


Excitation Control of Synchronous Generator Via Neural Network Based Controllers

Ekhlas M. Thajeel 

Electrical Engineering Department, University of Technology, Baghdad
Email:eklas mahawsh@yahoo.com

Received on: 8 /6 / 2011& Accepted on: 1/ 12 / 2011

ABSTRACT

Modern power systems are complex and non-linear and their operating can vary over a wide range. This paper presents a linear mathematical model of the synchronous generator to control the excitation system based on Neural Network to simulate an Automatic Voltage Regulator. The voltage regulator is used to modify terminal voltage for the purpose of tracking a reference voltage and comparative with PID controller. ANN (NARMA-L2) system is proposed as an effective controller model to achieve the desired enhancement. This model after training can be called as (Identifier).The proposed technique is evaluated on a single machine infinite bus under different operating conditions (no-load and full load condition) by using MATLAB simulink software.

Keywords: Synchronous generator, neural network based control, automatic voltage regulator (AVR), PID controller.

السيطرة على تحفيز المولد التزامني باستخدام الشبكات العصبية الاصطناعية الذكية

الخلاصة

تعتبر أنظمة القدرة الحديثة لأخطية وعالية التعقيد وان شروط عملها يمكن أن يتغير ضمن مجال واسع. هذا البحث يقدم نمودجا رياضيا خطيا لمولد متزامن للسيطرة على نظام الاثارة معتمدا الشبكات العصبية لمحاكاة منظم الفولطية الالي. ان منظم الفولطية يستعمل لتعديل فولطية الاقطاب لغرض تتبع فولطية المرجع ومقارنتها مع المسيطر PID (التناسي - التكاملي - التفاضلي). لقد تم اقتراح الشبكات العصبية الاصطناعية من نوع (NARMA-L2) كنموذج فعال و مسيطر و الذي يمكن ان يدرج بشكل دقيق لكي يكون اخراج هذا المسيطر هو الاخراج الحقيقي للمنظومة (المطابق). ان التقنية المقترحة طبقت على مولدة احادية مرتبطة بخط نقل لا نهائي (infinite bus) وتحت ظروف تشغيل مختلفة (حالة الحمل واللاحمل) باستخدام برنامج (MATLAB).

INTRODUCTION

POWER-SYSTEM control essentially requires a continuous balance between electrical power generation and a varying load demand, while maintaining system frequency, voltage levels, and power grid security. However, generator and grid disturbances can vary between minor and large imbalances in mechanical and electrical generated power, while the characteristics of a power system change significantly between heavy and light loading conditions, with varying numbers of generator units and transmission lines in operation at different times. The result is a highly complex and nonlinear dynamic electric power grid with many operational levels made up of a wide range of energy sources with many interaction points. [1]

** In most modern systems the automatic voltage regulator (AVR) is a controller that senses the generator output voltage (and sometimes the current) then initiates corrective action by changing the exciter control in the desired direction. The speed of the AVR is of great interest in studying stability. Because of the high inductance in the generator field winding, it is difficult to make rapid changes in field current. This introduces a considerable lag in the control function and is one of the major obstacles to be overcome in designing a regulating system. [2]

ANNs are good at identifying a nonlinear system and then controlling it, and they are suitable for multi-variable applications, where they can identify the interactions between the inputs and outputs. This removes the need for an accurate model of the power system. It has been shown that a multi-layer feedforward neural network using deviation signals as its inputs can identify the complex and nonlinear dynamics of a single-machine infinite-bus system (SMIB) system with sufficient accuracy to be used to design a generic controller, which yields optimal dynamic system response irrespective of the generator load and system configurations.[3,4]

SIMULINK as a software tool has been designed for solving nonlinear differential equation in either state space or block diagram form .It can be called from MATLAB using, often, a single command.

The results of the SIMULINK program can be automatically passed back to MATLAB for further analysis .Simulations in SIMULINK run three to ten times faster than similar program in MATLAB because SIMULINK program are compiled. Due to these advantages offered by MATLAB/SIMULINK, this environment was chosen for these studies.

MATHEMATICAL MODELING AND SIMULATION OF SYNCHRONOUS MACHINE

For modeling purpose, the theoretical analysis of single machine infinite-bus system is considered to simulate the terminal voltage of excitation system.

A single machine infinite bus power system is shown in fig.1.This system consists of a generator connected to the infinite bus through a transmission line. An infinite bus has a fixed voltage and frequency.

The mathematical description of the synchronous machine is the same as all types of AC machines, which have two main problems: first, is the complex 3-phase represented differential equations, and second, is the time varying mutual inductance between stator and rotor winding, through the dynamic response of the SG. Simply, the first problem can be solved by using axis transformation to transfer the 3-phase

parameters and quantities (like: voltage, current, flux...) to 2-phase parameters, which called Park's transformation or, Park model of SG. In which all stator quantities are transferred from phase a, b and c into equivalent dq axis new variables. Equations (1 to 4) show the approximate Park's transformation by neglecting the zero sequence parameters: [6, 7]

$$P(\theta) = \frac{2}{3} \begin{bmatrix} \cos(\theta) & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta + \frac{2\pi}{3}) \\ \sin(\theta) & \sin(\theta - \frac{2\pi}{3}) & \sin(\theta + \frac{2\pi}{3}) \end{bmatrix} \quad (1)$$

SO ,

$$\begin{bmatrix} V_q^s \\ V_d^s \end{bmatrix} = |P(\theta)| \cdot \begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} \quad \dots\dots (2)$$

$$\begin{bmatrix} I_q^s \\ I_d^s \end{bmatrix} = |P(\theta)| \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \quad \dots\dots (3)$$

$$\begin{bmatrix} \Psi_q^s \\ \Psi_d^s \end{bmatrix} = |P(\theta)| \cdot \begin{bmatrix} \Psi_a \\ \Psi_b \\ \Psi_{i_c} \end{bmatrix} \quad \dots\dots (4)$$

The time varying problem can be solved by using the synchronously rotating reference frame model, in which all stator variables associated with fictitious winding rotating with the rotor at synchronous speed [7]. The transformation equations are:

$$F(\theta_e) = \begin{bmatrix} \cos(\theta_e) & -\sin(\theta_e) \\ \sin(\theta_e) & \cos(\theta_e) \end{bmatrix} \quad \dots (5)$$

$$\begin{bmatrix} v_q^e \\ v_d^e \end{bmatrix} = |F(\theta_e)| \cdot \begin{bmatrix} v_q^s \\ v_d^s \end{bmatrix} \quad \dots (6)$$

Figure (2) shows the synchronous generator stator and rotor windings in the dq-axis model, it's obviously that the effect of the field winding appears only in the d-axis, whereas the effect of the damper winding is equivalent to the rotor cage winding of an induction motor, which appears in both dq-axis circuits.

