_k, \boldsymbol{w}_k) + \boldsymbol{e}_k \tag{3}$$

where the parameters $\boldsymbol{w}_k$ correspond to a stationary process with identity state transition matrix, driven by process noise $\boldsymbol{u}_k$. The output $\boldsymbol{y}_k$ corresponds to a nonlinear observation on $\boldsymbol{w}_k$, where $\boldsymbol{x}_k$ are the input measurements. The measurement is corrupted by a noise $\boldsymbol{e}_k$. The process noise $\boldsymbol{u}_k$ may represent our uncertainty in how the parameters evolve, modeling errors or unknown inputs such such as maneuvers in tracking applications [18].

Consider the state-space estimation framework given in (2) and (3). Given the noisy observation $\boldsymbol{y}_k$, a recursive estimation for $\boldsymbol{w}_k$, which is $\hat{\boldsymbol{w}}_k$, can be expressed in the form:

$$\hat{\boldsymbol{w}}_{k+1} = \hat{\boldsymbol{w}}_k + \mathcal{K}_k(\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k), \tag{4}$$

where $\hat{\boldsymbol{y}}_k$ is the estimated output from the NN at k^{th} time step. This recursion provides the optimal minimum mean-squared error (MMSE) estimate for $\boldsymbol{w}_k$, assuming the prior estimate $\hat{\boldsymbol{w}}_{k-1}$ and the current observation $\boldsymbol{y}_k$ are Gaussian Random Variables (GRV) in the cases of EKF, UKF, and EnKF. In the succeeding section, we discuss the four filtering techniques that we use in this work.

Filtering Techniques

EKF:

The Extended Kalman Filter approximates the nonlinear measurement function by a first-order linearization, usually a Taylor expansion. The key advantage of linearization lies in its efficiency. Each update requires time $O(n_y^{2.4} + n_x^2)$ where n_y is the dimension of the measurement vector, and n_x is the dimension of the state vector [19]. It is important to note however, that the first-order approximations used by EKF are capable of introducing large errors into the true posterior and co-variance of the transformed GRV, potentially leading to sub-optimal performance or worse, divergence [12]. Nonetheless, dynamics in parameter estimation is strictly linear, as evident in (2). This permits a judicious use of the EKF in training NN, despite the non-linearity of the output function, when the measurement noise co-variance is negligible.

Consider the nonlinear state-space model presented in (2) and (3), where $\boldsymbol{u}_k$ and $\boldsymbol{e}_k$ are independent Gaussian processes with means zero, and having co-variance matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ respectively. The algorithm is presented below:

Linearize (3) using a Taylor series expansion. Define:

$$\hat{\boldsymbol{C}}_k^w = \frac{d\boldsymbol{G}_k}{d\boldsymbol{w}}\Big|_{\boldsymbol{w}=\hat{\boldsymbol{w}}_k^-}$$

Compute $\hat{\boldsymbol{C}}_k^w$ recursively.
Initialize: For $k = 0$ set:

$$\begin{aligned}\boldsymbol{w}_0^+ &= \mathbb{E}[\boldsymbol{w}_0]\\ \Sigma_{\tilde{\boldsymbol{w}},\boldsymbol{0}}^+ &= \mathbb{E}[(\boldsymbol{w}_0 - \hat{\boldsymbol{w}}_0^+)(\boldsymbol{w}_0 - \hat{\boldsymbol{w}}_0^+)^T]\end{aligned}$$

Iterate: For $k = 1, 2, \ldots$ compute:

$$\begin{aligned}&\textit{Parameter Prediction:}\\ \hat{\boldsymbol{w}}_k^- &= \hat{\boldsymbol{w}}_{k-1}^+\\ &\textit{Error covariance Prediction:}\\ \Sigma_{\tilde{\boldsymbol{w}},k}^- &= \Sigma_{\tilde{\boldsymbol{w}},k-1}^- + \boldsymbol{Q}\\ &\textit{Output Estimate:}\\ \hat{\boldsymbol{y}}_k &= \boldsymbol{G}(\boldsymbol{x}_k, \hat{\boldsymbol{w}}_k^-)\\ &\textit{Kalman Gain:}\\ \mathcal{K}_k &= \Sigma_{\tilde{\boldsymbol{w}},k}^-(\hat{\boldsymbol{C}}_k^w)^T[\hat{\boldsymbol{C}}_k^w\Sigma_{\tilde{\boldsymbol{w}},k}^-(\hat{\boldsymbol{C}}_k^w)^T + \boldsymbol{R}]\\ &\textit{Parameter Update:}\\ \hat{\boldsymbol{w}}_k^+ &= \hat{\boldsymbol{w}}_k^- + \mathcal{K}_k[\boldsymbol{y}_k - \hat{\boldsymbol{y}}_k]\\ &\textit{Error covariance Update:}\\ \Sigma_{\tilde{\boldsymbol{w}},k}^+ &= (\boldsymbol{I} - \mathcal{K}_k\hat{\boldsymbol{C}}_k^w)\Sigma_{\tilde{\boldsymbol{w}},k}^-\end{aligned}$$

UKF:

