March, 2004

A Generalized Neuron Based Adaptive Power System Stabilizer

D. K. Chaturvedi, Dayalbagh Educational Institute
O. P. Malik, University of Calgary
P. K. Kalra

Available at: https://works.bepress.com/dk_chaturvedi/18/
Generalised neuron-based adaptive power system stabiliser


Abstract: Artificial neural networks (ANNs) can be used as intelligent controllers to control nonlinear, dynamic systems through learning, which can easily accommodate the nonlinearities and time dependencies. However, they require long training time and large numbers of neurons to deal with complex problems. To overcome these drawbacks, a generalised neuron (GN) has been developed that requires much smaller training data and shorter training time. Taking benefit of these characteristics of the GN, a new generalised neuron-based adaptive power system stabiliser (GNPSS) is proposed. The GNPSS consists of a GN as an identifier, which tracks the dynamics of the plant, and a GN as a controller to damp low-frequency oscillations. Results show that the proposed adaptive GNPSS can provide a consistently good dynamic performance of the system over a wide range of operating conditions.

1 Introduction

Use of a supplementary control signal in the excitation system and/or the governor system of a generating unit can provide extra damping for the system and thus improve the unit’s dynamic performance [1]. Power system stabilisers (PSSs) aid in maintaining power system stability and improving dynamic performance by providing a supplementary signal to the excitation system. This is an easy, economical and flexible way to improve power system stability. Over the past few decades, PSSs have been extensively studied and successfully used in the industry.

The commonly used PSS (CPSS) was first proposed in the 1950s and is based on a linear model of the power system at some operating point to damp the low frequency oscillations in the system. Linear control theory was employed as the design tool for the CPSS. After decades of theoretical studies and field experiments, this type of PSS has made a great contribution in enhancing the operating quality of the power system [2, 3].

To improve the power qualities in a large integrated power system, it is worthwhile looking into the possibility of using modern control techniques. The linear optimal control strategy is one possibility that has been proposed for supplementary excitation controllers [4]. Preciseness of the linear model to represent the actual system and the measurement of some variables are major obstacles to the application of the optimal controller in practice.

A more reasonable design of the PSS is based on adaptive control theory as it takes into consideration the nonlinear and stochastic characteristics of power systems [5, 6]. This type of stabiliser can adjust its parameters online according to the operating conditions. Many years of intensive studies have shown that the adaptive stabiliser can not only provide good damping over a wide operating range but also, more importantly, solve the co-ordination problem among stabilisers.

Power systems being dynamic systems, the response time of the controller is the key to a good closed-loop performance. Many adaptive control algorithms have been proposed in recent years. Generally speaking, the better the closed-loop system performance desired, the more complicated the control algorithm becomes, thus needing more online computation time to calculate the control signal.

More recently, an ANN (artificial neural network) and fuzzy set theoretic approach has been proposed for power system stabilisation problems. A number of papers have been published in the last decade. An illustrative list is given in [6–13]. Both techniques have their own advantages and disadvantages. The integration of these approaches can give improved results.

The common neuron model has been modified to obtain a generalised neuron (GN) model using fuzzy compensatory operators as aggregation operators to overcome the problems such as large number of neurons and layers required for complex function approximation, which not only affect the training time but also the fault-tolerant capabilities of the ANN [14]. Application of this GN as a PSS is described in this paper.

2 Generalised neuron model

The general structure of the common neuron is an aggregation function and its transformation through a filter. It is shown in the literature [15–17] that ANNs can be universal function approximators for given input–output data. The common neuron structure has summation as the aggregation function with sigmoidal, radial basis, tangent hyperbolic or linear limiters as the thresholding function, as shown in Fig. 1.

The aggregation operators used in the neurons are generally crisp. However, they overlook the fact that most of the processing in the neural networks is done with
incomplete information at hand. Thus, a generalised neuron model approach has been adopted that uses the fuzzy compensatory operators [18] that are partly sum and partly product, to take into account the vagueness involved.

