Chapter 5
Soft Computing Techniques in Wind Energy Conversion Systems

Learning Objectives
On completion of this chapter, the reader will have knowledge on:

- Importance of soft computing techniques in WECS.
- Prediction of power factor using soft computing techniques.
- Fuzzy based Pitch angle control.
- Soft computing based MPPT in WECS.
- Economic dispatch of WECS using soft computing algorithms.

The influence of soft computing on the different aspects of wind power systems especially in the field of operation and maintenance has received great focus during the recent years. This chapter reviews, analyses, discusses and summarizes the recent research, development and trends in the applications of soft computing in the field of wind power systems. This chapter provides the state of the art of soft computing techniques in wind energy conversion systems as a good guidance for future research work. Based on the observations from several applications, soft computing techniques are adequate for solving the different challenges at different phases of the life cycle processes of wind power systems. Soft computing can enhance the efficiency and effectiveness of the operation and maintenance of offshore wind power systems through improving the availability levels, thus, providing secure, sustainable and competitive energy supply for the future.

Soft computing is a collection of computational techniques which intend to complement each other. The aim of soft computing is to exploit the tolerance for imprecision, uncertainty, approximation, and partial truth to achieve tractability, robustness, and low solution cost. Components of soft computing include: fuzzy logic, neural computing, evolutionary computing and probabilistic computing. Applications based on soft computing such as evolutionary computation, neural networks and fuzzy logic for components of wind energy conversion systems are illustrated in this chapter.

S. Sumathi et al., *Solar PV and Wind Energy Conversion Systems*,
Green Energy and Technology, DOI 10.1007/978-3-319-14941-7_5

5.1 Prediction of Wind Turbine Power Factor

Wind energy conversion systems (WECS) are arising as smart alternative for electricity generation. The basic components of a typical wind energy conversion system include a wind turbine, a generator, interconnection apparatus, and control systems. The design of such components has great focus during recent years. The wind turbine is the most important part of the WECS, which transforms the wind kinetic energy into mechanical or electric energy. The wind turbine comprises of a blade, a mechanical part and an electric engine coupled to each other. The kinematical energy of wind is the function of wind speed, the specific mass of air, the area of air space where the wind is captured and the height at which the rotor is placed. The power generated by each wind turbine depends on parameters such as turbine type, the number of blades and the power factor. The power factor is also called blade yield and can be obtained from blade and wind properties. In this section, power factor of wind turbine is predicted using Artificial Neural Networks (ANN) and Adaptive Neuro-Fuzzy Inference System (ANFIS).

5.1.1 Problem Formulation

The definition of power factor is the ratio between the power in turbine shaft (Pp) and the wind power (Pr) due to its kinetic energy right before the turbine plane, which yields

$$C_p = \frac{P_p}{P_r} \tag{5.1}$$

The wind power, P_r, in the air flow passing through the circle with a radius of R immediately before the turbine plane can be defined as;

$$P_r = \frac{1}{2}\rho\pi R^2 V_r^3 \tag{5.2}$$

where ρ is the density of the air and Vr is the wind speed. In general, the maximum power factor of the wind turbine is 59.26 % known as the Betz limit and the practical value can be 45 % of the maximum. This decrease is due to the losses involved in practical systems such as profile losses, end losses, eddy losses, and blade number losses.

5.1.1.1 Profile Losses

This loss can be considered using

$$\eta_{profile} = 1 - (\lambda_A/\varepsilon) \tag{5.3}$$

where λ_A is the tip speed ratio and ε is the number of slip (Slide) and expressed as

$$\varepsilon = \frac{C_L}{C_D} \tag{5.4}$$

where C_L is the coefficient of lift force of chosen profile and C_D is the coefficient of drag force.

5.1.1.2 End Losses

In the end of a blade, airflow from the lower side of the profile to the upper side takes place. Coupling with the airflow coming towards the blade, this airflow widens gradually. In the calculations, this can be considered as

$$\eta_{end} = 1 - (1.84/z\lambda_A) \tag{5.5}$$

where z is the number of turbine blade.

5.1.1.3 Eddy Losses

There is no change in the wind before and after the turbine plate, based on the Betz theory. But the air mass encountering the blade changes its direction.

5.1.1.4 Blade Number Losses

If the turbine has more than four blades, the movement of air through the blades becomes complex, and hence theoretical analysis is difficult. Thus, the theory of Glauert-Shmitz previously applies to the turbines with four or less wind turbine blade.

Including the losses mentioned in the above sections, the power factor can be re-expressed as

$$C_p = f\left(A, \lambda_A, C_{pschmitz}, \eta_{end}, \eta_{profile}, \eta_{eddy}, \eta_{blade\ number}\right) \tag{5.6}$$

where A represents the type of profile used, λ_A is the tip speed ratio, $C_{pshmitz}$ is Shmitz coefficient and the η values are the associate losses. Based on the above equation, assessment of the power factor is quite cumbersome, for which an effective procedure is needed. Thus soft computing techniques are applied to predict the power factor in WECS.

5.1.2 *Artificial Neural Networks*

The typical back-propagation network has an input layer, an output layer, and at least one hidden layer. There is no theoretical limit on the number of hidden layers but typically there is just one or two. Some work has been done which indicates that a minimum of four layers (three hidden layers plus an output layer) are required to solve problems of any complexity. Each layer is fully connected to the succeeding layer, as shown in Fig. 5.1.

The in and out layers indicate the flow of information during recall. Recall is the process of putting input data into a trained network and receiving the answer. Back-propagation is not used during recall, but only when the network is learning a training set.

The number of layers and the number of processing element per layer are important decisions. These parameters to a feedforward, back-propagation topology are also the most ethereal. There are only general rules picked up over time and followed by most researchers and engineers applying this architecture of their problems.

Rule One: As the complexity in the relationship between the input data and the desired output increases, then the number of the processing elements in the hidden layer should also increase.

Rule Two: If the process being modeled is separable into multiple stages, then additional hidden layer(s) may be required. If the process is not separable into stages, then additional layers may simply enable memorization and not a true general solution.

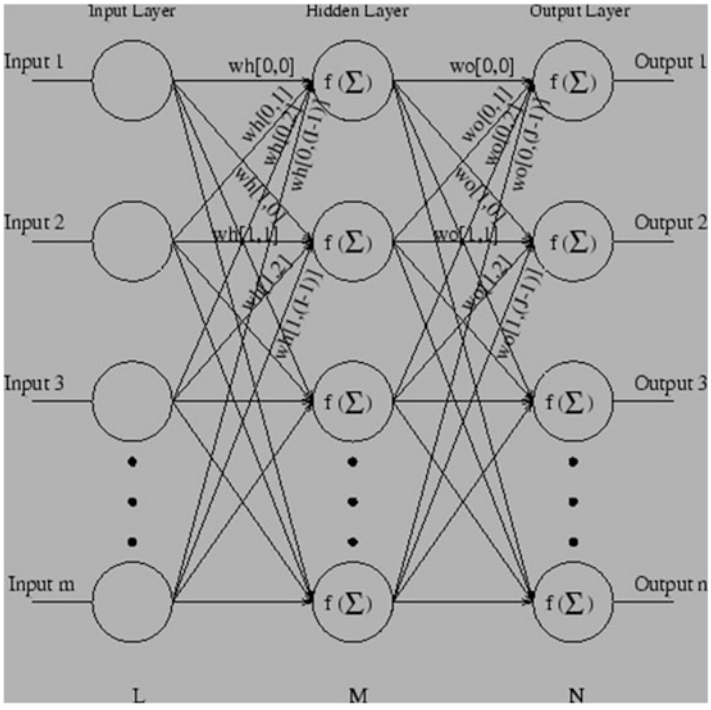


Fig. 5.1 Back-propagation network

Rule Three: The amount of training data available sets an upper bound for the number of processing elements in the hidden layers. To calculate this upper bound, use the number of input output pair examples in the training set and divide that number by the total number of input and output processing elements in the network. Then divide that result again by a scaling factor between 5 and 10. Larger scaling factors are used for relatively noisy data. Extremely noisy data may require a factor of 20 or even 50, while very clean input data with an exact relationship to the output might drop the factor to around 2. It is important that the hidden layers have few processing elements. Too many artificial neurons and the training set will be memorized. If that happens then no generalization of the data trends will occur, making the network useless on new data sets.

5.1.2.1 Training Algorithm

The training algorithm of back propagation involves four stages, viz.

1. Initialization of weights.
2. Feed forward.
3. Back propagation of errors.
4. Updation of the weights and biases.

During first stage, which is the initialization of weights, some small random values are assigned.

During feed forward stage each input unit (X_i) receives an input signal and transmits this signal to each of the hidden units $z_1 \ldots z_p$. Each hidden units then calculates the activation function and sends its signal z_j to each output unit. The output unit calculates the activation function to form the response of the net for the given input pattern.

During back propagation of errors, each output unit compares its computed activation y_k with its target value t_k to determine the associated error for that pattern with that unit. Based on the error, the factor δ_k ($k = 1,\ldots m$) is computed and is used to distribute the error at output unit y_k back to all units in the previous layer. Similarly the factor δ_j ($j = 1,\ldots p$) is computed for each hidden unit z_j.

During final stage, the weight and biases are updated using the δ factor and the activation.

5.1.2.2 Parameters

The various parameters used in the training algorithm is as follows:

x: Input training vector $x = (x_1,\ldots x_i,\ldots x_n)$
t: Output target vector $t = (t_1,\ldots t_k,\ldots t_m)$.
δ_k = error of output unit y_k
δ_j = error of output unit z_j
α = learning rate
v_{oj} = bias on hidden unit j
z_j = hidden unit j
w_{oK} = bias on output unit k
y_k = output unit k

The pseudocode of the training algorithm of the back propagation network is as follows. The algorithm is given with the various phases:

Initialization stage:
Initialize weights v and w to small random values
Initialize Learning rate
While not stopping condition for phase I **Do**
For each training pair
Feed forward stage:
Each input unit receives the input signal x_i and transmits to all units in the hidden layer
Each hidden unit (z_j, j=1,…,p) sums its weighted input signals

$$z_{inj} = v_{oj} + \sum_{i=1}^{n} x_i v_{ij}$$

applying activation function
$Zj = f(z_{inj})$
and sends this signal to all units.
Each output unit (y_k, k=1,…,m) sums its weighted input signals

$$y_{ink} = w_{ok} + \sum_{j=1}^{p} z_j w_{jk}$$

applies its activation function to calculate output signals
$Y_k = f(y_{ink})$
Back propagation of errors:
Each output unit (y_k, k=1,…, m) receives a target pattern corresponding to an input pattern, error information term is calculated as

$$\delta_k = (t_k - y_k) f(y_{ink})$$

Each hidden unit (z_j, j=1,…,p) sums its delta inputs from units in the layer above

$$\delta_{inj} = \sum_{k=1}^{m} \delta_j w_{jk}$$

The error information is calculated as
$\delta_j = \delta_{inj}\ f(z_{inj})$
Updation of weights and biases:
Each output unit (y_k, k=1,…,m) updates its bias and weights (j=0,…,p)
The weight correction term is given by

$$\Delta W_{jk} = \alpha \delta_k z_j$$

The bias correction term is given by

$$\Delta W_{ok} = \alpha \delta_k$$

$W_{jk(new)} = W_{jk(old)} + \Delta W_{jk}$, $W_{ok(new)} = W_{ok(old)} + \Delta W_{ok}$
Each hidden unit (z_j, j=1,…,p) updates its bias and weights (i=0,…,n)
The weight correction term

$$\Delta V_{ij} = \alpha \delta_j x_i$$

The bias correction term

$$\Delta V_{oj} = \alpha \delta_j$$

$V_{ij\ (new)} = V_{ij(old)} + \Delta V_{ij}$, $V_{oj(new)} = W_{oj(old)} + \Delta V_{oj}$
End For
Test the stopping condition
End while
End

The stopping condition may be the minimization of the errors, number of epochs etc.

5.1.3 Adaptive Neuro-fuzzy Inference System (ANFIS)

Both fuzzy logic systems and neural networks are natural complementary tools in building intelligent systems. Neural networks are basically low-level computational structures that perform well when dealing with raw data, fuzzy logic deals with reasoning on a higher level. However, fuzzy systems lack the ability to learn and cannot adjust themselves. A neuro-fuzzy system is, in fact, a neural network that is functionally equivalent to a fuzzy inference model.

The hybrid neuro-fuzzy model for potential mapping described here is a Takagi-Sugeno type fuzzy inference system, which is implemented in the framework of adaptive neural networks. It is an adaptation of "adaptive network-based fuzzy inference system" for mineral potential mapping. A typical fuzzy if-then rule in a generalized Takagi-Sugeno type fuzzy inference system has the following form:

$$\text{IF } x \text{ is } a \text{ AND } y \text{ is } b \text{ THEN } z = f(x, y)$$

where x and y are input variables, a and b are fuzzy membership values of x and y, respectively, in the antecedent part of the fuzzy if-then rule and $z = f(x, y)$ is a crisp function in the consequent part of the rule. Usually, $f(x, y)$ is a polynomial in the input variables x and y, but it can be any function as long as it can appropriately describe the output of the system. When $f(x, y)$ is a first-order polynomial, the resulting fuzzy inference system is called a first-order Takagi-Sugeno type fuzzy inference system, which was originally proposed by Takagi and Sugeno. When $f(x, y)$ is a constant, the resulting fuzzy inference system is called a zero-order Takagi-Sugeno type fuzzy inference system. The higher the order of the polynomial functions, in a fuzzy inference system, the larger is the number of the function parameters and, consequently, the larger is the number of training samples required for a robust estimation of these parameters. Therefore, the order of a fuzzy inference system used in an application is largely determined by the number of available training samples. In the case of the feature vector T, there are two input predictor patterns, x_{1j} and x_{2j}, and four membership functions, $\mu_{\tilde{A}_1^1}$, $\mu_{\tilde{A}_1^2}$, $\mu_{\tilde{A}_2^1}$ and $\mu_{\tilde{A}_2^2}$, which can be combined using a first-order Takagi-Sugeno type fuzzy inference system based on the following fuzzy if-then rules:

1. IF x_{1j} is $\mu_{\tilde{A}_1^1}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^1}(x_{2j})$ THEN $F_1 = P_{10} + P_{11}x_{1j} + P_{12}x_{2j}$
2. IF x_{1j} is $\mu_{\tilde{A}_1^1}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^2}(x_{2j})$ THEN $F_2 = P_{20} + P_{21}x_{1j} + P_{22}x_{2j}$
3. IF x_{1j} is $\mu_{\tilde{A}_2^1}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^1}(x_{2j})$ THEN $F_3 = P_{30} + P_{31}x_{1j} + P_{32}x_{2j}$
4. IF x_{1j} is $\mu_{\tilde{A}_1^2}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^2}(x_{2j})$ THEN $F_4 = P_{40} + P_{41}x_{1j} + P_{42}x_{2j}$

where P_{ki} (k = 1 to 4, i = 0 to 2) is the parameter of the polynomial function in the consequent part of the k^{th} fuzzy if-then rule. The above fuzzy inference system is characterized by four fuzzy if-then rules, each of which contains, in the consequent part, a first-order polynomial function characterized by three parameters. Therefore, a first-order Takagi- Sugeno type fuzzy inference system with two input predictor

maps and two fuzzy membership functions for each map results in $2^2(=4)$ fuzzy if-then rules and $2^2(2+1)(=12)$ function parameters. In general, a first-order Takagi- Sugeno type fuzzy inference system with n predictor maps and 2n fuzzy membership functions contains 2^n fuzzy if-then rules and $2^n(n+1)$ function parameters.

Even for a moderately-large n, robust estimation of the function parameters would require a large number of training samples of known mineral deposits, which may not always be available. In such cases, it is preferable to use a zero-order Takagi-Sugeno type fuzzy inference system, which, in the case of the feature vector T, comprises the following fuzzy if-then rules:

1. IF x_{1j} is $\mu_{\tilde{A}_1^1}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^1}(x_{2j})$ THEN $F_1 = P_1$
2. IF x_{1j} is $\mu_{\tilde{A}_1^1}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^2}(x_{2j})$ THEN $F_2 = P_2$
3. IF x_{1j} is $\mu_{\tilde{A}_2^1}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^1}(x_{2j})$ THEN $F_3 = P_3$
4. IF x_{1j} is $\mu_{\tilde{A}_1^2}(x_{1j})$ AND x_{2j} is $\mu_{\tilde{A}_2^2}(x_{2j})$ THEN $F_4 = P_4$

where P_k (k = 1 to 4) is the parameter of the constant function in the consequent part of the k^{th} fuzzy if-the rule. The number of function parameters is reduced to $2^2(=4)$ for the zero-order Takagi-Sugeno fuzzy inference system from $2^2(2+1)(=12)$ for the first-order Takagi-Sugeno fuzzy inference system. In general, a zero-order Takagi-Sugeno type fuzzy inference system with n predictor maps and 2n fuzzy membership functions contains 2^n fuzzy if-then rules and 2^n function parameters. The firing strengths s_i (i = 1 to 4) of the above fuzzy if-then rules are calculated using prod t-norm operator as follows:

$$\begin{aligned} s_1 &= \mu_{\tilde{A}_1^1}(x_{1j})\, x\, \mu_{\tilde{A}_2^1}(x_{2j}) \\ s_2 &= \mu_{\tilde{A}_1^1}(x_{1j})\, x\, \mu_{\tilde{A}_2^2}(x_{2j}) \\ s_3 &= \mu_{\tilde{A}_1^2}(x_{1j})\, x\, \mu_{\tilde{A}_2^1}(x_{2j}) \\ s_4 &= \mu_{\tilde{A}_2^1}(x_{1j})\, x\, \mu_{\tilde{A}_2^2}(x_{2j}) \end{aligned} \tag{5.7}$$

As a matter of fact, other t-norm operators that perform generalized AND can also be used for combining the fuzzy membership values in order to determine the firing strengths of the fuzzy if-then rules. The output, O_k, of the k^{th} (k = 1 to 4) fuzzy if-then rule is:

$$O_k = s_k . F_k \tag{5.8}$$

where F_k *(k = 1 to 4)* is the value of the function in the consequent part of the k^{th} fuzzy if-then rule. The overall output of the fuzzy inference system is the weighted average of the output of all the four fuzzy if-then rules:

$$\textit{Overall output} = \frac{\sum_{k=1}^{4} O_k}{\sum_{k=1}^{4} s_k} \tag{5.9}$$

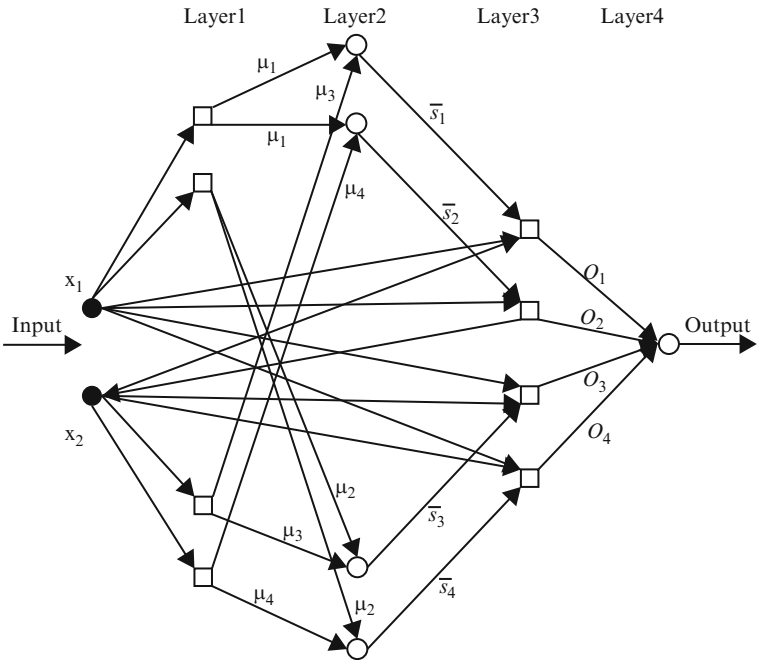


Fig. 5.2 Simplified ANFIS for mineral potential mapping. *Square* and *circular nodes* contain, respectively, adaptive and fixed functions

The overall output is a measure of combined favorability of the feature vector T with respect to a target mineral deposit-type. The above procedure can be easily extended for estimating the favorability of a feature vector comprising more than two predictor maps. In a simple Takagi-Sugeno type fuzzy inference system, the parameters of fuzzy membership functions and consequent polynomial functions are estimated heuristically. However, in a hybrid neuro-fuzzy model, an ANFIS is used for estimating these parameters.

