Enhancing Neural Network Prediction against Unknown Disturbances with Neural Network Disturbance Observer

Maxime Pouilly-Cathelain[1,2], Philippe Feyel[1], Gilles Duc[2] and Guillaume Sandou[2]
[1]*Safran Electronics & Defense, Massy, France*
[2]*L2S, CentraleSupélec, CNRS, Université Paris-Sud, Université Paris-Saclay, Gif-sur-Yvette, France*

Keywords: Neural Network, Prediction, Disturbance Observer.

Abstract: Neural network prediction is a very challenging subject in the presence of disturbances. The difficulty comes from the lack of knowledge about perturbation. Most papers related to prediction often omit disturbances but, in a natural environment, a system is often subject to disturbances which could be external perturbations or also small internal parameters variations caused, for instance, by the ageing of the system. The aim of this paper is to realize a neural network predictor of a nonlinear system; for the predictor to be effective in the presence of varying perturbations, we provide a neural network observer in order to reconstruct the disturbance and compensate it, without any a priori knowledge. Once the disturbance is compensated, it is easier to realize such a global neural network predictor. To reach this goal we model the system with a State-Space Neural Network and use this model, completed with a disturbance model, in an Extended Kalman Filter.

1 INTRODUCTION

Most of controlled processes are very sensitive to disturbances that may be of different natures; on the one hand, they can model real external disturbances or, on the other hand, internal parameter variations of the system provided that they are small enough (Chen et al., 2000). In this paper, we aim to model nonlinear systems by using neural networks in order to get a predictor. For instance, this predictor can be used in a Model Predictive Control (Yu and Gomm, 2003), being in that case initialized by the measure at each time step following the so-called receding horizon principle (Keviczky and Balas, 2006). Usually, for the training task, the neural network used for prediction purpose is a Feedforward Neural Network (FNN) preceded by tapped delay lines (TDL, defined in figure 1). For the prediction task, the neural network is then artificially looped into a Nonlinear AutoRegressive eXogeneous model (NARX). This method, summarized in figure 1, is similar to the one presented in (Hagan et al., 1996) (Chapter 27) and has proved its efficiency for different kinds of systems (Hedjar, 2013), (Diaconescu, 2008). This neural network structure is preferred because it is easy to be trained offline and online contrary to recursive structures as shown in (Pascanu et al., 2013). Moreover, training this structure online allows to learn small parameters variations that can be modeled by small disturbances, however we aim to deal with relatively high disturbances in this paper.

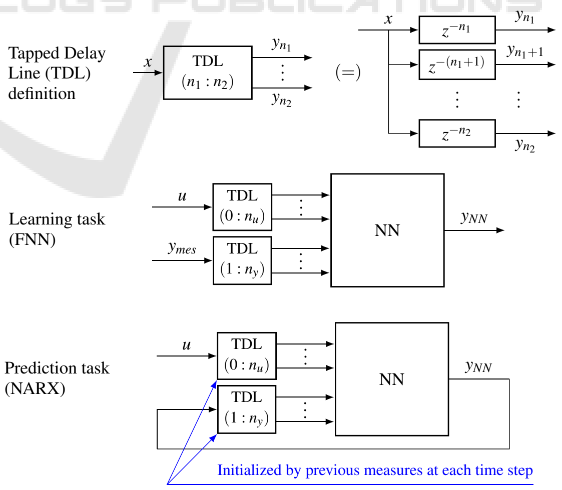

Figure 1: Neural network prediction using FNN/NARX models.

However, using such a method leads to inaccurate prediction results in the face of varying disturbance profiles or parameter variations. Indeed, the neural network is trained using some non-disturbed data or some particular disturbance inputs. As a result, the prediction remains well adapted to these particular situations, but some large prediction errors may arise in other real-life situations characterized by other realistic disturbance profiles or system parameter variations. This statement will be shown with the example at the end of the paper.

With the aim of improving the performance of the predictor against unknown disturbances that can vary or vanish, a natural idea is to estimate the current disturbance. The observed perturbation will then be used as a feedback compensator to make the system less sensitive to disturbances. Some solutions have already been proposed in the literature. (Vatankhah and Farrokhi, 2017) presents an observer based on a neural network inverse model to find the inverse model of the system with an offline identification: it allows to reconstruct the control signal and so to deduce the perturbation. The main drawback of this method is that it supposes that the perturbation signal is available during the training process. Moreover, the identification of the static inverse neural network model is not efficient in case of parameter variations. (Talebi et al., 2010) and (Lakhal et al., 2010) have developed an adaptive observer based on backpropagation which is a well-known neural network learning algorithm (Werbos, 1974). An idea could be to combine this method with a disturbance observer. However the steady state error, although bounded, does not asymptotically converge to zero and the disturbance may not be well reconstructed.

In this study, the aim is to asymptotically reconstruct the perturbation without any direct measurement of the perturbation.

To improve the predictor performance for a plant subject to a varying disturbance, the following three steps methodology, summarized in figure 2, is proposed in this paper:

- The plant is modelled as a State-Space Neural Network (SSNN), offline trained without disturbances. This neural network structure has been chosen because, contrary to the NARX structure, it can be easily used as an observer.
- The state of the SSNN model is augmented with the searched disturbance to be reconstructed. Then, by using this new augmented model, an Extended Kalman Filter (EKF) reconstructs the augmented state and thus the disturbance.
- The observed disturbance is finally used as a feedback compensator for the purpose of compensating the perturbation effect.

