12th IFAC International Symposium on
Advanced Control of Chemical Processes
July 14-17, 2024. Toronto, Canada

Approximation Of Constrained Sample-based Filters Using Neural Networks

Abhilash Dev, Sharad Bhartiya, Sachin Patwardhan

Department of Chemical Engineering, IIT Bombay, Mumbai-400076, India
(abhilash.cdev@iitb.ac.in, bhartiya@che.iitb.ac.in, sachinp@iitb.ac.in)

Abstract: For a non-linear system, sample-based state estimators have proven to be a better alternative to their derivative-based counterparts, due to their superior ability to obtain transformed probability densities. However, the computation relating to propagation of the numerous samples make these methods relatively expensive. In this work, a novel ML-based state estimation method is proposed to reduce computation time for a specific form of sample-based state estimator namely, constrained unscented transform based filters. The key feature of the proposed approach is that a trained neural network (NN) model is used as a map between initial posterior samples (or sigma points), manipulated inputs and new measurements to obtain filtered posterior sigma points. The efficacy of the proposed approach is demonstrated on a non-isothermal continuous stirred tank reactor and the Williams Otto reactor. It is shown that the proposed approach outperforms unscented transform-based state estimator in terms of computation time, while matching its performance.

Keywords: Neural networks, Estimation and filtering, Unscented Kalman filter

1. INTRODUCTION

State estimation is an essential element of model-based methods, commonly employed in monitoring, detection & diagnosis of faults and control of dynamic systems. In state estimation, model predictions of the system dynamics are merged with a datastream of available real-time measurements in a Bayesian framework to obtain an estimate of the plant. The filtered state estimates must satisfy two requirements: 1) the estimates must accurately describe the system behaviour, which is invariably non-linear, while complying with the system constraints, and 2) all computations related to the state estimation algorithm must be completed in a small fraction of the sampling period. Satisfying both of the above requires dynamic models and fast computations. An opportunity to solve this trade-off is available due to the recent advances in hardware and computational techniques involving AI/ML (Venkatasubramanian, 2019).

Bayesian methods for state estimation are predicated on the time propagation of probability density of associated states via the model and subsequent incorporation of real-time measurements to assimilate the effect of uncertainties. These methods can be broadly divided into derivative-based and sample-based estimation. Derivative-based filters like Extended Kalman Filter (EKF) use a first-order approximation of the nonlinear dynamic model and assume additive uncertainties to propagate the state densities in time. This approach endows EKF with computational efficiency, and have field deployability, but makes the state estimates inaccurate for significantly nonlinear systems. On the other hand sample-based filters, such as particle filters (PF) (Patwardhan et al., 2012). or Unscented Kalman filter (UKF) (Julier and Uhlmann, 2004), propagate samples (or realizations) of the probability density to reconstruct the propagated density. Using a large number of samples, such as in PF, affords an accurate estimate of the propagated density (Gelb, 2006) (Patwardhan et al., 2012). However, the computational costs make such methods impractical for certain systems. UKF solves the trade-off via deterministic sampling. It can be shown that the moments of Gaussian density can be captured by a small set of deterministic samples. While the small set of samples relieves the issue of computational complexity for small-sized problems, its implementation for medium and large-scale problems is still a bottleneck. Moreover, the inability of UKF/PF to be implemented in a real-time scenario becomes dramatically pronounced in the presence of constraints, as it involves repeated solutions of constrained optimization problems in real time.

The immense popularity and success of AI/ML methods have led to use of NN in systems engineering to offset the computational burden associated with using nonlinear dynamic models. Different modes of integration of AI/ML models with state estimators have been explored in the literature. The literature can be broadly classified based on roles of NN in state estimation algorithm into: 1) NN used to augment the state estimation algorithm (Sieberg et al., 2022; Revach et al., 2022; Jouaber et al., 2021) 2) NN used to obtain modelling and estimation errors (Wang et al., 2021; Vaidehi et al., 2001; Yun and Zanetti, 2021; Talla and Peroutka, 2011). In particular (Yun and Zanetti, 2021) have approximated the measurement update step of UKF using an NN. The approach, however, uses the mechanistic model for the prediction step which is similar to the conventional UKF. The prediction step in UKF is computationally intensive and can pose difficulties in the real-time implementation of UKF for a system with fast dynamics.

While existing literature often incorporates AI/ML methods for partial roles in model representation or measurement integration, notable gaps exist,

(1) reports of a complete replacement of the state estimation algorithm, including state propagation and measurement update, do not exist,

(2) use of NN to approximate constrained state estimator has not been reported.

In this work, an ML-based approach is proposed for non-linear state estimation aimed at enhancing state estimation speed while performing at par with sample-based estimators, namely UKF, in both constrained and unconstrained settings. Unlike the reviewed literature, wherein only a partial aspect of the filter implementation was replaced by an NN, a single NN replaces both the propagation and updation steps in the proposed approach. The efficacy of the proposed approach is demonstrated on a non-isothermal continuous stirred tank reactor (CSTR) and Williams Otto reactor (WOR). The rest of the paper is organized as follows: Section 2 reviews typical steps in a sample-based estimator algorithm; Section 3 presents the proposed approach; Section 4 presents two comparative case studies between sample-based state estimators and their NN approximations for a CSTR (2 states) and WOR (6 states).