The basic architecture of an ANFIS comprises a Takagi-Sugeno type fuzzy inference system in a five-layer feed-forward network. Figure 5.2 shows the simplified four-layer ANFIS architecture used in a hybrid neuro fuzzy model for mineral potential mapping. The basic functionality of each layer is summarized in the following paragraphs.

5.1.3.1 Layer 1

There are 2n (= the number of fuzzy sets defined in the superset X) nodes in this layer. One half of the nodes (= n) contain the adaptive Gaussian fuzzy membership function defined in Eq. 5.3, each of which receives one component (a predictor pattern x_{ij} encoded as a class score cs_{ij}) of an incoming n-dimensional feature vector and returns its membership value in the fuzzy set $\tilde{A}_i^1$ (i = 1 to n) containing 'favorable indicators of

target mineral deposit type'. The other half of the nodes (= n) contain the adaptive Gaussian fuzzy membership function, each of which also receives one component of the incoming feature vector and returns its membership value in the fuzzy set $\tilde{A}_i^2$ (i = 1 to n) containing 'unfavorable indicators of target mineral deposit-type'.

The parameters c and σ, which control the shape of a node function, and therefore output fuzzy membership values, are referred to as premise parameters.

5.1.3.2 Layer 2

Each of the 2^n (= the number of fuzzy if-then rules) nodes in this layer contains a prod t-norm operator as a node function, which synthesizes information transmitted by Layer 1 and returns a firing strength for each of the fuzzy if-then rules:

$$s_k = \mu_{\tilde{A}_1^q}(x_{1j}) \;\; x \;\; \mu_{\tilde{A}_2^q}(x_{2j}) \;\; x.....x \;\; \mu_{\tilde{A}_n^q}(x_{nj}) \tag{5.10}$$

where q = 1 or 2, depending on whether $\mu_{\tilde{A}_1^q}$ defines fuzzy membership value of x_{ij} in the fuzzy set $\tilde{A}_i^1$ or in the fuzzy set $\tilde{A}_i^2$ (i = 1 to n). The output of each node is the normalized firing strength $\overline{s}_k$ (k = 1 to 2^n) of the k^{th} fuzzy if-then rule given by:

$$\overline{s}_k = \frac{s_k}{\sum_{k=1}^{2^n} s_k} \tag{5.11}$$

5.1.3.3 Layer 3

Each of the 2^n (= the number of fuzzy if-then rules) nodes in this layer, contains the following adaptive function:

$$O_k = s_k.F_k = s_k\left(P_{k0} + P_{k1}x_{1j} + P_{k2}x_{2j} + ... + P_{kn}x_{nj}\right) \tag{5.12}$$

where O_k is the output of the k^{th} fuzzy if-then rule. The parameters P_{ki} (k = 1 to 2^n, i = 0 to n) are referred to as consequent parameters. In the case of a zero-order Takagi-Sugeno type fuzzy inference system, $F_k = P_{k0}$.

5.1.3.4 Layer 4

The single node in this layer synthesizes information transmitted by Layer 3 and returns the overall output using the following fixed function:

$$Overall\ Output \;=\; \sum_{k=1}^{2^n} O_k \tag{5.13}$$

The following description of the hybrid learning procedure is drawn from Jang and Sun.

Forward Pass: Least Square Estimator Method

For 2^n fuzzy if-then rules and Q n-dimensional training vectors (where n is the number of input predictor maps), Eq. 5.13 can be expressed as:

$$B = AX \tag{5.14}$$

where B is a column vector containing output values of training vectors, A is matrix containing one row for each training vector and X is an unknown vector whose elements are the consequent parameters P_{ki}. As the number of consequent parameters is $2^n(n+1)(=M$, say), the dimensions of A, X and B are $Q \times M$, $M \times 1$ and $Q \times 1$, respectively.

A least square estimate of X, denoted by X*, can be used to minimize squared error $\|AX - B\|^2$. It can be computed as below:

$$X^* = \left(A^T A\right)^{-1} A^T B \tag{5.15}$$

where A^T is the transpose of A and $(A^T A)^{-1} A^T$ is the pseudo-inverse of A, if $A^T A$ is non-singular. The above equation is expensive in computation when dealing with matrix inversion and, more over, becomes ill defined if $A^T A$ is singular. ANFIS uses a recursive least-square method for estimating X *as follows*.

If a_q^T is the q^{th} row vector of A and b_q^T is the qth element of B, then X can be calculated iteratively as follows:

$$X_{q+1} = X_q + \sum\nolimits_{q+1} a_{q+1} \left(b_{q+1}^T - a_{q+1}^T X_q\right) \tag{5.16}$$

$$\sum\nolimits_{q+1} = \sum\nolimits_q - \frac{\sum\nolimits_q a_{q+1} a_{q+1}^T \sum\nolimits_q}{1 + a_{q+1}^T \sum\nolimits_q a_{q+1}} \left(for\ q = 0, 1, \ldots\ldots\ldots.., Q-1\right) \tag{5.17}$$

where $\sum$ is called covariance matrix and the least square estimate X^* is equal to X_q. The initial conditions are $X_0 = 0$ and $\sum 0 = \gamma I$, where γ is a large positive number and I is an identity matrix of $M \times M$ dimensions.

Backward Pass: Back-Propagation Method

Premise parameters of ANFIS are estimated iteratively by using a modified back-propagation learning rule along with the chain rule as follows. In Eq. 5.14, if the estimated output of the qth row vector, a_q, of the matrix A is o_q and the actual output is b_q, the qth element of B, then an error margin for the q^{th} training vector a_q can be defined as:

$$E_q = \left(b_q - o_q\right)^2 \tag{5.18}$$

If E_q is zero, then the actual output exactly matches the estimated output. Thus, the objective is to minimize an overall error measure, which is defined as $\sum_{q=1}^{Q} E_q$, where Q is the total number of training vectors. In order to use the gradient descent method to minimize the error measure, a gradient vector is required to be calculated. It should be noted that a small change in the premise parameters (c or σ) will affect the output of the node containing the parameters, which, in turn, will affect the output of the single node in Layer 4 and, hence, the error measure will also change. Therefore in order to calculate the gradient vector of the parameters, a form of derivative information has to be passed, starting from Layer 4 and traveling back to Layer 1.

An error signal, $\varepsilon_{l,i}$, can be defined as the ordered derivative of the error measure E_q with respect to the output of node i in Layer l (l = 1 to 4) as follows

$$\varepsilon_{l,i} = \frac{\partial + E_q}{\partial o_{l,i}} \tag{5.19}$$

where $o_{l,i}$ is the output of the i^{th} node of Layer l. The error signal for the single output node of Layer 4 can be calculated as:

$$\varepsilon_4 = \frac{\partial + E_p}{\partial o_{(4,1)}} = \frac{\partial E_p}{\partial o_{(4,1)}} = -2\left(b_q - o_q\right) \tag{5.20}$$

For the i^{th} node of Layer $l(l = 1$ to $3)$, the error signal can be derived using the chain rule:

$$\varepsilon_{l,i} = \frac{\partial + E}{\partial o_{(l,i)}} = \sum_{m=1}^{N_{(l+1)}} \frac{\partial + E}{\partial o_{(l+1,m)}} \frac{\partial f_{(l+1,m)}}{\partial o_{(l,i)}} = \sum_{m=1}^{N_{(l+1)}} \varepsilon_{(l+1,m)} \frac{\partial f_{(l+1,m)}}{\partial o_{(l,i)}} \tag{5.21}$$

where $N_{(l+1)}$ is the number of nodes in Layer $(l+1)$, $o_{(l+1,m)}$ is the output of the m^{th} node in Layer $(l+1)$, $f_{(l+1,m)}$ is the nodal function of the m^{th} node in Layer $(l+1)$, $o_{(l,i)}$ is the output of the i^{th} node in Layer l and ε(l + 1,m) is the error signal at the mth node of Layer (l + 1). In other words, the error signal of an internal node at Layer l can be expressed as a linear combination of the error signal of the nodes at Layer (l + 1). Therefore, for the ith node of Layer 1, the error signal, ε (1,i), can be obtained by first applying Eq. 5.20 once to get error signals at the Layer 4 and then applying Eq. 5.21 iteratively until Layer 1 is reached.

Because the consequent parameters are fixed in the backward pass, the gradient vector is defined as the derivative of the error measure with respect to each of the two premise parameters, c and ε, which reside in the nodes of Layer 1. The chain rule is applied to determine the gradient vectors as follows:

$$\frac{\partial^+ E_p}{\partial c_i} = \frac{\partial^+ E_p}{\partial o_{(1,i)}} \frac{\partial^+ \mu_{1,i}}{\partial c_i} = \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial c_i} \tag{5.22}$$

and

$$\frac{\partial^+ E_p}{\partial \sigma_i} = \frac{\partial^+ E_p}{\partial o_{(1,i)}} \frac{\partial^+ \mu_{1,i}}{\partial \sigma_i} = \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial \sigma_i} \tag{5.23}$$

In the above equations c_i and σ_i are, respectively, center and spread of the gaussian membership function $\mu_{(1,i)}$ in the i^{th} node of layer 1.

The derivative of the overall error measure E with respect to c_i is:

$$\frac{\partial^+ E}{\partial c_i} = \sum_{q=1}^{Q} \frac{\partial^+ E_q}{\partial c_i} = \sum_{q=1}^{Q} \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial c_i} \tag{5.24}$$

where Q is the total number of training vectors. The update expression for the parameter c_i is given by:

$$\Delta c_i = -\eta \frac{\partial^+ E}{\partial c_i} \tag{5.25}$$

where η is the learning rate, which can be expressed as follows:

$$\eta = \frac{\kappa}{\sqrt{\sum_{i=1}^{N(1)} \left(\frac{\partial E}{\partial c}\right)^2}} \tag{5.26}$$

where $N(1)$ is the total number of nodes in Layer 1 and κ is the step size or the length of each transition along the gradient direction in the parameter space. Similarly, the derivative of the overall error measure E with respect to σ_i is:

$$\frac{\partial^+ E}{\partial \sigma_i} = \sum_{q=1}^{Q} \frac{\partial^+ E_q}{\partial \sigma_i} = \sum_{q=1}^{Q} \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial \sigma_i} \tag{5.27}$$

where Q is the total number of training vectors. The update expression for the parameter σ_i is given by:

$$\Delta \sigma_i = -\eta \frac{\partial^+ E}{\partial \sigma_i} \tag{5.28}$$

where η is the learning rate, which can be expressed as follows:

$$\eta = \frac{\kappa}{\sqrt{\sum_{i=1}^{N(1)} \left(\frac{\partial E}{\partial \sigma}\right)^2}} \tag{5.29}$$

where $N(1)$ is the total number of nodes in Layer 1 and κ is the step size or the length of each transition along the gradient direction in the parameter space.

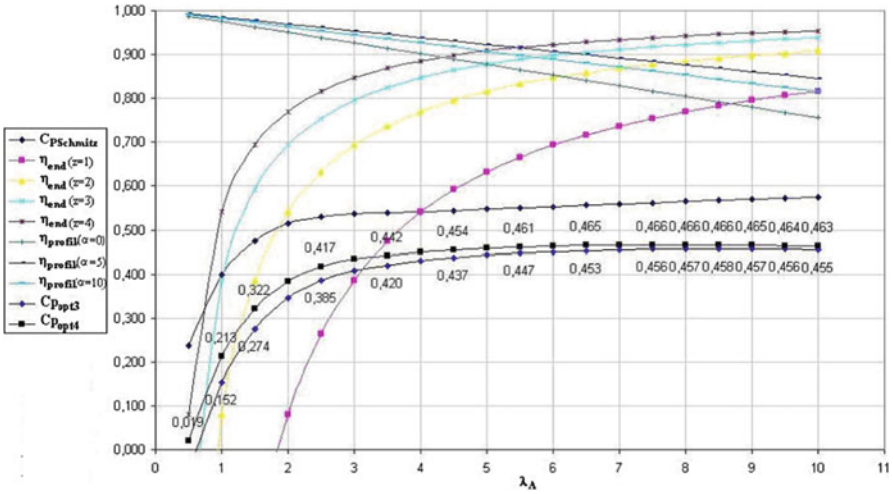


Fig. 5.3 Plot of the values of the profile type NACA 4415

5.1.4 Description of Profile Types

The experimental procedure based on ANN and ANFIS for optimum power factor prediction for three-blade and four-blade turbines. To carry out the analysis, the profile types such as LS-1 and NACA 4415 are considered. The properties of these profiles are presented Figs. 5.3 and 5.4. The input parameters are taken according to the Eq. 5.6 without considering the eddy losses. The blade number losses are not directly taken as an input parameter but considered during the preparation of training data.

5.1.5 Design of the ANN

The design process for estimating power factor using the ANN includes the following steps:

(i) Preparation of suitable training data.
(ii) Selection of a suitable ANN structure.
(iii) Training of the ANN.
(iv) Evaluation of the trained network.

The design process in ANN is an iterative procedure. If the user is not satisfied with the results, the selection of the ANN structure can be changed according to problem and then the network has to be retrained. Sometimes even a trained network may fail while testing the network. Even in such situations, the network structure should be changed and network is retrained and tested.

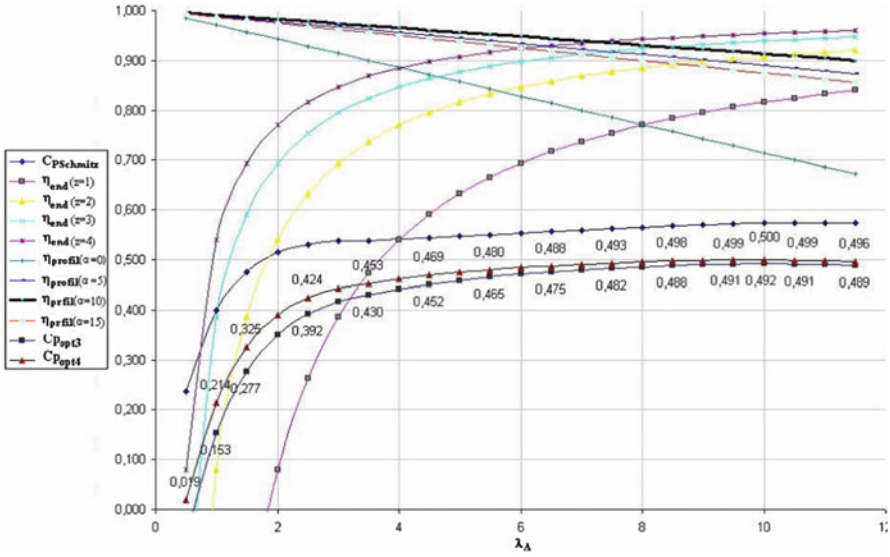


Fig. 5.4 Plot of the values of the profile type LS-1

The training patterns presented to the ANN structure should contain all the necessary information to generalize the problem. After collecting all the required data, normalization of data is mandatory, and this is usually done in the range [0,1]. Normalisation is done for both training and testing data. The testing data is usually chosen as a random percentage of the training sample.

The choice of type of neural network depends upon the number of layers, type of transfer function, number of nodes in the input, hidden and output layer. Based on previous research on neural networks, it is observed that a three-layer feed-forward network can model complex-mapping functions reasonably well and, therefore, is adopted for power factor prediction for the above mentioned profile types. The arbitrary shaped functions requires sigmoid non-linear mapping for modeling. The number of neurons in the input layer and hidden layers are decided by experimentation which involves training and testing different network configurations.

Training of the selected network is done using the training patterns and back propagation algorithm. Training is stopped when the mean squared error between actual outputs and desired outputs stops improving. However, at that point, if the designer is not satisfied with the training and performance of the ANN, the training data and structure of the ANN are modified and the design process is repeated.

5.1.6 ANFIS for Prediction of Power Factor

While implementing the ANFIS model in MATLAB, the basic parameters such as the number of inputs; the number of membership functions (MFs) and their type/shape; and the number of training epochs are to be defined. A small change

in any one of these parameters, can affect the system's operation thus leading to inefficient results. The major advantages of ANFIS are:

- ANFIS uses the neural network's ability to classify data and find patterns.
- A fuzzy expert system is more transparent to the user and also less likely to produce memorization errors than a neural network.
- ANFIS maintains the advantages of a fuzzy expert system, thus reducing the need for an expert.

In general, ANFIS requires a large amount of training data to develop an accurate system. In complex data analysis, fuzzy logic reduces the difficulty in modeling and behaves like a human in including the aspects in terms of its mapping rules. Likewise, ANNs are also more popular in identifying models of complex systems. Thus, ANNs and FL are combined, referred to as ANFIS, to take advantage of the learning capabilities of ANNs and modeling superiority of FL.

The model is designed to predict the power factor of wind turbines according to the following steps:

(i) the input and output data were divided into two groups for training and testing;
(ii) a fuzzy model was created using the ANFIS editor and data training was carried out;
(iii) the test data were utilized for the validation of the model.

A hybrid ANFIS algorithm based on the Sugeno system improved by Jang is used for acquiring optimal output data in the experiment. The algorithm consists of the least-squares method and the back-propagation algorithm. The first method was used for optimizing the consequent parameters, while the second method in relation to fuzzy sets was employed to arrange the premise parameters. The input parameters in the application are

– type of profile
– Schmitz coefficient
– end losses
– profile type losses based on the type of profile

The output parameter is the power factor. The experiment was conducted for two different profile types namely, LS-1 and NACA4415. The training and testing data used for ANFIS model were formed from the characteristic values shown in Figs. 5.3 and 5.4 for the two profile types considered. The membership function of the model outputs was selected to be Gaussian (gaussmf). A separate data set, not included in the training set, was employed for verifying the ANFIS model generalization capabilities.