Therefore, the synchronously rotating reference frame equivalent circuits of the SG in d^e-q^e axes can be shown in fig (3). Equations (7 to 18) show stator and rotor circuits equations in d^e-q^e axes:

■ *stator equations:* [5, 6]

$$V_{qs}^e = -I_{qs}R_s - \omega_e \Psi_{ds} - \frac{d\Psi_{qs}}{dt} \quad \dots (7)$$

$$V_{ds}^e = -I_{ds}R_s + \omega_e \Psi_{qs} - \frac{d\Psi_{ds}}{dt} \quad \dots (8)$$

■ *rotor equations:* [6, 7]

$$0 = I_{qr}R_r - \frac{d\Psi_{qr}}{dt} \quad \dots (9)$$

$$0 = I_{dr}R_r - \frac{d\Psi_{dr}}{dt} \quad \dots (10)$$

$$V_f = I_fR_f + \frac{d\Psi_f}{dt} \quad \dots (11)$$

Where all rotor parameters are referred to stator circuit and the mutual and self inductance of air gap (main) flux linkage are identical to L_{qm} and L_{dm} rotor to stator reduction.

$$\Psi_{qs} = L_{ls}I_{qs} + L_{qm}(I_{qs} + I_{qr}) \quad \dots (12)$$

$$\Psi_{ds} = L_{ls}I_{ds} + L_{dm}(I_{ds} + I_{dr} + I_f) \quad \dots (13)$$

$$\Psi_f = L_{lf}I_f + L_{dm}(I_{ds} + I_{dr} + I_f) \quad \dots (14)$$

$$\Psi_{qr} = L_{lr}I_{qr} + L_{qm}(I_{qs} + I_{qr}) \quad \dots (15)$$

$$\Psi_{dr} = L_{lr}I_{dr} + L_{dm}(I_{ds} + I_{dr} + I_f) \quad \dots (16)$$

■ *The electromagnetic torque:* [6, 7]

$$T_e = -\frac{3}{2}P_1(\Psi_{ds}I_{qs} - \Psi_{qs}I_{ds}) \quad \dots (17)$$

■ *The motion equation:* [6, 7]

$$T_{shaft} - T_e = \frac{J}{P_1} \frac{d\omega_r}{dt} \quad \dots (18)$$

The overall system simulation is shown in figure(4), and the inside subsystem1, subsystem2 are shown in figure (5) and figure(6). The output performance of the system under no-load and full load condition can be shown in figure(7) and figure(8).

PID CONTROLLER

An adaptive proportional-integral-derivative (PID) controller is the most powerful method to regulate terminal voltage of the S.G used with different operating loads and power factors [1]. The operating principle is to sense the terminal phase voltage and use it as a feedback signal through the PID controller to generate command setting signal which adapts the excited voltage of the S.G, which gives an acceptable behavior of the system for different conditions.

PID control is one of the earliest control strategies. It has been widely used in the industrial control fields. Its widespread acceptability can be recognized by:

the familiarity with which it is perceived amongst researchers and practitioners within the control community, simple structure and effectiveness of algorithm, relative ease and high speed of adjustment with minimal down-time and wide range of applications where its reliability and robustness produces excellent control performances. However, successful applications of PID controllers require the satisfactory tuning of three parameters - which are proportional gain (Kp), integral time constant (KI), and derivative time constant (KD) - according to the dynamics of the process. A proportional controller (Kp) will have the effect of reducing the rise time and will reduce, but never eliminate, the steady-state error. An integral control (Ki) will have the effect of eliminating the steady-state error, but it may make the transient response

worse . A derivative control (Kd) will have the effect of increasing the stability of the system , reducing the overshoot , and improving the transient response . Traditionally, these parameters are determined by a trial and error approach. Manual tuning of PID controller is very tedious, time consuming and laborious to implement, especially where the performance of the controller mainly depends on the experiences of design engineers. In recent years, many tuning methods have been proposed to reduce the time consumption on determining the three controller parameters. The most well known tuning method is the Ziegler-Nichols tuning formula; it determines suitable parameters by observing a gain and a frequency on which the plant becomes oscillatory. [9]

NEURAL CONTROLLER

The model of an artificial neuron that closely matches a biological neuron is given by an op-amp summer like configuration shown in figure (9).

Where $x_1, x_2, x_3...$ are input signals, each of the input signal flows through a gain called synaptic weight. The weight can be positive (excitatory) or negative (inhibitory) corresponding, respectively, to acceleration or inhibition [10].

The summing nodes accumulate all the input weighted signals and then pass to the output through the transfer function which is usually nonlinear. The transfer function can be step or threshold type, signum type, or linear threshold type. The transfer function can also be nonlinear continuously varying type, such as sigmoid, inverse-tan, hyperbolic, or Gaussian type. The sigmoidal transfer function is most commonly used, and it is given by

$$Y = \frac{1}{1+e^{-\alpha x}} \quad \dots (19)$$

Where α is the coefficient or gain which adjusts the slope of the function. With high gain, this function approaches a step function. The sigmoidal function is nonlinear, monotonic, differentiable, and has the largest incremental gain at zero signal, and these properties are of particular interest.

In general, neural networks can be classified as feedforward and feedback types depending on the interconnection of the neurons. At present, the majority of the problems use feedforward architecture, and it is of direct relevance to power electronics and motion control applications.

Figure (10) shows the structure of a feedforward multilayer network with two input and two output signals. The topology is based on Perceptron which was proposed by Rosenblatt in 1958. The circles represent neurons and the dots in the connections represent the weights.

The network has three layers, defined as input layer (a), hidden layer (b), and output layer (c). The hidden layer functions as a connection between the input and the output layers. The input and output layers have neurons equal to the respective number of signals. The input layer neurons do not have transfer functions, but there are scale factors, as shown, to normalize the input signals. The number of hidden layers and the number of neurons in each hidden layer depend on the network design considerations. The input layer transmits the signals to the hidden layer, and the hidden layer, in turn,

transmits the signals to the output layer, as shown. The network can be fully connected or partially connected.

back Propagation Training

Back-Propagation training algorithm is most commonly used in a feedforward neural networks as mentioned before.

For this reason, a feedforward network is often defined as “back-prop” network. Figure (11) shows the principle of back propagation training.

In the beginning, the network is assigned random positive and negative weights. For a given input signal pattern, step by step calculations are made in the forward direction to derive the output pattern. A cost functional given by the squared difference between the net output and the desired net output for the set of input patterns is generated and this is minimized by gradient descent method altering the weights one at a time starting from the output layer. The equations for the output of a single processing unit are given as:

$$Net_j^p = \sum_{i=1}^N W_{ij} X_i \quad \dots(20)$$

$$Y_j^p = f_j(Net_j^p) \quad \dots (21)$$

Where j is the processing unit under consideration, p is the input pattern number X_i is the output of the i^{th} neuron connected to the j^{th} neuron, W_{ij} is the connection weight between the i^{th} and j^{th} neurons. Net_j^p is the output of the summing node, i.e., the j^{th} neuron activation signal, N is the number of the neurons feeding the j^{th} neuron, f_j is the nonlinear differentiable transfer function (usually sigmoid), and Y_j^p is the output of the corresponding neuron. For the input pattern p , the squared output error for all the output layer neurons of the network is given as

$$E_p = \frac{1}{2} (d^p - y^p)^2 = \frac{1}{2} \sum_{j=1}^S (d_j^p - y_j^p)^2 \quad \dots (22)$$

Where d_j^p is the desired output of the j^{th} neuron in the output layer y_j^p , is the corresponding actual output, S is the dimension of the output vector y^p is the actual net output vector, and d^p is the corresponding desired output vector.

The total squared error E for the set of P patterns is then given by

$$E = \frac{1}{2} \sum_{p=1}^P E_p = \frac{1}{2} \sum_{p=1}^P \sum_{j=1}^S (d_j^p - y_j^p)^2 \quad \dots (23)$$

The weights are changed to reduce the cost functional E in a minimum value by gradient descent method, as mentioned. The weight update equation is then given as:

$$W_{ij}(t+1) = W_{ij}(t) - \eta \left[\frac{\delta E_p}{\delta W_{ij}(t)} \right] \quad \dots (24)$$

Where η is the learning rate, $W_{ij}(t+1)$ is the new weight and $W_{ij}(t)$ is the old weight. The weights are updated for all the P training patterns. Sufficient learning is achieved when the total error E summed over the patterns falls below a prescribed

threshold value. The iterative process propagates the error back-propagation [10, 11, 12].

NARMA –L2

In this work, the NARMA –L2 architecture is applied with the aid of the Neural Network Toolbox of MATLAB software. The identification can be summarized by the following steps :

a- The first step in using feedback linearization (or NARMA-L2 control) is to identify the system to be controlled.