2.1 Generalised neuron model

Use of the sigmoidal thresholding function and ordinary summation or product as aggregation functions in the existing models require more numbers of hidden neurons, which results in a long training time to cope with the non-linearities involved in real-life problems. To deal with these, the proposed model has both sigmoidal and Gaussian functions with weight sharing. The generalised neuron model has flexibility at both the aggregation and threshold function level to cope with the nonlinearity involved in the type of applications dealt with, as shown in Fig. 2. The neuron has both $\Sigma$ and $\pi$ aggregation functions. The $\Sigma$ aggregation function has been used with the sigmoidal characteristic function ($f_1$) while the $\pi$ aggregation function has been used with the Gaussian function ($f_2$) as a characteristic function.

$$O_{\Sigma} = f_1(s_{net}) = \frac{1}{1 + e^{-\lambda s_{\Sigma} s_{net}}} \quad (1)$$

where $s_{net} = \Sigma W_{\Sigma} X_i + X_{o\Sigma}$ and $\lambda s =$ gain scale factor for the $\Sigma$ part.

The output of the $\pi$ part with Gaussian characteristic function of the generalised neuron is:

$$O_{\Pi} = f_2(p_{\pi_{net}}) = e^{-\lambda p_{\Pi} p_{\pi_{\Pi}}^2} \quad (2)$$

where $p_{\pi_{net}} = \prod W_{\Pi} X_i + X_{o\Pi}$ and $\lambda p =$ gain scale factor for the $\Pi$ part.

The final output of the neuron is a function of the two outputs $O_{\Sigma}$ and $O_{\Pi}$ with the weights $W$ and $(1-W)$, respectively:

$$O_{pk} = O_{\Pi} \cdot (1 - W) + O_{\Sigma} \cdot W \quad (3)$$

The neuron model described above is known as the summation type compensatory neuron model, since the outputs of the sigmoidal and Gaussian functions are summed up. Similarly, the product type compensatory neuron models may also be developed. It is found that, in most of the applications, the summation type compensatory neuron model works well [18] and is the one used for the development of the adaptive GNPSS.

2.2 Advantages of GN

Comparison of GN with a multi-layer feedforward ANN (FFNN) is shown in Table 1. The weights are determined through training. Hence, by reducing the number of unknown weights, training time as well as minimum number of patterns required for training can be reduced. In the proposed GN the training time is significantly reduced by optimally selecting the number of aggregation functions and thresholding functions.

<table>
<thead>
<tr>
<th>Table 1: Comparison of GN with one hidden layer FFNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of inputs</td>
</tr>
<tr>
<td>four angular speeds+ three past control actions</td>
</tr>
<tr>
<td>Hidden layers</td>
</tr>
<tr>
<td>Number of neurons</td>
</tr>
<tr>
<td>Number of weights</td>
</tr>
<tr>
<td>Number of biases</td>
</tr>
</tbody>
</table>

In this paper, summation and product are used at the aggregation level for simplification, but one can also take other fuzzy aggregation operators, such as max, min or compensatory operators. Similarly, the thresholding functions are only sigmoidal and Gaussian function for the proposed GN, but other functions, like straight line, sine, cosine etc., can also be used. The weighting factor may be associated with each aggregation function and thresholding function. During training, these weights change and decide the best functions for the GN.

2.3 Learning algorithm of a generalised neuron

The following steps are involved in the training of a generalised neuron:

**Step 1:** Calculate output of the generalised neuron model as given in (1)-(3).

**Step 2:** After calculating the output of the generalised neuron in the forward pass, as in the feedforward neural network, it is compared with the desired output to find the error. Using the back-propagation algorithm the GN is trained to minimise the error. In this step, the output of the single flexible generalised neuron is compared with the desired output to get the error for the $i$th set of inputs:

$$\text{error } E_i = (Y_i - O_i) \quad (4)$$

Then the sum-squared error for convergence of all the patterns is:

$$E_p = \frac{1}{2} \sum E_i^2 \quad (5)$$

A multiplication factor of 0.5 has been taken to simplify the calculations.