Instead of analytically linearizing the dynamics and measurement model and using the Kalman filter equations, UKF implements the unscented transform (UT) [20]. If we have a non-linear transformation $\boldsymbol{y} = \boldsymbol{G}(\boldsymbol{x}, \boldsymbol{w})$, where $\boldsymbol{w}$ is a random variable with known mean $\bar{\boldsymbol{w}}$ and co-variance $\boldsymbol{P}^w$, UT approximates the posterior mean and co-variance of the random variable $\boldsymbol{y}$. UT uses a set of carefully chosen sample points, known as sigma points. These points completely capture the true mean and co-variance of the Gaussian random variable and when propagated through the nonlinear system, capture the posterior mean and co-variance accurately up to the third order for any non-linearity [12]. To capture the mean $\hat{\boldsymbol{w}}_{k-1|k-1}^a$ of the augmented state $\boldsymbol{w}_{k-1}^a := \begin{bmatrix}\boldsymbol{w}_{k-1}\\ \boldsymbol{u}_{k-1}\end{bmatrix}$, where $\boldsymbol{w}_{k-1}^a \in \mathbb{R}^{n_a}$ and $n_a := n + q$, as well as the augmented error co-variance $\boldsymbol{P}_{k-1|k-1}^{wwa} := \begin{bmatrix}\boldsymbol{P}_{k-1|k-1}^{ww} & \boldsymbol{0}_{n\times q}\\ \boldsymbol{0}_{q\times n} & \boldsymbol{Q}_{k-1}\end{bmatrix}$, the sigma point matrix $\boldsymbol{\chi}_{k-1|k-1} \in \mathbb{R}^{n_a\times(2n_a+1)}$ is chosen as:

$$\boldsymbol{\chi}_{k-1|k-1} = \hat{\boldsymbol{w}}_{k-1|k-1}^a 1_{1\times(2n_a+1)} + \sqrt{(n_a+\lambda)} \times \left[0_{n_a\times 1}\ \left(\boldsymbol{P}_{k-1|k-1}^{wwa}\right)^{1/2} - \left(\boldsymbol{P}_{k-1|k-1}^{wwa}\right)^{1/2}\right] \tag{5}$$

with weights:

$$\gamma_0^{(m)} \triangleq \frac{\lambda}{n_a+\lambda} \tag{6}$$

$$\gamma_0^{(c)} \triangleq \frac{\lambda}{n_a+\lambda} + 1 - \alpha^2 + \beta \tag{7}$$

$$\gamma_i^{(m)} \triangleq \gamma_i^{(c)} \triangleq \gamma_{i+n_a}^{(m)} \triangleq \gamma_{i+n_a}^{(c)} \frac{1}{2(n_a+\lambda)},\ i = 1, \ldots, n_a, \tag{8}$$

where $(\cdot)^{1/2}$ is the Cholesky square root, $0 < \alpha \leq 1$, $\beta \geq 0$, $\kappa \geq 0$, and $\lambda \triangleq \alpha^2(\kappa + n_a) - n_a > -n_a$. We set $\alpha = 1$ and $\kappa = 0$ [21] such that $\lambda = 0$ [22] and set $\beta = 2$ [21]. Alternative schemes for choosing sigma points are given in [22]. The notation $\hat{\boldsymbol{w}}_{k|k-1}$ indicates an estimate of $\boldsymbol{w}_k$ at time k based on information available up to and including time $k-1$. Likewise, $\hat{\boldsymbol{w}}_{k|k}$ indicates an estimate of $\boldsymbol{w}_k$ at time k using information available up to and including time k. The UKF prediction equations are given by the ones above together with:

$$\boldsymbol{\chi}^{w}_{i,k|k-1} = f(\boldsymbol{\chi}^{w}_{i,k-1|k-1}, u_{k-1}, \boldsymbol{\chi}^{u}_{i,k-1|k-1}, k-1),\ i = 0, \ldots, 2n_a, \tag{9}$$

$$\hat{\boldsymbol{w}}_{k|k-1} = \sum_{i=0}^{2n_a} \gamma_i^{(m)} \boldsymbol{\chi}^{w}_{i,k|k-1}, \tag{10}$$

$$\boldsymbol{P}^{ww}_{k|k-1} = \sum_{i=0}^{2n_a} \gamma_i^{(c)} [\boldsymbol{\chi}^{w}_{i,k|k-1} - \hat{\boldsymbol{w}}_{k|k-1}][\boldsymbol{\chi}^{w}_{i,k|k-1} - \hat{\boldsymbol{w}}_{k|k-1}]^T, \tag{11}$$

$$\boldsymbol{\mathcal{Y}}_{i,k|k-1} = h(\boldsymbol{\chi}^{w}_{i,k-1|k-1}, k),\ i = 0, \ldots, 2n_a, \tag{12}$$

$$\hat{\boldsymbol{y}}_{k|k-1} = \sum_{i=0}^{2n_a} \gamma_i^{(m)} \boldsymbol{\mathcal{Y}}_{i,k|k-1}, \tag{13}$$

$$\boldsymbol{P}^{yy}_{k|k-1} = \sum_{i=0}^{2n_a} [\gamma_i^{(c)} [\boldsymbol{\mathcal{Y}}_{i,k|k-1} - \hat{\boldsymbol{y}}_{k|k-1}][\gamma_i^{(c)} [\boldsymbol{\mathcal{Y}}_{i,k|k-1} - \hat{\boldsymbol{y}}_{k|k-1}]^T + R_k \tag{14}$$

$$\boldsymbol{P}^{wy}_{k|k-1} = \sum_{i=0}^{2n_a} [[\boldsymbol{\chi}^{w}_{i,k|k-1} - \hat{\boldsymbol{w}}_{k|k-1}][\gamma_i^{(c)} [\boldsymbol{\mathcal{Y}}_{i,k|k-1} - \hat{\boldsymbol{y}}_{k|k-1}]^T \tag{15}$$

where $\boldsymbol{\chi}_i$ is the ith column of $\boldsymbol{\chi}$,

$$\begin{bmatrix} \boldsymbol{\chi}^{w}_{k-1|k-1} \\ \boldsymbol{\chi}^{u}_{k-1|k-1} \end{bmatrix} \triangleq \boldsymbol{\chi}_{k-1|k-1}, \tag{16}$$

$$\boldsymbol{\chi}^{w}_{k-1|k-1} \in \mathbb{R}^{n \times (2n_a+1)}, \quad \boldsymbol{\chi}^{u}_{k-1|k-1} \in \mathbb{R}^{q \times (2n_a+1)} \tag{17}$$

$\boldsymbol{P}^{ww}_{k|k-1}$ is the prediction error covariance, $\boldsymbol{P}^{yy}_{k|k-1}$ is the innovation covariance, $\boldsymbol{P}^{wy}_{k|k-1}$ is the cross covariance, and $\boldsymbol{P}^{ww}_{k|k}$ is the data-assimilation error-covariance.