Neural network is trained to represent the forward dynamics of the system. One standard model that has been used to represent general discrete-time nonlinear systems is the NARMA-L2 model [13]:

$$y(k+d) = N[y(k),y(k-1),\dots, y(k-n+1), u(k),u(k-1),\dots,u(k-n+1)] \dots\dots(25)$$

where $u(k)$ is the system input, and $y(k)$ is the system output and k, d, n are integral number and N is the function of the output system after identification.

b- The next step is to make the output system follows some reference trajectory by developing a nonlinear controller of the form:

$$y(k+d) = yr(k+d) \dots\dots (26)$$

$$u(k) = G[y(k),y(k-1),\dots, y(k-n+1), yr(k+d),u(k-1),\dots,u(k-m+1)] \dots\dots (27)$$

The problem with using this controller is: Training neural network to minimize mean square error, needs to use dynamic back propagation which quite slow [14].

One solution is to use approximate models to represent the system. The controller used in this section is based on

the NARMA-L2 approximate model:

$$\hat{y}(k+d) = f[y(k),y(k-1),\dots,y(k-n+1),u(k-1),\dots,u(k-m+1)] + g[y(k),y(k-1),\dots,y(k-n+1),u(k-1),\dots,u(k-m+1)]u(k) \dots\dots (28)$$

Where the next controller input is not contained inside the nonlinearity. The advantage of this form is that controlled

input make the system output follows the reference equation(16). The resulting controller is:

$$u(k) = \frac{y_r(k+1) - f[y(k),y(k-1),\dots,y(k-n+1),u(k-n+1)]}{g[y(k),\dots,y(k-n+1),u(k-n+1)]} \dots\dots (29)$$

Using this equation directly can cause realization problems, because must determine the control input based on the output at the same time, i.e:

$$y(k+d)=f[y(k),y(k-1),\dots,y(k-n+1),u(k),u(k-1),\dots,u(k-n+1)]+g[y(k),\dots,y(k-n+1),u(k),\dots,u(k-n+1)]u(k+1) \quad \dots\dots (30)$$

Figure (8) is referred to block diagram of the proposed AVR system synchronous generator with NARMA-L2 controller.

SIMULATION RESULT

The model is inserted in the Simulink diagram and run firstly for the case with PID controller to calculate values of overshooting and settling time from the output response. Parameters used in the simulation studies are given below: Machine Parameters:

3-phase,
380V,5KVA,50HZ, $L_s=13.6\text{mH}$, $L_f=33.4\text{mH}$, $R_s=1.2\Omega$, $R_f=50\Omega$,
Excitation voltage=110V,
Rotor inertia=0.1Kg/m².

The simulation model for the AVR of the synchronous generator for load change with PID controller is shown in the figure (12). It is very interesting to investigate the effects of PID controller's parameters K_p, K_i, and K_d on the terminal voltage response that exist in the excitation system. Tuning the PID controller by setting the proportional gain K_p to 10, K_i to 8, and K_d to 0.001.

The response for terminal voltage and current are shown in figure (13) and figure(14), we note that the overshoot of the terminal voltage is (0.14), the rise time is (0.8sec) and the settling time is (4.3sec).

The field voltage and current can be shown as in Fig(15) and Fig(16). Figure (17) illustrates simulation system of SG with ANN. The controlling steps and output response is discussed in the following. The graphs shown in figure (18) and figure (19) show the performance results of the ANN and the response for the terminal voltage and current, we note that the overshoot is approximately zero, the rise time and the settling time is (0.8sec), we can see the field voltage and current as in Fig(20) and fig(21). Fig(22) show the comparative of the terminal voltage (V_t) step response between PID & ANN.

CONCLUSIONS

A proportional-integral-derivative (PID) and ANN controller for a synchronous generator is presented in this paper. Simulation studies for a single-machine power system environment are presented to demonstrate the effectiveness of using the PID controller, we compared PID controller result that with ANN to show the performance of this controller. The ANN is more effective than the PID. The improved damping performance by the neurocontrollers allows the generator to be operated closer to its stability limit during steady state, and still remain stable after severe disturbances.

7. REFERENCES

- [1] Adkins, B. and Harley, R. G., The General Theory of Alternating Current Machines. London, U.K.: Chapman and Hall, 1975.

-
- [2] Anderson ,G., "Dynamics and Control of Electric Power Systems" , ETH Zurich press, March 2003
- [3]Hunt, KJ., Sbarbaro, D., Zbikowski, R, Gawthrop, PJ, "Neural networks for control systems – a survey", *Automatica*, Vol 28, No 6, pp 1083 – 1112.31992,.
- [4] Venayagamoorthy, GK, Harley RG, "A continually online trained artificial neural network identifier for a turbogenerator", accepted for publication in the Proceedings of IEEE International Electric Machines and Drives Conference IEMDC' 99, Seattle, USA, 9 – 12 May, 1999.
- [5]Venayagamoorthy GK, Harley RG, "Simulation studies with a continuously online trained artificial neural network controller for a micro-turbogenerator", Proceedings of IEE International Conference on Simulation, University of York, UK, 30 September – 2 October, pp 405 – 412,1998.
- [6]Taylor and Francis Group, "Synchronous Generators" , Handbook,2006.
- [7]Bimal K. Bose, " Modern Power Electronic and AC Drives", Prentice Hall, 2002.
- [8] Saadat, H., "Power System Analysis", McGraw-Hill Inc., 1999.
- [9]Ziegler, J.G., and Nichols, "Optimum settlings for automatic controllers", Trans. On ASME., vol. 64, pp.759- 768 ,N.B. 1942.,
- [10]B. K. Bose, "Expert System, Fuzzy logic, and neural network Applications in power Electronics and motion control", Proceedings of the IEEE, Vol. 82, NO. 8, 1303-1321. World Academy of Science, Engineering and Technology 46 2008 899 August 1994.
- [11] Aissaoui, A. G., M. Abid, H. Abid, And A. Tahour," A Neural Controller for Synchronous Machine" World Academy of Science, Engineering and Technology 46,pp 893- 900,2008.
- [12] Chow, M., Sharpe, R. N. and Hung, J. C. "On the application and design of artificial neural networks for motor fault detection – Part I", IEEE Transaction on Industry Electronics, Vol. 40, NO. 2, 181- 188, April 1993.
- [13] Becerra ,V. M., "Process Control Simulation using Simulink", Sept. 2000, Updated for Matlab 6.5 – June,2003.
- [14] Mehpotra ,K. and Mohan ,C. K., " Elements of Artificial Neural Networks" , proceeding Japan text book pp.41 ,2001.

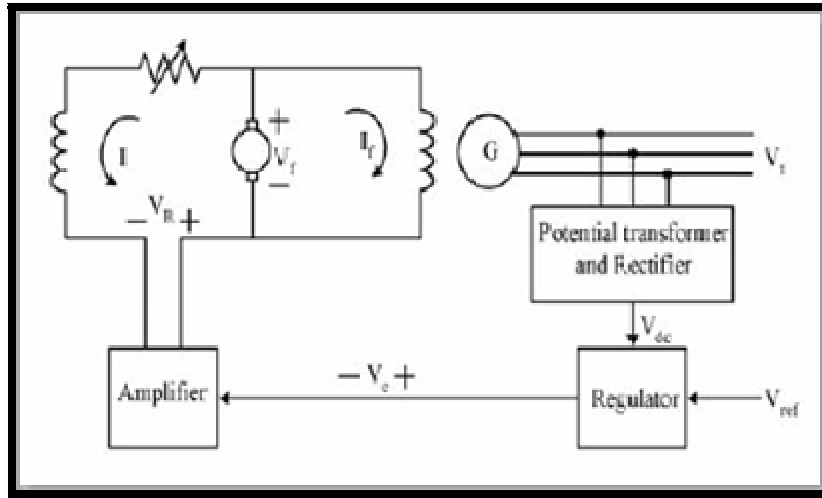


Figure (1) a close loop control of S.G. and its excitation system.