This paper is organized as follows. In section 2, corresponding to the first step of the methodology, the structure of the neural network used to model the plant and the associated training method is presented. The second step is presented, in section 3 where the EKF formulation is reminded and applied to a simple example of SSNN. Finally, the compensator design is the core of section 4. Numerical results to prove the viability of the approach are given in section 5 and section 6 concludes and gives some forthcoming works.

This study can easily be extended to MIMO (Multi Input Multi Output) systems but, for simplicity of equations and schemes, we will only present the method for SISO (Single Input Single Output) systems. By the way, the example corresponds to a SIMO system.

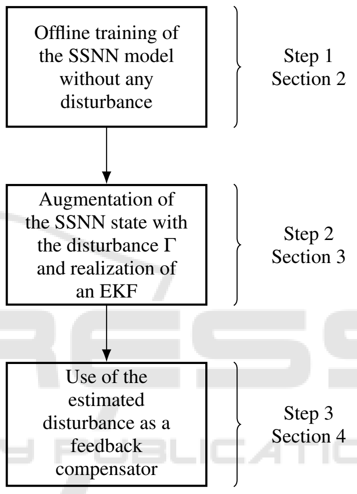
Offline training of the SSNN model without any disturbance
Step 1 Section 2
Augmentation of the SSNN state with the disturbance Γ and realization of an EKF
Step 2 Section 3
Use of the estimated disturbance as a feedback compensator
Step 3 Section 4

Figure 2: Three steps proposed method.

2 STATE-SPACE NEURAL NETWORK MODELING

2.1 State-Space Neural Network (SSNN)

In order to be able to model any kind of nonlinearity we have chosen to use a neural network (Hagan et al., 1996) due to its ability to approximate any kind of static function (Cybenko, 1989). Because of the dynamical property of the considered system, it is required to use recurrent neural networks (RNN). It exists many different structures of recurrent neural networks, (De Jesus and Hagan, 2007) gives some examples. The SSNN structure (Figure 3) seems us more suitable to be used in nonlinear observer techniques such as extended Kalman filter. Some properties of the SSNN structure can be found in (Zamarreño and Vega, 1998).

Figure 3 defines the following notation: u is the input signal, T_e is the sampling period, k the sampling index, X the internal state vector of the SSNN, y_{NN} the output of the SSNN, y the output of the plant, e the error between the plant and the model and z^{-1} the delay operator. The hidden layer has a nonlinear activation function and the output layer has a linear activation function.

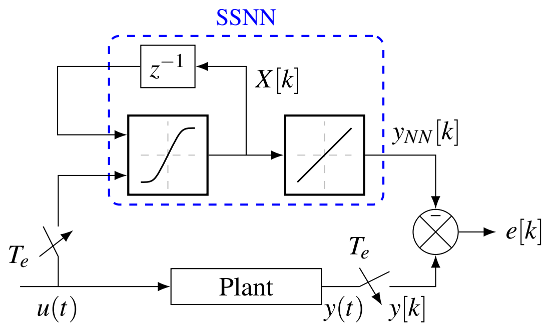

Figure 3: SSNN model of the plant.

The SSNN structure contains internal states that model the dynamics of the system. The number of neurons of the hidden layer, noted n, is equal to the number of internal states. These states do not have any physical meaning and n is a compromise between accuracy of the model and complexity (essentially in terms of computation time and memory needed for the learning process). The SSNN is represented by:

$$\begin{cases} X[k+1] = \sigma(W_{h,u}u[k] + W_{h,X}X[k] + B_h) \\ y_{NN}[k] = W_f X[k] + B_f \end{cases}, \quad (1)$$

where $X \in \mathbb{R}^{n\times 1}$ is the internal state vector, $W_{h,u} \in \mathbb{R}^{n\times 1}$ are the weights of the first layer related to the input u, $W_{h,X} \in \mathbb{R}^{n\times n}$ the weights of the first layer related to the state X, $B_h \in \mathbb{R}^{n\times 1}$ the bias of the first layer, $W_f \in \mathbb{R}^{1\times n}$ the weights of the final layer, $B_f \in \mathbb{R}$ the bias of the final layer. The activation function of the hidden layer σ is the sigmoid function defined by:

$$\sigma(x) = 1/(1+\exp(-x)). \quad (2)$$

This activation function has to be evaluated for each component of the vector in (1). Equation (1) is directly a state-space representation of a nonlinear system. The identification process (also named learning or training process) aims to determine the values of $W_{h,u}$, $W_{h,X}$, B_h, W_f and B_f.

Figure 4 shows the details of the internal SSNN structure. We note $B_h = [b_{h_1}, \dots, b_{h_n}]^{\mathrm{T}}$.