The data-assimilation equations are given as:

$$\mathcal{K}_k = \boldsymbol{P}^{wy}_{k|k-1} (\boldsymbol{P}^{yy}_{k|k-1})^{-1}, \tag{18}$$

$$\hat{\boldsymbol{w}}_{k|k} = \hat{\boldsymbol{w}}_{k|k-1} + \mathcal{K}_k (\boldsymbol{y}_k - \hat{\boldsymbol{y}}_{k|k-1}), \tag{19}$$

$$\boldsymbol{P}^{ww}_{k|k} = \boldsymbol{P}^{ww}_{k|k-1} - \mathcal{K}_k \boldsymbol{P}^{yy}_{k|k-1} \mathcal{K}_k^T, \tag{20}$$

where $\mathcal{K}_k \in \mathbb{R}^{n \times m}$ is the Kalman gain matrix. Model information is used during the prediction step, while measurement data are injected into the estimates during the data-assimilation step.

EnKF:

The Ensemble Kalman Filter (EnKF) introduced by Evensen [23] represents error statistics using an ensemble of model states. It was designed to solve two major problems with EKF, the first being approximation issues involved with discarding higher order states in the Taylor series expansion, and the other being the huge computational burden associated with storage and propagation of the full error covariance matrix [14]. In EnKF, probability density functions are represented using a large cloud of points in state space, known as an ensemble. By integrating these states forward in time, it is possible to approximately estimate moments of the PDF at different time levels. Computational complexity of the EnKF is lower than that of the other extensions of the Kalman filter because the ensemble size N is typically less than n (the dimension of the state vector), thereby reducing storage costs and time spent in covariance update, which is the bottleneck of the other filters [24]. The summary of the update and the prediction steps are provided below.

$\boldsymbol{\chi}^{+}_k$ is a matrix (ensemble) with N posterior samples at time k. The dynamic update is performed in the following manner:

$$\boldsymbol{\chi}^{+}_k = [\boldsymbol{w}^{1+}_k \ \boldsymbol{w}^{2+}_k \ \ldots \ \boldsymbol{w}^{N+}_k]$$

The posterior mean from the samples is approximated as:

$$\boldsymbol{\mu}_k^+ := \mathbb{E}[\boldsymbol{w}_k^+] \approx \frac{1}{N}\boldsymbol{\chi}_k^+ 1_N$$

where $1_N \in \mathbb{R}^N$ is a column vector of N ones. We define:

$$\bar{\boldsymbol{\chi}}_k^+ = \boldsymbol{\mu}_k^+ 1^T$$

Variance from the samples is computed as:

$$\begin{aligned}\Sigma_{xx,k}^+ &= \mathbb{E}[(\boldsymbol{w}_k^+ - \boldsymbol{\mu}_k^+ \mathbf{1}_N)(\boldsymbol{w}_k^+ - \boldsymbol{\mu}_k^+ \mathbf{1}_N)^T] \\ &\approx \boldsymbol{\chi}_k^+ A \boldsymbol{\chi}_k^{+T}\end{aligned}$$

where:

$$A := \left[\frac{1}{N-1}\left(\boldsymbol{I}_N - \frac{\mathbf{1}\mathbf{1}^T}{N}\right)\left(\boldsymbol{I}_N - \frac{\mathbf{1}\mathbf{1}^T}{N}\right)\right]$$

The state of each ensemble member at the next time step is computed using (2):

$$\boldsymbol{w}_k^- = \boldsymbol{w}_{k-1}^+ + \boldsymbol{u}_{k-1}$$

The Kalman gain is computed as:

$$\mathcal{K}_k = \Sigma_{wy,k}^-(\Sigma_{yy,k}^-) + \mathcal{R}_{k-1}$$

where $\Sigma_{wy,k}^-$ is defined as:

$$\Sigma_{wy,k}^- = \frac{1}{N-1}(\boldsymbol{\chi}_k^- - \bar{\boldsymbol{\chi}}_k^-) \times (\boldsymbol{G}_k(\boldsymbol{x}_k, \boldsymbol{\chi}_k^-) - \boldsymbol{G}_k(\boldsymbol{x}_k, \bar{\boldsymbol{\chi}}_k^-))^T$$

and $\Sigma_{yy,k}^-$ is defined as:

$$\Sigma_{yy,k}^- = \frac{1}{N-1}\left\{\boldsymbol{G}_k(\boldsymbol{x}_k, \boldsymbol{\chi}_{k+1}^-) - \boldsymbol{G}_{k+1}(\boldsymbol{x}_k, \bar{\boldsymbol{\chi}}_k^-)) \times (\boldsymbol{G}_k(\boldsymbol{x}_k, \boldsymbol{\chi}_k^-) - \boldsymbol{G}_{k+1}(\boldsymbol{x}_k, \bar{\boldsymbol{\chi}}_k^-))^T\right\}$$

The measurement update therefore, is formulated as:

$$\boldsymbol{w}_k^{i+} = \boldsymbol{w}_k^{i+} + \mathcal{K}_k(y_k - \boldsymbol{G}_k(\boldsymbol{w}_k^{i-}, \boldsymbol{x}_k) + \boldsymbol{\epsilon}_k^i)$$

The covariance update is:

$$\Sigma_{xx,k}^+ = \Sigma_{xx,k}^- - \Sigma_{xy,k}^-(\Sigma_{yy,k}^- + \mathcal{R}_k)^{-1} \times \Sigma_{xy,k}^{-T}$$

OT-based Filter:

The theory of Optimal Transport (OT) introduced originally by Monge [25] in 1781 essentially involves finding a mapping between two probability density functions that minimizes a chosen cost function [26]. Filtering techniques based on the theory of optimal transport are also sample-based, as in the EnKF and UKF. The primary difference between these algorithms however, lies in the manner of updating priors to posteriors using observations [27].