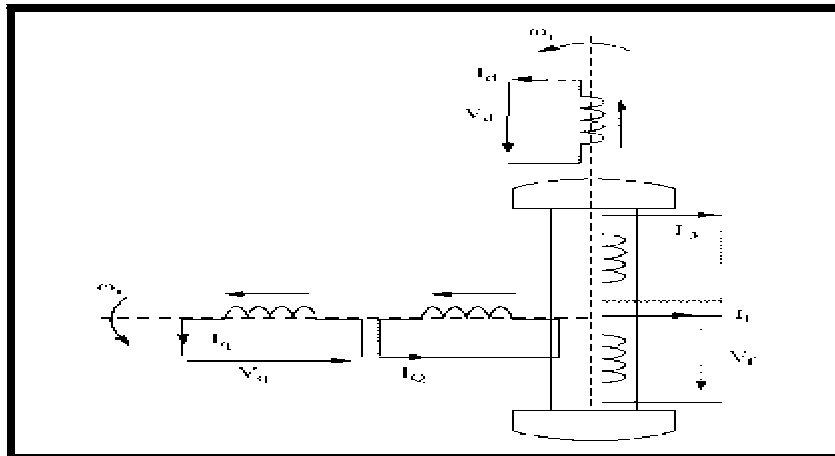


Figure (2) S.G. windings in qd-axis

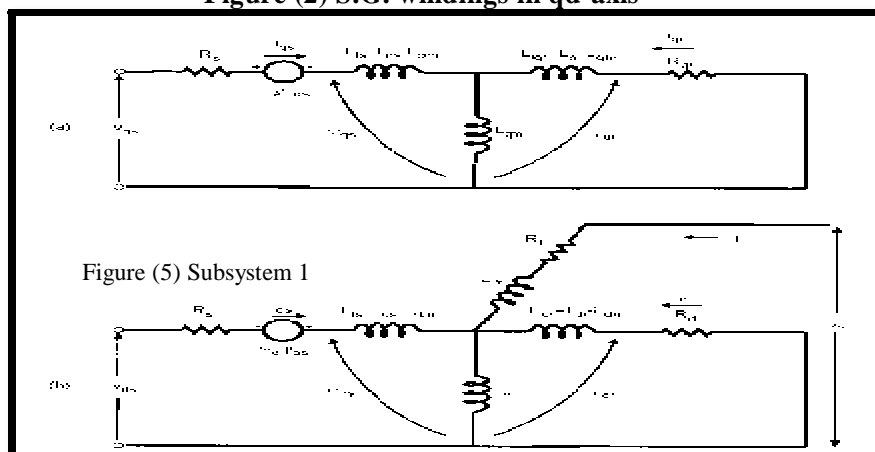


Figure (3) Stator and rotor equivalent circuits in qd-axis

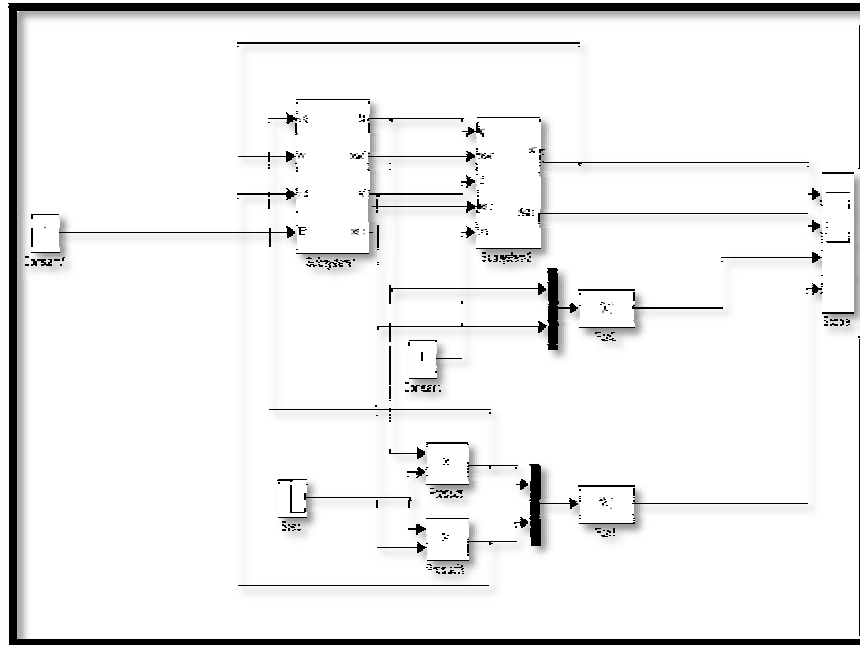


Figure (4) Block diagram of a synchronous machine with AVR simulated with

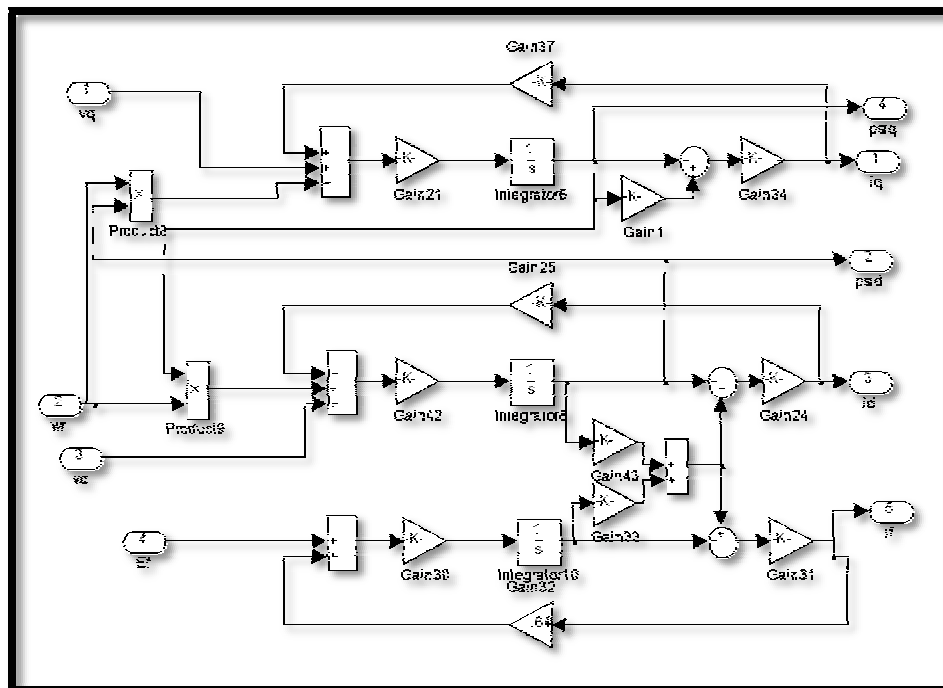


Figure (5) Subsystem 1

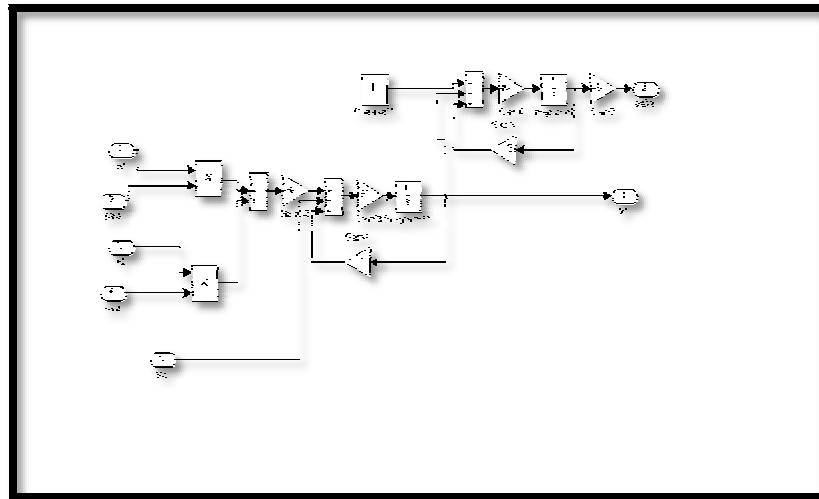


Figure (6) Subsystem 2

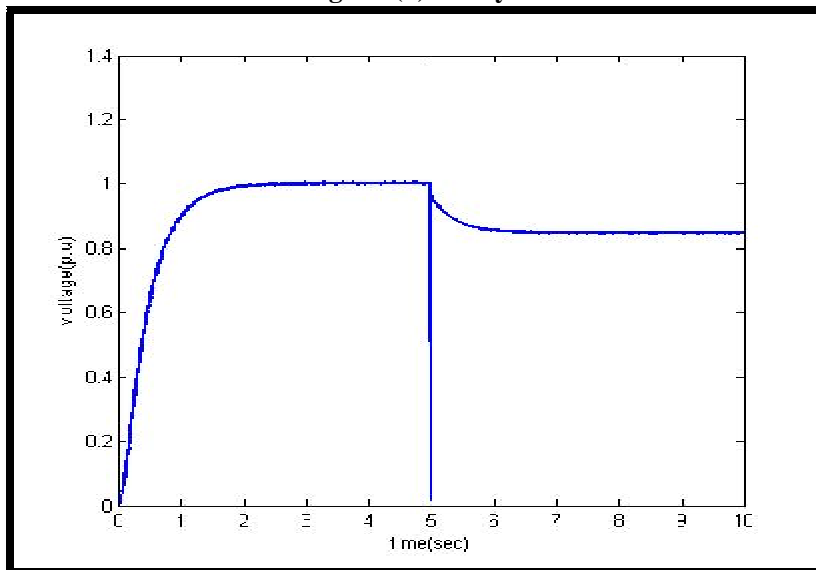


Figure (7) Terminal voltage (V_t) step response simulated with matlab