2. PRELIMINARIES

2.1 Dynamic model and assumptions

Consider a plant whose model is represented by a non-linear discrete state space model Eq. (1).

$$\mathbf{x}_{k+1} = \mathbf{F}(\mathbf{x}_k, \mathbf{u}_k, \bar{\mathbf{d}} + \mathbf{w}_k) \tag{1}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \tag{2}$$

where $\mathbf{x}_k \in \mathbb{R}^{n_x}$ represents the state vector, $\mathbf{u}_k \in \mathbb{R}^{n_u}$ is the known piecewise continuous input to the system, $\mathbf{y}_k \in \mathbb{R}^{n_y}$ represents the measured states, $\bar{\mathbf{d}} \in \mathbb{R}^{n_w}$ represents the mean value of unmeasured disturbance, $\mathbf{w}_k \in \mathbb{R}^{n_w}$ represents a zero mean gaussian white noise with covariance $\mathbf{Q}$, $\mathbf{v}_k \in \mathbb{R}^{n_y}$ represents a zero mean gaussian white noise to the measurements with covariance $\mathbf{R}$. The sampling interval is T such that $t_k = kT$ where k represents the sampling instant. $\mathbf{F}$: $\mathbb{R}^{n_x \times n_u \times n_w} \rightarrow \mathbb{R}^{n_x}$ represents the non-linear dynamic system. $\mathbf{h}$: $\mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_y}$ represents the measurement model.

2.2 Constrained Recursive Sample Based Bayesian Estimator

Consider a sample-based constrained state estimator. The typical steps involved in a constrained recursive sample-based Bayesian estimator for a non-linear process are as follows (Vachhani et al., 2006; Kadu et al., 2010; Kottakki et al., 2016).

(1) **Constrained sample generation**:
Samples of states and noise are generated from their respective probability distribution. The process of sampling can be deterministic or non-deterministic. Let $\Omega[.]$ represent the sampling operator. Given the initial posterior for the system states $p(\mathbf{x}_k|\mathbf{y}_1,\ldots,\mathbf{y}_k)$ and densities for state noise and measurement noise namely $p(\mathbf{w}_k)$, $p(\mathbf{v}_k)$ respectively, obtain n samples of random vectors $\mathbf{x}_{k|k}^{(i)}$, $\mathbf{w}_k^{(i)}$, $\mathbf{v}_k^{(i)}$ for $i = 1,\ldots,n$ using Eq.(3,4,5) such that constraints as represented by Eq. (6,7) are satisfied are as follows, where $\mathbf{x}_L, \mathbf{x}_U$ represent the lower and upper bounds on states, while $\mathbf{g}(.)$ represent the non-linear constraints,

$$\mathbf{x}_{k|k}^{(i)} = \Omega[p(\mathbf{x}_k|\mathbf{y}_1,\ldots,\mathbf{y}_k)] \tag{3}$$

$$\mathbf{w}_k^{(i)} = \Omega[p(\mathbf{w}_k)] \tag{4}$$

$$\mathbf{v}_k^{(i)} = \Omega[p(\mathbf{v}_k)] \tag{5}$$

$$\mathbf{x}_L \leq \mathbf{x}_{k|k}^{(i)} \leq \mathbf{x}_U \tag{6}$$

$$\mathbf{g}(\mathbf{x}_{k|k}^{(i)}) \leq 0 \tag{7}$$

The sample mean $\hat{\mathbf{x}}_{k|k}$ and sample covariance $\mathbf{P}_{k|k}$ are calculated using Eq. (8,9) as follows,

$$\hat{\mathbf{x}}_{k|k} = \sum_{i=1}^{n_x} \omega^{(i)} \mathbf{x}_{k|k}^{(i)} \tag{8}$$

$$\mathbf{P}_{k|k} = \sum_{i=1}^{2n_x+1} \omega^{(i)} [\mathbf{x}_{k|k}^{(i)} - \hat{\mathbf{x}}_{k|k}][\mathbf{x}_{k|k}^{(i)} - \hat{\mathbf{x}}_{k|k}]^T \tag{9}$$

where, $\mathbf{x}_{k|k}^{(i)}$ is associated with weight $\omega^{(i)}$ such that $\sum_{i=1}^{n} \omega^{(i)} = 1$. $\mathscr{W}$ denotes a vector of weights $\mathscr{W} = [\omega^{(1)}, \omega^{(2)}, ..., \omega^{(n)}]$.

(2) **Sample Propagation**: The samples generated in Step 1 are propagated through the non-linear process $\mathbf{F}(.)$ (Eq.1), to obtain $\mathbf{x}_{k+1|k}^{(i)}$ i=1,2,..n using Eq.(10),

$$\mathbf{x}_{k+1|k}^{(i)} = \mathbf{F}[\mathbf{x}_{k|k}^{(i)}, \mathbf{u}_k, \bar{\mathbf{d}} + \mathbf{w}_k^{(i)}] \tag{10}$$

The propagated mean $\hat{\mathbf{x}}_{k+1|k}$ and covariance $\mathbf{P}_{k+1|k}$ are calculated using similar procedure as described in Eq. (8,9). Let $\mathbf{X}_{k|k}$ be defined as follows,

$$\mathbf{X}_{k|k} = [\mathbf{x}_{k|k}^{(1)^T} \mathbf{x}_{k|k}^{(2)^T} ... \mathbf{x}_{k|k}^{(n)^T}]^T$$

