Physics Constrained Learning in Neural Network based Modeling

Rahul S. Patel*, Sharad Bhartiya and Ravindra D. Gudi*****

Department of Chemical Engineering, Indian Institute of Technology Bombay,

Powai, Mumbai 400076, India

** (E-mail: rahul.patel@iitb.ac.in)*

*** (E-mail: sharad_bhartiya@iitb.ac.in)*

**** (E-mail: ravigudi@iitb.ac.in)*

Abstract: Neural Network (NN) models based on training solely using data are limited in their use due to issues related to extrapolability and interpretability. On the other hand, while mechanistic models based on governing physical laws can overcome these limitations, the unavailability of accurate mechanistic models render them unsuitable for critical applications. In this paper, we propose an approach to develop an NN model that is trained to exploit available data while also being regularized by physics based information; in other words the loss function of NN is augmented by constraints associated with the system physics. This approach, also known as PINNs (Raissi et al. 2019) has been applied to representative problems in process systems engineering (PSE) to evaluate its efficacy to represent the knowledge about the physics of the system while also exploiting the information in the data. It has been shown that in the presence of noisy data and partially known physics model, this approach can give better predictions compared to the conventional training methodology. It has also been shown that the constraints given by the physics based model are also satisfied to a greater extent as compared to models trained only on data.

Keywords: Artificial intelligence and machine learning; Modeling and identification; Dynamic modelling and simulation for control and operation

1. INTRODUCTION

Neural networks (NN) have been used as universal function approximators which are primarily trained using data on cause and effect (C-E) measurements from physical systems. As such, they have found eminent use in various process systems engineering (PSE) applications. However, the training and the resulting knowledge representation about the variable inter-relationships seldom utilize the significant prior knowledge available in the form of physical and empirical laws. The NN based approaches have been constrained to represent knowledge based on information present in the data alone, and have therefore been associated with drawbacks such as poor interpretability & limited extrapolability; this has affected their acceptance in critical applications to some extent.

On the other hand, models based on prior knowledge about the physics are of course richer and help in relatively accurate knowledge representation about the C-E relationships. While they have relatively better extrapolability due to structural information present in them, such models are also incomplete due to relatively lesser understanding of the physics and the related first-principles relationships.

Approaches to exploit knowledge from the physics / first-principles as well as from data have been quite prevalent in the PSE literature. Depending on the relative accuracies of the model and the measurements, as well as the intended application at hand, estimates of system parameters as well as critical variables are generated by suitable weighing information from the sources, for realizing decisions in optimization and control. For instance, approaches in estimation & filtering (such as the Kalman filter and its various nonlinear extensions) have adopted methods to combine information from both the sources, *viz*. models and measurements to arrive at statistically optimal estimates of the key variables of interest.

While approaches to learn the C-E relationships using NN (both shallow as well as deep learning) structures has been primarily an exercise in empirical model identification, there are merits to bring in improved interpretability by augmenting the NN structures with prior physics-based knowledge of the C-E relationships. In the recent literature, data driven learning frameworks have been augmented with physics based models to give rise to a new class of deep learning approach known as physics-informed neural networks (PINN)(Raissi et al. 2017a, 2017b). PINNs have been successful for the solution and inversion of equations governing the physical systems.

Raissi et al. (2019) have used PINNs to solve PDEs for the applications in domains such as fluid mechanics & solid mechanics. Rudy et al. (2019) have used PINNs for data driven identification of parametric partial differential equations. Li et al. (2021) developed a electrochemical thermal model and have used it to generate data to train the PINNs for the task of electrode level state estimation in lithium ion batteries. Arnold and King (2021) have used PINNS for state space modelling of dynamical system and used them for control-based applications. It is important to note that all of the above

approaches are oriented towards generating an NN structure (including their weights/ biases) that solves and interprets such differential equations (*i.e.* they are data driven solution approaches to the differential equations). These developments have been accelerated by the advances in automatic differentiation and availability of open source platforms such as Theano, Tensorflow (Abadi et al., 2016) and Keras (Chollet et al.,2015). Haghighat and Juanes (2021) have introduced SciANN package that is well suited for neural networks application for scientific computation based on PINN.