In [28], it was demonstrated that the performance of the OT-based filtering algorithm in space situational awareness problems was more accurate, consistent, and robust than that observed with the EnKF. The distribution agnostic behavior of OT lends it favorability when compared to algorithms that assume system uncertainty to be in $\mathbb{R}^n$.

Given the measurement model and assuming the prior distribution $p(\boldsymbol{w}^-)$ for the state variable $\boldsymbol{w}$, we can compute its posterior distribution using Bayes' theorem as

$$p(\boldsymbol{w}^+) \propto g(\boldsymbol{y}|\boldsymbol{w}^-)p(\boldsymbol{w}^-), \tag{21}$$

where g is the distribution of the measurement error ϵ, also known as the likelihood function. Our goal is to generate N samples $\boldsymbol{\chi}^+ = [\boldsymbol{x}_1^+, \boldsymbol{x}_2^+, ..., \boldsymbol{x}_N^+] \in \mathbb{R}^{n\times N}$ from the posterior distribution, given N prior samples $\boldsymbol{\chi}^- = [\boldsymbol{x}_1^-, \boldsymbol{x}_2^-, ..., \boldsymbol{x}_N^-] \in \mathbb{R}^{n\times N}$, with $\boldsymbol{w}_i \in \mathbb{R}^n$. In the OT framework, this is done using a transformation

$$\boldsymbol{w}^+ = \phi(\boldsymbol{w}^-),$$

where $\phi(\cdot)$ is obtained from optimization with respect to a suitably defined cost function. The optimization must also constrain $\phi(\cdot)$ to be measure preserving and monotone. This is an infinite dimensional problem. For computational tractability, the map $\phi(\cdot)$ is often parameterized in finite dimensional space.

Let the weighted samples of the posterior be $\hat{\boldsymbol{\chi}}^+ \in \mathbb{R}^{n\times N}$, where $\hat{\boldsymbol{\chi}}^+ = \boldsymbol{\chi}^-$, i.e. the sample locations are the same, but each sample in $\hat{\boldsymbol{\chi}}^+$ has weight given by

$$\lambda_i = \frac{g(\boldsymbol{y}|\boldsymbol{w}_i^-)}{\sum_{i=1}^N g(\boldsymbol{y}|\boldsymbol{w}_i^-)}. \tag{22}$$

We seek a coupling $\boldsymbol{A}$ between the two random variables $\boldsymbol{\chi}^-$ and $\hat{\boldsymbol{\chi}}^+$. But the matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ is not unique. A unique matrix is determined by optimizing some specific cost function. Here, we maximize the correlation between $\boldsymbol{\chi}^-$ and $\hat{\boldsymbol{\chi}}^+$, which is equivalent to minimization of the expected distance between $\boldsymbol{w}^-$ and $\hat{\boldsymbol{w}}^+$, i.e.

$$\min_{\boldsymbol{A}} \sum_{i=1}^N \sum_{j=1}^N A_{ij} D(\boldsymbol{w}_i^-, \hat{\boldsymbol{w}}_j^+),$$

where A_{ij} are elements of $\boldsymbol{A}$, and

$$D(\boldsymbol{w}_i^-, \hat{\boldsymbol{w}}_j^+) = D(\boldsymbol{w}_i^-, \boldsymbol{w}_j^-) = \|\boldsymbol{w}_i^- - \boldsymbol{w}_j^-\|_2, \tag{23}$$

We normalize all the state variables before calculating the distance $D(.)$ between each of the samples. Normalization is done using the diameter of the sample set.

The map $\boldsymbol{A}$ must also be measure preserving, which is enforced using the following constraints

$$\sum_{i=1}^N A_{ij} = 1/N, \ \sum_{j=1}^N A_{ij} = \lambda_i, \text{ and } \ A_{ij} \geq 0; \tag{24}$$

where $\lambda_i \propto g(\boldsymbol{y}|\boldsymbol{x}_i^-)$. Therefore, the optimal map can be obtained by solving the following linear programming problem with problem size N^2,

$$\boldsymbol{A}^* = \underset{\boldsymbol{A}}{\operatorname{argmin}} \sum_{i=1}^N \sum_{j=1}^N A_{ij} D(\boldsymbol{w}_i^-, \boldsymbol{w}_j^+) \tag{25}$$

subject to :

$$\sum_{i=1}^N A_{ij} = 1/N,$$

$$\sum_{j=1}^N A_{ij} = \lambda_i,$$

$$A_{ij} \geq 0.$$

We can think of (25) as a network flow problem where probability masses of $1/N$ are optimally transported from sites $\boldsymbol{w}_i^-$ to sites $\boldsymbol{w}_j^-$ so that $\boldsymbol{w}_j^-$ have probability mass λ_j.