5.1.7 Estimation of the Optimal Power Factor

The ANN and ANFIS approaches are applied to profile types, LS-1 and NACA4415. The number of input neurons was chosen as seven inputs and the input variables are described according to

$$x(t) = A, \lambda_A, C_{pschmitz}, \eta_{end}, \eta_{profile}, \eta_{eddy}, \eta_{blade\ number}, \tag{5.30}$$

where A is an integer number representing the type of profile, 1 for LS-1 and 2 for NACA4415; λ_A is the tip speed ratio, $C_{pschmitz}$ is the Schmitz coefficient mentioned previously; η_{end} represents the end losses for 3 and 4 blade turbines. $\eta_{profile}$ is the profile type losses for the type of profiles considered, LS-1 and NACA4415. On the other hand, the output variables take the form:

$$y(t) = \left[C_{popt3}, C_{popt4}\right], \tag{5.31}$$

where C_{popt3}, C_{popt4} are the power factors for the wind turbines with 3 and 4 blades. The appropriate ANN structure, back propagation is chosen and the network is formed by presenting the inputs to the network. The network is then trained to verify of the structure is efficient in producing the output variables from the given inputs. From the chosen profile, for all input and output variables 30 training samples were formed from the characteristic values given in Figs. 5.3 and 5.4.

The observations of the ANFIS and ANN methods in terms of power factors are presented in Tables 5.1 and 5.2. The performance of the network is satisfactory with

Table 5.1 Comparison of power factor obtained by ANFIS and conventional method

	ANFIS		Conventional method			
Test no	C_{popt3}	C_{popt4}	C_{popt3}	C_{popt4}	Error (%)	
1	0.4530	0.4650	0.4528	0.4646	−0.04	−0.08
2	0.4576	0.4665	0.4576	0.4665	0.00	0.00
3	0.4550	0.4630	0.4553	0.4627	−0.06	−0.06
4	0.3515	0.3893	0.3510	0.3900	−0.14	0.17
5	0.4520	0.4690	0.4520	0.4690	0.00	0.00
6	0.4641	0.4800	0.4650	0.4800	0.19	0.00
7	0.4895	0.4970	0.4890	0.4960	0.01	−0.20
8	0.3695	0.3896	0.3700	0.3890	0.13	−0.15
9	0.3309	0.3381	0.3320	0.3390	0.33	0.26

Table 5.2 Comparison of power factor obtained by ANN and conventional method

	By ANN		By CM			
Test no	Cp_{opt3}	Cp_{opt4}	Cp_{opt3}	Cp_{opt4}	Error (%)	
1	0.4531	0.4664	0.4528	0.4646	−0.06	−0.38
2	0.4576	0.4676	0.4576	0.4665	0.00	−0.23
3	0.4566	0.4649	0.4553	0.4627	−0.28	−0.47
4	0.3525	0.3888	0.3510	0.3900	−0.42	0.30
5	0.4505	0.4702	0.4520	0.4690	0.33	−0.25
6	0.4637	0.4808	0.4650	0.4800	0.27	−0.16
7	0.4890	0.4991	0.4890	0.4960	0.00	−0.62
8	0.3689	0.3916	0.3700	0.3890	0.29	−0.66
9	0.3298	0.3374	0.3320	0.3390	0.66	0.47

small deviation from the values obtained from the curves given in Figs. 5.3 and 5.4. The conventional values are also presented for computation of the percentage of error.

Therefore, it can be stated that the presented methodology provides more detailed values in an attempt to obtain the optimal power factor wind turbines rather than using the small number of data obtained from the curves whose derivations require rather complicated process. The results indicate that both ANN and ANFIS models can give good predictions of the power factor. However, the ANFIS model performs better than ANN model.

5.2 Pitch Angle Control

Generally, wind turbines are capable of operating with either fixed speed or variable speed. The variable-speed wind turbines involving pitch adjustment have become more dominant during the recent years. The reasons for the increase in choice of variable speed wind turbines during the recent years are the reduction of both the mechanical structure stresses and acoustic stresses, and the possibility to control active and reactive power. In fact, variable speed operation increases the energetic efficiency and reduces the drive train torque and the energy of generating flow unsteadiness. Thus pitch angle control is necessary when the rotational speed is kept constant. Very small changes in the pitch angle can lead to a massive change on the output power. Pitch angle control is expressed as:

1. Optimization of power output: Below rated wind speed, the pitch setting should be at its optimum value to give maximum power.
2. Prevention of input mechanical power from exceeding design limits: Above rated wind speed, pitch angle control provides a very effective means of regulating the aerodynamic power and loads produced by the rotor.
3. Minimizing fatigue loads of the turbine mechanical component.

While designing the pitch angle controller, the effect on loads must be considered and the controller should ensure that excessive loads will not result from the control action. For explicit design of the controller with the objective of reducing the effect of soft-computing methodologies, such as artificial intelligence and genetic algorithms are proving to be more effective during recent years. In this section, conventional pitch angle control strategy, using the proportional and integral (PI) controller is discussed. In addition, fuzzy logic and genetic algorithms are applied for pitch angle control since these intelligent algorithms do not need a well known system and are potential for the system with strong non-linearity.

5.2.1 Problem Definition

The power in the wind is proportional to the cube of the wind speed and may be expressed as

$$P = 0.5\rho A\nu_w^3 \tag{5.32}$$

where ρ is air density, A is the area swept by blades and ν_w is wind speed. A wind turbine can only extract part of the power from the wind, which is limited by the Betz limit (maximum 59 %). This fraction is described by the power coefficient of the turbine, Cp, which is a function of the blade pitch angle and the tip speed ratio. Therefore the mechanical power of the wind turbine extracted from the wind is

$$P_w = 0.5C_p(\beta, \lambda)\rho A\nu_w^3 \tag{5.33}$$

where Cp is the power coefficient of the wind turbine, β is the blade pitch angle and λ is the tip speed ratio. The tip speed ratio is defined as the ratio between the blade tip speed and the wind speed

$$\lambda = \frac{\Omega R}{\nu_w} \tag{5.34}$$

where Ω is the turbine rotor speed and R is the radius of the wind turbine blade.

Thus any change in the rotor speed or the wind speed induces change in the tip speed ratio leading to power coefficient variation. In this way, the generated power is affected.

5.2.2 Fuzzy Logic Controllers

The ability to model the fuzzy, or imprecise, membership of an element to a set enables inference based on linguistic terms. Production rules governing a fuzzy controller can be described using words or simple sentences in natural language as opposed to formal predicate calculus statements. This enables a domain expert, who typically would not have an advanced knowledge of first-order predicate logic, to describe the rules that govern a given system using domain specific linguistic terms, which may be better understood. Figure 5.5 outlines a simple architecture for a fuzzy controller consisting of three primary components. First, the condition interface, which is responsible for converting outputs from the system into a fuzzy form, hence the term fuzzifier, utilized by the fuzzy inference engine. Next, the engine performs inference, based on linguistic rules, to determine an appropriate control action. Finally, the action interface is responsible for interpreting the output of the inferencing process and converting it back into system specific actions

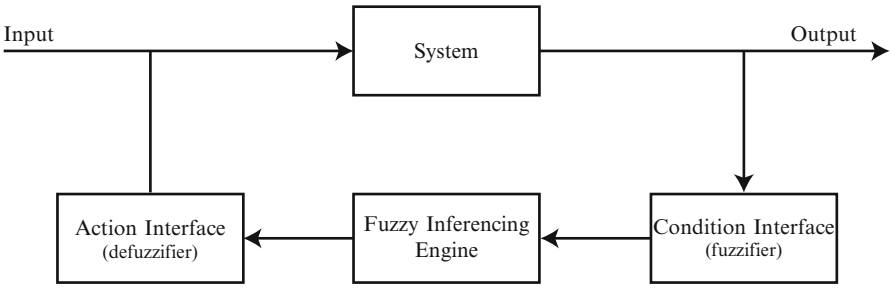


Fig. 5.5 Fuzzy controller architecture

through a process known as defuzzification. Thus, a feedback loop is realized where the controller constantly monitors the system while effecting control actions on the system according to its rule base.

After fuzzifying the inputs, the next step is to perform inferencing using the fuzzy rule base. Typically, the rule base is made up of a list of rules of the form:

$$if\ antecedent \rightarrow consequent \tag{5.35}$$

where the *antecedent* consists of one or more fuzzy sets combined using the operators to form a logical expression. In the case of a Mamdani controller, the *consequent* consists of a single target fuzzy set. The value of the antecedent, also known as the firing strength of the rule, determines the degree of membership to the target set in the *consequent*.

5.2.3 Genetic Algorithms

Genetic Algorithms (GAs) are adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetic. The basic concept of GAs is designed to simulate processes in natural system necessary for evolution, specifically those that follow the principles first laid down by Charles Darwin of survival of the fittest. As such they represent an intelligent exploitation of a random search within a defined search space to solve a problem. First pioneered by John Holland in 1960s, Genetic Algorithms has been widely studied, experimented and applied in many fields in engineering worlds. Not only does GAs provide alternative methods to solving problem, it consistently outperforms other traditional methods in most of the problems link. Many of the real world problems involved finding optimal parameters, which might prove difficult for traditional methods but ideal for GAs. However, because of its outstanding performance in optimization, GAs has been wrongly regarded as a function optimizer. In fact, there are many ways to view genetic algorithms. Perhaps most users come to GAs looking for a problem solver, but this is a restrictive view.

GAs are used as problem solvers, as challenging technical puzzle, as basis for competent machine learning, as computational model of innovation and creativity, as computational model of other innovating systems and for guiding philosophy.

GAs were introduced as a computational analogy of adaptive systems. They are modeled loosely on the principles of the evolution via natural selection, employing a population of individuals that undergo selection in the presence of variation-inducing operators such as mutation and recombination (crossover). A fitness function is used to evaluate individuals, and reproductive success varies with fitness.

5.2.3.1 The Algorithms

1. Randomly generate an initial population.
2. Compute and save the fitness for each individual in the current population.
3. Define selection probabilities for each individual in so that is proportional to the fitness.
4. Generate the next population by probabilistically selecting individuals from current population to produce offspring via genetic operators.
5. Repeat step 2 until satisfying solution is obtained.

The paradigm of GAs described above is usually the one applied to solving most of the problems presented to GAs. Though it might not find the best solution. more often than not, it would come up with a partially optimal solution.

5.2.4 Conventional Pitch Angle Control

Under strong wind speeds, the performance of the wind turbine can be regulated by adjusting the pitch angle of the blades. During normal operation, the expected blade pitch adjustments are rotational speeds of approximately 5–10 %. Here the chosen pitch rate is 8°/s which avoids excessive loads during normal regulation procedures. The conventional blade pitch angle control strategies are shown in Fig. 5.6. The pitch angle reference, β_{ref}, is controlled by the input values, such as the wind speed, generator rotor speed and the generator power described as follows:

1. Wind speed (Refer Fig. 5.6a). In general the pitch angle reference can be obtained by plotting a curve between pitch angle and wind speed. This type of pitch angle control strategy is simple since the wind speed is measured directly. This method does not measure the wind speed directly and hence it is not an appropriate procedure.
2. Generator rotor speed (Refer Fig. 5.6b). The difference between the rotor speed and reference is determined. The error signal is then sent to the PI controller and produces the reference value of the pitch angle, β_{ref}.
3. Generator power (Refer Fig. 5.6c). The error signal of the generator power is sent to a PI controller. The PI controller produces the reference pitch angle β_{ref}.

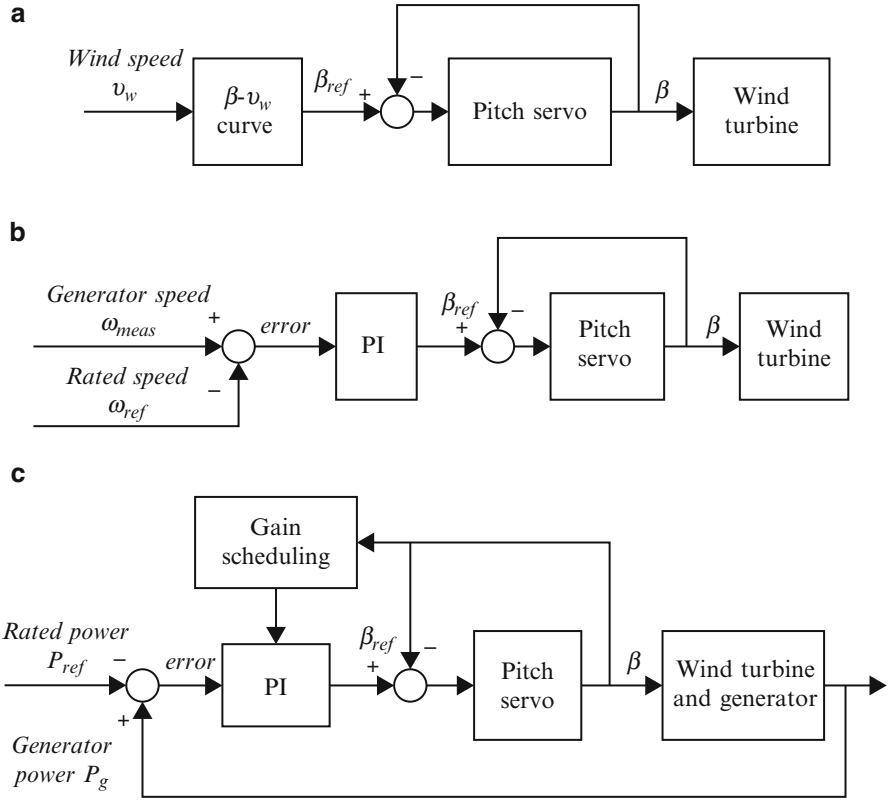


Fig. 5.6 Pitch angle control strategy. (**a**) Wind speed; (**b**) generator rotor speed; (**c**) generator power

The variation of the pitch angle and wind speed during large wind speeds is non-linear in fashion and hence requires a non-linear control strategy. For a wind speed close to the rated speed, the sensitivity of aerodynamic torque to pitch angle is very small. Hence a larger controller gain is required. A very small change in pitch can have a large effect on torque. Frequently the torque sensitivity changes almost linearly with pitch angle, and so can be compensated by varying the overall gain of the controller linearly in inverse proportion to the pitch angle. This process of modifying the gain with operating point is termed as gain scheduling. This type of conventional pitch angle control strategy with gain scheduling is best suitable when the system dynamic is not strong. On the other hand, the sensitivity of aerodynamic torque to pitch angle varies in a different way which is treated as linear variation in gain scheduling, and because of its effect on turbine dynamics which couples strongly with the pitch controller, it may be necessary to modify the gain scheduling further to ensure good performance.

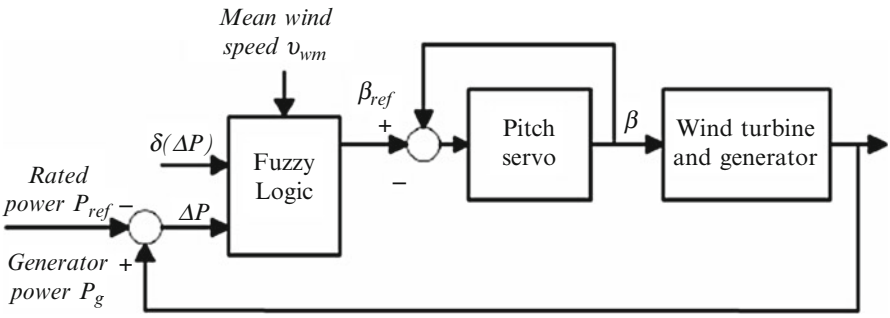


Fig. 5.7 Fuzzy logic control strategy

5.2.5 *Fuzzy Logic for Pitch Control*

The major steps in designing a pitch control strategy based on fuzzy logic consists of determining the inputs, setting up the rules and designing a method to convert the fuzzy result of the rules into output signal, known as defuzzification. The fuzzy logic controller for pitch control should be modeled in order to reduce the fatigue loads. The model for pitch angle control based on fuzzy logic system is shown in Fig. 5.7. For both input and output triangular symmetrical membership functions are more suitable, since they are capable of providing more sensitivity especially as variables approach to zero value. The width of variation can be adjusted according to the system parameters.

As shown in Fig. 5.7, the proposed fuzzy logic controller is based on the power deviation from its reference value ΔP, its variation during a sampled time $\delta(\Delta P)$ as follows:

$$\begin{cases} \Delta P = P_{ref} - P_g \\ \delta(\Delta P) = \Delta P_n - \Delta P_{n-1} \end{cases} \tag{5.36}$$

where *Pref* is the rated power of the system and *Pg* is the measured generator power. The mean wind speed ν_{wm}, used as the third input variable, is useful to compensate the non-linear sensitivity of pitch angle to the wind speed. Table 5.3 lists the control rules for the input and output variable. In the proposed fuzzy system, nine fuzzy sets have been considered for variables: negative large (NL), negative medium large (NML), negative medium (NM), negative small (NS), zero (ZE), positive small (PS), positive medium (PM), positive medium large (PML), positive large (PL). At lower mean wind speed, the low sensitive of the pitch angle control means that a large reaction of pitch angle to the control variants will be needed than at higher mean wind speed.

Table 5.3 Rules of fuzzy logic controller

υ_m	PS					PM					PL				
ΔP	NL	NS	ZE	PS	PL	NL	NS	ZE	PS	PL	NL	NS	ZE	PS	PL
δΔP															
NL	NL	NML	NM	NM	PS	NL	NM	NM	NS	PS	NML	NM	NS	NS	PS
NS	NL	NM	NS	PS	PM	NML	NM	NS	PS	PM	NML	NM	NS	ZE	PS
ZE	NML	NS	ZE	PS	PML	NM	NS	ZE	PS	PM	NM	NS	ZE	PS	PM
PS	NM	NS	PS	PM	PL	NM	NS	PS	PM	PML	NS	ZE	PS	PM	PML
PL	NS	PM	PM	PML	PL	NS	PS	PM	PM	PL	NS	PS	PS	PM	PML

The fuzzy rule base is defined based on Table 5.3. An immediate and easy understanding of the controller logic can be obtained as following considerations:

1. If ΔP and $\delta(\Delta P)$ are negative large, the output power angle is too large and its amplitude is growing, consequently current pitch angle must be fast decreased.
2. If ΔP is negative large and $\delta(\Delta P)$ is positive large, the output power is higher than its reference, but since its amplitude is decreasing, pitch angle variation must be small.
3. If ΔP is small pitch angle variation must be smoothed because too large variations excite oscillatory modes.

5.2.6 *Genetic Algorithm Controller for Pitch Angle Control*

Genetic algorithms (GA) are commonly used as optimization techniques, thus providing high quality solutions. In this section, the GAs are used as controllers rather than optimizers. The proposed Genetic algorithm controller approach can be better explained by considering the turbine power curves shown in Fig. 5.8.