(3) **Constrained Sample update**: The samples are updated using the new measurement $\mathbf{y}_{k+1}$ and some appropriate operator $\Upsilon[.]$.

$$\mathbf{x}_{k+1|k+1}^{(i)} = \Upsilon[\mathbf{x}_{k+1|k}^{(i)}, \mathbf{P}_{k+1|k}, \mathbf{y}_{k+1}, \mathbf{v}_{k+1}^{(i)}, \mathbf{R}, \mathbf{g}(.), \mathbf{x}_L, \mathbf{x}_U] \tag{11}$$

Finally, the filtered estimate $\hat{\mathbf{x}}_{k+1|k+1}$ and its covariance $\mathbf{P}_{k+1|k+1}$ are obtained using Eq. (12,13),

$$\hat{\mathbf{x}}_{k+1|k+1} = H[\mathbf{X}_{k+1|k+1}, \mathscr{W}] \tag{12}$$

$$\mathbf{P}_{k+1|k+1} = \Pi[\mathbf{X}_{k+1|k+1}, \mathbf{P}_{k|k+1}, \mathbf{R}, \mathscr{W}] \tag{13}$$

where $H[.]$ and $\Pi[.]$ are appropriate operators.

In case of UKF Eq. (3,4,5) represent unscented transform, wherein the mean $\hat{\mathbf{x}}_{k|k}$ and covariance $\mathbf{P}_{k|k}$ are represented by the $n = 2n_x + 1$ samples (or sigma points) as follows.

$$\mathbf{x}_{k|k}^{(1)} = \hat{\mathbf{x}}_{k|k} \tag{14}$$

$$\omega_1 = \frac{\kappa}{n_x + \kappa} \tag{15}$$

$$\mathbf{x}_{k|k}^{(i)} = \hat{\mathbf{x}}_{k|k} + \sqrt{(n_x + \kappa)\mathbf{P}_{k|k}\xi_i} \tag{16}$$

$$\omega_i = \frac{1}{2(n_x + \kappa)} \tag{17}$$

$$\mathbf{x}_{k|k}^{(i+N_x)} = \hat{\mathbf{x}}_{k|k} - \sqrt{(n_x + \kappa)\mathbf{P}_{k|k}\xi_i} \tag{18}$$

$$\omega_{i+n_x} = \frac{1}{2(n_x + \kappa)} \tag{19}$$

here, ξ_i represents a unit vector with all elements except i^{th} component as zeroes.

Remark. *If constraints Eq. (6,7) in Eq. (3 - 13) are omitted, the estimation corresponds to an unconstrained version.*

The following section presents the neural network approximation of the unscented transform based state estimator.

3. MAIN CONTRIBUTION: NN APPROXIMATION OF UNSCENTED TRANSFORM BASED FILTERS

The mainstay of the proposed approach is use of NN to predict the posterior density samples $\mathbf{x}_{k+1|k+1}^{(i)}$ (Eq.11) using the initial posterior density samples $\mathbf{x}_{k|k}^{(i)}$ (Eq.3), measurement $\mathbf{y}_{k+1}$ and input $\mathbf{u}_k$ as the input to the network in a single step, thereby replacing the two-step procedure, namely propagation (Step 2) and update (Step 3) as noted in the previous section. In this work, it is considered that the input-output data for training of the NN is obtained by implementation of UKF. The input-output structure of UKF can be represented as in Fig. 1. Thus, the proposed NN is a direct map between UKF inputs $\mathscr{X}_k = [\mathbf{X}_{k|k}{}^T \, \mathbf{u}_k{}^T \, \mathbf{y}_{k+1}{}^T]^T \in \mathbb{R}^{n\times n_x+n_u+n_v}$ and the samples corresponding to UKF posterior at sample $k+1$, $\mathbf{X}_{k+1|k+1} \in \mathbb{R}^{n\times n_x}$. Note that while the posterior mean and covariance are obtained directly from UKF, it is proposed to train the NN to predict the posterior samples that are consistent with UKF posterior mean and covariance.

Consider a feedforward NN consisting of p hidden layers and an output layer, with corresponding weight matrices $\mathbf{W}^{(i)}$ and bias vectors $\mathbf{b}^{(i)}, i = 1,\cdots,p+1$. Let the intermediate layer outputs and associated activation function be denoted by $\eta^{(i)}$ and $\rho^{(i)}$ respectively. Given the NN inputs $\mathscr{X}_k$ and NN model with parameter $\theta = \{\mathbf{W}^{(i)}, \mathbf{b}^{(i)} : i = 1,2..p+1\}$ the NN output consists of the set of the propagated samples and is denoted as $\tilde{\mathbf{X}}_{k+1|k+1} = \mathscr{NN}(\mathscr{X}_k, \theta)$ and is calculated as,

$$\begin{aligned}
\eta^{(1)} &= \rho^{(1)}(\mathbf{W}^{(1)}\mathscr{X}_k + \mathbf{b}^{(1)}) \\
\eta^{(2)} &= \rho^{(2)}(\mathbf{W}^{(2)}\eta^{(1)} + \mathbf{b}^{(2)}) \\
&\cdots \\
\eta^{(p)} &= \rho^{(p)}(\mathbf{W}^{(p)}\eta^{(p-1)} + \mathbf{b}^{(p)}) \\
\tilde{\mathbf{X}}_{k+1|k+1} &= \rho^{(p+1)}(\mathbf{W}^{(p+1)}\eta^{(p)} + \mathbf{b}^{(p+1)})
\end{aligned} \tag{20}$$