Chen et. al (2018) developed a new family of deep neural network models by parameterizing the derivative of the hidden state using NN and calculating the network output using a blackbox differential equation solver. Their study discusses about training the NN by converting them into a form similar to ODE-IVP where the inputs are considered as values at time 0 and outputs at values as time T. The hidden states of the NN are represented as values at the intermediate time steps t ($0 < t < T$). Whereas, PINNs proposed in our work are focused on approximating the solution of PDEs/ODEs by a neural network which are trained to find the weights and biases that minimises the residuals of the differential equation. Krishnan et al. (2015) introduced techniques to learn causal generative temporal models from noisy high-dimensional data for medical applications. In such medical applications, a mechanistic model to correlate the inputs and outputs is not available to be used in Kalman filters. They developed a generative model by using healthcare claims data to look into effect of anti-diabetic drugs on a population of 8000 diabetic and pre-diabetic patients. They used this generative temporal model to perform counterfactual inference in Kalman filter setting.

An alternate approach to combine first-principles knowledge with the data generated from the system is to formulate an optimization problem for training the neural network using the data (i.e. minimizing the loss function) but also subject the solution to adhere to the constraints that are posed from the physics of the system. The associated NN parameters could then be expected to more closely reflect the physics-based relationships and therefore lend the composite model to better accuracy for predictions. In this context, Degroote et al., (2021) have used a similar approach for predicting unknown friction phenomena in crank servo mechanisms. Such augmentation approaches could offer several merits in terms of overcoming the drawbacks of relatively unknown or uncertain parameters in the physics, as well as noise/ biases in the measurements. The composite model could help complement knowledge from both these sources, and thereby serve to generate relative accurate predictions of the key variables.

In this paper, we propose to develop and evaluate such an augmentation of the physical laws based model with data generated from the system, for the task of developing a composite model. We revisit some of the typical, important PSE problems and propose methods to complement knowledge from physics as well as data. We demonstrate that such approaches can result in generating more accurate estimates for decision making related to optimization and control.

The remainder of the paper is structured as follows: Section 2 outlines the methodology applied. In Section 3, we cast representative PSE problems in the augmentation based optimization framework. Section 4 discusses results from the augmented learning and optimization approach, followed by summarizing conclusions in Section 5.

2. METHODOLOGY

A single-layer feed-forward neural network with inputs $x \in R^m$, outputs $y \in R^n$, and d hidden units is written as:

$$y = W_1 \sigma\ (W^0 x + b^0) + b^1 \qquad (1)$$

where ($W^0 \in \mathbb{R}^{dxm}$, $b^0 \in \mathbb{R}^d$), ($W^1 \in \mathbb{R}^{nxd}$, $b^1 \in \mathbb{R}^n$) are parameters known as weights and biases, and σ is the activation function. Any approach to NN training is typically formulated to identify the weights W^i and bias b^i of all layers in terms of minimizing a loss function of the form.

$$MSE_d = \frac{1}{N_d}\sum_{i=1}^{N_d} |y_i - \hat{y}_i|^2 \qquad (2)$$

Where $\hat{y}$ is the prediction of the neural network and y is the actual value that has been used for training.

In the PINN framework, the loss optimization framework is recast with an additional constraint and results in an optimization problem with a new loss function that can be written as

$$MSE = MSE_d + \lambda * MSE_m \qquad (3)$$

$$MSE_m = \frac{1}{N_f}\sum_{i=1}^{N_f} |f(y_i, x_i)|^2 \qquad (4)$$

Here *f* is the function derived from physics that captures the relationship between outputs y and inputs x. The first term in the total loss term MSE given by Equation (3) is used to capture the C-E relationships from the data and the second term captures the knowledge inferred from the physics based model. The parameter λ can be tuned to give more importance to data or model as per the requirements. Since the optimisation problem is non-convex, suitable optimisation algorithm and optimisation parameters must be chosen. One can also use choices of loss function other than mean squared error (MSE) such as mean absolute error (MAE) or cross entropy based on end applications.

The NN models are model are trained mainly by the back-propagation algorithm by taking the derivatives with respect to weights and biases. The ability of Opensource python packages such as TensorFlow and Pytorch have been exploited for Automatic differentiation of neural networks to obtain the model parameters (Güne¸ et al., 2018).