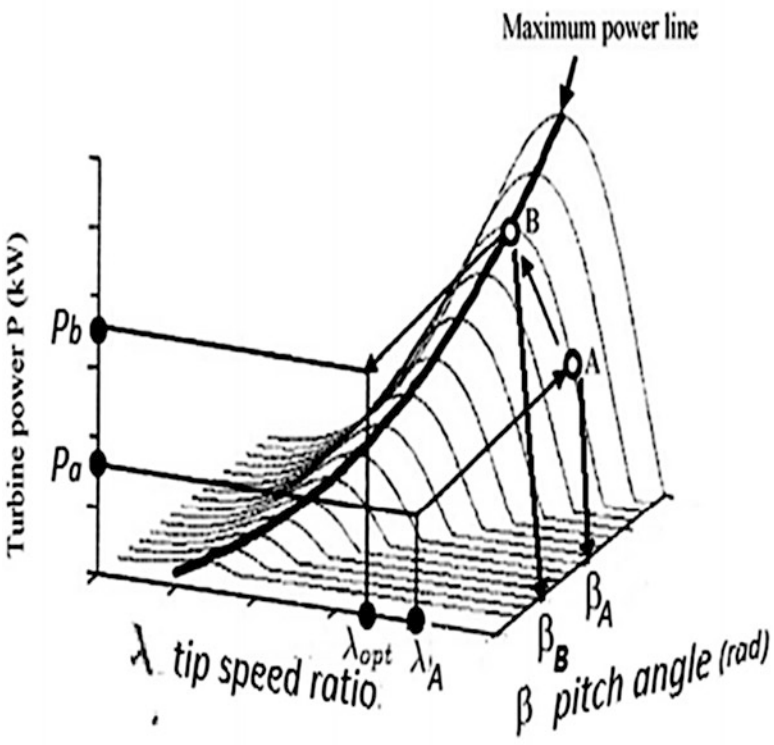


Fig. 5.8 Turbine power curves

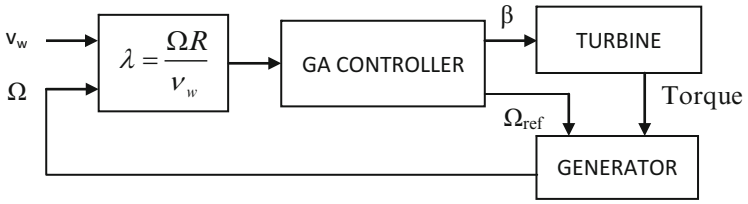


Fig. 5.9 Genetic algorithm based pitch control

Assuming that the wind turbine operates initially at point A, the fuzzy control system from measured speed ratio $\lambda_A = \frac{\Omega_A R}{v_w}$ and β_A pitch angle, turbine power P_w can derive the corresponding optimum operating point B, giving the desired rotor speed reference $\lambda_B = \frac{\Omega_B R}{v_w}$ and send a control signal for new β_B. The GA based pitch control strategy is shown in Fig. 5.9. Therefore, the generator speed will be controlled in order to reach the speed Ω_A and β allowing the extraction of the maximum power from the turbine.

Pitch angle control is the most common means to control the aerodynamic power generated by the wind turbine rotor. Pitch angle control also has an effect on the aerodynamic loads which may be controlled by the controller to achieve lower torque peak as well as lower fatigue loads. The conventional pitch angle control strategy using different controlling variables can be implemented. However, fuzzy logic and genetic algorithm pitch angle control strategy need not well know about the wind turbine dynamics and these algorithms are more suitable when wind turbine contains strong non-linearities. The soft computing techniques can results in lowest fatigue loads and lower torque peak and lower power peak. The pitch angle of the fuzzy logic controller is less active than conventional pitch angle control with power or wind speed controlling variable, which causes less dynamic torque.

5.3 MPPT for WECS

MPPT controllers are used for extracting maximum power from the WECS using different generators such as permanent magnet synchronous generators (PMSG), squirrel cage induction generators (SCIG) and doubly fed induction generator (DFIG). The MPPT controllers can be classified into three main control methods, namely tip speed ratio (TSR) control, power signal feedback (PSF) control and hill-climb search (HCS) control. The energy produced from wind source depends upon the accuracy with which the peak power points are tracked by the MPPT controller of the WECS control system irrespective of the type of generator used. Based on the control schemes, the maximum power extraction algorithms are tip speed ratio (TSR) control, power signal feedback (PSF) control and hill-climb search (HCS) control.

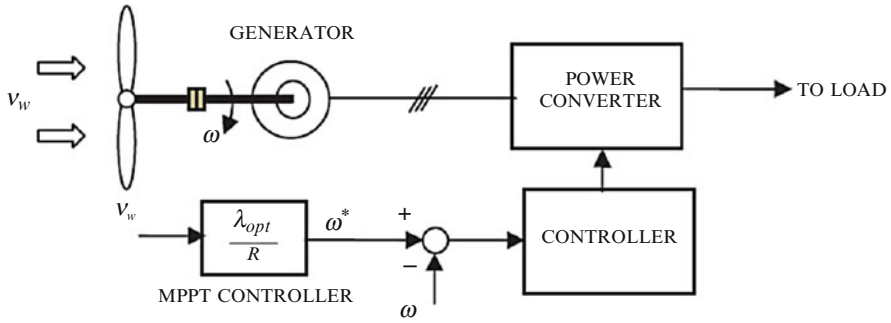


Fig. 5.10 Tip speed ratio control of WECS

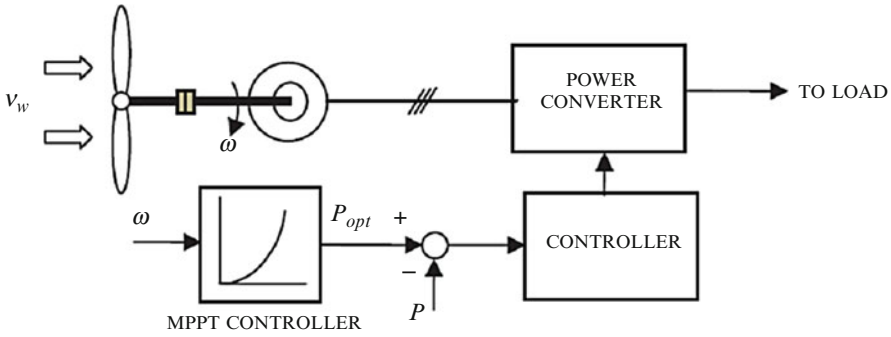


Fig. 5.11 Power signal feedback control

In the TSR control method, the rotational speed of the generator is regulated in order to maintain the tip speed ratio to an optimum value during which the power extracted attains the maximum. In this method in order to obtain the maximum output power, the speed from both wind and the turbine are required along with the optimal value of the TSR of the turbine. The block diagram of a WECS with TSR control is shown in Fig. 5.10.

In PSF control, it is required to have the knowledge of the wind turbine's maximum power curve, and track this curve through its control mechanisms. The maximum power curves need to be obtained via simulations or off-line experiment on individual wind turbines. In this method, reference power is generated either using a recorded maximum power curve or using the mechanical power equation of the wind turbine where wind speed or the rotor speed is used as the input. Figure 5.11 shows the block diagram of a WECS with PSF controller for maximum power extraction.

The HCS control algorithm continuously searches for the peak power of the wind turbine. It can overcome some of the common problems normally associated with the other two methods. The tracking algorithm, depending upon the location of the operating point and relation between the changes in power and speed, computes

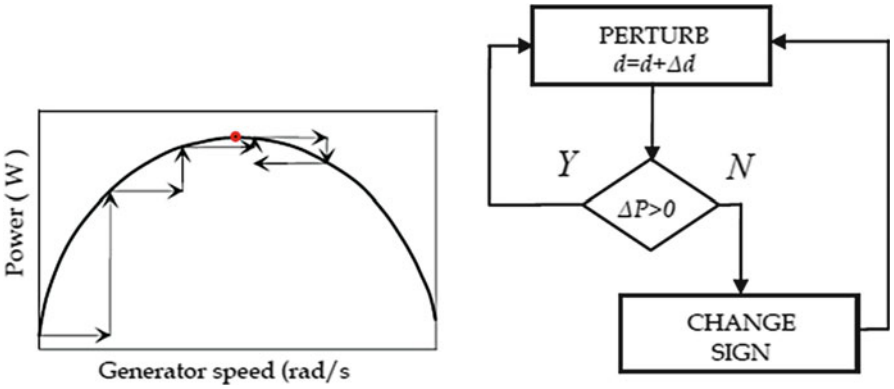


Fig. 5.12 HCS control principle

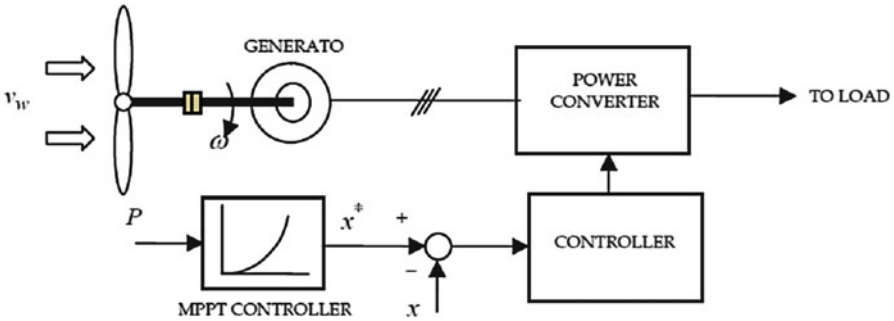


Fig. 5.13 WECS with hill climb search control

the desired optimum signal in order to drive the system to the point of maximum power. Figure 5.12 shows the principle of HCS control and Fig. 5.13 shows a WECS with HCS controller for tracking maximum power points.

5.3.1 Fuzzy Logic Based MPPT Controller

Though the implementation of the traditional hill climb search algorithm is simple and independent of turbine characteristics, the selection of step size is a very important concern. MPP values can be tracked with a large step size at a faster rate, but leads to severe oscillations around the maximum power point. A small value of step size is capable of decreasing the oscillations around the MPP to a much minimum value, but the MPPT process slows down during varying wind speeds. Thus fuzzy logic control algorithms with variable step size are used recently for tracking the MPP effectively and smoothly.

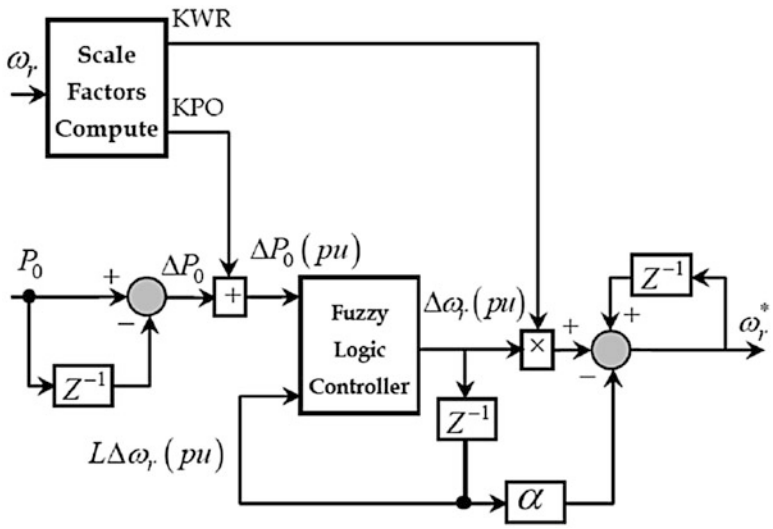


Fig. 5.14 Block diagram of fuzzy logic MPPT controller

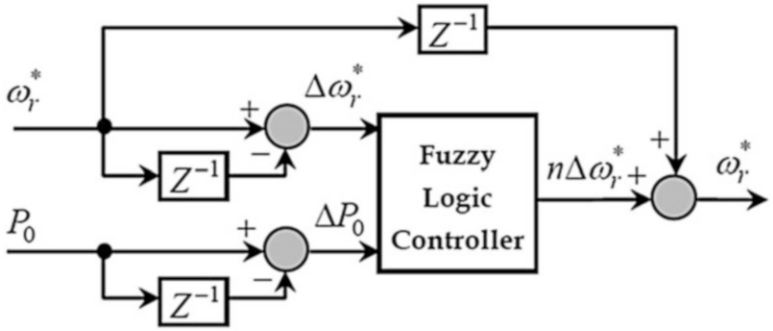


Fig. 5.15 Fuzzy MPPT controller

The fuzzy logic controller is used in MPPT to perturb the generator reference speed and to estimate the corresponding change of output power *P0*. The decision process is based on the output power. The searching process continues in the same direction, if the output power increases with the last increment. Vice versa, if the output power reduces with the last increment, then the direction of search is reversed. The block diagram of the fuzzy logic controller is shown in Fig. 5.14. The fuzzy logic controller is efficient to track the maximum power point, especially in case of frequently changing wind conditions. Two inputs $\Delta\omega_r^*$, Δw_{ref} and ΔP_0 in Fig. 5.15 are used as the control input signals and the output of the controller is the new speed reference speed which, after adding with previous speed command, forms the present reference speed.

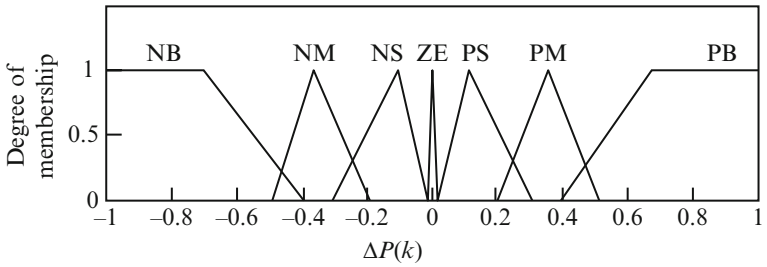


Fig. 5.16 Member functions of input variables $\Delta P(k)$

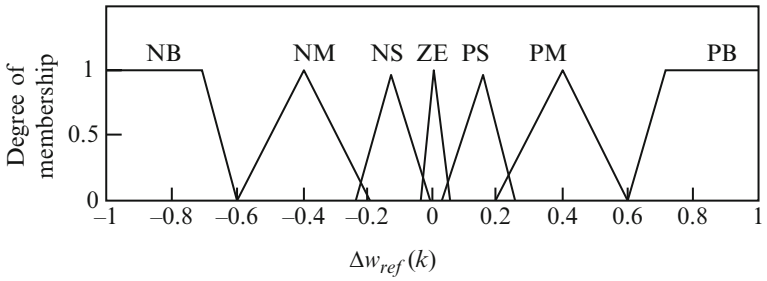


Fig. 5.17 Member functions of output variable $\Delta w_{ref}(k)$

The membership functions corresponding to the input variable ΔP(k) for the implemented FLC is defined as follows (Figs. 5.16 and 5.17): PB (positive big), PM (positive medium), PS (positive small), ZE (zero), NS (negative small), NM (negative medium), and NB (negative big), respectively; the member functions corresponding to the input variable Δw(k) are P (positive), Z (zero), and N (negative), respectively.

The member functions of output variable $\Delta w_{ref}(k)$, are PB, PM, PS, ZE, NS, NM, NB, respectively almost similar to that of the input variable. The following equations define the relationship between turbine mechanical power and turbine rotational speed depending on the $P - w$ curve.

$$dP/dw > 0, \ (w < w_{mpp})$$
$$dP/dw = 0, \ (w = w_{mpp})$$
$$dP/dw < 0, \ (w > w_{mpp})$$

where w_{mpp} denotes the turbine rotational speed corresponding to the MPP value. The fuzzy rule base is formed based on properties of wind turbine, as shown in Table 5.4. The reference rotational speed is updated according to $w_{ref}(k) = w(k-1) + \Delta w(k)$. Thus the FLC is capable of changing the step size dynamically based on the wind speed, thus enabling the turbine to track the MPP efficiently.

Table 5.4 Rules for the fuzzy logical controller

	$\Delta P(k)$						
$\Delta w(k)$	NB	NM	NS	ZE	PS	PM	PB
N	PB	PM	PS	NS	NS	NM	NB
Z	NM	NS	NS	ZE	PS	PS	PM
P	NB	NM	NS	PS	PS	PM	PB

5.4 Economic Dispatch for Wind Power System

The increased rate of power consumption has lead to development of alternate energy source. Several renewable energy sources are responsible for today's electric network. The characteristics of these energy resources are based on operational costs and reliability. In this section, Economic Environmental Dispatching (EED) of power system including wind energy is discussed based on evolutionary algorithms. Renewable energy resources depend on the data of the climate such as the wind speed for wind energy, solar radiation and the temperature for solar thermal energy. The problem focuses on minimizing the cost and reducing of the emissions of the polluting gases.

In traditional power systems, power can be dispatched economically using static and dynamic methods. The static optimal dispatch only seeks to achieve an optimal objective for the power system at a specific time, but will not take into account of the intrinsic link between system at different time moments; while the dynamic optimal dispatch takes into account of the coupling effect of system at different time moments, such as the limit on the climbing rate of a generator. Thus the computation process is more complex in the dynamic dispatch while the computation results are more in line with actual requirements.

Moreover, economic dispatch of power systems containing wind power farms adopts dynamic models due to the random changes in wind speed. In addition, dynamic economic dispatch requires the knowledge of output data of the wind power farm at every moment in the optimal duration. Thus artificial intelligent methods have been developed to minimize the error of predicting output for a wind power farm, thus reducing the difficulty for economic dispatch. In this section, techniques such as Quantum Genetic Algorithm (QGA) and Strength Pareto Evolutionary Algorithm (SPEA) are adopted to solve the problem.

5.4.1 Mathematical Model of Economic Dispatch for Power System Based on Wind Energy

The wind power consumed from nature has to dispatched immediately, since there is no fuel consumption in wind power systems. The economic dispatch of power

systems containing wind energy is to minimize the cost of generating power using a traditional power generator group. The objective function is defined as:

$$Min\ F_T = \min \sum_{t=1}^{T} \sum_{i=1}^{N} F_{it}(P_{Git}) + F_{cp} \tag{5.37}$$

where F_T is the total cost for generating power; T is the number of hours in the research period; N is the total number of conventional generators within the system; P_{Git} is the output of active power for the i^{th} generator at time t; $F_{it}(P_{Git})$ is the corresponding cost of consumption and is usually expressed using the following polynomial equation:

$$F_i(P_{Gi}) = a_i + b_i P_{Gi} + ciP_{Gi}^2 \tag{5.38}$$

where a_i, b_i, and c_i are the coefficients for the cost function $F_{it}(P_{Git})$, F_{cp} is the environmental cost function. In considering the emission characteristics of thermal generating unit in the economic dispatch model of electricity market, the environmental cost of thermal generating units is added on top of the cost of generating electricity for these units. This added cost:

$$F_{cp} = \sum_{i=1}^{n} M_{cpi} \times f_{di} \tag{5.39}$$

where M_{cpi} is the environmental cost coefficient, f_{di} is the emission amount of the ith thermal generating unit, n is the number of thermal generating units in the system. From the point of view of environmental protection, the emission characteristics for each unit can be represented by emission per unit time converting to the weight of the amount of NOx discharged. The emission equation:

$$f_{di} = \alpha_i + \beta_i P_{Gi} + \gamma_i P_{Gi}^2 + \delta_i \times \exp(\theta_i \times P_{Gi}) \tag{5.40}$$

where $\alpha_i, \beta_i, \gamma_i, \delta_i$ *and* θ_i are the emission characteristics coefficients for the i^{th} thermal generating unit, which can be measured.