The NN approximation error with respect to UKF targets can be quantified by $\mathbf{r}_{k+1}$ (Eq.21),

$$\mathbf{r}_{k+1} = \mathbf{X}_{k+1|k+1} - \mathscr{NN}(\mathscr{X}_k, \theta) \tag{21}$$

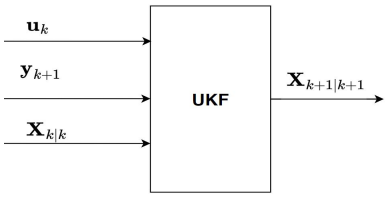

Fig. 1. Input output structure of UKF for NN training

3.1 Data Generation and Pre-Treatment

NN requires an enormous amount of data to be trained for a good approximation of any system or function. In the current work, the training data is generated by collecting multiple realizations of UKF by estimating the process states via simulations over a wide range of operating conditions obtained by random process inputs, for multiple noise realizations, and different initial states. Multilevel random input (For example, see Fig. (3)) coupled with different steady-state operating points ensures that a large enough input/output space is covered. Such a strategy of use of simulators to generate data for neural network training is discussed in Dufour et al. (2005). The "Constrained Sample Generation" step from section 2.2 is carried out after the updated states are obtained to generate a complete set of input and output pairs. The input/output data is standardized before training, and the parameters are preserved to be used while implementation to scale the data.

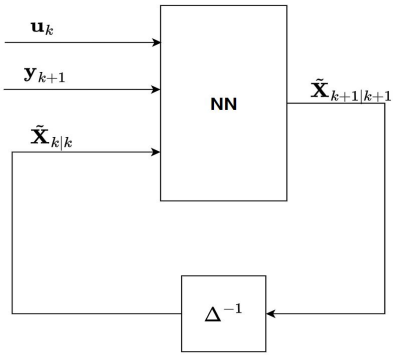

Fig. 2. NN for online deployment

3.2 NN Training

The loss function used for the NN training $J(\theta)$ to estimate θ for n_e epochs with n_s data points is,

$$E_k = \frac{1}{n_s}\sum_{k=1}^{n_s} \mathbf{r}_k^T \mathbf{r}_k \tag{22}$$

$$E_\theta = \frac{1}{2}\sum_{i=1}^{p+1} \theta_i^2 \tag{23}$$

$$J(\theta) = \alpha E_\theta + \beta E_k \tag{24}$$

Note that the objective function is regularized by Eq. (23), to reduce the extent of overfitting. α, β are initialised to $[0\ 1]$ respectively and are estimated per epoch as a part of the training algorithm. The training is carried out using MATLAB NN training toolbox (MATLAB, 2022a).

3.3 Online Implementation of NN-UKF

Algorithm 1 outlines the online implementation of the proposed approach, while Fig. 2 is a pictorial representation of the online implementation. As shown in Fig.2, the NN output $\tilde{\mathbf{X}}_{k+1|k+1}$ is used as input for the next time instant. The training of the NN is based on a single step-ahead prediction. However, during deployment, the predicted $\tilde{\mathbf{X}}_{k+1|k+1}$ is used in the NN input for the next time instant. The predicted samples are used to calculate the state estimates $\hat{\tilde{\mathbf{x}}}_{k+1|k+1}$.

Algorithm 1 Online Implementation of NN

1: At each instant k
2: Generate Samples and associated weights: $\tilde{\mathbf{x}}_{k|k}^{(i)}, \omega_i$ using $\hat{\tilde{\mathbf{x}}}_{k|k}$ and associated covariance $\mathbf{P}_{k|k}$ ▷ from Eq.(3,14-19)
3: Generate vector of input for NN: $\mathscr{X}_k = [\tilde{\mathbf{X}}_{k|k}, \mathbf{u}_k, \mathbf{y}_{k+1}]$
4: Sample Propagation: $\tilde{\mathbf{X}}_{k+1|k+1} = \mathscr{NN}(\mathscr{X}_k, \theta)$
5: Compute sample mean: $\hat{\tilde{\mathbf{x}}}_{k+1|k+1} = \sum_{i=1}^{n_x} \omega^{(i)} \tilde{\mathbf{x}}_{k+1|k+1}^{(i)}$
6: Compute sample covariance: $\mathbf{P}_{k+1|k+1} = \sum_{i=1}^{2n_x+1} \omega^{(i)} [\tilde{\mathbf{x}}_{k+1|k+1}^{(i)} - \hat{\tilde{\mathbf{x}}}_{k+1|k+1}][\tilde{\mathbf{x}}_{k+1|k+1}^{(i)} - \hat{\tilde{\mathbf{x}}}_{k+1|k+1}]^T$
7: Set: $k \leftarrow k+1$

4. SIMULATION CASE STUDIES

In this section, efficacy of the proposed approach is demonstrated using two simulation case studies. The first system is a two state CSTR system while the second system is the benchmark Williams Otto reactor system. Stochastic simulation experiments are carried out to generate data for UKF implementation and generation of data sets for NN training. The performance metrics used for assessing the performance of the NN based estimator are chosen as follows:

(1) **Computation Time**:

$$T = \frac{1}{L}\frac{1}{N_s}\sum_{i=1}^{N_s}\sum_{j=1}^{L} t_i^j \tag{25}$$

T is the average computation time over L realizations, N_s represents number of samples in one run.