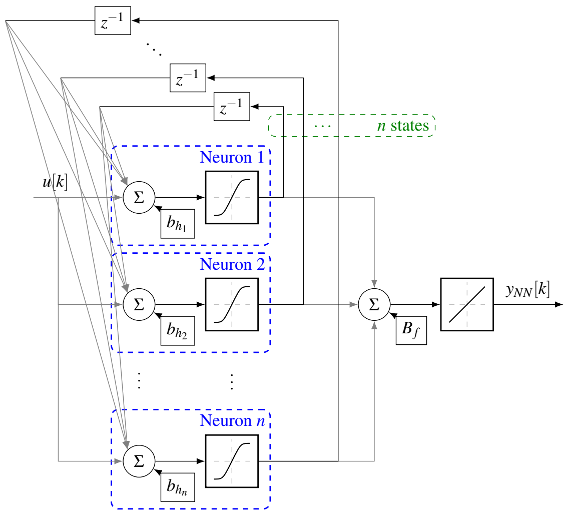

Figure 4: SSNN (grey arrows are weighted connections).

2.2 Training task of the State-Space Neural Network

The identification process is realized offline once for all. All training data can be acquired with or without disturbances. However, in the former case, the estimated disturbance will not correspond to the real one because the SSNN model will take into account the disturbance included in the training data. For instance, if the training set contains an additive input disturbance Γ_{train} and if we use the system in presence of another disturbance Γ_{use} then the estimated disturbance $\hat{\Gamma}$ will be defined by (3).

$$\hat{\Gamma} = \Gamma_{use} - \Gamma_{train}. \quad (3)$$

For clarity, we suppose in the sequel that the training data are obtained without disturbances.

Because the SSNN is a recurrent neural network, classical learning algorithms such as Backpropagation [8] cannot be used. Many learning algorithms have been developed to deal with recurrent neural networks such as Real Time Recurrent Learning (RTRL) (Williams and Zipser, 1989), BackPropagation Through Time (BPTT) (Werbos, 1990) and Decoupled Extended Kalman Filter (DEKF) (Haykin, 2004). The two main learning algorithms for RNN are RTRL and BPTT which are both presented in (Hagan et al., 1996). The RTRL, combined with the well-known Levenberg-Marquardt (LM) method (Levenberg, 1944), (Marquardt, 1963), is used in this study. These algorithms use the cost function defined by (4).

$$J_c = \frac{1}{n_s}\sum_{k=1}^{n_s} e^2[k] = \frac{1}{n_s}\sum_{k=1}^{n_s}(y[k]-y_{NN}[k])^2, \quad (4)$$

where n_s is the number of samples.

Usually, the input signal is chosen to be square with random amplitudes and durations (between minimum and maximum values defined by the user) in order to stimulate the system for all possible configurations. For that purpose, it is supposed that bounds on the admissible input amplitudes are known as well as an order of magnitude of the system time response. For instance, a possible input signal for a system with input levels in the range $[-2;2]$ and a time response equals to 1s is presented in figure 5.

In the sequel, all inputs and outputs data have to be normalized.

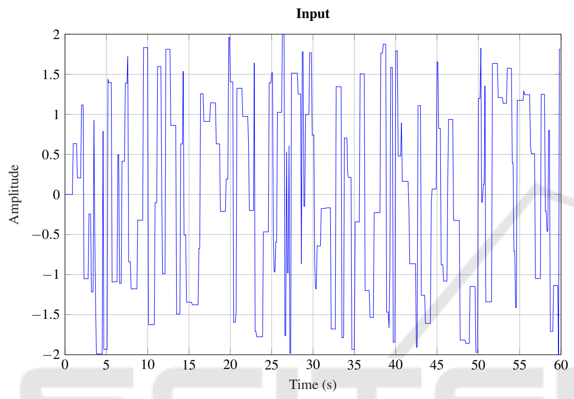

Figure 5: Example of input signal.

RTRL is a deterministic algorithm but a starting point is needed and randomly chosen. Thus, several runs with different initializations can be done, hoping the result to be closer to the global minimum. Moreover, in order to avoid overfitting (and so to improve the prediction capacity with regard to different operating conditions) an early stopping has been performed by using a validation set as proposed in (Sarle, 1995). The data set is thus divided in three different sets: a training set, a validation set and a test set. The training set is used to train the system, the validation set is used for generalization purpose and the test set is used to test the neural network at the end of the training. Finally, the stopping criterion consists in one of these conditions:

- the maximum number of iterations has been reached,
- the cost function has reached the minimum value defined by the user,
- the cost function evaluated with the validation set has increased for n_v consecutive times, where n_v is an integer defined by the user.

The reader can get more information about RTRL and LM respectively in (Williams and Zipser, 1989) and (Gavin, 2017). The training of RNN can be a challenging task for some systems. (Pascanu et al., 2013) presents some difficulties that can occur during the training and some solutions to improve the learning.

3 STATE-SPACE NEURAL NETWORK DISTURBANCE OBSERVER

3.1 Disturbance Model

As shown in figure 6, we consider in this paper that a disturbance is added at the input of the system. As a reminder, this disturbance can model external perturbations or internal parameter variations providing that, depending on the system, they are small enough.

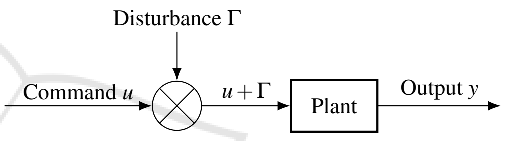

Figure 6: Disturbed plant.