5.4.1.1 Constraints

Power conservation constraint is given as

$$\sum_{i=1}^{N} P_{Git} + \sum_{j=1}^{N_w} P_{Gjt}^{w} = P_{Dt} + P_L \tag{5.41}$$

where N_w is the total number of wind power plants in the system; P_{Gjt}^{w} is the output of active power for the jth wind power plant at time t; P_{Dt} is the load at time t.

$$P_L = \sum_{i=1}^{N_G}\sum_{j=1}^{N_G} P_{Gi}B_{ij}P_{Gj} + \sum_{i=1}^{N_G} B_{0i}P_{Gi} + B_{00} \tag{5.42}$$

where P_L is the line loss, B_{ij}, B_{0i} and B_{00} are the loss parameters also known as the B parameters.

Operation constraints are given by

$$P_{Git}^{\min} \leq P_{Git} \leq P_{Git}^{\max} \tag{5.43}$$

where $P_{Git}^{\min}$ and $P_{Git}^{\max}$ are the minimum and maximum generator limits corresponding to the i^{th} unit.

5.4.2 Quantum Genetic Algorithm (QGA) for Economic Dispatch of Wind Power System

QGA embeds the concept of quantum bit and quantum superposition state into traditional genetic algorithm to form Quantum genetic algorithm. The chromosomes are encoded using quantum bits and quantum chromosomes are generated using quantum encoding. The quantum probability amplitude implies that a quantum chromosome carries information about multiple states, a chromosome will be in a quantum superposition state of many determined states before an observation is made on it. Therefore, generating a new object through the quantum probability amplitude, the deciding variable, in a sense, is no longer fixed information; rather, it has become a kind of information carrying a superposition of different information. Thus it can bring a richer population than the simplistic application of genetic method. In addition, mutations can be induced with the information of current best individual object so that the population will evolve toward a good schema with high probability to speed up the convergence and, at the same time, prevent it from being trapped in a local optimal solution effectively and prevent the premature phenomenon from occurring.

5.4.2.1 Step 1: Initialize Population

Quantum Bit The smallest information unit in QGA is a quantum bit. The state of a quantum bit can be either 0 or 1, which can be represented as:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{5.44}$$

where α, β, represent two complex numbers representing the probability of occurrence for the corresponding state: $\left(|\alpha|^2 + |\beta|^2\right) = 1$, $|\alpha|^2 and\ |\beta|^2$ represent the probability of the quantum bit in the state of 0 and 1 respectively.

Quantum Chromosome Frequently used coding methods in EA are binary, decimal, and symbolic coding. In QGA, a new coding method based on quantum bit is adopted, i.e. using a pair of complex numbers to define a quantum bit. A system with m quantum bits can be described as

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \ldots & \alpha_m \\ \beta_1 & \beta_2 & \ldots & \beta_m \end{array}\right] \tag{5.45}$$

This representation method can be used to express any linear superposition of states.

5.4.2.2 Step 2: Perform Coding and Individual Measurement for the Generating Units in the Population

QGA is a probabilistic algorithm similar to EA. The algorithm flowchart is shown in Fig. 5.18. $H(t) = \left\{Q_1^t, Q_2^t, \ldots Q_h^t, \ldots Q_l^t\right\}$, (h = 1, 2,. . ., l) where h is the size of the population, $Q_l(t) = \left\{q_1^t, q_2^t, \ldots q_j^t, \ldots q_n^t\right\}$, where n is the number of generator unit, t is the evolution generation, q_j^t is the binary coding of the generation volume of the jth generator unit. Its chromosome is defined as follows:

$$q_j^t = \left[\begin{array}{c|c|c|c} \alpha_1^t & \alpha_2^t & \ldots & \alpha_m^t \\ \beta_1^t & \beta_2^t & \ldots & \beta_m^t \end{array}\right] \tag{5.46}$$

For (j = 1, 2,. . ., n) (m is the length of the quantum chromosome).

During the "initialization of H(t)," if α_i^t, β_i^t (i = 1, 2,. . ., m) in $Q_l(t)$ and all the q_j^t are initialized, it means that all the possible linear superposition states will occur with the same probability. During the step of "generating S(t) from H(t)", a common solution set S(t) is generated by observing the state of H(t), where in the t^{th} generation, $S(t) = \left\{P_1^t, P_2^t, \ldots P_h^t, \ldots P_l^t\right\}, P_l = \left\{x_1^t, x_2^t, \ldots x_j^t, \ldots x_n^t\right\}$, every x_j^t (j = 1, 2,. . ., n) is a series, (x_1, x_2,. . ., x_i. . ., x_m), of length m, which are obtained from the amplitude of quantum bit $|\alpha_i^t|^2$ or $|\beta_i^t|^2$, (i = 1, 2,. . ., m). The corresponding process in the binary situation is to generate a [0, 1] number randomly. Take "1" if it is larger than $|\alpha_i^t|^2$ take "0" otherwise.

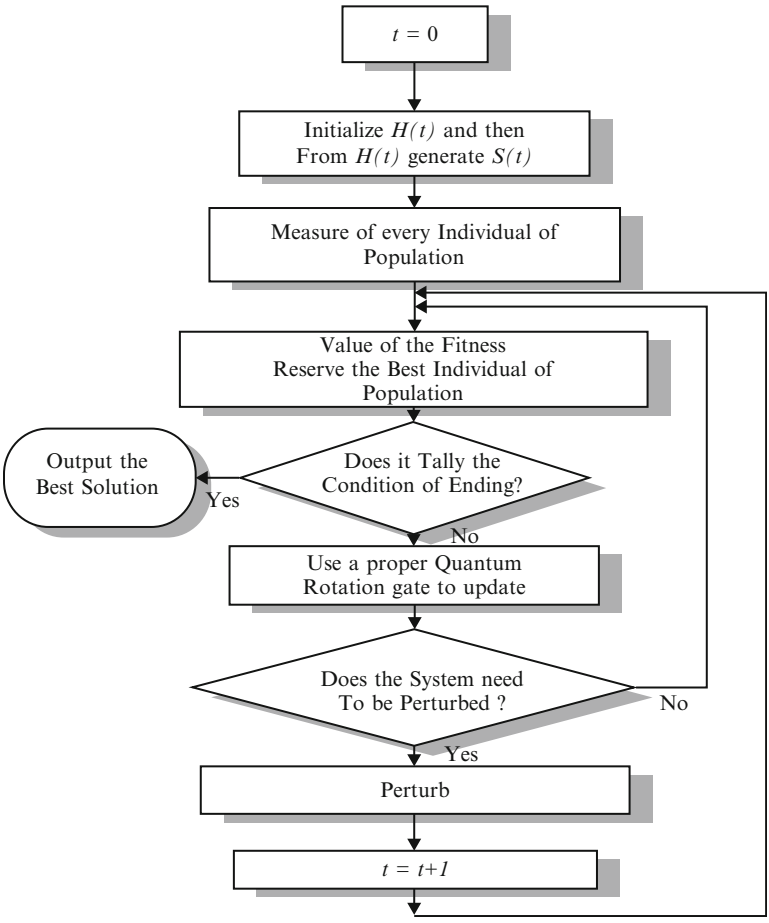


Fig. 5.18 Flowchart of QGA

5.4.2.3 Step 3: Perform Individual Measurement to Each Object in S(t)

Use a fitness evaluation function to evaluate each individual object in S(t) and keep the best object in the generation. If a satisfactory solution is obtained, stop the algorithm; otherwise, continue to the fourth step.

5.4.2.4 Step 4: Use a Proper Quantum Rotation gate U(t) to Update S(t)

The traditional genetic algorithm uses mating, mutation, etc. operations to maintain the diversity of the population. Quantum genetic algorithm applies logic gate to the probability amplitude of quantum state to maintain the diversity of the population. Therefore, the update method using a quantum gate is the key to the quantum

genetic algorithm. In the traditional genetic algorithm, binary system, adaptation values, and probability amplitude comparison method are used for update using a quantum gate. This update method using a quantum gate is suitable to find solutions for combinatorial optimization problems with known optimal solution in principle. However, for the actual optimization problems, in particular, those multi-variable continuous function optimization problems, their optimal solutions are not known beforehand in principle. Therefore, here a quantum rotation gate of quantum logic gate is adopted for the new quantum genetic algorithm.

$$U = \begin{bmatrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{5.47}$$

where θ is the quantum gate rotation angle. Its value is taken as,

$$\theta = k.f(\alpha_i, \beta_i) \tag{5.48}$$

$$k = \pi.\exp\left(-\frac{t}{iter_{\max}}\right) \tag{5.49}$$

We define k as a variable related to the generation of the evolution so that it will adjust the size of the grid self-adaptively. Where t is the evolution generation, p is an angle, $\text{iter}_{\max}$ is a constant depending on the complexity of the optimization problem. The purpose of the function $f(\alpha_i, \beta_i)$ is to make the algorithm search along the optimal direction. Its principle is to make the current search solution approach the optimal solution gradually and thereby, determine the direction of the quantum rotation gate. In this way, the procedure for applying quantum rotation gate to all probability amplitude for individual object in the population, i.e. using quantum rotation gate U(t) to update S(t), in quantum genetic algorithm can be expressed as:

$$S(t+1) = U(t) \times S(t) \tag{5.50}$$

In the above equation: t is the evolution generation, U(t) is the t^{th} generation quantum rotation gate, S(t) is the t^{th} generation probability amplitude of a certain object, S(t + 1) is the $(t + 1)^{th}$ generation probability amplitude of the corresponding object.

5.4.2.5 Step 5: Perturbation

In order to solve the QGA problem being prone to be trapped in local extreme value better, we perturb the population. It is found that by using the QGA analysis that when the best individual of the current generation is a local extreme value, it is very hard for the algorithm to extricate itself. Therefore, when the best individual does not change in successive generations, the algorithm is trapped in the local extreme. At this point of time, a perturbation should be applied to the population to extricate itself out of the local optimal and start a new search.

5.4.3 Strength Pareto Evolutionary Algorithm (SPEA) Approach

Zitzler and Thiele developed an elitist evolutionary approach to solve a multi objective problem called Strength Pareto Evolutionary Algorithm (SPEA). The elitism in SPEA is introduced by an external Pareto set. The external set stores the non-dominant solutions funded during the resolution of the problem. In order to reduce the size of the external set, an average linkage based on hierarchical clustering algorithm is used without destroying the characteristics of the trade-off front.

The notations are denoted as:

P: the current population.
P_t: the external population.
N_{pop}: the size of current population.
F_i the fitness of an individual i.
S_i the strength of an individual i.

The assignment procedure to calculate the fitness values is the following:

Step 1: For each individual $i \epsilon P_t$ is assigned a reel value $S_i \epsilon (0, 1)$ called strength. S_i is proportional to the number of individuals in the current population dominated by the individual i in the external Pareto set. It can be calculated as follows:

For an individual $j \epsilon P_t$

$$S_i = \frac{|\{j/j \in P_t \ \ and \ \ individual \ \ j \ \ is \ \ dominated \ \ by \ \ i\}|}{N_{pop} + 1} \tag{5.51}$$

The strength of a Pareto solution is also its fitness: $F_i = S_i$

Step 2: The fitness of an individual $j \epsilon P_t$ is the sum of the strengths of all external Pareto individuals $i \epsilon P_t$ dominated by $j \epsilon P_t$. We add one in odder to guarantee that Pareto solutions are most likely to be produced.

$$F_j = 1 + \sum_{i \in P_t, i \ \ dominate \ \ j} S_i \tag{5.52}$$

where $F_j \epsilon (1, N_{pop})$

The clustering algorithm is described by the following steps:

Step 1: To initialise clustering set C; each individual $i \epsilon P_t$ constitutes a distinct cluster:

$$C = Y_{i \in P_t}\{i\} \tag{5.53}$$

Step 2: If the number of cluster is lower or equal to maximum size of external set (N_{pop}), go to step 5. Else, go to step 3.

Step 3: Calculate the distance between each pair of clusters. The distance d_c between two clusters c_1 and $c_2 \epsilon\, C$ is defined as the average distance between two pairs of individuals from each cluster:

$$d_c = \frac{1}{n_1 n_2} \sum_{i_1 \in c_1,\, i_2 \in c_2} d_{i_1, i_2} \tag{5.54}$$

n_1 and n_2 are respectively the numbers of individuals in clusters c_1 and c_2.

Step 4: Find the pair of clusters corresponding to the minimal distance d_c between them and return to step 2.

Step 5: Find the centroid of each cluster. Select the nearest individual in this cluster to the centroid as a representative individual and remove all other individuals from the cluster.

Step 6: Thus, the reduced Pareto set P_{t+1} is computed by uniting these representatives:

$$P_{t+1} = Y_{c \in C} c. \tag{5.55}$$

In this section, a method allowing the resolution of the problem of the Environmental Economic Dispatching of an electrical network including wind energy based on QGA and SPEA was discussed. The optimization was performed without auxiliary elements to extract the maximum of power from the random wind energy and to distribute the remainder of the power on the power stations.

5.5 SEIG Driven by WECS

Various research work and case studies were investigated in self-excited induction generators (SEIG) since 1935. SEIG have been commonly applied due to the basic advantages such as lower maintenance costs, better transient performance, lack of a DC power supply for field excitation, and brushless construction (squirrel-cage rotor), etc. In general, induction generators have been commonly used to operate as wind-turbine generators and small hydroelectric generators of isolated power systems since they can be easily connected to large power systems, to inject electric power.

The generator action takes place, when the rotor speed of the induction generator is greater than the synchronous-speed of the air-gap-revolving field. This section concentrates on the dynamic performance of an isolated SEIG, driven by wind

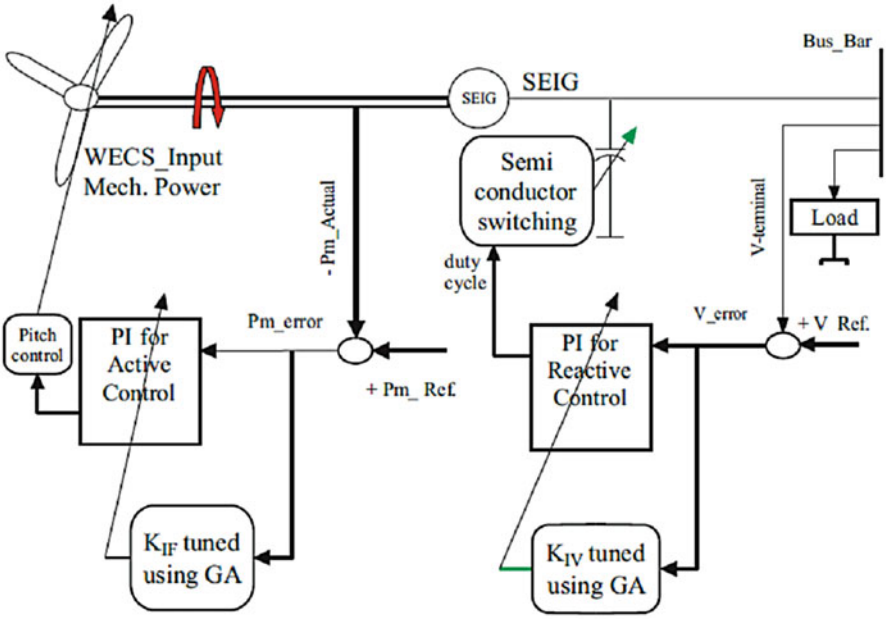


Fig. 5.19 SEIG driven by WECS

energy, to supply an isolated static load. A D-Q axis equivalent circuit model based on various reference frames, extracted from fundamental machine theory, is created to study SEIG performance in dynamic case. The performance of the SEIG, when equipped with a switching capacitor bank is discussed with a controller based on Genetic Algorithm (GA). GA is used to adjust the duty cycle and the stator frequency via the pitch control.

5.5.1 Mathematical Model for SEIG Driven by WECS

The *d-q* axis equivalent-circuit model without any load for a three-phase symmetrical induction generator is presented in Fig. 5.19. The stator and rotor voltage equations are derived based on Krause transformation, with a stationary reference frame.

5.5.1.1 Reactive Power Control and Switching Capacitor Bank Technique

Switching

During the past decades, the switching of capacitors has been neglected due to the practical difficulties such as the occurrence of voltage and current transients.

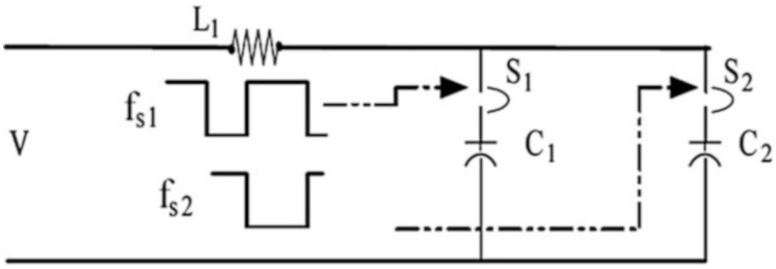


Fig. 5.20 Semi conductor switches (S1, S2) circuit for capacitor bank

Discussions in the literature prove that current 'spikes', would inevitably exceed the maximum current rating as well as the (di/dt) value of a particular semiconductor switch. To overcome this drawback, a semiconductor switch is designed to withstand the transient value at the switching instant. The equivalent circuit in Fig. 5.20 shows the application of switching capacitor bank due to duty cycle. In this circuit, the operation of the switches are in anti-phase, i.e., the switching function fs2 which controls switch S2 is the inverse function of fs1 which controls switch S1. In simple terms, switch S2 is closed when switch S1 is open and vice versa.

Reactive Power Control

The error in voltage is taken as the input to the controller and the output is the duty cycle (λ). The computed λ is applied as an input to the semiconductor switches to change the value of the capacitor bank. The input is applied based on the requirement of the excitation. The terminal voltage in turn is controlled by adjusting the self-excitation through automatic switching of the capacitor bank.

Active Power Control

Active power control is applied by adjusting the pitch angle of the wind turbine blades. Such control strategy is used to maintain the constant and steady operation of SEIG thus reducing the effect of the speed annoyance. The pitch angle is usually a function of the power coefficient "Cp" of the wind turbine WECS. The best optimal value of pitch angle leads to improved mechanical power regulation, which, in turn, achieves a better frequency adaptation of the overall system. Thus the active power control regulates the mechanical power of the wind turbine.

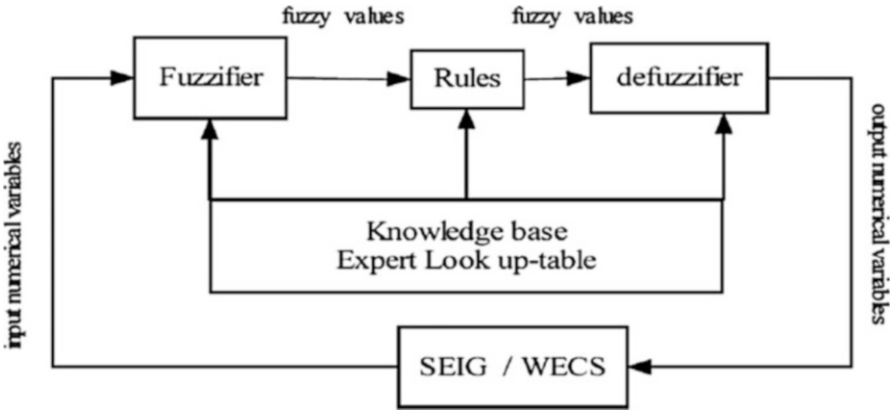


Fig. 5.21 FLC with SEIG driven WECS

5.5.2 Controllers

With SEIGs in this section, two different types of controller strategies have been discussed. The application of GA and FLC controllers are used to adjust the value of KI for both active and reactive controllers.