**Step 3:** Reverse pass for modifying the connection strength.
(a) Weight associated with the $\Sigma_1$ and $\Sigma_2$ part of the generalised neuron is:

$$W(k) = W(k-1) + AW$$

(6)

where $AW = \eta \delta_i (O_{\Sigma} - O_{\Sigma}) \dot{x} + xW(k-1)$ and $\delta_k = \sum (Y_i - O_i)$. 

(b) Weights associated with the inputs of the $\Sigma_1$ part of the generalised neuron are:

$$W_{\Sigma 1}(k) = W_{\Sigma 1}(k-1) + AW_{\Sigma 1}$$

(7)

where $AW_{\Sigma 1} = \eta \delta_{\Sigma 1} \dot{x} + xW_{\Sigma 1}(k-1)$, and $\delta_{\Sigma 1} = \sum \delta_k W/(1 - O_{\Sigma}) + O_{\Sigma}$.

(c) Weights associated with the input of the $\pi$-part of the generalised neuron are:

$$W_{\Pi}(k) = W_{\Pi}(k-1) + AW_{\Pi}$$

(8)

where $AW_{\Pi} = \eta \delta_{\Pi} \dot{x} + xW_{\Pi}(k-1)$, $\delta_{\Pi} = \sum \delta_k (1 - W)^* + (-2 \times p_{\text{net}}) + O_{\Pi}$, $x =$ momentum factor for better convergence and $\eta =$ learning rate.

The range of these factors is from 0 to 1 and is determined by experience.

3 Conventional PSS

The most commonly used PSS, referred to as the CPSS, is a fixed parameter device. It is based on the use of a transfer function designed using classical control theory. It uses a lead/lag compensation network to compensate for the phase shift caused by the low-frequency oscillation of the system during perturbation. By appropriately tuning the parameters of a lead/lag compensation network, it is possible to make a system have the desired damping ability.

However, power systems are highly nonlinear systems. The linearised system models used to design conventional power system stabilisers are valid only at the operating point that is used to linearise the system. A practical CPSS with the shaft speed input may take the form as mentioned in the Appendix. As a fixed parameter controller, CPSS cannot provide optimal performance under the very wide operating range of the power system. The main limitation of adaptive control is that it takes a large amount of computing time for online parameter identification using conventional techniques.

To overcome this problem, researchers have proposed the use of ANNs for system identification. As mentioned in Section 1, the ANN has drawbacks in terms of selection of suitable ANN structure, large training time and the amount of training data required. In this work, a GN is used for system identification. The number of connections is small in the GN as compared to the ANN and hence less time and data are required to update the weights online.

4.1 GN identifier

The problem of identification includes setting up a suitably parameterised identification model and adjusting the parameters of the model to optimise a performance function based on the error between the plant and the identification model outputs. A schematic diagram of the GN based plant identification using forward modelling is shown in Fig. 3. A GN identifier is placed in parallel with the system and has the following inputs:

$$X_i(t) = [\omega_{\text{vector}}, u_{\text{vector}}]$$

(9)

where $\omega_{\text{vector}} = [\omega(t), \omega(t-2^\ast T), \omega(t-3^\ast T)]$, $u_{\text{vector}} = [u(t-T), u(t-2^\ast T), u(t-3^\ast T)]$ and $T$ is the sampling period.

The dynamics of the change in angular speed of the synchronous generator can be viewed as a nonlinear mapping:

$$\omega(t + T) = f_i(X_i(t))$$

(10)

Therefore, the GN identifier for the plant can be represented by a nonlinear function $F_i$:

$$\omega_i(t + T) = F_i(X_i(t), W_i(t))$$

(11)

where, $W_i(t)$ is the matrix of the GN identifier weights at time instant $t$. 