3.2 Augmented State with Disturbance

A common method to estimate disturbances is to complete the state with a new state Γ that represents the disturbance by enforcing:

$$\frac{\mathrm{d}\Gamma(t)}{\mathrm{d}t} = 0, \tag{5}$$

The new augmented state vector X_e can now be defined from the model proposed in figure 6 in which the plant is the SSNN and $X_e = (X^{\mathrm{T}}, \Gamma^{\mathrm{T}})^{\mathrm{T}}$. In discrete time we get (6).

$$\begin{cases} X_e[k+1] = \begin{pmatrix} \sigma\{W_{h,u}(u[k] + C_1 X_e[k]) \\ \quad + W_{h,X} C_2 X_e[k] + B_h\} \\ C_1 X_e[k] \end{pmatrix} \\ y_{NN}[k] = W_f C_2 X_e[k] + B_f \end{cases}, \tag{6}$$

where $C_1 = [0_{1\times n} \quad 1]$ and $C_2 = [I_{n\times n} \quad 0_{n\times 1}]$.

Given X_e from (6), we can implement a state observer that will estimate X_e and thus Γ.

3.3 Extended Kalman Filter (EKF)

The system (6) can be observed by using many non-linear observation methods such as sliding mode observer (Alessandri, 2000), EKF (Terejanu, 2008), Unscented Kalman Filter (UKF) (Wan and Van Der Merwe, 2000), high-gain observer (Bullinger and Allgöwer, 1997) or extended Luenberger observer (Grossman, 1999). The EKF, which is a nonlinear derivation of the original Kalman filter (Kalman, 1960), is used in this paper, for its simplicity of implementation even if tuning the weights may be sometimes difficult. Considering a system defined by:

$$\begin{cases} X_e[k+1] = f(X_e[k], u[k], w[k]) \\ y_{NN}[k] = h(X_e[k], v[k]) \end{cases} . \tag{7}$$

$w[k]$ and $v[k]$ are two white Gaussian noises with respective covariance matrices $Q[k]$ and $R[k]$ that can be used as tuning weights for the filter. One iteration of the Extended Kalman Filter is concerned with two steps, the prediction and the update ones, where the use of the "hat" notation denotes the estimated values.

- Prediction step:

$$\begin{cases} X_e[k \mid k-1] = f(\hat{X}_e[k-1], u[k], 0) \\ P[k \mid k-1] = F[k]P[k-1]F^{\mathrm{T}}[k] + Q[k] \end{cases} . \tag{8}$$

- Update step:

$$\begin{cases} \tilde{y}_{NN}[k] = y[k] - h\left(\hat{X}_e[k \mid k-1], 0\right) \\ S[k] = H[k]P[k \mid k-1]H^{\mathrm{T}}[k] + R[k] \\ K[k] = P[k \mid k-1]H^{\mathrm{T}}[k]S^{-1}[k] \\ \hat{X}_e[k] = X_e[k \mid k-1] + K[k]\tilde{y}_{NN}[k] \\ P[k] = (I - K[k]H[k])\,P[k \mid k-1] \end{cases} . \tag{9}$$

Where $F[k] = \left.\frac{\partial f}{\partial X_e}\right|_{\hat{X}_e[k-1], u[k]}$ and $H[k] = \left.\frac{\partial h}{\partial X_e}\right|_{X_e[k|k-1]}$. $F[k]$ and $H[k]$ can be easily computed from (6) since the sigmoid function is differentiable:

$$\sigma^{(1)}(x) = \frac{\mathrm{d}\sigma(x)}{\mathrm{d}x} = \frac{\exp(-x)}{(1+\exp(-x))^2} \tag{10}$$

and h is a linear function in the considered case (see (6)). For instance, F and G are given in section 3.4 for a 2-neurons SSNN.

As states of the SSNN have no physical meaning, an accurate reconstruction of all states is not the priority. It is suggested to tune the filter to give more importance to the disturbance, that is the last component of X_e.

3.4 Example for a Two-neurons SSNN

To illustrate the computation of matrices $F[k]$ and $H[k]$ of the Kalman filter, a model with two neurons and $\mathrm{d}\Gamma(t)/\mathrm{d}t = 0$ is considered in this section.

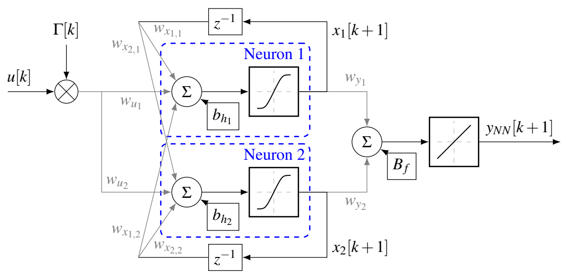

Figure 7: Two neurons SSNN.

Figure 7 presents such a two-neurons SSNN. We use the following notations: $X_e = (x_1\ x_2\ \Gamma)^{\mathrm{T}}$, $W_{h,u} = (w_{u_1}\ w_{u_2})^T$, $B_h = (b_{h_1}\ b_{h_2})^T$, $W_f = (w_{y_1}\ w_{y_2})$ and $W_{h,x} = \begin{pmatrix} w_{x_{1,1}} & w_{x_{1,2}} \\ w_{x_{2,1}} & w_{x_{2,2}} \end{pmatrix}$.