5.5.3 Fuzzy Logic Controller

With the information on artificial decision making in a closed loop environment, a fuzzy logic controller is designed using knowledge base. Figure 5.21 shows the basic construction of the FLC. The FLC is designed using input fuzzy sets, fuzzy rules and output fuzzy sets. The relation between the input and output variables are governed by a set of rules. The rules are formed based on prior knowledge base of the system. Each parameter in the input and output variable set has a corresponding membership function. The better adaptation of fuzzy set parameters leads to fine tuning of the fuzzy output. The FLC is used to compute and adapt the variable integral gain KI of PI controller.

The fuzzy input vector corresponding to reactive control consists of two variable; the terminal voltage deviation eV and the change of the terminal voltage deviation ΔeV. Table 5.5 shows the linguistic variables formed for both the inputs. The membership functions for voltage deviation and change in voltage deviation are shown in Figs. 5.22 and 5.23 respectively. Similarly, mechanical power deviation eF and the change of the mechanical power deviation ΔeF are chosen as the fuzzy inputs and five linguistic variables are defined. While, the output variable fuzzy set is shown in Fig. 5.24 and Fig. 5.25 shows the fuzzy surface. The linguistic variables are defined as follows: for example PB is Positive Big and NS is Negative

Table 5.5 Look up table

	Voltage deviation change (Δe_v)				
Voltage deviation (e_v)	NB	NS	AV	PS	PB
NB	NB	NB	NB	NS	AV
NS	NB	NB	NS	AV	PS
AV	NB	NS	AV	PS	PB
PS	NS	AV	PS	PB	PB
PB	AV	PS	PB	PB	PB

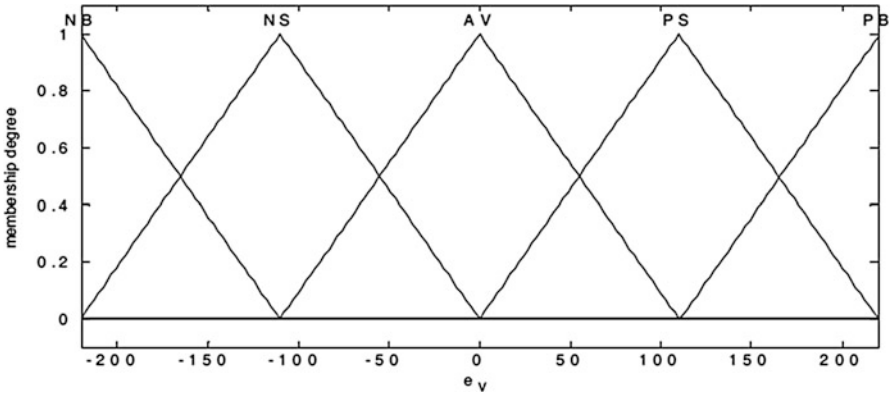


Fig. 5.22 Membership function of voltage error e_v

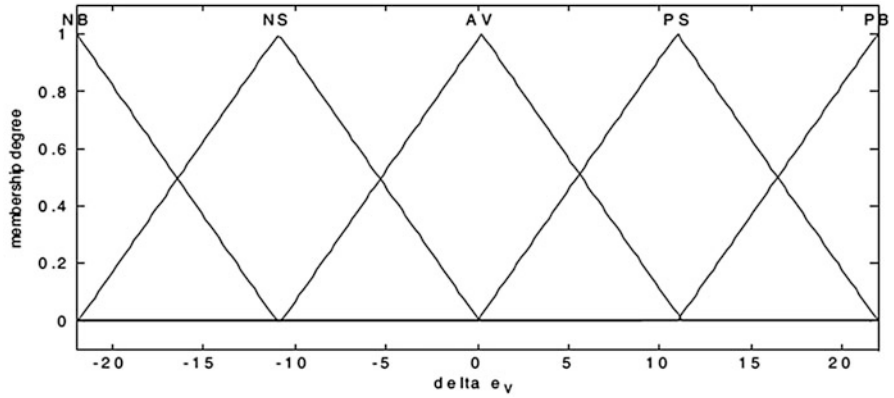


Fig. 5.23 Membership function of change in voltage error e_v

Small . . ..etc. Once the linguistic variables for the input and output variables are defined, the set of rules are developed, also known as the look-up table (Table 5.5). The rule base states the relation between the input variables, eV, eF, ΔeV and ΔeF and the output variable of the fuzzy logic controller. The integral gain value KI is the output of the fuzzy controller further applied in the PI controller.

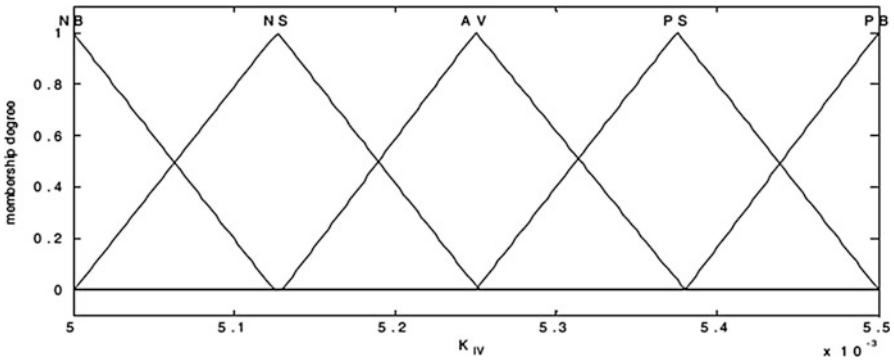


Fig. 5.24 Membership function of variable K_{IV}

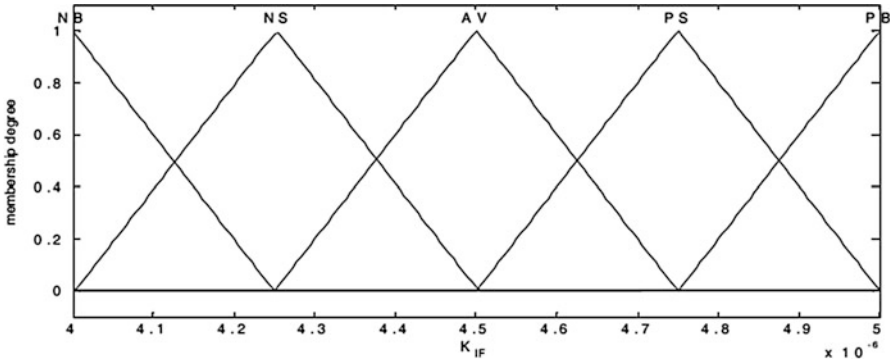


Fig. 5.25 Membership function of variable K_{IF}

The Defuzzification process is performed using the minimum of maximum method to determine the output from the fuzzy rules. This method, in general represents a polyhedron map. Initially, the minimum membership grade is computed from the intersection between two input variables, say x_1 and x_2. The min value is computed to rescale the output rule, and then the max value is computed. Finally, the centroid or center of area is used to compute the fuzzy output, which is defined as follows:

$$KI = \frac{\int y\mu(y)dy}{\int \mu(y)dy} \tag{5.56}$$

Thus, two FLCs are used to represent the reactive power and active power. The first controller is applied to tune the KI value of PI controller, using the terminal voltage error (eV) and its rate of change (ΔeV) as input variables. The second controller is

used to regulate the mechanical power via the blade angle adaptation of the wind turbine. The terminal voltage error (eV) varies between (−220 and 220) and its change (ΔeV) varies between (−22 and 22), and the output of the FLC is the KIV which changes between (5e-003 and 5.5e-003). The same technique is applied for active power controller where the two inputs for FLC are the mechanical power error (eF) which varies between (−1 and 1) and its change (ΔeF) varies between (−0.1 and 0.1). The output of the FLC is the KIF which changes between (4e-006 and 5e-006). The output of the PIFLC in the active controller adapts the pitch angel value to enhance the stator frequency.

The FLCs can be applied to active and reactive power controls of the system under study to enhance its dynamic performance. The FLC is used to regulate the duty cycle of the switched capacitor bank to adjust the terminal voltage of the induction generator. Another FLC is applied to regulate the blade angle of the wind energy turbine to control the stator frequency of the overall system.

5.5.4 Genetic Algorithm

The integral gain of the second PI controller is optimized based on the Genetic Algorithm. GA is used to calculate the optimum value of the variables based on the best dynamic performance and a domain search of the variable. The objective function used in the GA technique is $F = 1/(1 + J)$, where J is the minimum cost function. GA uses its operators and functions to find the values of *KIV* and *KIF* of the PI controllers to achieve better dynamic performance of the overall system. These values of gains lead to the optimum value of gains for which the system achieves the desired values by improving the P. O. S, rising time and oscillations. The main aspects of the proposed GA approach for optimizing the gains of PI controllers, and the flowchart procedure for the GA optimization process, are shown in Fig. 5.26.

5.5.4.1 Solution Representation

The PI gains are formulated using the GA approach, where all the gains are represented in a chromosome. The chromosome representation determines the GA structure. The parameter gains are encoded in a chromosome. The PI controller gains are initially started using minimum values of the domain search for PI gains. Based on the simulation results given in the previous sections, the values of [*KIV*max, *KIV*min] are shown in Fig. 4, while the values of [*KIF*max, *KIF*min] are shown in Fig. 11. The acceptable domain search for each gain is defined as [*KIV*max, *KIV*min] and [*KIF*max, *KIF*min] based on five times less and five times more than the gains obtained using the Ziegler-Nichols rule to satisfy minimum cost function J, as given in the following equation:

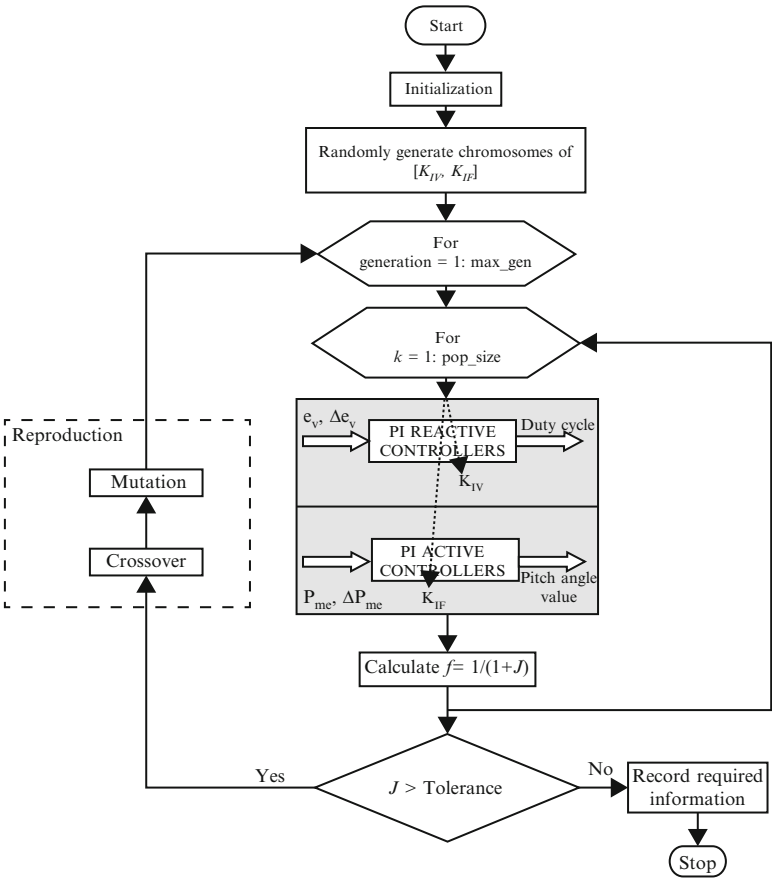


Fig. 5.26 Flowchart of GA approach for optimizing PIC gains

$$J = \int_0^T \left(\alpha_1 |e(t)| + \beta_1 \left|e^{'}(t)\right| + \gamma_1 \left|e^{''}(t)\right|\right) dt \tag{5.57}$$

where; $e(t)$ is equal to eV or Pme; eV is the voltage error used for the reactive power control and Pme is the mechanical power error used for active power control, as shown in Fig. 1. The parameters α_1, β_1, and λ_1 are weighting coefficients.

5.5.4.2 Coding of PI Controller Gains

The coded parameters are arranged on the basis of their constraints, as shown in Fig. 5.27, to form a chromosome of the population. The binary representation, given in Fig. 5.27, is the coded form for parameters with chromosome length equal

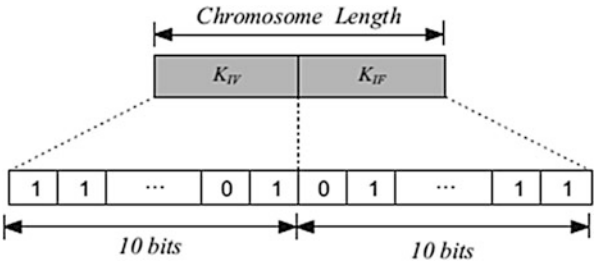


Fig. 5.27 Coded parameters of PI controller gains

to the sum of the bits for all parameters. In binary coding, the relation between the bit length L_i and the corresponding bit resolution R_i is given in the following equation:

$$R_i = \frac{UB_i - LB_i}{2^{L_i} - 1}, \tag{5.58}$$

where *UBi* and *LBi* are the upper and lower bounds of parameter *i*, respectively. In the present case study, we assume bit resolution $R = 10^5$ for all parameters. Figure 5.27 shows the coded parameters of the PI controller gains for reactive and active power controllers, respectively. The chromosome length used in this paper was 20 bits, where the bit length of *KIV* equal 10 bits and the bit length of *KIF* equals 10 bits.

5.5.4.3 Selection Function

The selection strategy decides how to select individuals to be parents for new '*children*'. The selection usually applies some selection pressure by favoring individuals with better fitness. After procreation, the suitable population consists, for example, of L chromosomes, which are all initially randomized. Each chromosome has been evaluated and associated with fitness, and the current population undergoes the reproduction process to create the next population. Then, the "roulette wheel" selection scheme is used to determine the member of the new population. The chance on the roulette-wheel is adaptive and is given as $P_\ell / \sum P_\ell$, where P_ℓ is defined as:

$$P_\ell = \frac{1}{J_\ell}, \ell \in \{1, \ldots, L\} \tag{5.59}$$

and J_ℓ is the performance of the model encoded in the chromosome measured in the terms used in Eq. 5.56. Maximizing the fitness function of each chromosome, which is inversely proportional to the performance criteria, Eq. 5.56 will damp the overshoot or the oscillations.

5.5.4.4 Crossover and Mutation Operators

The mating pool is formed, and crossover is applied. Then the mutation operation is applied, followed by the proposed GA approach. Finally, after these three operations, the overall fitness of the population is improved. The procedure is repeated until the termination condition is reached. The termination condition is the maximum allowable number of generations, or a certain value of J. This procedure is shown in the flowchart given in Fig. 5.26.

This section provided a GA based approach to optimize the PI controller gains in order to improve the dynamic response of the overall system. Optimal gains for the PI controller were determined using the GA procedure. The two types of controllers improve the dynamic performance when tested against various types of disturbances.

5.6 FLC Based STATCOM

Reactive power compensation is an important issue in the control of electric power system. Reactive power from the source increases the transmission losses and reduces the power transmission capability of the transmission lines. Moreover, reactive power should not be transmitted through the transmission line to a longer distance. Hence Flexible AC Transmission Systems (FACTS) devices such as static compensator (STATCOM) unified power flow controller (UPFC) and static volt-ampere compensator (SVC) are used to alleviate these problems.

5.6.1 Modeling of STATCOM

The STATCOM is based on the principle that a voltage source inverter generates a controllable AC voltage source behind a transformer leakage reactance so that the voltage difference across the reactance produces active and reactive power exchange between the STATCOM and the transmission network. By injecting a current of variable magnitude in quadrature with the line voltage, the STATCOM can inject reactive power into the power system which is done by making the inverter output voltage greater than the bus voltage. At the same time it will absorb the reactive power from the power system when the inverter output voltage is less than the bus voltage. The STATCOM does not employ capacitor or reactor banks to produce reactive power as does the SVC, but instead uses a capacitor to maintain a constant DC voltage for the inverter operation. The DC capacitor voltage can be adjusted by controlling the phase angle difference between line voltage and VSC voltage. The angle of line voltage is taken as reference. If the phase angles ‘α’ is slightly advanced, beyond the system bus voltage the STATCOM injected voltage

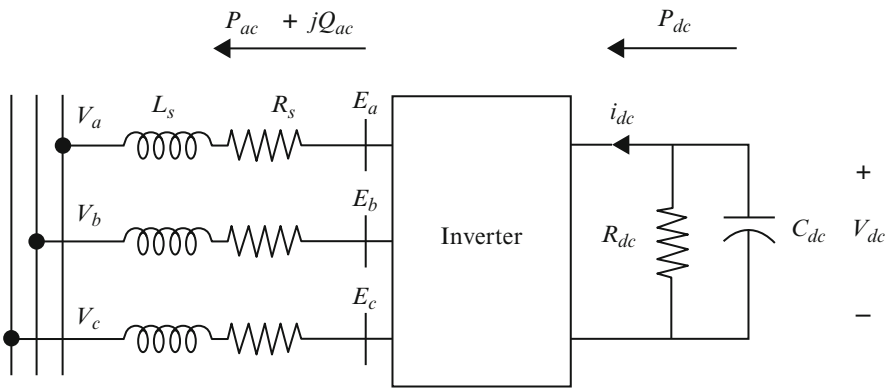


Fig. 5.28 Equivalent circuit model of STATCOM

decreases and reactive power flows into STATCOM. Conversely, if the phase angle 'α' is slightly delayed, the STATCOM injected voltage increases and STATCOM supplies reactive power to the bus. By controlling the phase angles of VSC, the reactive power can be generated from or absorbed by STATCOM and the voltage regulation can be achieved. The DC voltage maintained by the Cdc will act as voltage source for the VSC and it is connected to the transmission line through a shunt transformer. Figure 5.28 shows the equivalent circuit of the STATCOM. The loop equations for the circuit may be written in vector form as

$$\frac{d}{dt}i_{abc} = -\frac{R_s}{L_s}i_{abc} + \frac{1}{L_s}(E_{abc} - V_{abc}) \tag{5.60}$$

where R_s and L_s represent the STATCOM transformer resistance and inductance respectively, E_{abc} are the inverter AC side phase voltages, V_{abc} are the system-side phase voltages, and i_{abc} are the phase currents. The output of the STATCOM is given by

$$E_a = KV_{dc}\cos(\omega t + \alpha) \tag{5.61}$$

where V_{dc} is the voltage across the DC capacitor, K is the modulation index, and 'α' is the injected voltage phase angle. Figure 5.29 consists of one IGBT based intelligent power module (IPM) which acts as VSC, three phase transformer as AC source, display and sensing unit, RL loads, LC filters, and dSPACE kit with PC.