(2) **Relative Average root mean squared error**:

$$R-ARMSE_i = \frac{1}{L}\sum_{j=1}^{L}\sqrt{\frac{\sum_{k=1}^{Ns}(\hat{x}_{i,k|k}^j - \hat{\hat{x}}_{i,k|k}^j)^2}{N_s}} \tag{26}$$

$R-ARMSE_i$ represents Relative Average root mean squared error over L realizations, $\hat{x}_{j,i,k|k}$ represents the states estimated by the Bayesian estimator and $\hat{\hat{x}}_{j,i,k|k}$ represents states estimated by NN model at k'^{th} instant for j^{th} realization of i^{th} state.

All computations were performed in *MATLAB* version 2022a on Windows 7 OS with Intel i7-7700, 4 Cores, 8 logical processors, 32 GB RAM computer. The average computational time is calculated from the sampling of $\mathbf{x}_{k|k}^{(i)}$ to the estimation of $\hat{\mathbf{x}}_{k+1|k+1}$ for all state estimation approaches.

4.1 Case Study A: Continuous Stirred Tank Reactor

A non-isothermal Continuous Stirred Tank Reactor (CSTR) in which an irreversible first-order reaction Eq. (27) is taking place is considered for demonstrating efficacy of the proposed approach.

$$A \rightarrow B \tag{27}$$

$$\frac{dC_A}{dt} = \frac{F}{V}(C_{A0}-C_A) - k_0\exp(\frac{-E}{RT})C_A \tag{28}$$

$$\frac{dT}{dt} = \frac{F}{V}(T_0-T) + \frac{(-\Delta H_r)k_0}{\rho C_p}\exp(\frac{-E}{RT})C_A - \frac{Q}{V\rho C_p} \tag{29}$$

$$Q = \frac{aF_c^{b+1}}{F_c + \frac{aF_c^b}{2\rho_c C_{pc}}}(T-T_{cin})$$

The system is governed by mass balance Eq. (28) and heat balance equation Eq. (29). The system parameters used in the simulation are taken from Marlin (1995). Here, $[C_{A0}\ T_{cin}]^T$ acts as a disturbance with its mean fixed at $[0.2\ 365]^T$ and is assumed to be corrupted by a zero mean Gaussian white noise characterized by the covariance matrix $\mathbf{Q}$. The measurement T is assumed to be corrupted by a zero mean white noise $\mathbf{v}_k$ with covariance $\mathbf{R}$. The classification associated with the variables is presented in Table 1. The reactor is simulated in open loop conditions and the reactor temperature is the only measurement. State and measurement noise model parameters are as follows,

Table 1. Case Study A: Variable description

Classification of Variable	
State Variable ($\mathbf{x}$)	C_A, T
Measured Variable ($\mathbf{y}$)	T
Manipulated inputs ($\mathbf{u}$)	F, F_C
Disturbances ($\mathbf{w}$)	C_{A0}, T_{cin}

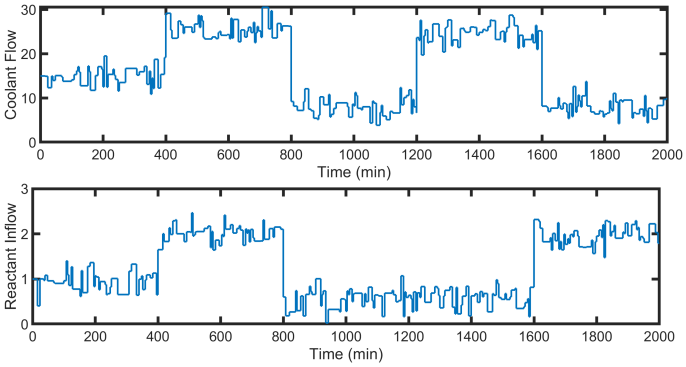

Fig. 3. Case Study A: Multilevel Process Inputs

$$\mathbf{P}_{0|0} = \begin{bmatrix} 6.278e-05 & -0.0133 \\ -0.0133 & 2.972 \end{bmatrix}$$

$$\mathbf{Q} = \begin{bmatrix} 0.0025 & 0 \\ 0 & 0.0225 \end{bmatrix}, \mathbf{R} = 0.0225$$