Using (6) and $\mathrm{d}\Gamma(t)/\mathrm{d}t = 0$, ones can get:

$$\begin{cases} x_1[k+1] = \sigma\{w_{x_{1,1}} x_1[k] + w_{x_{1,2}} x_2[k] \\ \qquad + w_{u_1}(u[k] + \Gamma[k]) + b_{h_1}\} = \sigma(\alpha[k]) \\ x_2[k+1] = \sigma\{w_{x_{2,1}} x_1[k] + w_{x_{2,2}} x_2[k] \\ \qquad + w_{u_2}(u[k] + \Gamma[k]) + b_{h_2}\} = \sigma(\beta[k]) \\ \Gamma[k+1] = \Gamma[k] \end{cases} . \tag{11}$$

We obtain

$$F[k] = \begin{pmatrix} w_{x_{1,1}}\sigma^{(1)}(\alpha[k]) & w_{x_{1,2}}\sigma^{(1)}(\alpha[k]) & w_{u_1}\sigma^{(1)}(\alpha[k]) \\ w_{x_{2,1}}\sigma^{(1)}(\beta[k]) & w_{x_{2,2}}\sigma^{(1)}(\beta[k]) & w_{u_2}\sigma^{(1)}(\beta[k]) \\ 0 & 0 & 1 \end{pmatrix} \tag{12}$$

and

$$H[k] = (w_{y_1}\ w_{y_2}\ 0)\,. \tag{13}$$

Obviously, the computation of F and G can be easily extended for a network with more neurons.

4 ROBUST NEURAL NETWORK BASED PREDICTOR

In this section, the observed disturbance is used as a feedback signal in order to compensate perturbations or small parameter variations of the plant. A new global model, which is less disturbed, is obtained. Thus, it will be easier to train this model with a FNN used as a predictor as shown in figure 1.

The proposed solution is shown in figure 8 where $\hat{\Gamma}$ corresponds to the estimated disturbance. The neural network can be tuned online as it has been done in (Bao et al., 2017). Note that this solution may cause instability due to the feedback created with the EKF. Proving the stability of the closed-loop with the EKF is not a trivial task, and forthcoming works will aim to use other observer techniques such as sliding mode observer or high-gain observer in order to study the global stability.

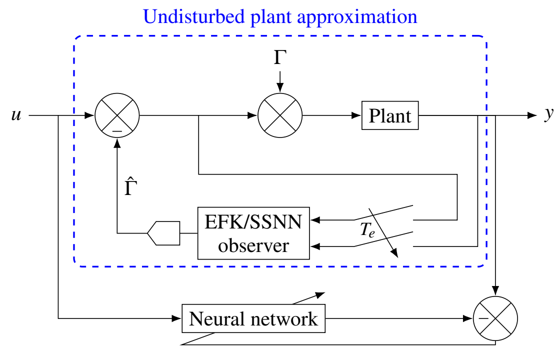

Figure 8: Feedback compensation of disturbances.

It appears to be difficult to train offline the neural network presented in figure 8 because adaptive gain K (see (9)) is varying and a fixed neural network will not be able to get the dynamics.

One drawback of this structure is that neural network predictor trained online (using FNN as presented in figure 1) on the global plant requires more neurons than a neural network predictor would need for the plant alone. Moreover, the online training can only be accurate if the input signal u has a sufficiently rich content of frequency and amplitude variations.

5 NUMERICAL EXAMPLE

In this section, the numerical example presented in figure 9 (where s is the Laplace variable) is chosen to illustrate the proposed approach. Table 1 gives all parameter values for this example. To show the genericity of the model with regard to nonlinearities, the example has been chosen to include differentiable and non-differentiable nonlinearities. To begin with, the inefficiency of the predictor trained without perturbation in a disturbance environment is shown. Moreover, this predictor is not accurate when it is trained with a disturbance if this disturbance varies or vanishes. Finally, the three steps defined in figure 2 are applied.

5.1 False Prediction in Case of Disturbance

As said in the introduction section, the method proposed in figure 1 leads to poor prediction results when a disturbance appears and the predictor has been trained using non-disturbed data. Figure 10(a) presents prediction on undisturbed system ($\Gamma = 0$, in figure 9) and figure 10(b) presents prediction on the system disturbed by: $\Gamma(t) = 0.2\sin(2\pi t)$.

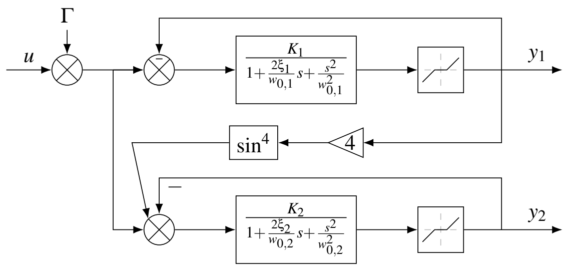
Γ
u
$\frac{K_1}{1+\frac{2\xi_1}{w_{0,1}}s+\frac{s^2}{w_{0,1}^2}}$
y_1
$\sin^4$
4
−
$\frac{K_2}{1+\frac{2\xi_2}{w_{0,2}}s+\frac{s^2}{w_{0,2}^2}}$
y_2

Figure 9: Studied system.

Table 1: Parameters.