The augmentation of physics based loss term ensures that the training of NN Model constrains the weights and biases to obey the physics and capture the knowledge from data as well. This approach can work with simple feed-forward NN architectures on small amounts of data as the physics based loss term introduces a regularization mechanism (Raissi et al. 2019).

3. CASE STUDIES

This section discusses the representative problems in the augmentation based optimization framework for application in process systems engineering.

3.1 Case 1: Approximation of an arbitrary Nonlinear function representation

Neural networks have been extensively used to find mappings between outputs and inputs features for the given set of data. However, such training methods do not take into consideration the underlying constraints that influence the data evolution. By augmenting the loss term with prior knowledge, the extent of constraint violation in the predictions could be minimised.

Let y_1 and y_2 be nonlinear functions of x_1, x_2, and x_3 and nonlinear relationship between them is given as:

$$y_1 = x_1^2 x_3 + x_2^2 + x_1 x_2 + x_3 \qquad (5)$$

$$y_2 = x_3 x_2^3 + x_1 x_3 + x_1^2 \qquad (6)$$

Further let us assume, that there is an additional constraint on the output as Equation (7).

$$y_1 + y_2 = 1 \qquad (7)$$

For mapping this nonlinear function, a dataset of 1500 samples was generated with x_1, x_2, x_3, y_1 and y_2 such that it satisfies the Equations (5),(6) and (7). Of these 1200 samples were taken as training dataset and 300 samples for test data set. A feed forward neural network with two hidden layers and 20 neurons each with rectified linear unit (ReLu) as activation unit was chosen. An NN trained on this clean data could be expected to satisfy the constraint (Equation 7) too; however, in presence of the noisy inputs the constraint violations would be unacceptable. To test this constraint satisfaction of the augmented model with noisy data, Random noise is added to the inputs x_1, x_2, and x_3 and the NN is trained on the loss function augmented by physics, with a view to satisfy the constraints. The penalty factor can be tuned as a hyperparameter to meet the extent of constraint fulfilment.

The NN is trained on the loss function:

$$MSE_d = \frac{1}{N_d} \sum_{i=1}^{N_d} \left| Y_i - \hat{Y}_i \right|^2 \qquad (8)$$

$$MSE_m = \frac{1}{N_m} \sum_{i=1}^{N_m} \left| 1 - \left(\hat{y}_{1,i} + \hat{y}_{2,i} \right) \right|^2 \qquad (9)$$

Where Y=[y_1,y_2] are actual target values and $\hat{Y} = [\hat{y}_1, \hat{y}_2]$ are the NN predictions.

3.2 Case 2: Data reconcilation problem

Process Data reconciliation (PDR) is an approach which uses process information represented in the form of models to ensure data validation and reconciliation by the correcting measurements in industrial processes. The use of PDR allows for extracting knowledge about the processes from noisy measurement data and produces a set of refined data representing the most likely process operation.

Consider a distillation process in which the flow rates of feed, distillate and bottoms are represented by F_1, F_2 and F_3 respectively. The estimated flow rates must satisfy the steady state mass balance equation:

$$F_1 = F_2 + F_3 \qquad (10)$$

Let x_1, x_2 and x_3 be the feed, distillate and bottoms composition; these introduce an additional constraint as shown by Equation 11.

$$F_1 x_1 = F_2 x_2 + F_3 x_3 \qquad (11)$$

However, in the presence of noisy measurements of the input features – F_2, F_3, x_2, x_3 and targets F_1 and x_1 the constraints given by Equation (10) and (11) are less likely to be satisfied.

Here we aim to train the NN from data and the partial knowledge represented by Equation (10) to check the satisfaction of constraints represented by Equation (11) in presence of noisy data.

A dataset of 1500 samples was generated such that it satisfies the Equations (10) and (11). The measurements F_1, F_2, F_3, x_1, x_2, *and* x_3 are corrupted with random noise.1200 samples were taken as training dataset and 300 samples for test data set. A feed forward neural network with two hidden layers and 20 neurons each with ReLu unit as activation unit was trained using Adam optimiser for 1000 epochs.