Once we solve (25) and obtain $\boldsymbol{A}^*$, we define a Markov chain on $\hat{\boldsymbol{\chi}}^+$ (or $\boldsymbol{\chi}^-$) with a left stochastic matrix $\boldsymbol{P} := N\boldsymbol{A}^*$, such that equally weighted samples $\boldsymbol{w}_i^+$, representing the posterior distribution, is obtained using

$$\boldsymbol{w}_i^+ = \sum_{j=1}^N P_{ij} \boldsymbol{w}_j^-, \tag{26}$$

where P_{ij} are the elements of $\boldsymbol{P}$.

Experimental Results

Our main objective was to experimentally analyze the performance of training a neural network using an OT-based filter by comparing it with three other filtering techniques over different kinds of measurement noise. For the first set of experiments, we chose the measurement noise to be i.i.d. Gaussian with zero mean and 0.005 as the co-variance. Fig.(2) shows the plots comparing all the training techniques for a Gaussian additive noise. We can observe from the RMSE plots that both UKF and EKF obtain good accuracy in the output when compared to OT filter and EnKF filters. We notice that the RMSE performance does not change appreciably after 13 epochs for EKF and UKF. Among the 4, our proposed OT filter performs the worst, followed by EnKF. Since both OT and EnKF are sample based, increasing the sample size might translate to improvement in RMSE performance. Moreover, EnKF and OT filters also require a sizable number of epochs to perform comparably with EKF or UKF.

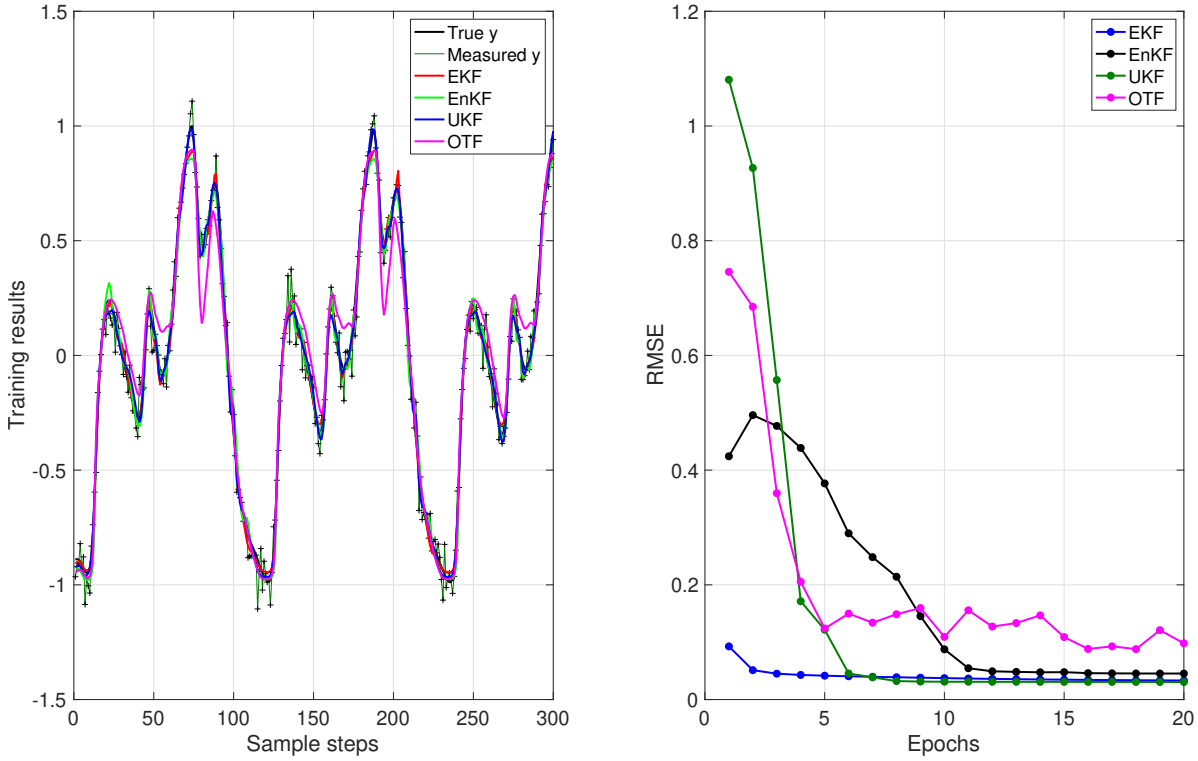

Fig. 2 Training performance comparison for Gaussian additive noise

In the next set of experiments, we use non-Gaussian noise measurement noise instead of Gaussian noise. We choose a *Bimodal* Gaussian noise with varying mean positions and varying individual co-variances. They are:

1) Mean: $[-0.1, 0.1]$, Standard Deviation : $[0.2, 0.2]$, and Weights: $[0.4, 0.6]$
2) Mean: $[-0.1, 0.1]$, Standard Deviation : $[1.0, 1.0]$, and Weights: $[0.4, 0.6]$
3) Mean: $[-0.2, 0.2]$, Standard Deviation : $[2.0, 2.0]$, and Weights: $[0.3, 0.7]$

Fig. (3) shows the performance of OT filter compared to the others for the first set of bi-modal Gaussian distribution parameters. For a bi-modal Gaussian noise we see OT filter and EKF performing better compared to others. UKF's performance is comparable to that of EKF, but we can see that OT filter visibly outperforms EKF. The training results show the prediction of OT filter and EKF filter compared with the real output and the noise corrupted output. Note that in presence of considerable non-Gaussian measurement noise, OT based training is capable or faithfully recovering the true output from the trained NN.