Parameter	Value	Unit
K_1	2	-
ξ_1	0.5	-
$w_{0,1}$	150	rad.s^{-1}
K_2	4	-
ξ_2	0.05	-
$w_{0,2}$	300	rad.s^{-1}
T_e	10^{-3}	s
Start of the dead zones	-0.2	-
End of the dead zones	0.2	-

Figure 10(a) shows that the neural network is well trained and gives accurate results in the nominal case ($\Gamma = 0$). Figure 10(b) shows how the prediction is affected by the input disturbance; first output gives acceptable results but second output prediction is far from real data.

The next section will show that the same phenomenon arises if the neural network is trained with a disturbance.

5.2 Offline Training with a Disturbance

A neural network offline trained with a disturbance cannot accurately predict the output if the disturbance varies or vanishes. To this end, a FNN is trained on the system presented in figure 9 with a sinusoidal input disturbance ($f = 1$Hz). Figure 11 presents the prediction (obtained from the method explained in figure 1) in the presence of the same disturbance, the prediction without any disturbance and the prediction with a sinusoidal input disturbance with $f = 10$Hz.

Figure 11 shows that the trained neural network is specific to the sinusoidal input disturbance with $f = 1$Hz as without any disturbance and for another frequency the prediction is incorrect for the second output and less accurate with regard to the first input.

Section 5.1 and 5.2 have shown that using the method presented in figure 1 leads to poor prediction results in the face of varying disturbance profiles. The

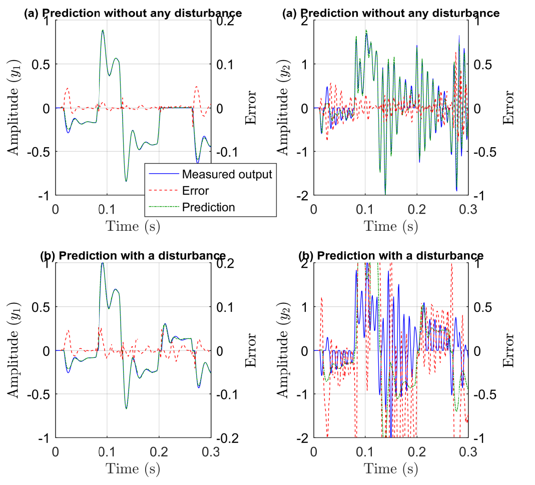

Figure 10: Influence of a disturbance on the prediction.

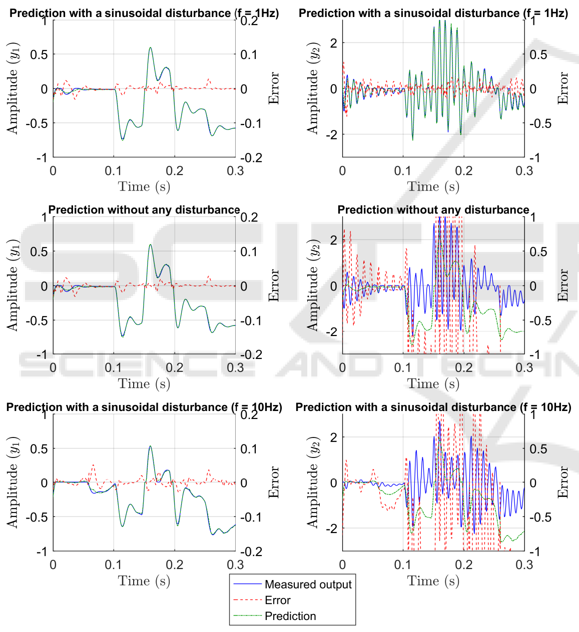

Figure 11: Influence of different disturbances on the prediction.

following part corresponds to the proposed method to enhance prediction against unknown disturbances.

5.3 Enhancing Prediction with the Proposed Method

5.3.1 Offline Training of the State-Space Neural Network

This section corresponds to the first step of the method described in figure 2. A SSNN with 50 neurons in the hidden layer is chosen. The system has been sampled at sampling time T_e and we have used 10000 samples for the training set and 10000 samples for the validation set. As explained previously, data have been obtained without any disturbances, that is $\Gamma = 0$ in figure 9, and have been normalized. The input training sequence is constrained to $[-1;1]$.

Figure 12 shows the results for training (several runs have been performed but only the best result in terms of cost function is presented) by presenting a simulation on a test set which is different from the training set. Regression, which corresponds to the plot of the output of the model as a function of the measured output, shows the training accuracy since it is close to the first bisector (for a perfect training the regression would have been exactly on the first bisector). It can be seen that the training is somewhat better for the first measure than it is for the second one. Using more neurons would lead to more accurate results.

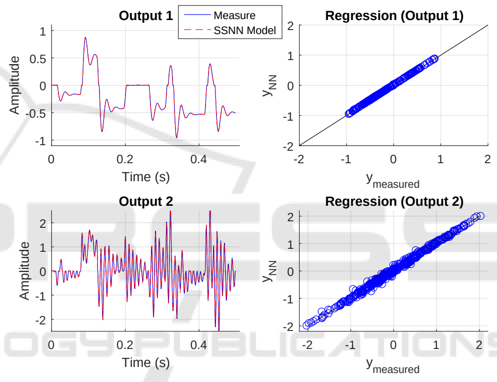

Figure 12: Learning results on a test scenario (different from the training scenario).