$$\mathbf{x}_L = [0\ 0]^T$$

Initial state $\mathbf{x}_0 = [0.26\ 393.95]^T$ is assumed to be a normally distributed variable with covariance chosen to be $\mathbf{P}_{0|0}$. $\mathbf{x}_L$ represent the lower bound imposed on the state estimates. The system is excited by injecting multilevel excitation at multiple operating points and the resulting simulation data is used for generation of the training data set using CUKF. Fig.(3) depicts one realization of manipulated inputs. In this case study, the NN model is used to approximate constrained unscented Kalman filter (CUKF) (Kottakki et al., 2016). The Input to the NN is $\mathscr{X}_k \in \mathbb{R}^{13}$ while output from the NN is $\hat{\mathbf{X}}_{k|k} \in \mathbb{R}^{10}$. The NN is a $p = 4$ hidden layered NN with 18 nodes in each layer. The activation function $\rho^{(i)}$ for the hidden layer for i=1,2,3,4 is a tansigmoid function $tansig(N) = \frac{exp^N - exp^{-N}}{exp^N + exp^{-N}}$, while the output layer $\rho^{(p+1)}$ has a linear activation function. The model is trained on a dataset of 1×10^6 instances (with $L = 30$) of input/output pair while tested on 2.5×10^5 instances. Details on training algorithm and metrics for measuring performance have been outlined in the Section. 3.2. A limit of 1×10^{-6} is set on performance metric for training.

Table. 2 depicts the comparison metrics between the state estimators. R-ARMSE values indicate that NN-CUKF generates reasonably accurate estimation performance. The one metric where NN-CUKF significantly outperforms CUKF is on computation time. NN-CUKF is ∼8.9 times faster than CUKF. Fig (4,5) depict temperature and concentration estimates, respectively for the validation dataset. It can be seen that NN-CUKF is able to track the states as well as CUKF. From Fig.(5) it can be seen that NN-CUKF is able track the states, but the estimation error is relatively higher when compared to CUKF.

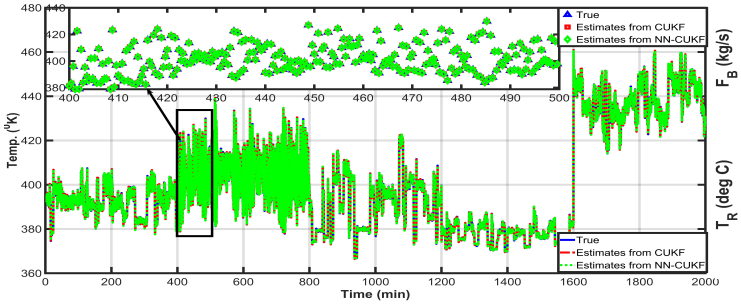

Fig. 4. Case Study A: Temperature estimates

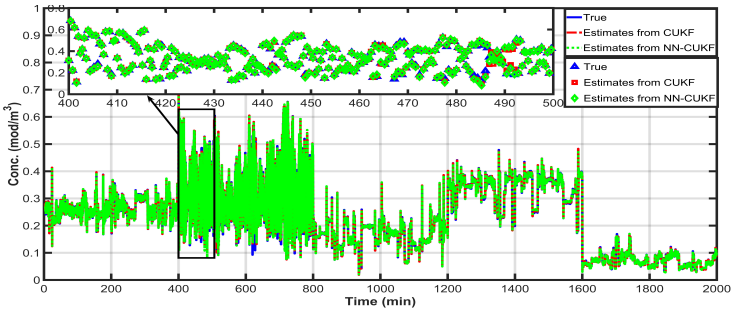

Fig. 5. Case Study A: Concentration estimates

Table 2. Case Study A: Comparision of performance metrics metrics

Parameter	Performance Metrics
$R-ARMSE_1$ (Concentration)	0.0082
$R-ARMSE_2$ (Temperature)	0.0309
T (CUKF) (sec)	0.0579
T (NN-CUKF) (sec)	0.0065

4.2 Case Study B: Williams Otto Reactor

The performance of the proposed approach is demonstrated using the Williams Otto reactor (Valluru and Patwardhan, 2019). In the reactor, irreversible exothermic reactions are taking place. The plant dynamics are modelled by Eq.(30-35). Here, $X_A, X_B, X_C, X_E, X_G, X_P$ are mass fractions of components A, B, C, E, G, P respectively. $r_1 = k_1 X_A X_B W, r_2 = k_1 X_B X_C W, r_3 = k_1 X_C X_P W$ are the reaction rates of the respective reactions. W is the mass of the reacting mixture. k_1, k_2, k_3 are the reaction rate constants. System parameters and associated reaction equations are in (Valluru and Patwardhan, 2019).

$$W\dot{X}_A = F_A - (F_A + F_B)X_A - r_1 \quad (30)$$

$$W\dot{X}_B = F_B - (F_A + F_B)X_B - r_1 - r_2 \quad (31)$$

$$W\dot{X}_C = -(F_A + F_B)X_C + 2r_1 - 2r_2 - r_3 \quad (32)$$

$$W\dot{X}_E = -(F_A + F_B)X_E + 2r_2 \quad (33)$$

$$W\dot{X}_G = -(F_A + F_B)X_G + 1.5r_3 \quad (34)$$

$$W\dot{X}_P = -(F_A + F_B)X_P + r_2 - 0.5r_3 \quad (35)$$

$$\mathbf{x}_L = [0\ 0\ 0\ 0\ 0\ 0]^T$$

The measurements are assumed to be corrupted by a zero mean white noise $\mathbf{v}_k$ with covariance $\mathbf{R}$. F_A acts as a disturbance (d)with its mean fixed at $1.827 kg/s$ and is assumed to fluctuate around it. F_A, F_B and T_R are assumed to be corrupted by a zero mean Gaussian white noise characterized by the covariance matrix $\mathbf{Q}$. A typical input profile (without noise) is depicted in Fig.6. Initial state covariance is chosen to be $\mathbf{P}_{0,0} = \mathbf{I}_{6\times 6} \times 10^{-6}$. The states cannot be negative as they represent