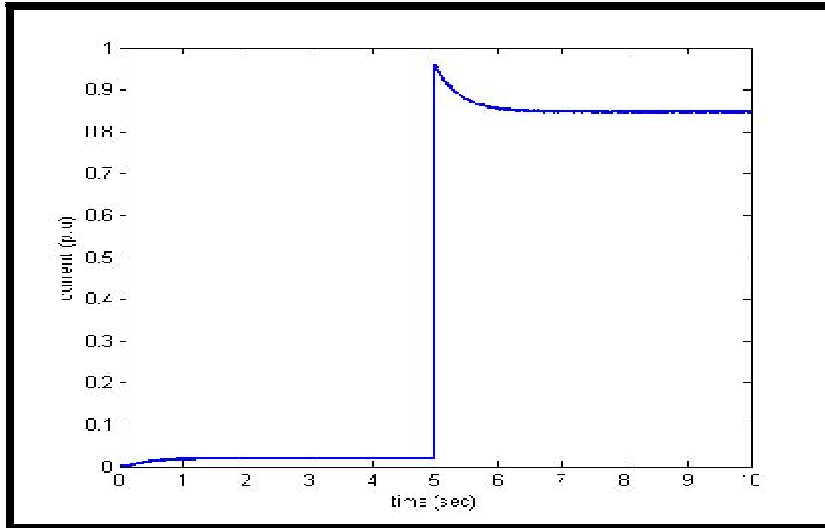


Figure (8) current waveform step response simulated with matlab

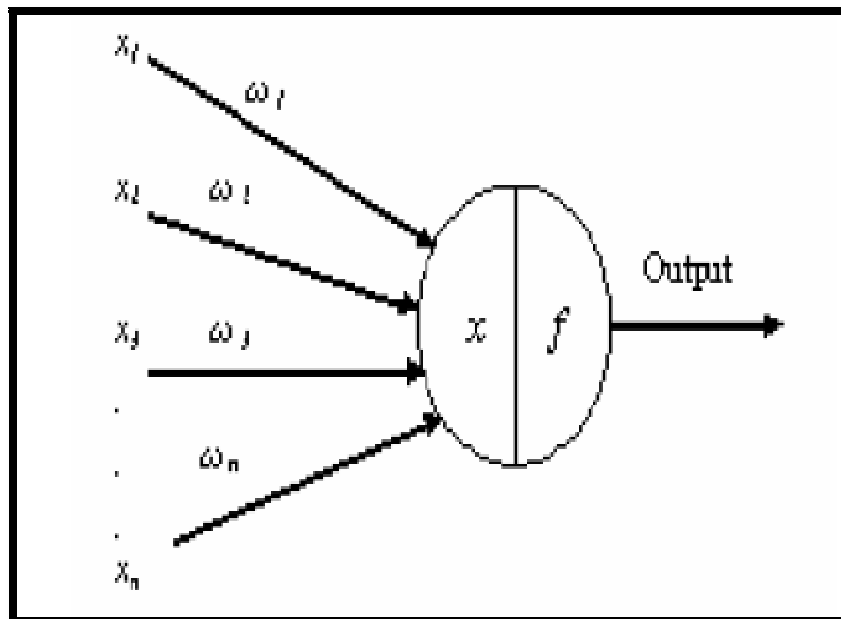


Figure (9) Structure of an artificial neuron

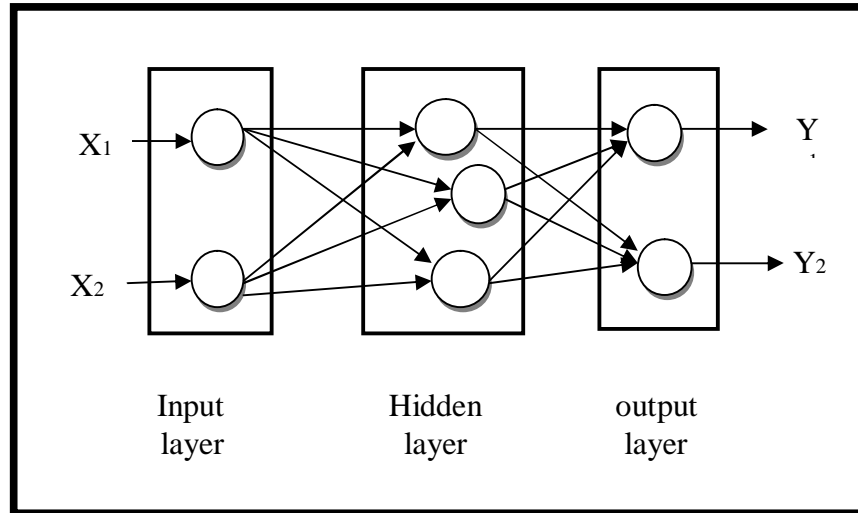


Figure (10) Structure of a feedforward multilayer network

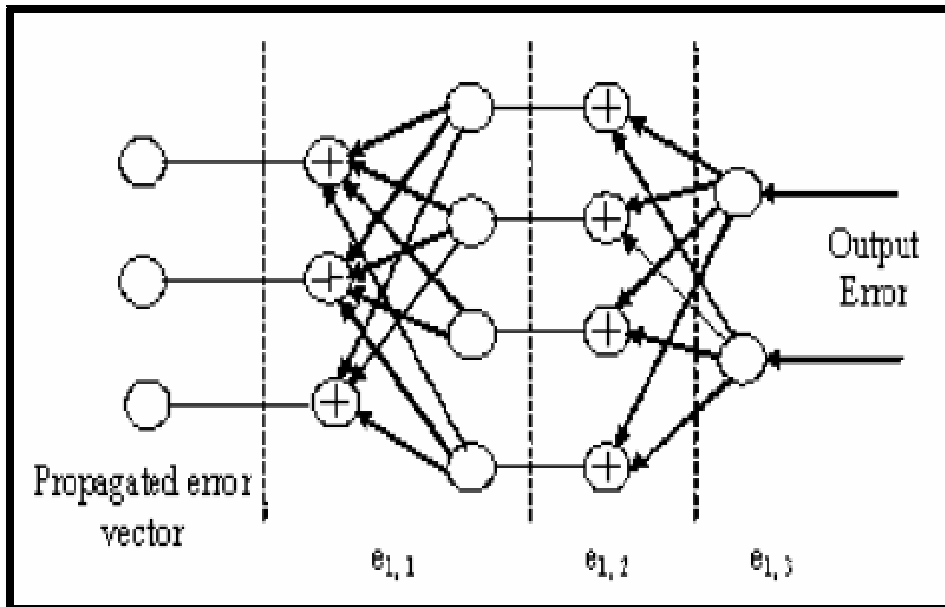


Figure (11) Principle of Back-Propagation training

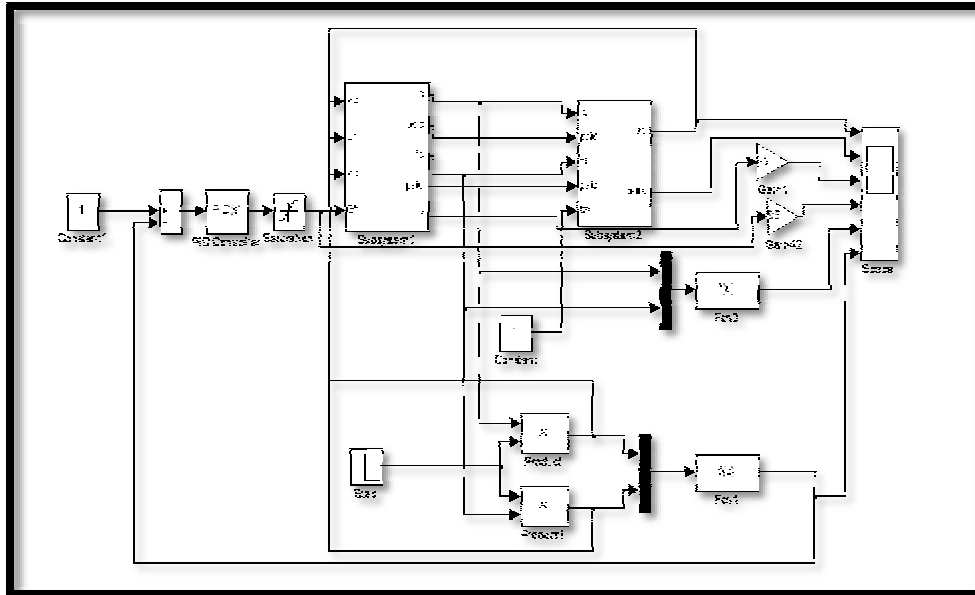


Figure (12) Block diagram of a synchronous machine with PID.