4.2 Training of GN identifier

Training of an ANN is a major exercise, because it depends on various factors, such as the availability of sufficient and accurate training data, suitable training algorithm, number of neurons in the ANN, number of ANN layers and so on. The GN identifier with only one neuron is able to cope with the problem complexity, as selection of the number of neurons and layers is not required. The training of the proposed GN identifier has two steps: offline training and online update. In offline training, the GN identifier is trained for a wide range of operating conditions, i.e. output ranging from 0.1 p.u. to 1.0 p.u. and the power factor ranging from 0.7 lag to 0.8 lead. Similarly, a variety of disturbances are also included in the training, like change in reference voltage, governor input torque variation, one transmission line outage and three phase fault on one circuit of the double circuit transmission line. Offline training data for the GN identifier has been acquired from the system controlled by the CPSS, although any suitable controller can be used.

The error between the system and the GN identifier output at a unit delay is used as the GN identifier training signal, which is also called the performance index of the GN identifier:

\[ J_i(t) = \frac{1}{2} (\epsilon_i(t) - \omega_i(t)) \]  

(12)

The weights of the GN identifier are updated as:

\[ W_i(t) = W_i(t - T) + \Delta W_i(t) \]

where \( \Delta W_i(t) \), the change in weight depending on the instantaneous gradient, is calculated by:

\[ \Delta W_i(t) = -\eta_i J_i(t + T) \frac{\partial J_i(t)}{\partial W_i(t)} + \alpha \Delta W_i(t - T) \]  

(13)

where \( \eta_i \) = learning rate for the GN identifier and \( \alpha \) = momentum factor for the GN identifier. The offline training is performed with 0.1 learning rate and 0.4 momentum factor.

After offline training is finished, i.e. the average error between the plant and the GN identifier outputs converges to a small value, and the GN identifier represents the plant characteristics reasonably well,

\[ \epsilon_i(t + T) = f_i(X_i(t)) \approx \omega_i(t + T) = F_i(X_i(t), W_i(t)) \]

the proposed GN identifier will be connected to the power system for online update of weights. The learning rate and momentum factor are very crucial factors in online training and greatly affect the performance of the GN identifier. Hence, the values of these factors are considered very carefully for online training. Figure 4 shows the online GN identifier response under different learning rates and momentum factors in comparison with the actual plant response.

The online performance of the adaptive and non-adaptive GN identifier at 30 ms sampling time is shown in Fig. 5.

4.3 GN controller

Figure 6 shows the schematic diagram of the GN controller. The plant consists of a single machine connected to an infinite bus through a double-circuit transmission line. The last four values of the angular speed of the synchronous machine, sensed at fixed time intervals (30 ms), are used as input to the GN controller. Besides the angular speed, the past three control actions are also given to the GN controller as inputs. These inputs are normalised in the range 0.1–0.9. The range of normalisation is chosen from 0.1 to 0.9 so that the inputs are well distributed in this range rather concentrating near a particular value. The output of the GN controller is the control signal \( u(t) \) in the normalised range. The actual value of \( u(t) \) is obtained from the normalised value.

\[ u(t) = F_i(X_i(t), W_i(t)) \]  

(14)
where, $W_c(t)$ is the matrix of neural controller weights at time instant $t$.

$a(t)$ is denormalised to get the actual control action and then sent to the plant and the GN identifier simultaneously.

### 4.4 Training of GN controller

Training of the proposed GN controller also has two steps: offline training and online update. In offline training, the GN controller is trained for a wide range of operating conditions and a variety of disturbances. Offline training data for the GN controller has been acquired from the system controlled by the CPSS, which is tuned for each operating condition.

The performance index of the neural controller is:

$$J_c(t) = \frac{1}{2}[\omega_d(t + T) - \omega_d(t + T)]^2$$

(15)

where $\omega_d(t+T)$ is the desired plant output at time instant $(t + T)$; in this study it is set to zero. $T$ is the sampling time (30 ms).