In the next set of results for non-Gaussian measurement noise, with parameters: Mean: $[-0.1, 0.1]$, Standard Deviation : $[1.0, 1.0]$, Weights: $[0.4, 0.6]$, we see an increase in RMSE error denoting a decline in NN training in fig.(4). This is expected, owing to the increase in the co-variance of the noise components. We notice from the RMSE plot that EKF and OT filters are still capable of faithfully following the true outputs. Rest of the techniques have high RMSE error. In fig.(5) we show plots comparing all the fours techniques with further increase in the component noise co-variance. The mean position of each of the component Gaussian distribution is also moved away from each other. The

corresponding parameters are: Mean: $[-0.2, 0.2]$, Standard Deviation : $[2.0, 2.0]$, and Weights: $[0.3, 0.7]$. We notice that for this set for bi-modal parameters, the EKF performance degrades considerably and starts diverging, whereas OT filter can still faithfully recover the true output signal with RMSE error much lower than the rest of the techniques. The EnKF filter after 17 epochs shows convergence in the samples, thus rendering zero predicted measurement co-variance. The Kalman gain calculation in EnKF involved inverse of this measurement co-variance matrix, which is zero at the 18th epoch. This is the reason we do not have EnKF predictions from 18th epoch as shown in the RMSE plot of fig.(5).

The input and true output for the discretized Mackey-Glass equation forms a repeating pattern. If we are to show the testing results for our trained neural network, it will be identical to the results for the training data. Hence, only training results are shown in this work and are used to evaluate the performance of our proposed OT filter based training against three other methodologies.

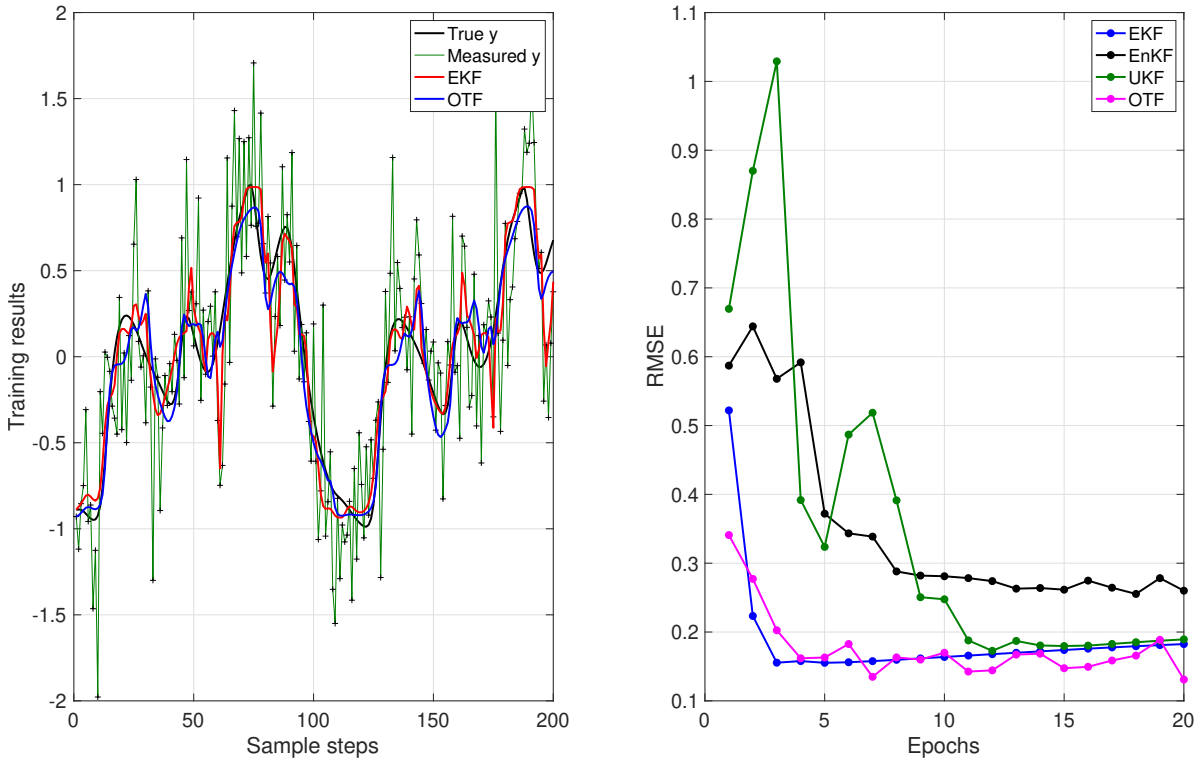

Fig. 3 Results for bi-modal Gaussian with Mean: $[-0.1, 0.1]$**, Standard Deviation :** $[0.2, 0.2]$ **, and Weights:** $[0.4, 0.6]$

Conclusion

In this work, an input vector of a chaotic signal of a given length is used in a neural network whose weights and biases are trained using filtering techniques to accurately predict given output observations. The training data contains outputs that have been corrupted with measurement noise. The filtering dynamics established how the weights and biases needs to be updated, fusing the information on the measurement noise. We developed a novel technique based on Optimal transport based filter to train a neural network in presence of considerable non-Gaussian noise in the output. The output predictions and the accuracy are empirically established using the Mackey-Glass chaotic time series data. We show that for Gaussian noise, EKF and UKF shows the best and comparable RMSE performance. EKF based training will be a preferred choice for real time training since it is considerably faster than UKF, although UKF gives the lowest RMSE error. Using OT filter will be an overkill in this situation, since it requires considerable computational time compared to EKF and UKF and might need more samples and epochs to give RMSE error comparable to that of EKF or UKF. The effectiveness of OT filter training is prominent when the measurement noise is non-Gaussian. With increase in non-Gaussianity we see a degrading performance in EKF. OT filter performs significantly better than the rest of the three techniques for all of the three non-Gaussian noise results presented in this paper. The OT filter based