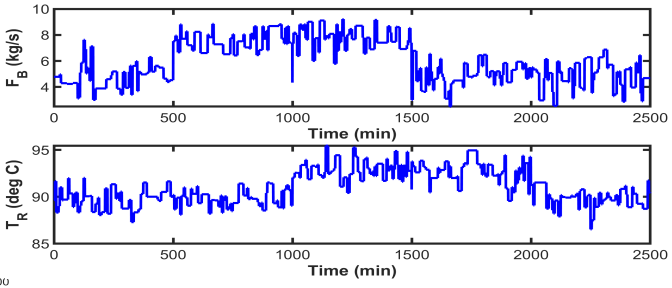

Fig. 6. Case Study B: Multilevel Process Inputs

Table 3. Case Study B: Variable description

Classification of Variable	
State Variable ($\mathbf{x}$)	$X_A, X_B, X_C, X_E, X_G, X_P$
Measured Variable ($\mathbf{y}$)	X_B, X_C, X_E, X_P
Manipulated inputs ($\mathbf{u}$)	F_B, T_R
Disturbances ($\mathbf{d}$)	F_A

Table 4. Case Study B: Comparision of performance metrics metrics

Parameter	Performance Metrics
$R-ARMSE_1$ (X_A)	0.0126
$R-ARMSE_2$ (X_G)	0.0063
T (ICUKF) (sec)	0.0235
T (NN-ICUKF) (sec)	0.0070

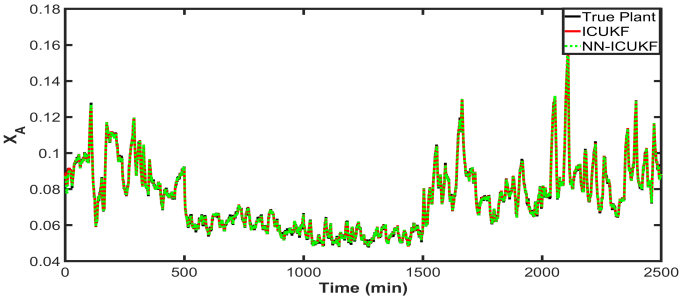

Fig. 7. Case Study B: X_A estimates from estimators

the mass fractions of components. Hence, interval-constrained UKF (ICUKF), as outlined in Kadu et al. (2010) is used to generate state estimates, whose mapping is then approximated using a NN. $\mathbf{x}_L$ represents the lower bound on the states. A description of associated variables can be found in Table.3

$$\mathbf{R} = diag[0.0078^2, 0.0003^2, 0.0058^2, 0.0022^2]$$

$$\mathbf{Q} = diag[0.0183^2, 0.0479^2, 0.8970^2]$$

The Input to the NN is $\mathscr{X}_k \in \mathbb{R}^{84}$ while output from the NN is $\tilde{\mathbf{X}}_{k|k} \in \mathbb{R}^{78}$. The NN selected is a three-hidden layered NN with 10, 20, 10 in respective layers. Hidden layer activation function is a tansigmoid function, while the output layer $\rho^{(4)}$ has a linear activation function. The model is trained on a dataset of 4.5×10^4 instances of input/output pair while tested on 5×10^3 instances (with L = 10). A limit of 1×10^{-10} is set on performance metric for training.

Table. 4 presents the comparison of performance metrics. The performance of NN-ICUKF to estimate X_G is significantly better when compared to the estimation of X_A. NN-ICUKF is $\sim$3.3 times faster than ICUKF. Fig (7,8) depict estimates of X_A and X_G respectively. It can be seen that NN-ICUKF is able to track the states as well as ICUKF. From both the case studies it

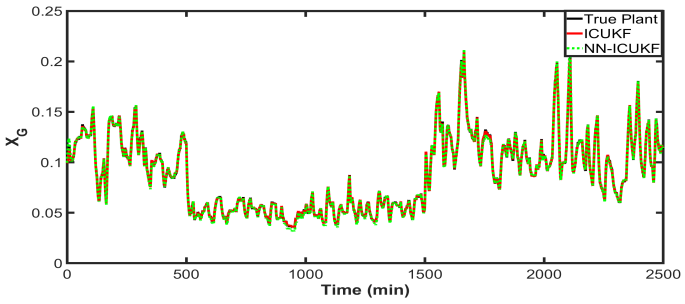

Fig. 8. Case Study B: X_G estimates from estimators

can be seen that the proposed approach is superior in terms of speed of execution and is at par in estimating states. The proposed approach can approximate a constrained sample-based state estimator, though the estimates from the NN were found to violate constraints at few instances. Simulations revealed that the NN implementations violated constraints 0.0347% and 0.004% times for the case studies A and B, respectively.

5. CONCLUSION

This paper explores approximation of sigma point-based estimators by NNs. The NN-CUKF and NN-ICUKF are approximated representations of CUKF and ICUKF using ML methods. The approach directly maps the initial posterior to the filtered posterior by exploiting the information of mean and covariance stored in sigma points. While the NN representations mimic constrained estimation, they do not enforce the constraints explicitly.