5.6.2 MATLAB/SIMULINK Model

Figure 5.29 shows the SIMULINK diagram of the PI controller part of the STATCOM. The real time performance analysis of STATCOM with PI controller

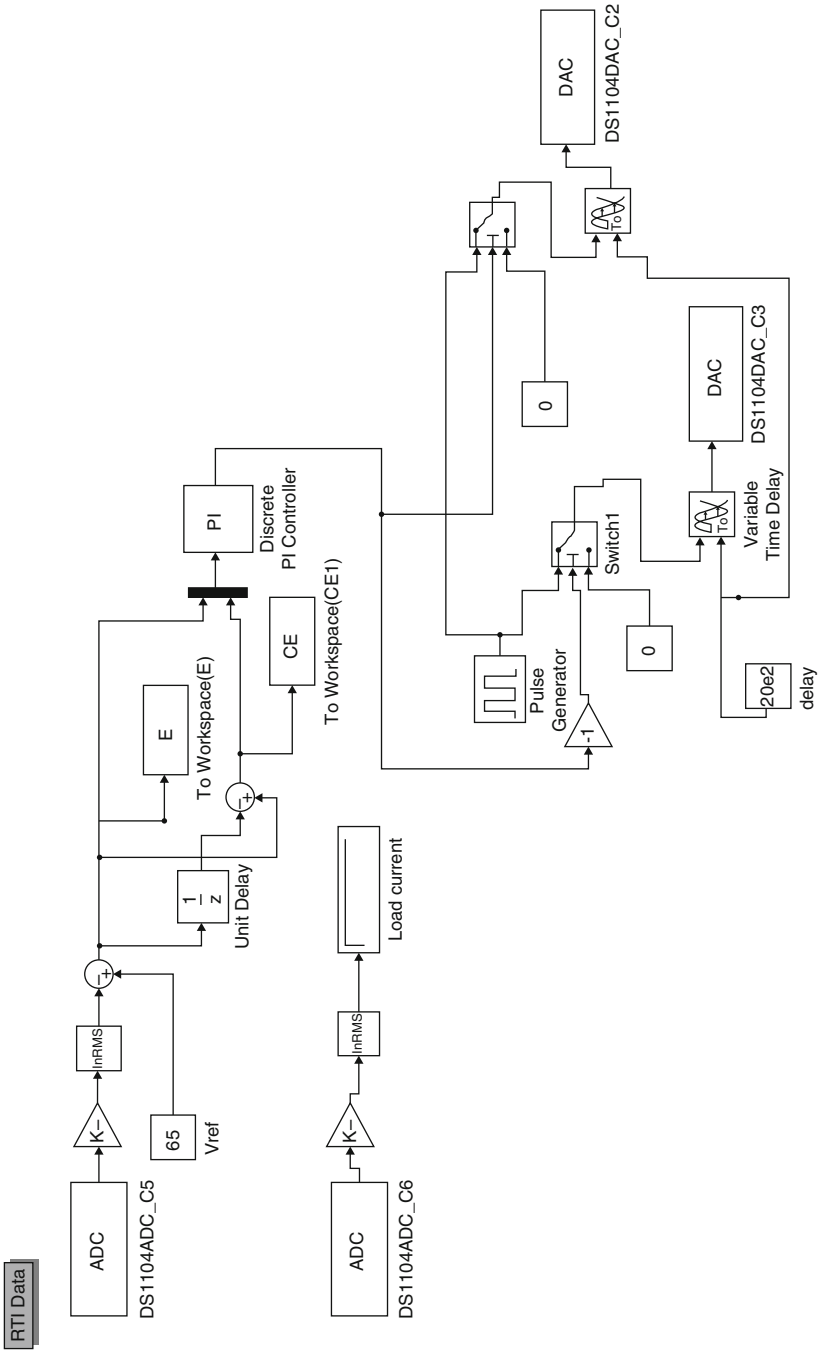


Fig. 5.29 SIMULINK diagram of the PI controller part of the STATCOM

is done by giving a load disturbance. In order to control V_{pcc}, its value is received in the dSPACE environment through one ADC. The command to change the phase angle 'α' so as to change the V_{pcc} is given to the I/O pins of the dedicated DSP TMS320F2407 through DAC from the dSPACE. Once the PI controller for STATCOM has been built in SIMULINK block-set, machine codes are generated by the dSPACE controller that runs on the inbuilt DSP processor. While the experiment is running, the dSPACE DS1104 provides a reference Vpcc as knob for the user to change its value online. Thus, it is possible for the user to view the process in real time, while the experiment is in progress. The Vpcc and the load current value have been taken from the ADCs. The reference value of Vpcc is set as 70v. The PI controller gives the output to DAC which is used to change the phase angle of the injected voltage by STATCOM till it reaches the reference value. Kp and Ki values are 0.1 and 500 respectively in the PI controller.

Fuzzy logic is used in STATCOM control loop. The V_{pcc} is fed back and is compared with the V_{pccref}. After comparison, the error and the change in error are calculated and are given as input to fuzzy controller. In this work, the error is normalized to one with V_{pccref}. This helps in using the fuzzy controller for any reference V_{pcc}. The fuzzy controller will attempt to reduce the error to zero by giving the pulse to S1 or S2. Figure 5.30 shows the SIMULINK diagram of the fuzzy controller part.

5.6.2.1 Sugeno Fuzzy Controller

There are two types of fuzzy controllers, viz., Mamdani and Sugeno type fuzzy controllers. In this work, Sugeno fuzzy controller is used. It uses singleton membership functions for the output variables. The Sugeno type controller is used because it can be easily implemented in any embedded system and reduces calculations. Furthermore, the reduction in calculation can result in real-time operation.

5.6.2.2 Fuzzification

In the present work, the error and change in error of Vpcc are fuzzified. Seven linguistic fuzzy sets with triangular membership function are used. The seven sets used for fuzzy variables 'error' and 'change in error' are negative big (NB), negative medium (NM), negative small (NS), zero (Z), positive big (PB), positive medium (PM), and positive small (PS).

5.6.2.3 Defuzzification

The reverse of fuzzification is called defuzzification. Weighted average method of defuzzification suitable for Sugeno type controllers is used in this work. The defuzzified output is the duration of the pulse.

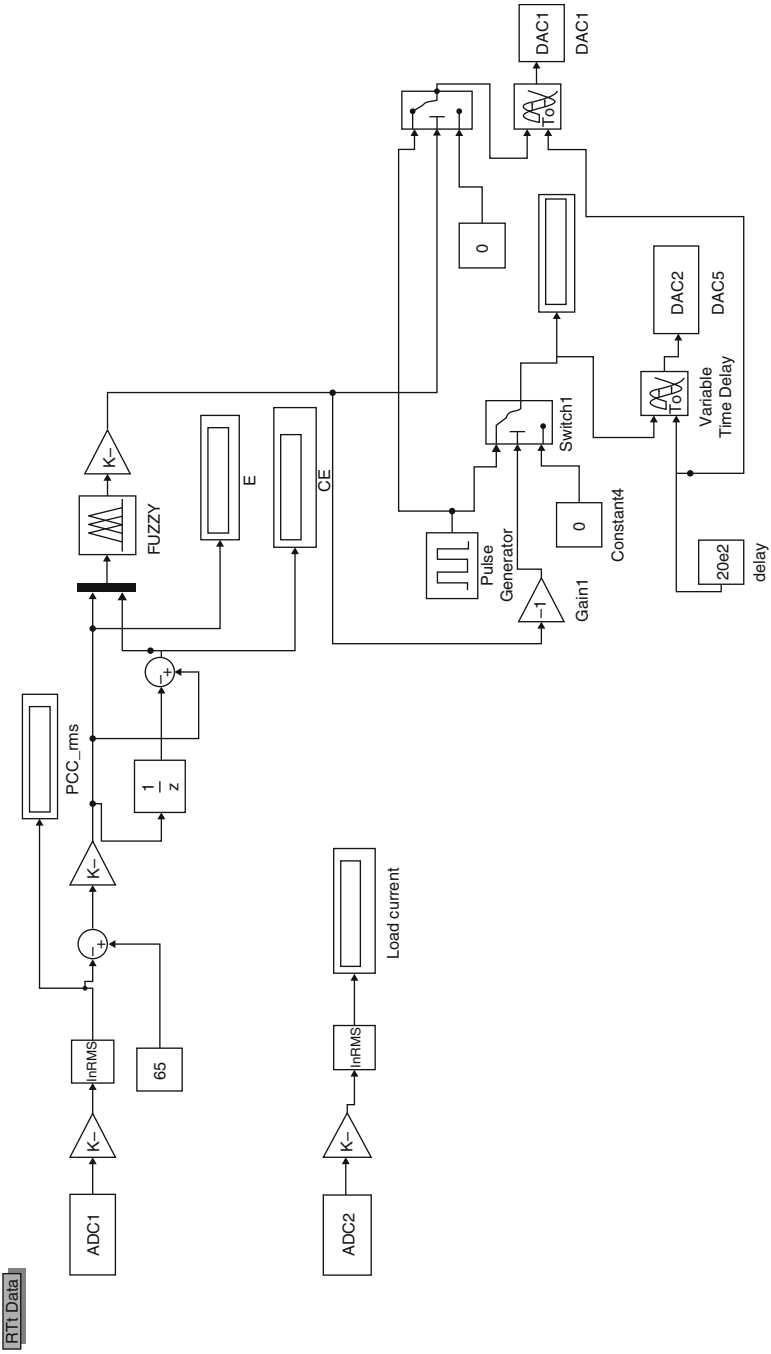


Fig. 5.30 SIMULINK diagram of the fuzzy controller part of the STATCOM

5.6.2.4 Rule Base

The fuzzy rules used in the design are in the general "If–Then" format. If error is Ai, and change in error is Bi, then output is Ci . Here the "if" part of a rule is called the rule-antecedent and is a description of a process state in terms of a logical combination of fuzzy linguistic sets. The "then" part of the rule is called the rule consequent and is a description of the control output in terms of logical combinations of fuzzy sets. The designed fuzzy controller increases the pulse duration when the error is positive and decreases the same when the error is negative. Figure 8 shows the fuzzy membership function.

5.6.3 Simulation Results

Simulation results of the fuzzy controller is presented in this section. Figure 5.31 shows the Active power, reactive power, power factor and current in the load side. The DC-link voltage regulation during sudden voltage changes is presented in Fig. 5.32. The active power, reactive power and power factor on the source side are shown in Fig 5.33.

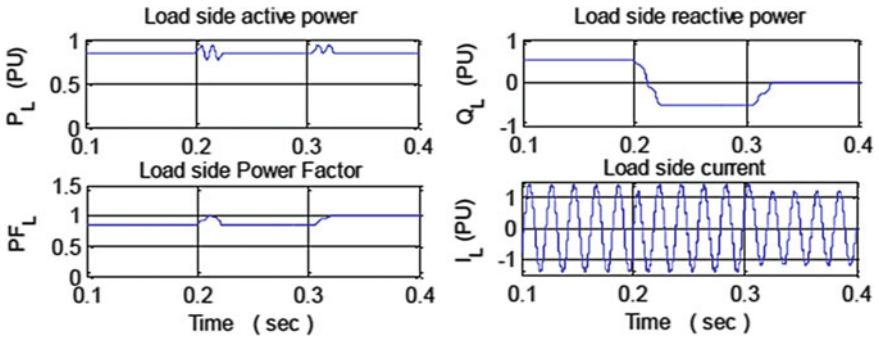


Fig. 5.31 Active power, reactive power, power factor and current in the load side

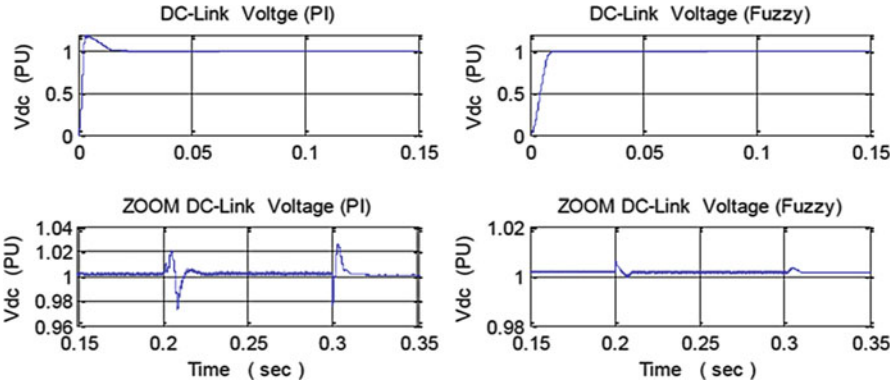


Fig. 5.32 DC-link voltage regulation during sudden voltage changes

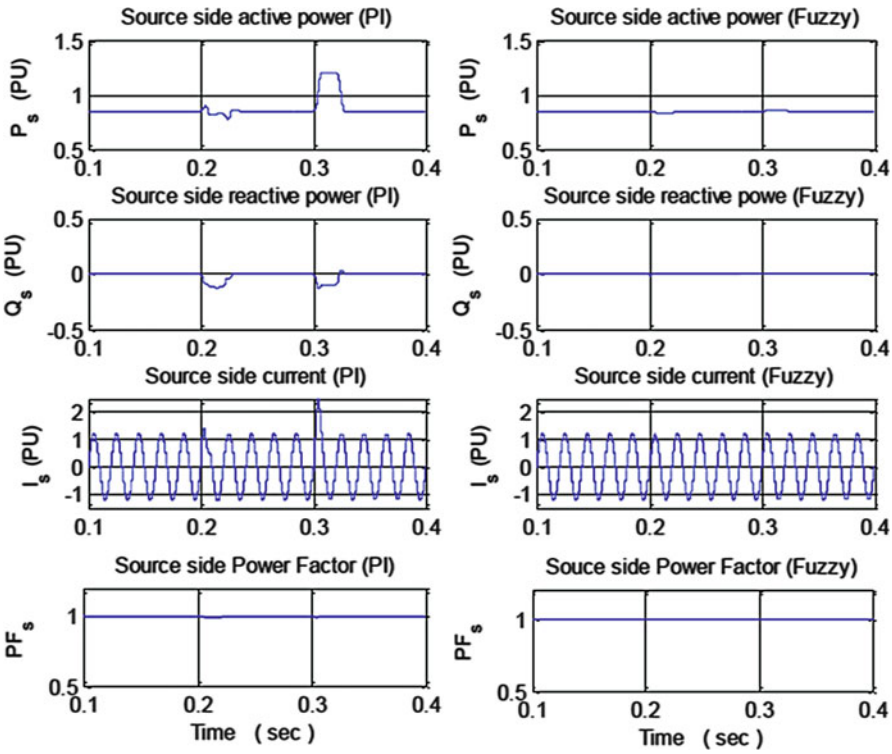


Fig. 5.33 Monitored results in the source side

5.7 FLC Based Wind Energy Production System

In this section, the fixing of voltage amplitude at a constant value of 380 V by using a controller for the loads supplied from a wind/battery energy production system is explained. The wind speed was continuously changed and the behavior of the system was analyzed. For the controller, both a PI controller and a fuzzy logic controller (FLC) were used; the results were compared. The quality of the energy was observed by taking measurements of the loads frequently. The value of the total harmonic distortion (THD) was held at the standard intervals. The entire system and all of its subcomponents were designed in a MATLAB/SIMULINK environment.

5.7.1 Wind/Battery Energy Production System

The overall system structure, consisting of a three-phase load, wind/battery hybrid energy system, interfacing converters, filters, and control unit, is shown in Fig. 5.34. When the wind speed is sufficient, the whole system is supplied by the wind turbine; otherwise, the batteries are turned on to supply the system. The direct-current (DC) voltage produced by the wind/battery system is converted to alternative-

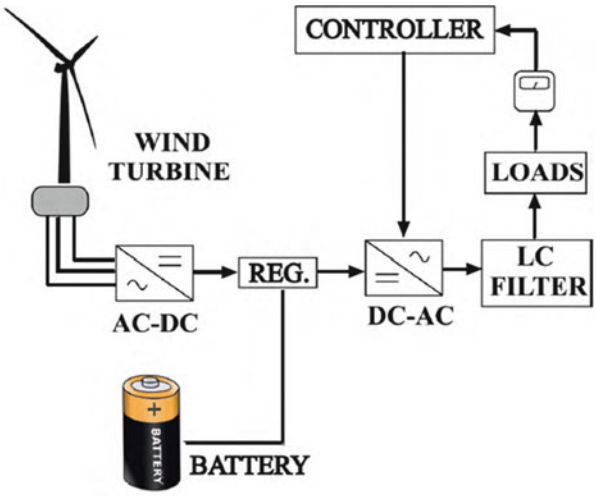


Fig. 5.34 PV/battery renewable source

current (AC) voltage using a three-phase inverter in order to supply power to the three-phase load. The voltage output of the inverter is controlled to keep the voltage at 380 V, line to line, at a fixed frequency of 50 Hz. The three-phase voltages are converted to d-q axis layouts as direct-axis voltage Vd and quadrature voltage Vq, to be used by the PI controller and the FLC. The output of the controller is then used to generate a pulse-width modulated (PWM) signal, which is used to control the output voltage of the inverter. The frequency of the inverter is controlled and kept constant at 50 Hz through a phase-locked loop (PLL) process.

The filters in the system are used to supply clean AC voltage to the loads. The value of the THD in the system was measured and compared with the correct standard values. All of the subsystem components (wind turbine, batteries, FLC, PI controller, inverter, three-phase load, regulator, and filters) were sub modeled individually using the MATLAB/SIMULINK GUI (graphical user interface) environment and were combined to establish the overall system model. The simulation of the proposed scheme was done in MATLAB/SIMULINK/SimPower to establish model validation and component compatibility. The used regulator is a device that measures the wind speed. This controller is a mechanism that uses an on-off control technique. When the wind speed is less than a specific value, it cannot supply the guaranteed power to the system; the system then turns off the wind turbine and turns on the batteries. In this study, a wind/battery system supplying power to three-phase AC loads was examined. To fix the voltage on the loads at 380 V, both a PI controller and a FLC were used, and the results were compared.

5.7.2 The Wind Turbine Model

Wind turbines convert mechanical energy produced by the wind to electrical energy. To use this electrical energy, voltage and frequency regulation are needed.