The loss functions are:

$$\text{MSE}_{d1} = \frac{1}{N_d} \sum_{i=1}^{N_d} \left| F_{1i} - \hat{F}_{1,i} \right|^2 \qquad (12)$$

$$MSE_{d2} = \frac{1}{N_d} \sum_{i=1}^{N_d} \left| x_{1,i} - \hat{x}_{1,i} \right|^2 \qquad (13)$$

$$MSE_m = \frac{1}{N_m} \sum_{i=1}^{N_m} \left| \hat{F}_{1,i} - \left(F_{2,i} + F_{3,i} \right) \right|^2 \qquad (14)$$

$$MSE_m = \frac{1}{N_m} \sum_{i=1}^{N_m} \left| \hat{F}_{1,i} \hat{x}_{1,i} - \left(F_{2,i} x_{2,i} + F_{3,i} x_{3,i} \right) \right|^2 \qquad (15)$$

Where $\hat{F}_1$ and $\hat{x}_1$ are the predictions from the NN. The PINN loss function will be the sum of all the four losses.

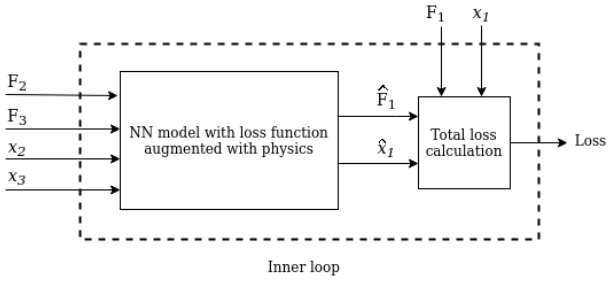

Figure 1: Flowchart to obtain the loss from NN model

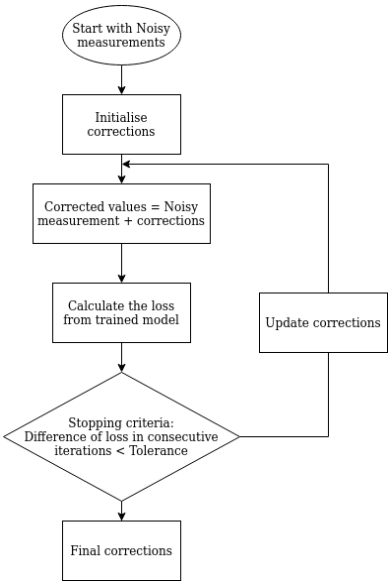

Figure 2: Predicting the corrections in case of noisy data.

The trained NN model is used to make predictions and calculate the total loss as shown in Figure 1. The higher the noise in the test data the higher will be the terms of the los functions. An optimiser is used to find corrections that minimises the prediction loss as shown in Figure 2. Here we have used the Nelder-Mead optimiser of Python's Scipy.optimise package.

3.3 Case 3: Kalman like adaptive estimation

Filtering and smoothening have multiple applications in PSE to produce estimates of unknown variables based on the series of noisy measurement observed over time. These applications require a model for estimation. An accurate and complete model that replicates the true state of plant is usually unavailable.
In presence of noisy measurements and partial knowledge about the model both of which express the ground truth to some extent, a physics constrained NN can be developed to generate noise free estimates.

The loss functions in all the three cases discussed below are:

$$MSE_d = \frac{1}{N_d} \sum_{i=1}^{N_d} \left| y_i - \hat{y}_{,i} \right|^2 \qquad (16)$$

$$MSE_m = \frac{1}{N_d} \sum_{i=1}^{N_d} |\hat{y}_i - Cx_i|^2 \qquad (17)$$

Where y and x are defined according model equations dicussed in case studies below.

Case 3.1: Bias in measurements

Consider the following one dimensional, discrete-time, nonlinear dynamical system, where y_k is the measurement of states x_k and u_k is the external input.