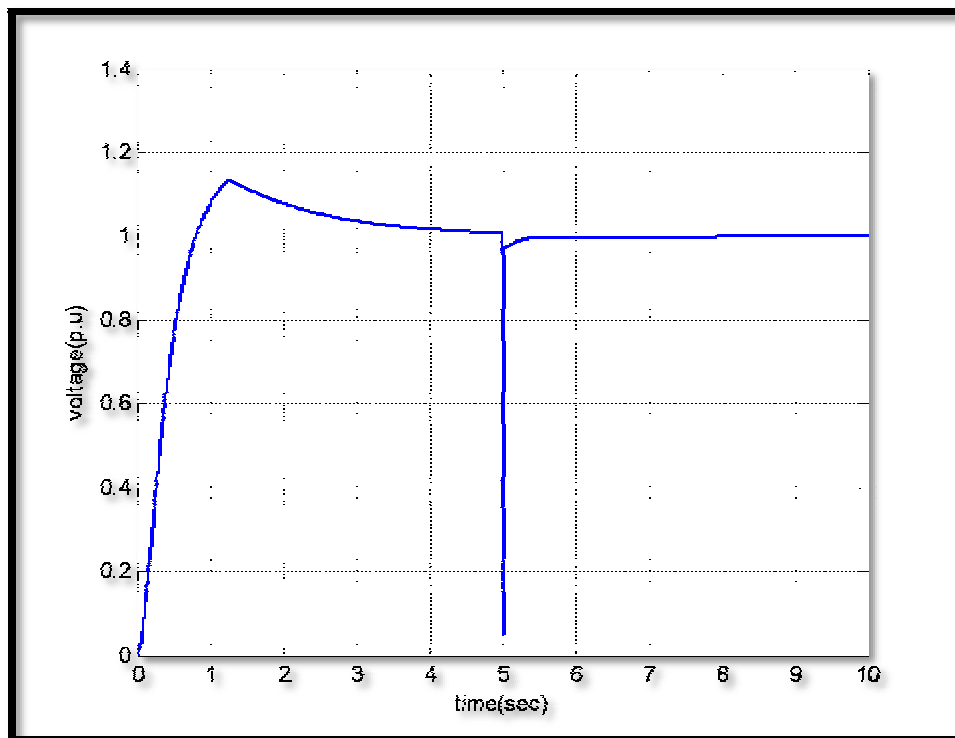


Figure (13) Terminal voltage (V_t) step response with PID

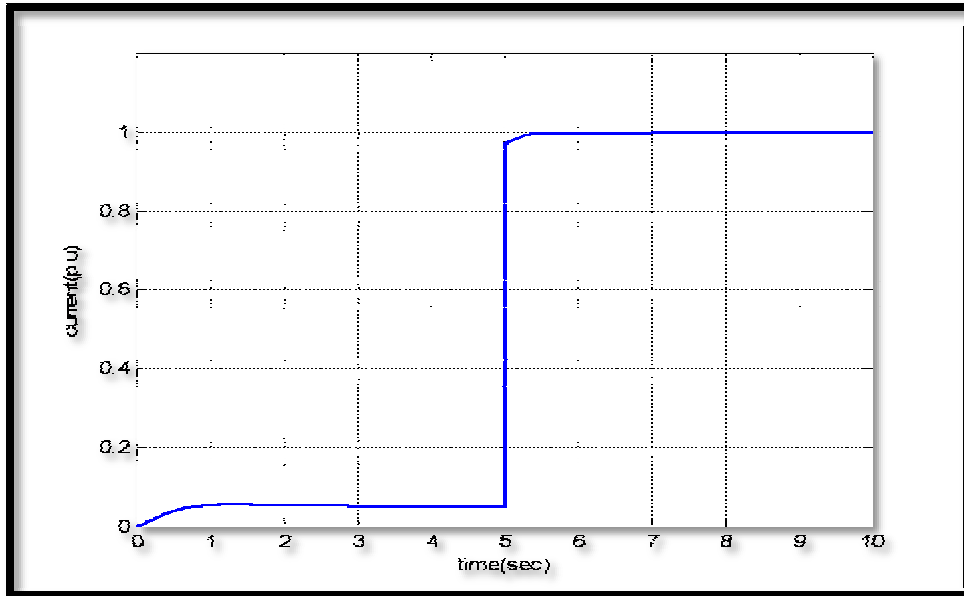


Figure (14) current waveform step response with PID

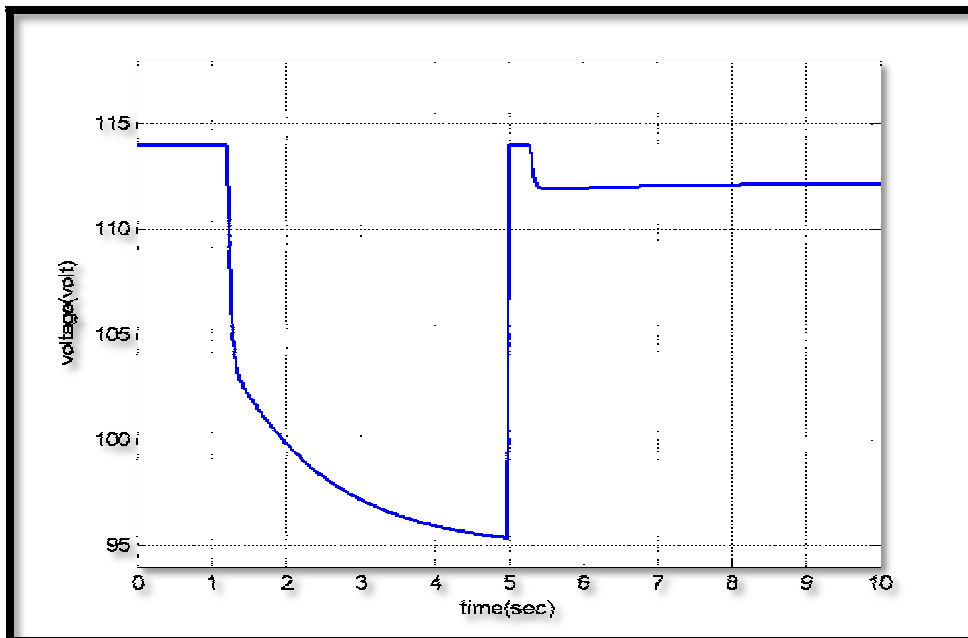