The results demonstrate that the benefit of the proposed approach relative to UKF lies in terms of computational efficiency while retaining similar estimation performance. An obvious disadvantage is the requirement of large amount of good quality data required to train the NNs and this may not be available for a practical case. However, if a digital twin of the system is available, it can act as the source of data or augment available data to train the NN. Our future research will focus on utilizing the proposed NN approach to speed-up particle filter and other sample-based approaches in presence of polytopic state constraints for problems of larger scale to enable practical implementation.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge funding for this work from Science & Engineering Research Board, Department of Science and Technology India through grant number CRG/2022/002587. The authors also gratefully acknowledge Prof. Mani Bhushan, ChE, IITB for insightful discussions.

REFERENCES

Dufour, P., Bhartiya, S., Dhurjati, P.S., and Doyle III, F.J. (2005). Neural network-based software sensor: training set design and application to a continuous pulp digester. *Control Engineering Practice*, 13(2), 135–143. doi:https://doi.org/10.1016/j.conengprac.2004.02.013.

Gelb, A. (ed.) (2006). *Applied optimal estimation*. M.I.T. Press, Cambridge, Mass., 19. printing edition.

Jouaber, S., Bonnabel, S., Velasco-Forero, S., and Pilte, M. (2021). NNAKF: A Neural Network Adapted Kalman Filter for Target Tracking. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4075–4079. IEEE, Toronto, ON, Canada. doi:10.1109/ICASSP39728.2021.9414681.

Julier, S. and Uhlmann, J. (2004). Unscented Filtering and Nonlinear Estimation. *Proc. IEEE*, 92(3), 401–422. doi:10.1109/JPROC.2003.823141.

Kadu, S.C., Bhushan, M., Gudi, R., and Roy, K. (2010). Modified unscented recursive nonlinear dynamic data reconciliation for constrained state estimation. *Journal of Process Control*, 20(4), 525–537. doi:10.1016/j.jprocont.2010.02.006.

Kottakki, K.K., Bhushan, M., and Bhartiya, S. (2016). Constrained Unscented Gaussian Sum Filter for state estimation of nonlinear dynamical systems. *Computers & Chemical Engineering*, 91, 352–364.

Marlin, T. (1995). *Process Control: Designing Processes and Control Systems for Dynamic Performance*. McGraw-Hill.

MATLAB (2022a). MATLAB train-train shallow neural network. `https://in.mathworks.com/help/deeplearning/ref/network.train.html`. Accessed: 2023-11-28.

Patwardhan, S.C., Narasimhan, S., Jagadeesan, P., Gopaluni, B., and L. Shah, S. (2012). Nonlinear Bayesian state estimation: A review of recent developments. *Control Engineering Practice*, 20(10), 933–953.

Revach, G., Shlezinger, N., Ni, X., Escoriza, A.L., van Sloun, R.J.G., and Eldar, Y.C. (2022). KalmanNet: Neural Network Aided Kalman Filtering for Non-Linear Dynamics with Partial Domain Knowledge. *IEEE Transactions on Signal Processing*, 70, 15. doi:10.1109/tsp.2022.3158588.

Sieberg, P.M., Blume, S., Reicherts, S., Maas, N., and Schramm, D. (2022). Hybrid State Estimation–A Contribution Towards Reliability Enhancement of Artificial Neural Network Estimators. *IEEE Trans. Intell. Transport. Syst.*, 23(7), 6337–6346. doi:10.1109/TITS.2021.3055800.

Talla, J. and Peroutka, Z. (2011). Neural network aided unscented kalman filter for sensorless control of pmsm. In *Proceedings of the 2011 14th European Conference on Power Electronics and Applications*, 1–9.

Vachhani, P., Narasimhan, S., and Rengaswamy, R. (2006). Robust and reliable estimation via Unscented Recursive Nonlinear Dynamic Data Reconciliation. *Journal of Process Control*, 16(10), 1075–1086.

Vaidehi, V., Chitra, N., Chokkalingam, M., and Krishnan, C. (2001). Neural network aided kalman filtering for multitarget tracking applications. *Computers & Electrical Engineering*, 27(2), 217–228. doi:https://doi.org/10.1016/S0045-7906(00)00013-6.

Valluru, J. and Patwardhan, S.C. (2019). An integrated frequent rto and adaptive nonlinear mpc scheme based on simultaneous bayesian state and parameter estimation. *Industrial & Engineering Chemistry Research*, 58(18), 7561–7578. doi:10.1021/acs.iecr.8b05327.

Venkatasubramanian, V. (2019). The promise of artificial intelligence in chemical engineering: Is it here, finally? *AIChE J*, 65(2), 466–478. doi:10.1002/aic.16489.

Wang, J., Alipouri, Y., and Huang, B. (2021). Dual Neural Extended Kalman Filtering Approach for Multirate Sensor Data Fusion. *IEEE Trans. Instrum. Meas.*, 70, 1–9.

Yun, S. and Zanetti, R. (2021). Bayesian Estimation with Artificial Neural Network. In *2021 IEEE 24th International Conference on Information Fusion (FUSION)*, 1–7. IEEE, Sun City, South Africa.