Ground truth:

$$x_k = \frac{x_{k-1}}{2} + \frac{25x_{k-1}}{1+x_{k-1}^2} + 5u_{k-1} \qquad (18)$$

Measurement model:

$$x_k = \frac{x_{k-1}}{2} + \frac{25x_{k-1}}{1+x_{k-1}^2} + 5u_{k-1} + w_k \qquad (19)$$

$$y_k = Cx_k + v_k \qquad (20)$$

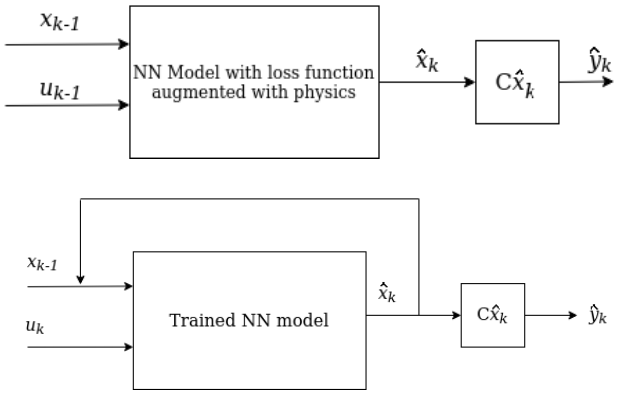

Figure 3: a) Training the NN model b) Using the trained model for forecasting.

The actual measurements obtained from the plant will have noise and bias as compared to the ground truth shown in Equations (18), (19) and (20). For the given system, C is assumed to be 1, w_k is assumed gaussian noise of mean 0 and standard deviation 2 and v_k is a constant bias of 5. The data set of 1900 samples is generated by providing x_{k-1} and u_{k-1} as the input to the measurement model with the corresponding output y_k. 1500 samples are used for training the NN model and 400 samples are used for validation. A single layer NN with 20 neurons with ReLu activation function is used for training. For the forecasting of the future measurement y_k, we start with a known value of x_k and use the ANN model that is trained on augmented loss function to predict the future 400 steps as shown in Figure 3b.

Case 3.2: Bias in both model and measurements

Plant model:

$$x_k = \frac{x_{k-1}}{2} + \frac{0.1x_{k-1}}{1+x_{k-1}^2} + 5u_{k-1} + w_k \qquad (21)$$

$$y_k = Cx_k + v_k \qquad (22)$$

In the real-world, we do not have an accurate measurement model to reflect the true states as estimated by the ground truth. Therefore, we corrupt our model by changing the prefactor 25 in the model to 0.1 as shown in Equation (21) and same tests are performed as in the previous section by changing the value

of penalty factor. Here, we have trained the NN model on the measurement generated from Equation (19) and the loss function is augmented with Equation (21).

Case 3.3: Parameter estimation

Parameter estimation:

$$x_k = \frac{x_{k-1}}{2} + \frac{\alpha * x_{k-1}}{1+x_{k-1}^2} + 5u_{k-1} + w_k \quad (23)$$

$$y_k = Cx_k \quad (24)$$

Consider the case as shown in Equation (23), where the model parameter α is unknown. We can estimate the parameter α by using the noisy data measurement generated by Equations (19) and (20) to train the NN model. The NN model will optimize the weights and the parameter α to fit the training dataset by minimising the loss functions given by Equations (16) and (17). The NN model in this case was trained using 1500 datapoints and the NN structure had single layer with 20 neurons for which the ReLu activation function was used.

4. RESULTS AND DISCUSSION

4.1 Approximation of an arbitrary Nonlinear function representation:

Table 1 shows the performance of trained NN models on the validation dataset. The extent of constraint violations in the NN model that is trained without any noisy data, are seen to be minimum. The trained NN model will have parameters such that it will only obey the constraints and won't give any priority to predicting the data, which can be seen from Figure 5. When the NN model is trained only on data i.e $\lambda = 0$, the extent of constraint violations will be larger than the case when the NN loss function is augmented with constraints. With an increase in penalty factor the extent of constraint violations will decrease as seen in Figure 4. This improvement in extent of constraint violation will be at the cost of loss in prediction accuracy given by MSE. However, the parameter λ can be tuned as per requirement to maintain trade-off between MSE and the constraint violations.

Table 1 : Validation errors for nonlinear approximation

Training criteria	MSE	Constraint violation (MSE)
Only data	0.084	0.00494
Only model	276.52	4.87E-05
Data and model	0.502	0.000389

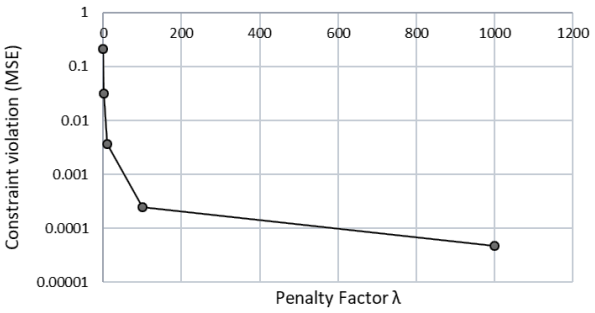

Figure 4: Effect of λ on extent of constraint violations.