Figure (15) Field voltage with PID

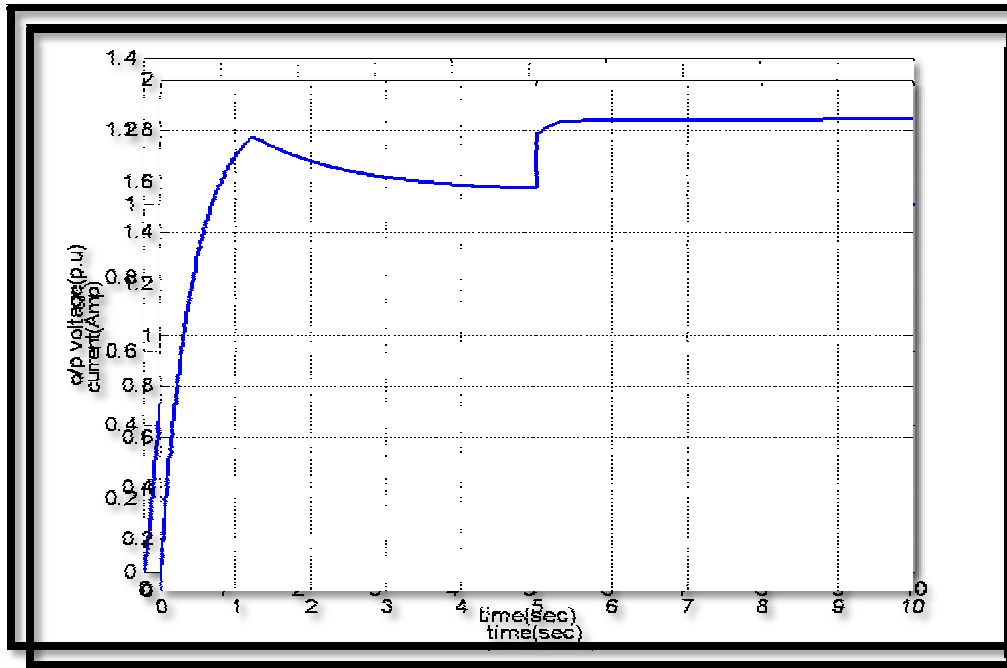


Figure (16) Field current with PID

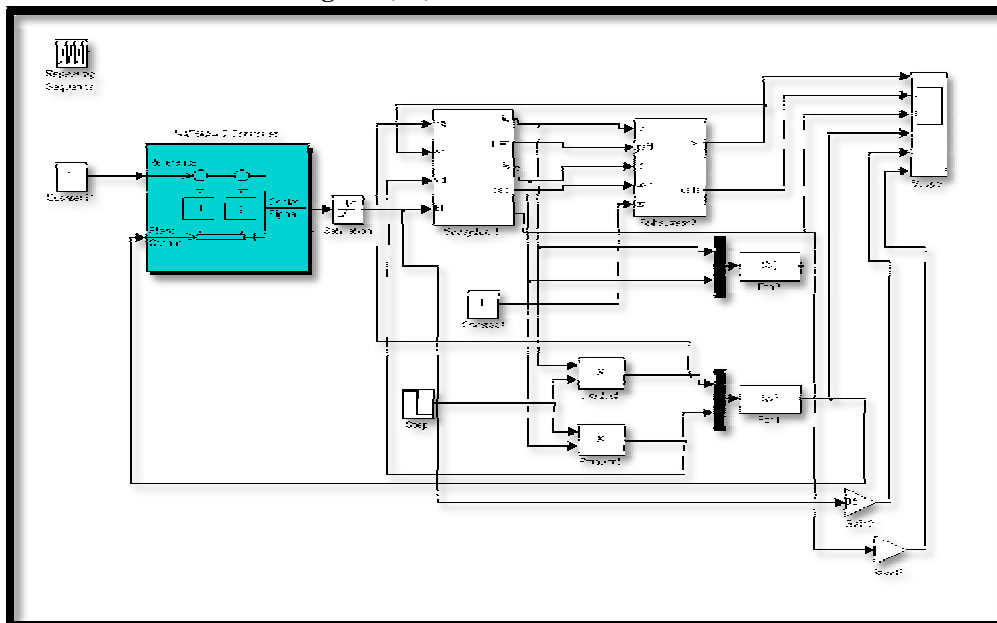


Figure (17) Block diagram of a synchronous machine with AVR simulated with ANN controller.