5.3.2 Extended Kalman Filter Implementation

The following part corresponds to the second step of the method described in figure 2. We implement the EKF by using a reasoning similar to the one done in section 3.4 by using the SSNN model.

Experiment results done using the system presented in figure 9 are presented in figure 13 for two different perturbations: the first one corresponds to a sinusoidal signal and the second one to a square signal.

The modelling error has a direct influence on the estimated disturbance. The result shows that the method is able to reconstruct the disturbance with a good accuracy.

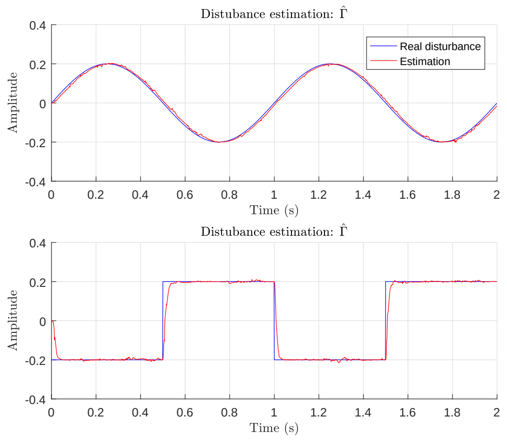

Figure 13: Estimated disturbance.

5.3.3 Implementation of the Feedback Compensator

The solution proposed in section 4 and figure 8 is now tested to see the robustness of the plant against different disturbances. It corresponds to the third step of figure 2. Three kinds of experiments have been done: the first one corresponds to a system subject to an additive sinusoidal disturbance input (Figure 6), the second one to a system subject to an internal sinusoidal disturbance for the first measure (Figure 14) and the last one corresponds to a parameter variation. Results are respectively presented in figure 15 (a), (b) and (c). For the case (c) the gain K_1 is multiplied by a factor 1.2 at time $t = 0.125$s in order to simulate an internal parameter variation. Ideally, the solid blue line and the dash-dot green line should be identical.

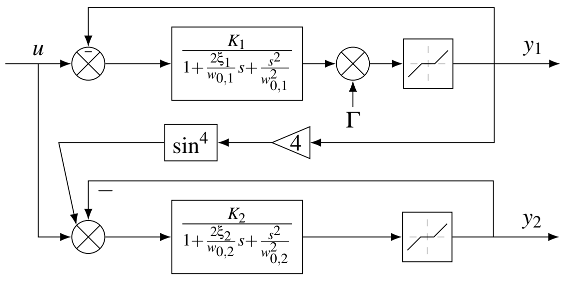
u
$-$
$\frac{K_1}{1+\frac{2\xi_1}{w_{0,1}}s+\frac{s^2}{w_{0,1}^2}}$
y_1
Γ
$\sin^4$
4
$-$
$\frac{K_2}{1+\frac{2\xi_2}{w_{0,2}}s+\frac{s^2}{w_{0,2}^2}}$
y_2

Figure 14: Internal disturbance.

A cost function is introduced in order to compare the invariance of the system as:

$$I_{v,p} = \frac{1}{n_s}\sum_{i=1}^{n_s}(y_{I,p} - y_p)^2, \qquad (14)$$

where the index p refers the output p, $y_{I,p}$ is the output of the disturbed system, corresponding to the dash red line in figure 15 or the output of the system using the feedback, corresponding to the dash-dot green line. Results are summed up in Table 2 where the case (a),(b) and (c) correspond to those presented in figure 15.

We can see from figure 15 (a-c) and Table 2 that the system is less sensitive to disturbances even if the disturbance applied to the plant is not an additive input disturbance as considered in the model used in the EKF. As shown with figure 13, the estimated disturbance also contains the model error thus, results presented in figure 15 can be improved if a better model is used, which means using more neurons or by retraining the neural network with other starting points.

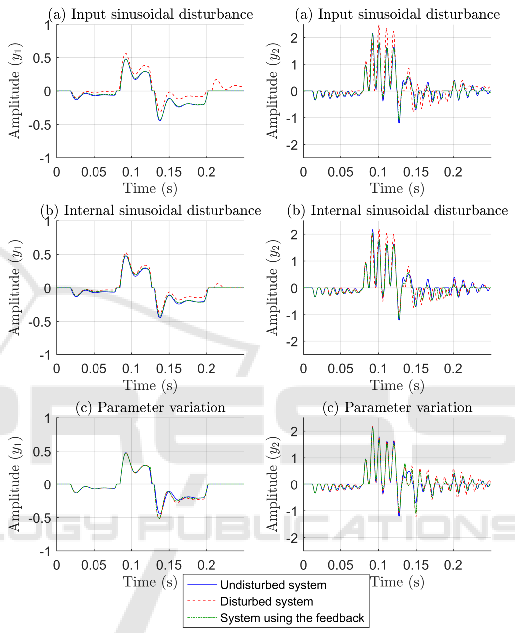

Figure 15: Invariance of the system using the feedback structure proposed in section 4.

Because the system is unvarying against disturbances, we can conclude that the global new system can be modeled by a FNN trained online as it has been done in (Bao et al., 2017). The predictor that arises from this FNN will be less sensitive to disturbances and thus be efficient.