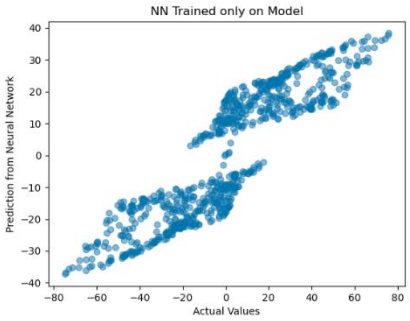

Figure 5: Actual vs Predicted values for NN trained only on model

4.2 Data reconcilation problem:

The NN model was trained with a penalty factor of 1 where the constraints were given by Equations (10) and (11). Model 1 is the total mass balance equation given by Equation (10) and Model 2 is component mass balance equation given by Equation (11). Table 2 shows that even in absence of model 2 i.e in presence of partial knowledge given by Model 1, the extent of constraint violations is better (lower) than the model trained only on data. The errors MSE1, MSE2, constraint violation 1 and constraint violation 2 are given by Equations 12-15 respectively. This trained model can now be used for reconciliation purposes of noisy data and to find the corrections using the scheme shown in Figure 2. The measurement F_1, F_2 and F_3 were corrupted by adding a bias and then the optimisation scheme was used to find those corrections such that the constraints are satisfied. Figure 6 shows that this scheme is able to predict these corrections accurately.

Table 2: Validation errors for Data reconciliation

Training criteria	MSE1	MSE2	Constraint violation 1 (MSE)	Constraint violation 2 (MSE)
Only data	0.09355	0.00228	0.03553	0.0164
Model 1	0.10006	1.08E-4	0.01004	0.00591
Model 1 and 2	0.10003	0.00117	0.01387	0.004807

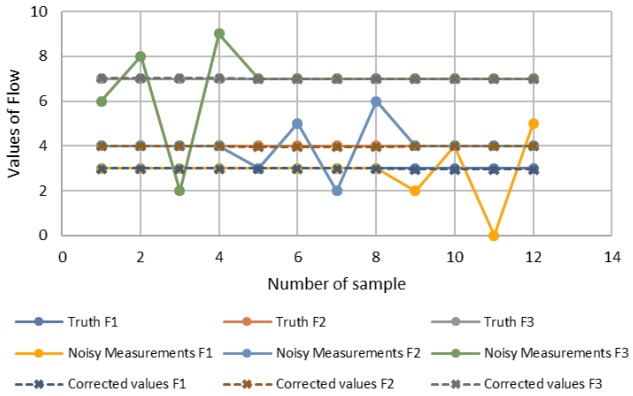

Figure 6: Reconciliation of the biased data

4.3 Kalman like adaptive estimation

For all the 3 subcases shown below, the external inputs U, the measurements y_k and the ground truth are taken to be the same as shown in the Figure (7) and (8).

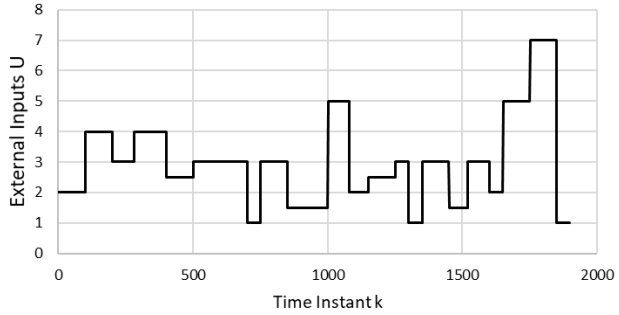

Figure 7: External inputs given to the system.