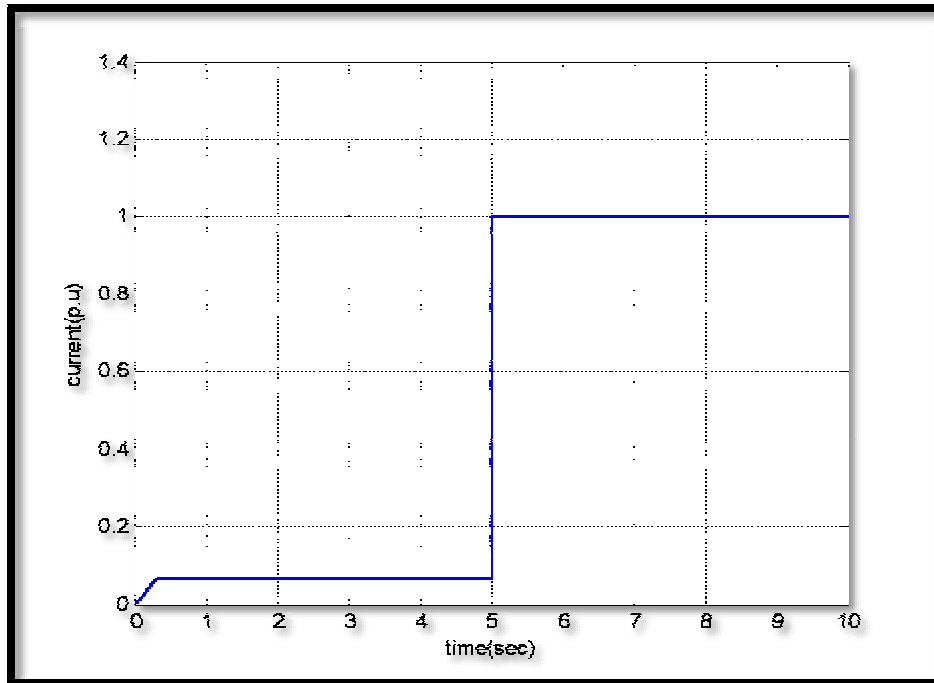


Figure (18) Terminal voltage (V_t) step response with ANN

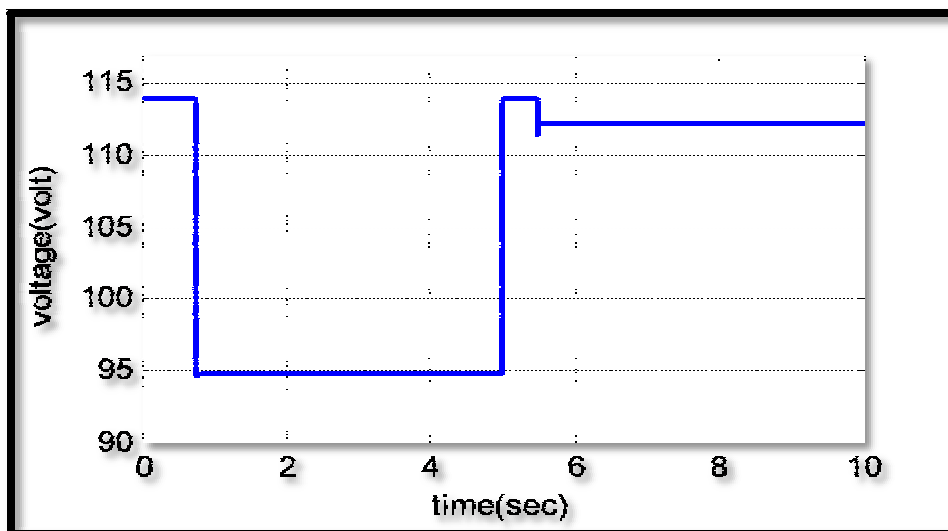


Figure (19) current waveform step response with ANN

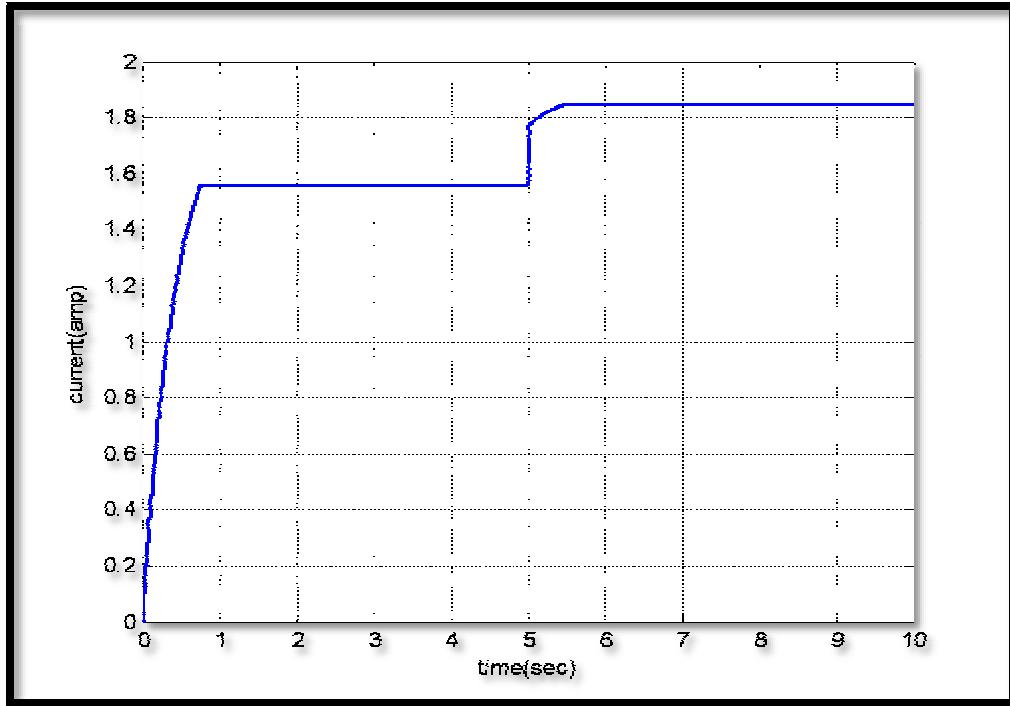


Figure (20)Field voltage with ANN

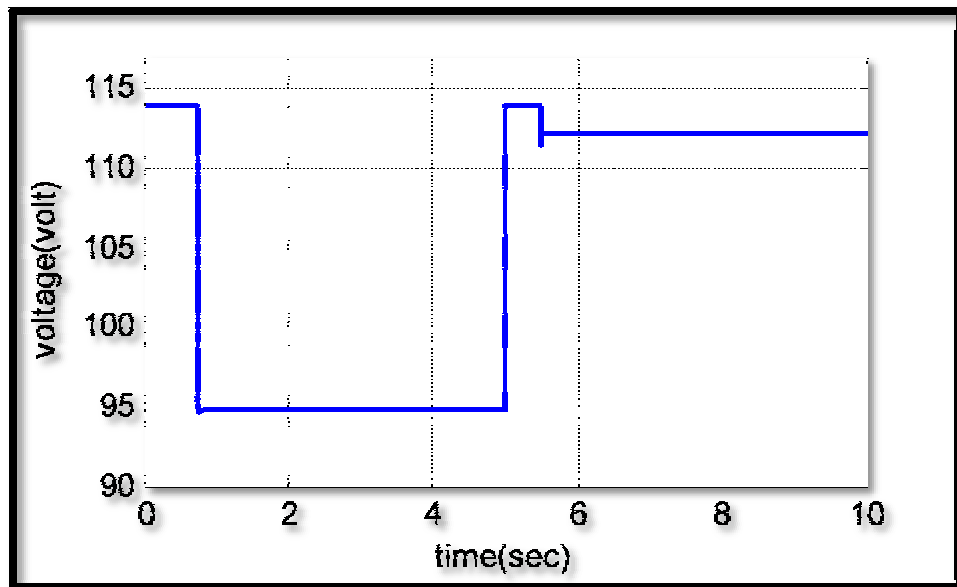


Figure (21) Field current with ANN

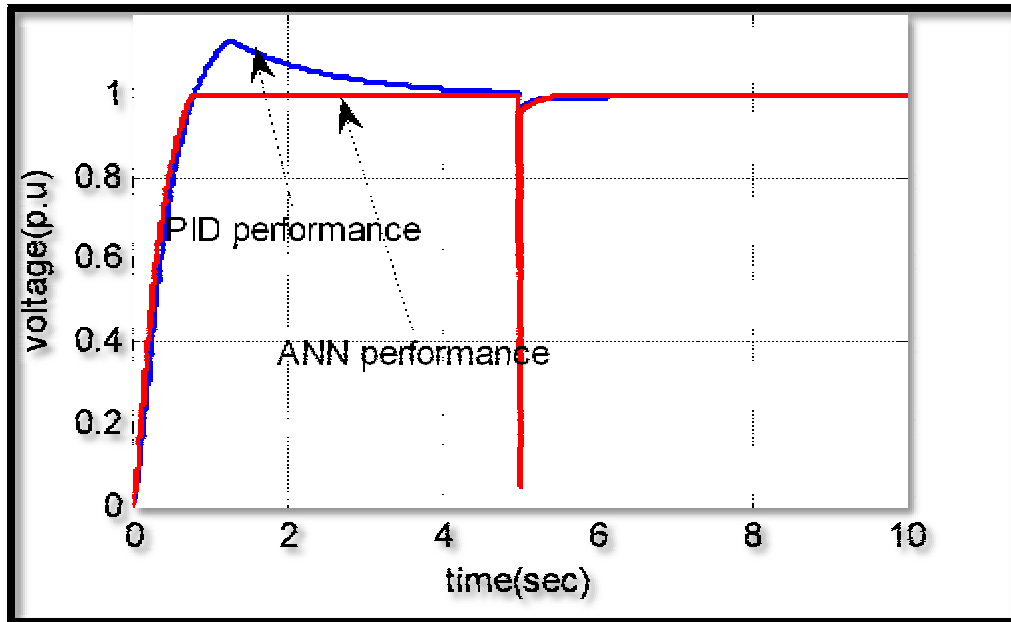


Figure (22) Comparative between the performances of the terminal voltage by using PID &ANN