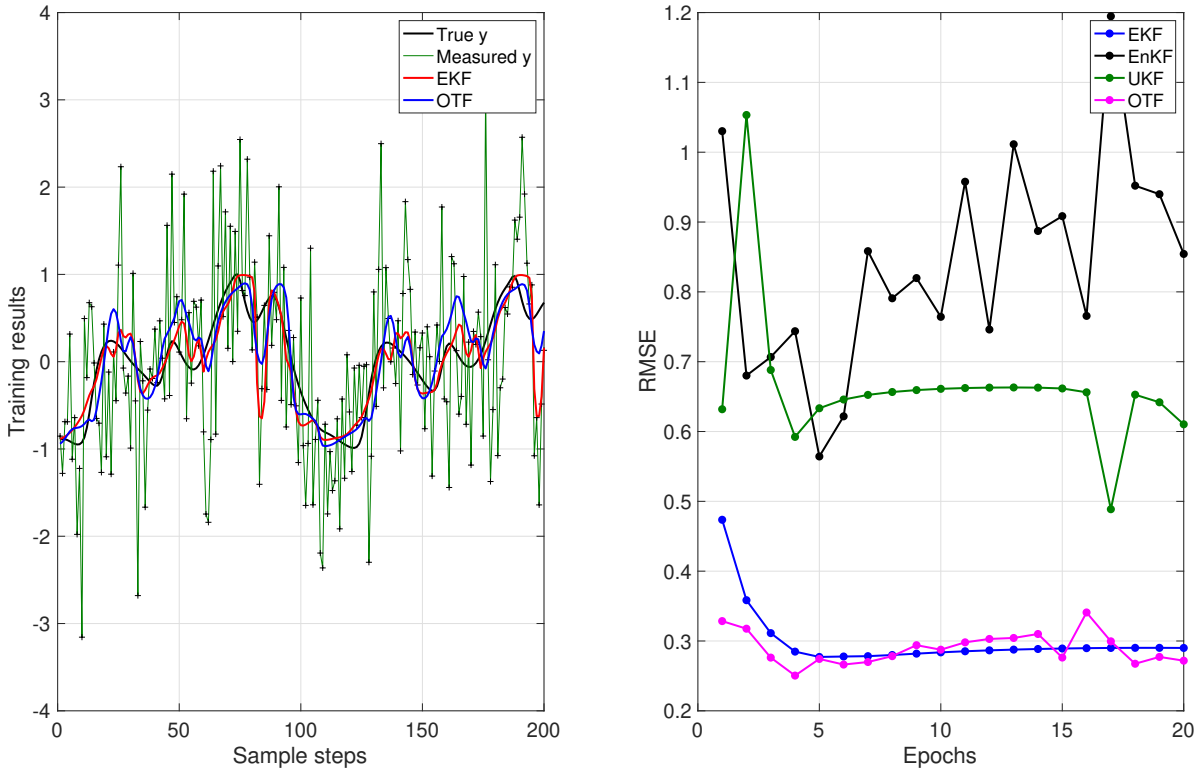

Fig. 4 Results for bi-modal Gaussian with Mean: $[-0.1, 0.1]$**, Standard Deviation :** $[1.0, 1.0]$ **, Weights:** $[0.4, 0.6]$

training takes large amounts of time depending on the choice of sample size and epoch number. In that respect, EKF based training might be favored for real-time training at the cost of increase in RMSE error. However, note that OT filter can still be used for real-time training if EKF is first used to get a good initial guess of the initial PDF, thus ensuring fast convergence of the OT filter based training. Augmenting EKF with OT filter for improving the NN training performance with reduction in run-time is a topic that needs further investigation.

Acknowledgments

The authors are all supported by NSF grant 1762825.

References

[1] Berke, L., Patnaik, S., and Murthy, P., "Optimum design of aerospace structural components using neural networks," *Computers & Structures*, Vol. 48, No. 6, 1993, pp. 1001–1010.

[2] Mazhar, F., Khan, A. M., Chaudhry, I. A., and Ahsan, M., "On using neural networks in UAV structural design for CFD data fitting and classification," *Aerospace Science and Technology*, Vol. 30, No. 1, 2013, pp. 210–225.

[3] Sun, G., Sun, Y., and Wang, S., "Artificial neural network based inverse design: Airfoils and wings," *Aerospace Science and Technology*, Vol. 42, 2015, pp. 415–428.

[4] Wang, S., Sun, G., Chen, W., and Zhong, Y., "Database self-expansion based on artificial neural network: An approach in aircraft design," *Aerospace Science and Technology*, Vol. 72, 2018, pp. 77–83.

[5] Sum, J., Leung, C.-S., Young, G. H., and Kan, W.-K., "On the Kalman filtering method in neural network training and pruning," *IEEE Transactions on Neural Networks*, Vol. 10, No. 1, 1999, pp. 161–166.

[6] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning internal representations by error propagation," Tech. rep., California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