4.3.1 Bias in measurements

Since the data on which the NN is trained is corrupted with noise and a bias, the predictions of the test dataset using the NN model trained solely on the data will be far from the ground truth as shown in Figure 9. Also as the equations representing the ground truth are augmented with the NN loss function, with increase in the penalty factor, it can be seen that the prediction made by composite models are more accurate. The extent of constraint violations and the mean absolute error of predictions are compared in Table 3 for the penalty factor values of 0,1 and 10. For the case of penalty factor 10, The predictions are seen to be closer to the ground truth and constraint violations are fewer. This shows that the overall composite model performance can be improved by augmenting the loss function with model.

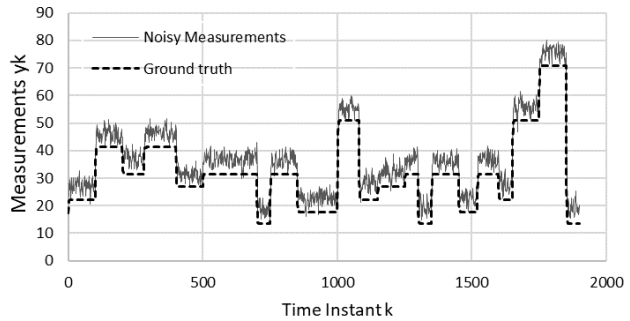

Figure 8: Comparison of Noisy measurements and ground truth.

Table 3: Performance of NN on Validation dataset

Training criteria	MAE	Constraint violation (MAE)
λ = 0	8.65	15.92
λ = 1	4.31	12.87
λ = 10	0.15	12.01

4.3.2 Bias in both model and measurements

The deviation of the states given by corrupted model and measurements from ground truth are shown as model truth in Figure 11. The noisy measurements with bias will lie above the ground truth and the model is corrupted in a manner such that it lies below the ground truth.

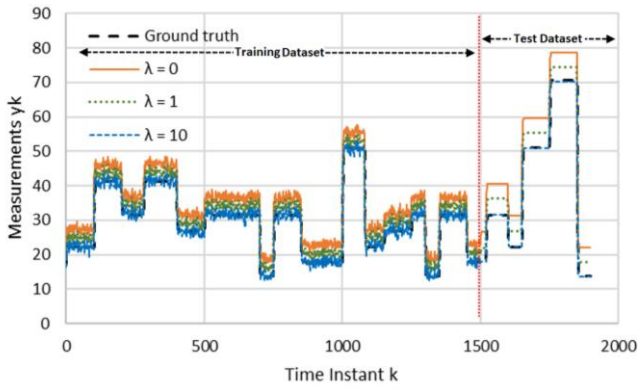

Figure 9: Training and test predictions for Case 3.1

The NN model is trained with different values of λ as mentioned in Table 4 to find the one which minimises the loss. The training and test predictions for values of λ 0, 3 and 50 are plotted in Figure 11. It can be observed that for λ = 0, and λ = 50 the predictions will deviate far away from the ground truth since, they will prioritise noisy measurements and corrupted model respectively. Of the chosen values the penalty factor λ = 3 gives the closest estimate of the ground truth and it can be further tuned so that the predictions match the ground truth.

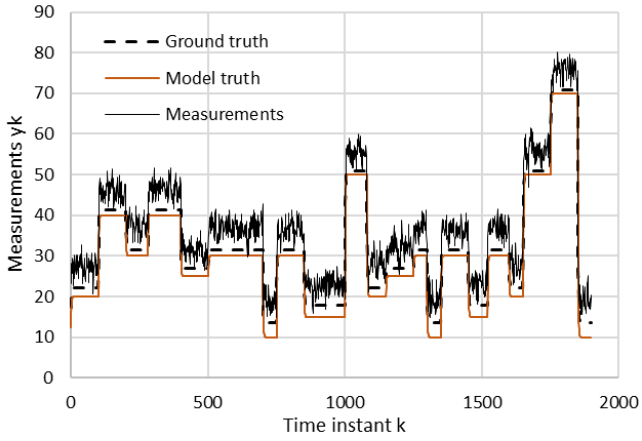

Figure 10: Model truth and measurements as compared to ground truth.

With this it can be concluded that by choosing a suitable value of penalty factor, aspects related to model inadequacies possible including bias can be suitably dealt with towards evolving more accurate estimates/predictions. even the presence of corrupted model and noisy data set with bias we can make predictions that are closer to ground truth.