Table 2: Performance comparison.

Case		$I_{v,1}$	$I_{v,2}$
(a)	Disturbed system	$7.8 * 10^{-3}$	$1.5 * 10^{-1}$
	System using the feedback	$1.0 * 10^{-4}$	$1.1 * 10^{-2}$
(b)	Disturbed system	$2.0 * 10^{-3}$	$5.2 * 10^{-2}$
	System using the feedback	$3.7 * 10^{-4}$	$2.8 * 10^{-2}$
(c)	Disturbed system	$8.1 * 10^{-4}$	$5.1 * 10^{-2}$
	System using the feedback	$6.3 * 10^{-4}$	$4.4 * 10^{-2}$

6 CONCLUSION

In this paper, a way to enhance neural network prediction against unknown disturbances has been presented, thanks to a feedback structure that uses the observed disturbance reconstructed by an extended Kalman filter based on a state-space neural network model. Numerical results obtained for a system with non-differentiable and differentiable nonlinearities have proven the interest in the proposed approach, exhibiting satisfactory results in terms of prediction errors and robustness against variations of the disturbance input profiles or parameter variations. These results have been obtained at the price of a slight increase in the predictor complexity, as the neural network used for the prediction for the global system, containing both the system and the observer, generally requires more neurons than a neural network predictor for the original system alone.

Future works will deal with improving learning methods for SSNN and combining this work with the Decoupled Extended Kalman Filter neural network learning method (Puskorius and Feldkamp, 1997) in order to get an adaptive filter. Other observer techniques will also be tested in order to achieve a fair comparison of the possible approaches. Finally, the estimated disturbance can be used to obtain a disturbance predictor in the case of a control law design.

REFERENCES

Alessandri, A. (2000). Design of sliding-mode observers and filters for nonlinear dynamic systems. In *Proceedings of the 39th IEEE Conference on Decision and Control*, volume 3, pages 2593–2598. IEEE.

Bao, X., Sun, Z., and Sharma, N. (2017). A recurrent neural network based MPC for a hybrid neuroprosthesis system. In *IEEE 56th Annual Conference on Decision and Control*, pages 4715–4720. IEEE.

Bullinger, E. and Allgöwer, F. (1997). An adaptive high-gain observer for nonlinear systems. In *36th IEEE Conference on Decision and Control*, pages 4348–4353.

Chen, W.-H., Ballance, D. J., Gawthrop, P. J., and O'Reilly, J. (2000). A nonlinear disturbance observer for robotic manipulators. *IEEE Transactions on industrial Electronics*, 47(4):932–938.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314.

De Jesus, O. and Hagan, M. T. (2007). Backpropagation algorithms for a broad class of dynamic networks. *IEEE Transactions on Neural Networks*, 18(1):14–27.

Diaconescu, E. (2008). The use of narx neural networks to predict chaotic time series. *Wseas Transactions on computer research*, 3(3):182–191.

Gavin, H. (2017). The levenberg-marquardt method for nonlinear least squares curve-fitting problems. *Department of Civil and Environmental Engineering, Duke University*.

Grossman, W. D. (1999). *Observers for discrete-time nonlinear systems*.

Hagan, M. T., Demuth, H. B., Beale, M. H., et al. (1996). *Neural network design*, volume 20. Pws Pub. Boston.

Haykin, S. (2004). *Kalman filtering and neural networks*, volume 47. John Wiley & Sons.

Hedjar, R. (2013). Adaptive neural network model predictive control. *International Journal of Innovative Computing, Information and Control*, 9(3):1245–1257.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45.

Keviczky, T. and Balas, G. J. (2006). Receding horizon control of an f-16 aircraft: A comparative study. *Control Engineering Practice*, 14(9):1023–1033.

Lakhal, A., Tlili, A., and Braiek, N. B. (2010). Neural network observer for nonlinear systems application to induction motors. *International Journal of Control and Automation*, 3(1):1–16.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441.

Pascanu, R., Mikolov, T., and Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318.

Puskorius, G. and Feldkamp, L. (1997). Extensions and enhancements of decoupled extended kalman filter training. In *International Conference on Neural Networks*, volume 3, pages 1879–1883. IEEE.

Talebi, H. A., Abdollahi, F., Patel, R. V., and Khorasani, K. (2010). Neural network-based state estimation schemes. In *Neural Network-Based State Estimation of Nonlinear Systems*, pages 15–35. Springer.

Terejanu, G. A. (2008). Extended kalman filter tutorial. *Department of Computer Science and Engineering, University at Buffalo*.

Vatankhah, B. and Farrokhi, M. (2017). Nonlinear model-predictive control with disturbance rejection property using adaptive neural networks. *Journal of the Franklin Institute*, 354(13):5201–5220.

Wan, E. A. and Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pages 153–158. IEEE.

Werbos, P. (1974). Beyond regression: New tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Williams, R. J. and Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280.

Yu, D. and Gomm, J. (2003). Implementation of neural network predictive control to a multivariable chemical reactor. *Control Engineering Practice*, 11(11):1315–1323.

Zamarreño, J. M. and Vega, P. (1998). State space neural network. properties and application. *Neural networks*, 11(6):1099–1112.