The weights of the GN controller are updated as:

$$W_c(t) = W_c(t - T) + \Delta W_c(t)$$

(16)

where $\Delta W_c(t)$ is change in weight depending on the instantaneous gradient and is calculated by:

$$\Delta W_c(t) = -\eta_c \omega_{identifier}(t + T) \frac{\partial J_c(t)}{\partial u(t)} + \alpha_c \Delta W_c(t - T)$$

(17)

where $\eta_c$ = learning rate for the GN controller and $\alpha_c$ = momentum factor for the GN controller.

Offline training is started with small random weights ($\pm 0.01$) and then updated with relatively high learning rate and momentum factor ($\eta_c = 0.1$ and $\alpha_c = 0.4$).

After offline training is finished, the proposed controller is connected to the power system for online update with smaller learning rate ($\eta_c = 0.001$) and momentum factor ($\alpha_c = 0.01$). In online training of the GN controller, the expected error is calculated from the predicted output $\omega_c(t+T)$ of the GN identifier at one step ahead. The expected error is then used to update the weights online. Parameters of the GN identifier and controller are adjusted every sampling period. This allows the controller to track the dynamic variations of the power system and provide the best control action.

### 5 Simulation results

The performance of the proposed GN-based adaptive PSS has been investigated on a synchronous generator connected to a constant voltage bus through two parallel transmission lines, as shown in Fig. 7. Parameters of the machine, exciter, AVR and the CPSS have been given in [26].

#### 5.1 CPSS parameter tuning

With the generator operating at $P = 0.9$ p.u. and $Q = 0.4$ p.u. lag, a 100 ms three phase to ground fault is applied at 0.5 s at the generator bus. The CPSS is carefully tuned under the above conditions to yield the best performance and its parameters are kept fixed for all studies.

#### 5.2 Performance under three-phase to ground fault

The results are compared for the GN based adaptive PSS and the conventional PSS for a 100 ms three-phase to ground fault at the generator bus under different operating conditions. The performance of the GN based PSS, GN based adaptive PSS and the CPSS is shown in Fig. 8 under three-phase to ground fault at $P = 0.7$ p.u. and $Q = 0.3$ p.u. (lag).

#### 5.3 Performance with one line removed

Performance under different operating conditions such as $P = 0.9$ p.u. and $Q = 0.4$ p.u. lag, when one line is removed from the system with two parallel transmission lines operating initially, is shown in Fig. 9. It can be seen that the GNPPS damps out the oscillations very effectively. Because the GNPPS is trained for a wide range of operating conditions, it is able to adjust the control output to that suitable for the working conditions.

### 6 Conclusions

Performance of an adaptive PSS based on a generalised neuron has been investigated in a single machine, infinite bus power system. The GN-based adaptive PSS is not designed for a fixed operating point. It is first trained offline for a wide range of generator operating conditions.
Parameters (i.e. the weights of GN) of the controller are updated online. With the learning ability through online update of weights, the GN-based PSS can track the changes in operating conditions. In this control architecture, no reference model is needed since the GN identifier tracks the system output and predicts the output one step ahead. The predicted output of the GN identifier helps in tuning the GN controller weights online.

Because of its adaptation capability, the adaptive GNPS can incorporate the nonlinearities involved in the system. Because it has a much smaller number of weights than the common multilayer feedforward ANN, the training data as well as training time required are drastically reduced.

Studies described in the paper show that the performance of the GN-based adaptive PSS can provide very good performance over a wide range of operating conditions.

7 Acknowledgment

This work was supported by the Dept. of Science and Technology, Govt. of India, New Delhi, India, under the BOYSCAST fellowship scheme to the first author.

8 References


9 Appendix

The generating unit is modelled by seven first-order nonlinear differential equations. Parameters for generator, exciter, AVR, governor and CPSS used in the simulation studies of the single machine infinite bus system are given in [6].