Table 4: Effect of λ on extent of constraint violations.

Training criteria	MAE	Constraint violation (MAE)
λ = 0	8.65	15.92
λ = 0.1	8.56	15.86
λ = 1	3.8	12.99
λ = 2	0.76	11.52
λ = 3	0.57	11.2
λ = 5	1.18	11.23
λ = 10	1.46	12.77
λ = 50	1.58	12.807

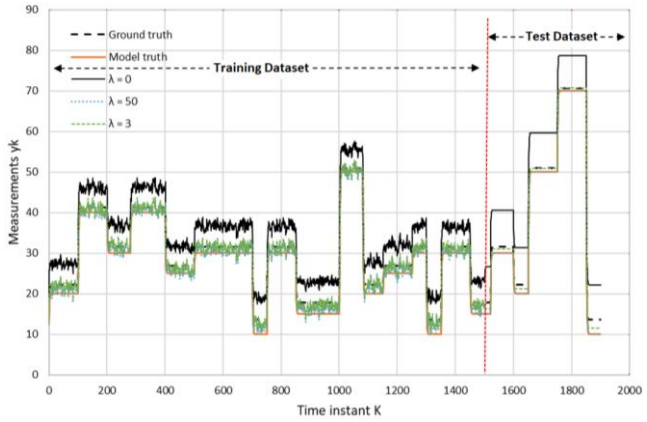

Figure 11: Training and test predictions for Case 3.2

4.3.3 *Parameter estimation*

For this case corresponding to formulation described in section in 3.3, the parameter α is estimated to be 24.61 by using the noisy data set in absence of bias, which is close to the actual value of 25. This shows that the approach can be used to identify unknown model parameters.

5. CONCLUSION

The training of NN by the use of data and physical models to regularize the predictions such that it obeys the physical constraints have been discussed in the mentioned case studies. It has been shown that by tuning the penalty factor, the extent of constraint violations can be satisfied as per the requirements. This approach can also be used to identify the unknown model parameters. The issues related to extrapolability of the neural network can be explored by this approach. Such physics constrained NN can find applications in areas such as fault detection and diagnosis, system model identification and process control in the field of process system engineering. Here we have only used feed forward NN for all the cases; however, one can also explore the possibility of using recurrent neural network and other deep learning architectures for estimation of states.

REFERENCES

Abadi, M. *et al.* (2016) "TensorFlow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation.*

Arnold, F. and King, R. (2021) "State–space modeling for control based on physics-informed neural networks," *Engineering Applications of Artificial Intelligence*.

Chen, Ricky TQ, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. (2018) "Neural ordinary differential equations." *Advances in neural information processing systems 31*.

Chollet, F. & others, 2015. Keras. Available at: https://github.com/fchollet/keras.

Degroote, W. *et al.* (2021) "Neural Network Augmented Physics Models for Systems with Partially Unknown Dynamics: Application to Slider-Crank Mechanism," *IEEE/ASME Transactions on Mechatronics* [Preprint].

Güne¸, A. *et al.* (2018) Automatic Differentiation in Machine Learning: a Survey, *Journal of Machine Learning Research*.

Haghighat, E. and Juanes, R. (2021) "SciANN: A Keras/TensorFlow wrapper for scientific computations and physics-informed deep learning using artificial neural networks," *Computer Methods in Applied Mechanics and Engineering*.

Krishnan, Rahul G., Uri Shalit, and David Sontag.(2015) "Deep kalman filters." arXiv preprint arXiv:1511.05121.

Li, W. *et al.* (2021) "Physics-informed neural networks for electrode-level state estimation in lithium-ion batteries," *Journal of Power Sources*.

Raissi, M., Perdikaris, P. and Karniadakis, G.E. (2017a) "Inferring solutions of differential equations using noisy multi-fidelity data," *Journal of Computational Physics*.

Raissi, M., Perdikaris, P. and Karniadakis, G.E. (2017b) "Machine learning of linear differential equations using Gaussian processes," *Journal of Computational Physics*.

Raissi, M., Perdikaris, P. and Karniadakis, G.E. (2019) "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*.

Rudy, S. *et al.* (2019) "Data-driven identification of parametric partial differential equations," *SIAM Journal on Applied Dynamical Systems*.