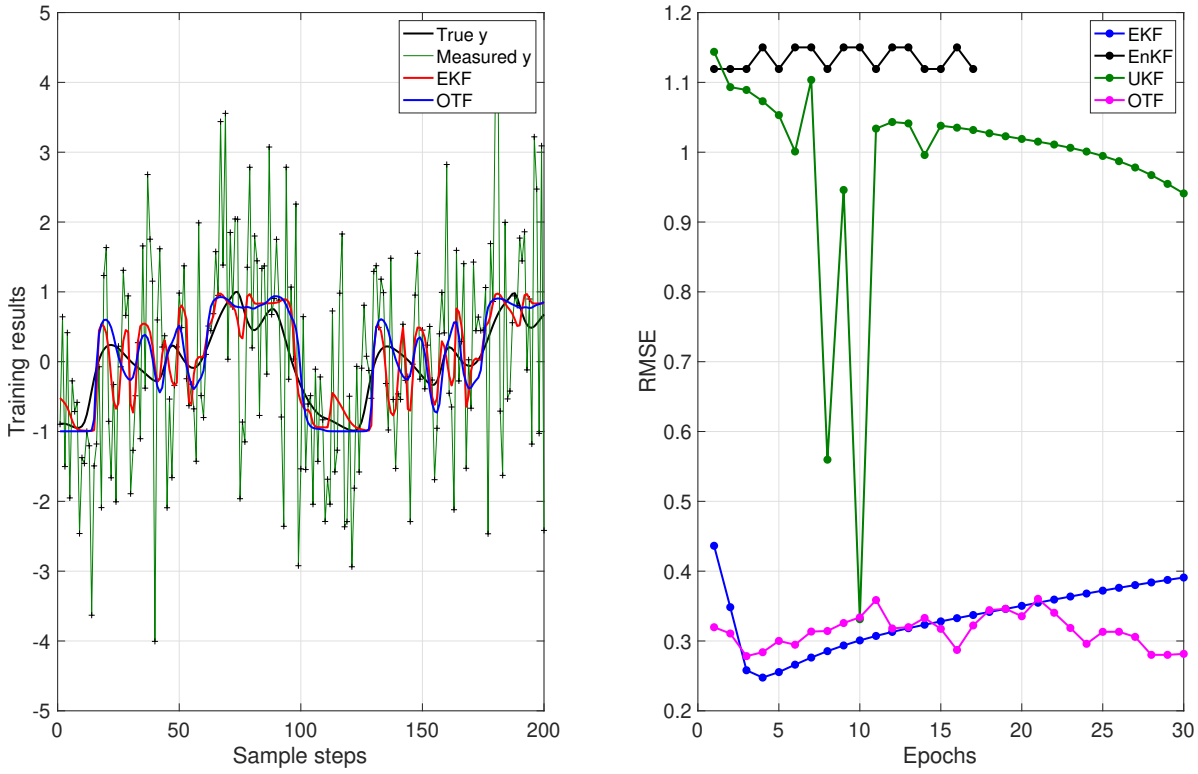

Fig. 5 Results for bi-modal Gaussian with Mean: $[-0.2, 0.2]$**, Standard Deviation :** $[2.0, 2.0]$ **, Weights:** $[0.3, 0.7]$

[7] Singhal, S., and Wu, L., "Training multilayer perceptrons with the extended Kalman algorithm," *Advances in neural information processing systems*, 1989, pp. 133–140.

[8] Haykin, S., *Neural networks: a comprehensive foundation*, Prentice Hall PTR, 1994.

[9] Anderson, B. D., and Moore, J. B., *Optimal filtering*, Courier Corporation, 2012.

[10] Wu, X., and Wang, Y., "Extended and Unscented Kalman filtering based feedforward neural networks for time series prediction," *Applied Mathematical Modelling*, Vol. 36, No. 3, 2012, pp. 1123–1131.

[11] Ruck, D. W., Rogers, S. K., Kabrisky, M., Maybeck, P. S., and Oxley, M. E., "Comparative analysis of backpropagation and the extended Kalman filter for training multilayer perceptrons," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, , No. 6, 1992, pp. 686–691.

[12] Wan, E. A., and Van Der Merwe, R., "The unscented Kalman filter for nonlinear estimation," *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*, Ieee, 2000, pp. 153–158.

[13] Sorenson, H. W., *Kalman filtering: theory and application*, IEEE, 1985.

[14] Evensen, G., "The ensemble Kalman filter: Theoretical formulation and practical implementation," *Ocean dynamics*, Vol. 53, No. 4, 2003, pp. 343–367.

[15] McCulloch, W. S., and Pitts, W., "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, Vol. 5, No. 4, 1943, pp. 115–133.

[16] Zurada, J. M., *Introduction to artificial neural systems*, Vol. 8, West publishing company St. Paul, 1992.

[17] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, Vol. 15, No. 1, 2014, pp. 1929–1958.

[18] Freitas, J. d., Niranjan, M., Gee, A. H., and Doucet, A., "Sequential Monte Carlo methods to train neural network models," *Neural computation*, Vol. 12, No. 4, 2000, pp. 955–993.

[19] Thrun, S., Burgard, W., and Fox, D., *Probabilistic robotics*, MIT press, 2005.

[20] Julier, S., Uhlmann, J., and Durrant-Whyte, H. F., "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on automatic control*, Vol. 45, No. 3, 2000, pp. 477–482.

[21] Haykin, S., *Kalman filtering and neural networks*, Vol. 47, John Wiley & Sons, 2004.

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

[23] Evensen, G., "Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics," *Journal of Geophysical Research: Oceans*, Vol. 99, No. C5, 1994, pp. 10143–10162.

[24] Roth, M., Hendeby, G., Fritsche, C., and Gustafsson, F., "The Ensemble Kalman filter: a signal processing perspective," *EURASIP Journal on Advances in Signal Processing*, Vol. 2017, No. 1, 2017, p. 56.

[25] Monge, G., "Mémoire sur la théorie des déblais et des remblais," *Histoire de l'Académie royale des sciences de Paris*, 1781.

[26] Santambrogio, F., "Optimal transport for applied mathematicians," *Birkäuser, NY*, Vol. 55, 2015, pp. 58–63.

[27] Das, N., Deshpande, V., and Bhattacharya, R., "Optimal-Transport-Based Tracking of Space Objects Using Range Data from a Single Ranging Station," *Journal of Guidance, Control, and Dynamics*, 2019, pp. 1–13.

[28] Das, N., Ghosh, R. P., Guha, N., Bhattacharya, R., and Mallick, B., "Optimal Transport Based Tracking of Space Objects in Cylindrical Manifolds," *preprint*, 2018.