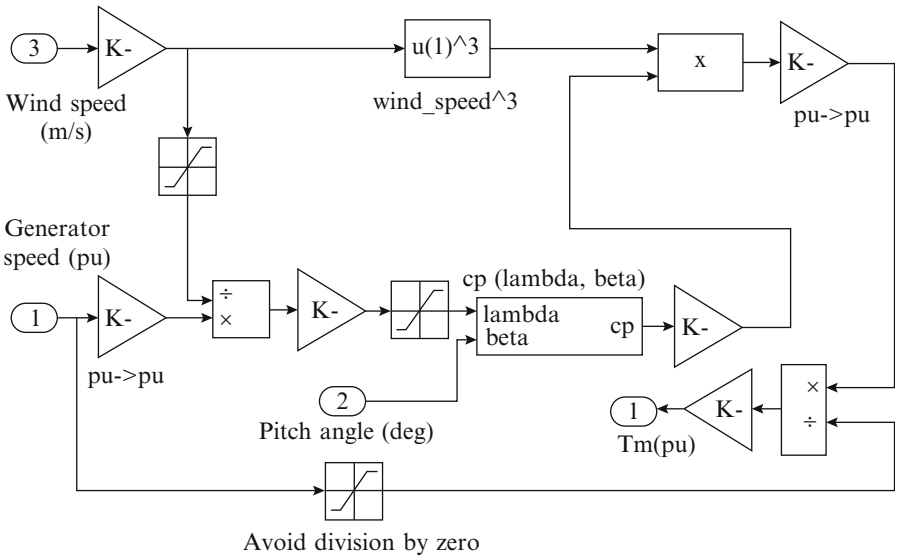


Fig. 5.35 Wind turbine model

The model of a wind turbine is developed on the basis of the steady-state power characteristics of the turbine. The SIMULINK model of the studied wind turbine is shown in Fig. 5.35.

5.7.3 Battery Model

The nickel-metal hydride battery was modeled using a simple controlled voltage source in a series with constant resistance, as shown in Fig. 5.36. This model assumes the same characteristics for the charge and the discharge cycles. The open voltage source was calculated with a nonlinear equation based on the actual state of charge (SOC) of the battery.

5.7.4 Fuzzy Logic Controller

The FLC's input values are generally the control error and the variation of this error in one sampling time. According to these variables, a rule table is produced in the FLC's rule-base unit.

The holding of the voltage and the amplitude of the load at the desired values of 380 V and 50 Hz is done by the FLC. The three-phase voltage is transferred from

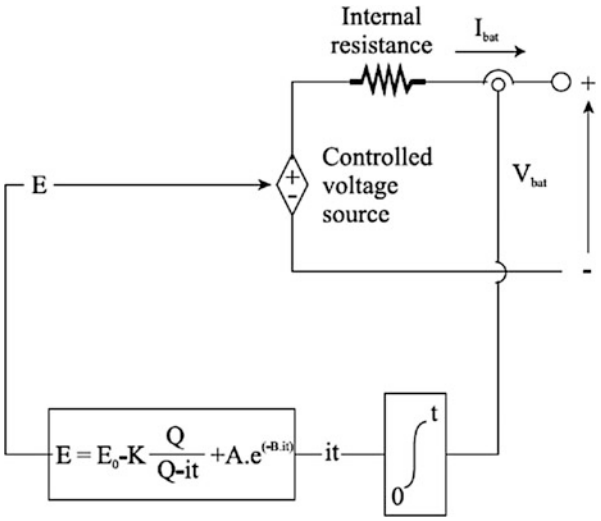


Fig. 5.36 Nonlinear battery model

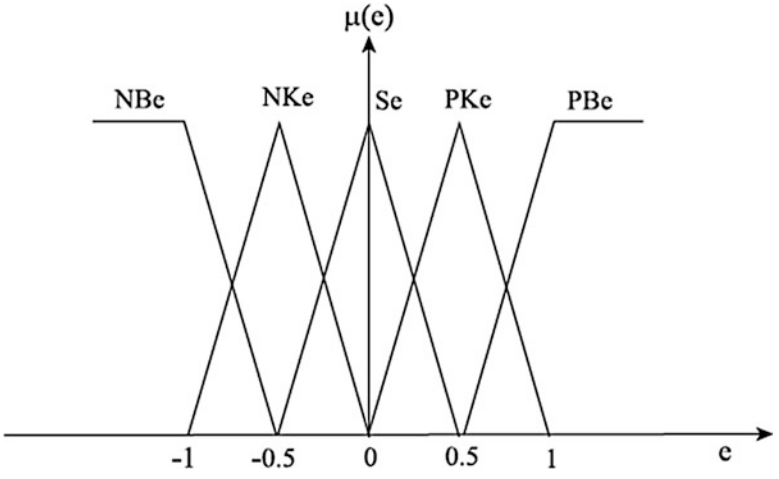


Fig. 5.37 Error membership functions

the a-b-c axes to the d-q axes first. The voltage divides into its components, which are controlled separately. At the output of the control, the voltage is transferred in reverse, from the d-q axes to the a-b-c axes, and, at the end, the PWM is driven by that voltage. These controls are done in the regulator block.

In Figs. 5.37 and 5.38, the error and the error variation of the input data of the FLC's input variables are shown. Triangle membership functions were used. These functions are called NB, NK, S, PK, and PB, and the data vary between –1 and 1.

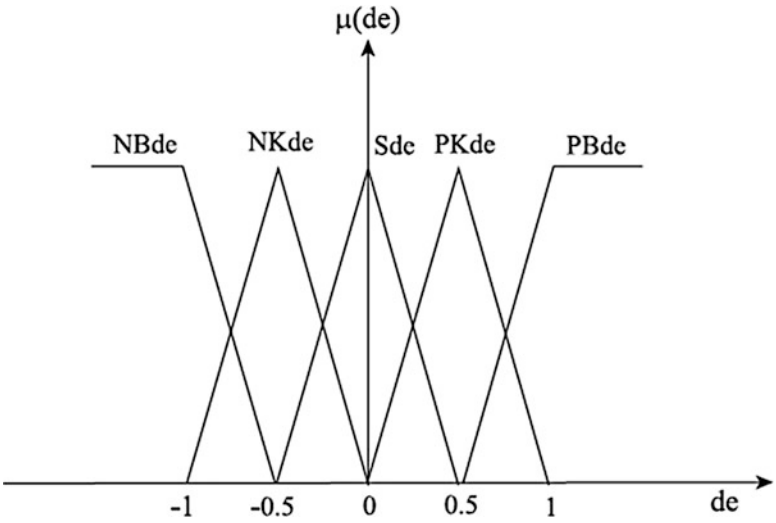


Fig. 5.38 Variation of error membership functions

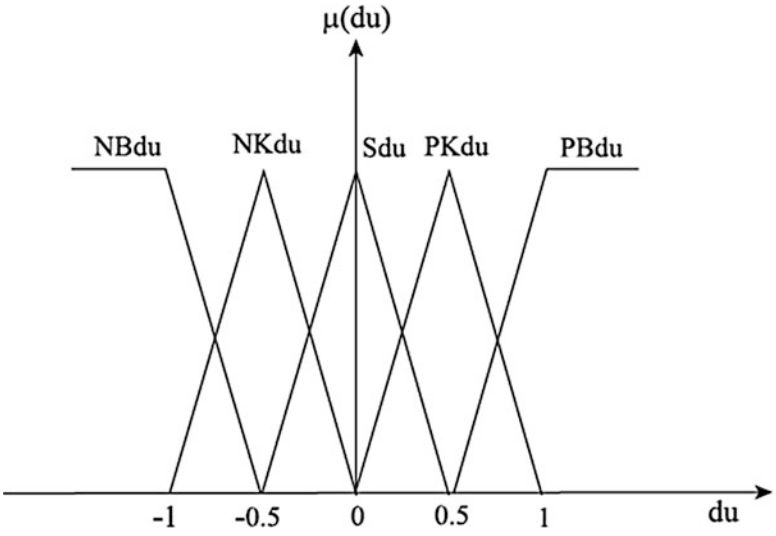


Fig. 5.39 Output space

In Fig. 5.39, the output space of the FLC is shown. These data also vary between –1 and 1. These subsystems are shown in Figs. 5.40, 5.41, 5.42 and 5.43. The triangle membership function is defined in Eq. 5.61, and its SIMULINK subsystem is shown in Fig. 5.40.

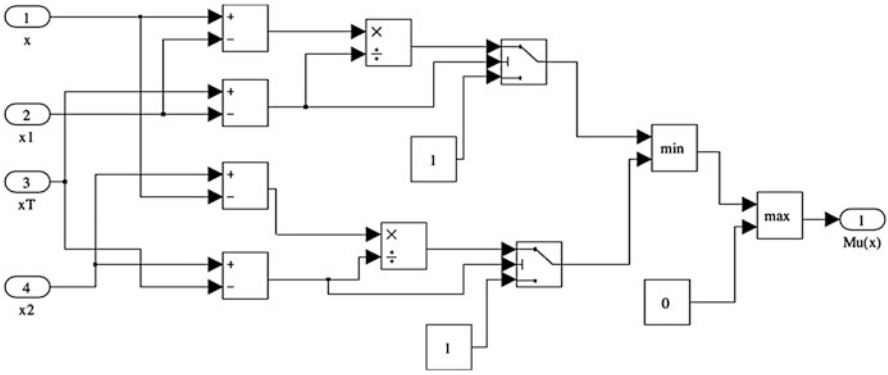


Fig. 5.40 Triangular membership function

Table 5.6 Fuzzy rules

e/de	NB	NK	S	PK	PB
NB	NB	NB	NK	NK	S
NK	NB	NK	NK	S	PK
S	NK	NK	S	PK	PK
PK	NK	S	PK	PK	PB
PB	S	PK	PK	PB	PB

$$\mu_{MU}(x) = \max\left(\min\left(\frac{x - x_1}{x_T - x_1}, \frac{x_2 - x_1}{x_2 - x_T}\right), 0\right) \tag{5.62}$$

In Fig. 5.41, the fuzzification unit is shown, with five rules. The minimums of the error and its variations according to the input variables are calculated here. In Table 5.6, the rules of the FLC are given. Due to the 5-ruled input variables, there are 25 rules in total. The unit for the rule processing is seen in Fig. 5.42.

In Fig. 5.43, certain variables are obtained with defuzzification by using the center of the area method. In Eq. 5.62, the mathematical expression of this method is shown.

$$z_0 = \frac{\sum_{j=1}^{n} \mu_C(z_j).z_j}{\sum_{j=1}^{n} \mu_C(z_j)} \tag{5.63}$$

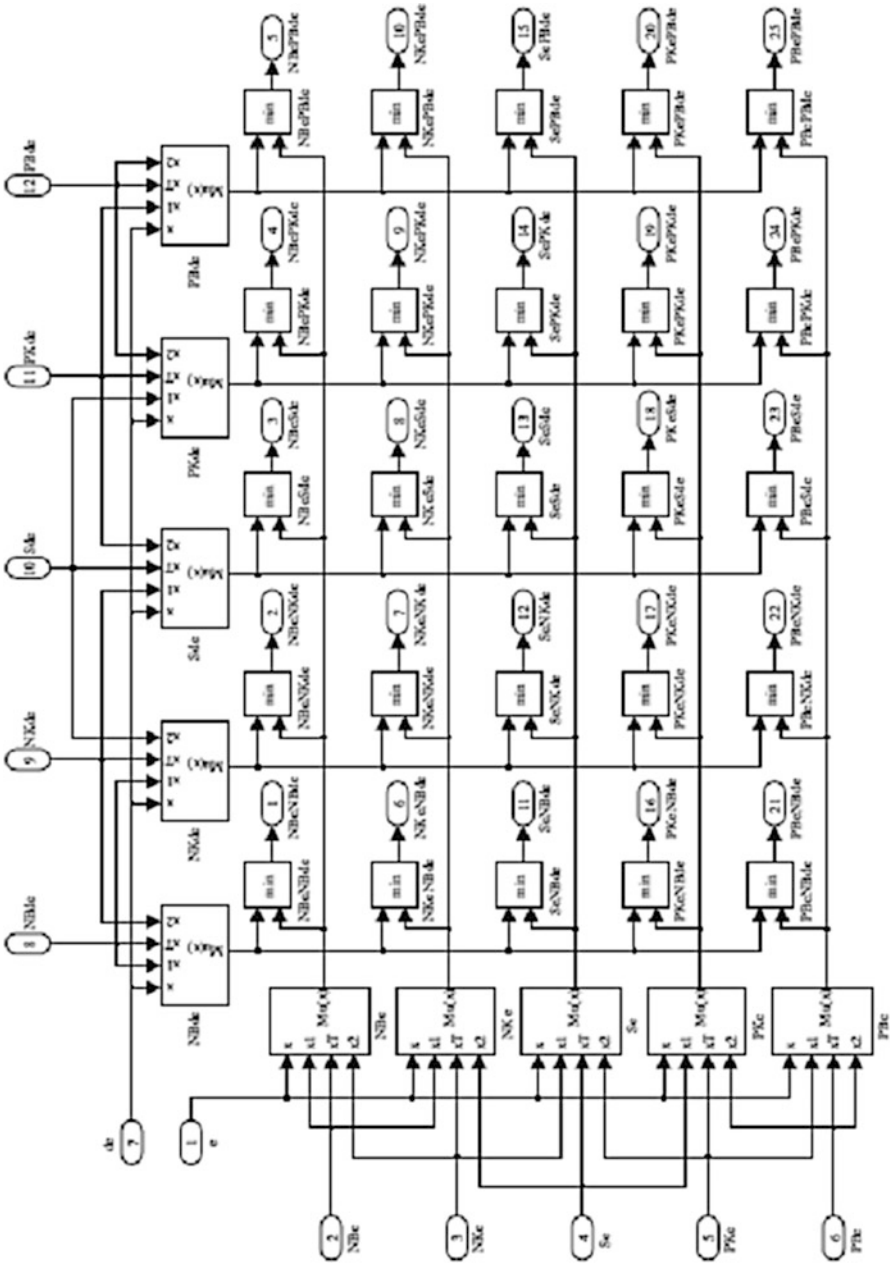


Fig. 5.41 Fuzzification unit

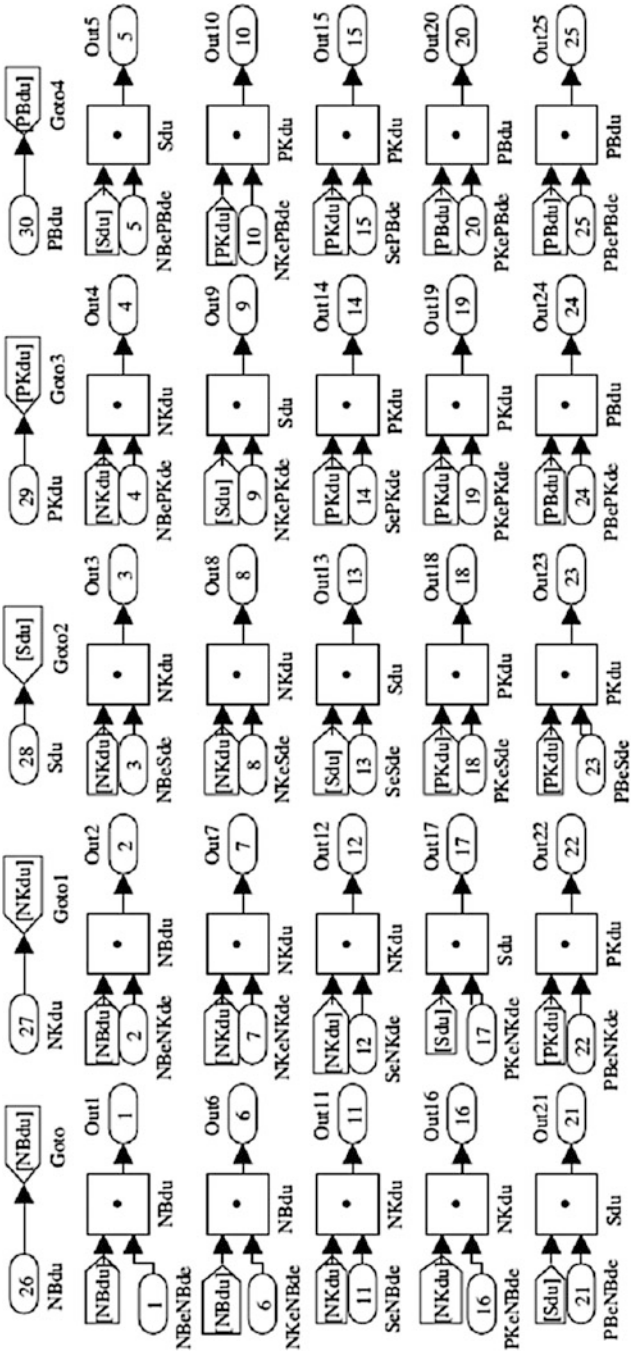


Fig. 5.42 Fuzzy rules in SIMULINK

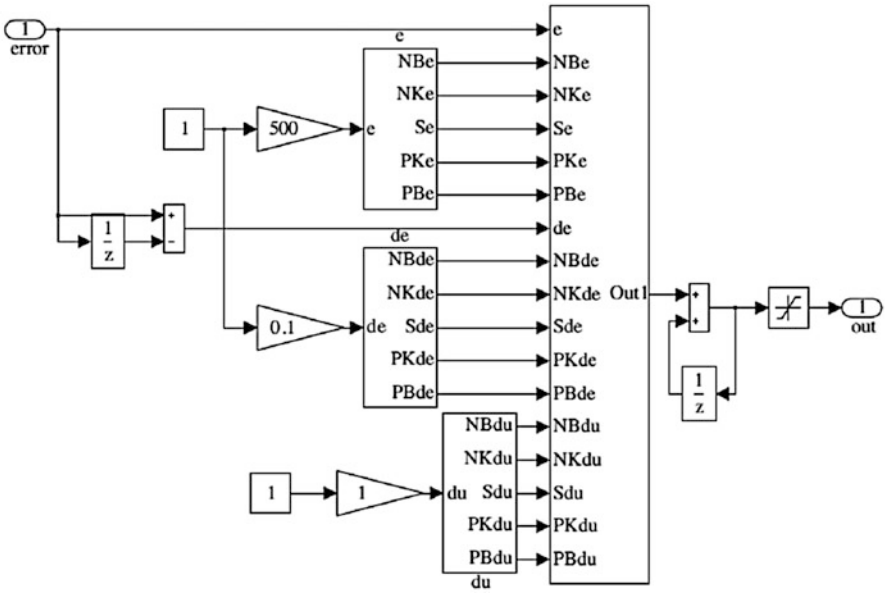


Fig. 5.43 FLC with input and output values

5.7.5 MATLAB SIMULINK Model

The wind turbine is affected by the wind speed. The simulated system is shown in Fig. 5.44. The system includes a 60-kW wind turbine and induction machine, a 55-kW battery, a regulator, an inverter, filters, a total 50-kW load, measuring units, switches, and controllers. During the simulation process, the voltages on the loads are fixed at the values of 380 V and 50 Hz.

5.7.6 Simulation Results

The PWM used in the system's inverter has a 2-kHz switching frequency. Therefore, in the simulation, discrete time control is done at a sampling time of 2 μs. The alternative voltage obtained from the wind turbine is converted to direct current by passing it through a rectifier and then filtering it. The direct current is also connected to the battery unit. It is converted to the three-phase voltage by an inverter and then filtered again, and the loads are supplied by this voltage. To investigate the system's behavior at different loads, first a 10-kW and then a 20-kW load is added to the 20-kW reference load. The total load is 50 kW. In the system, the regulator decides when the loads will be supplied by the wind turbine or by the battery unit. According to the decision, the signals are sent to the switches that turn

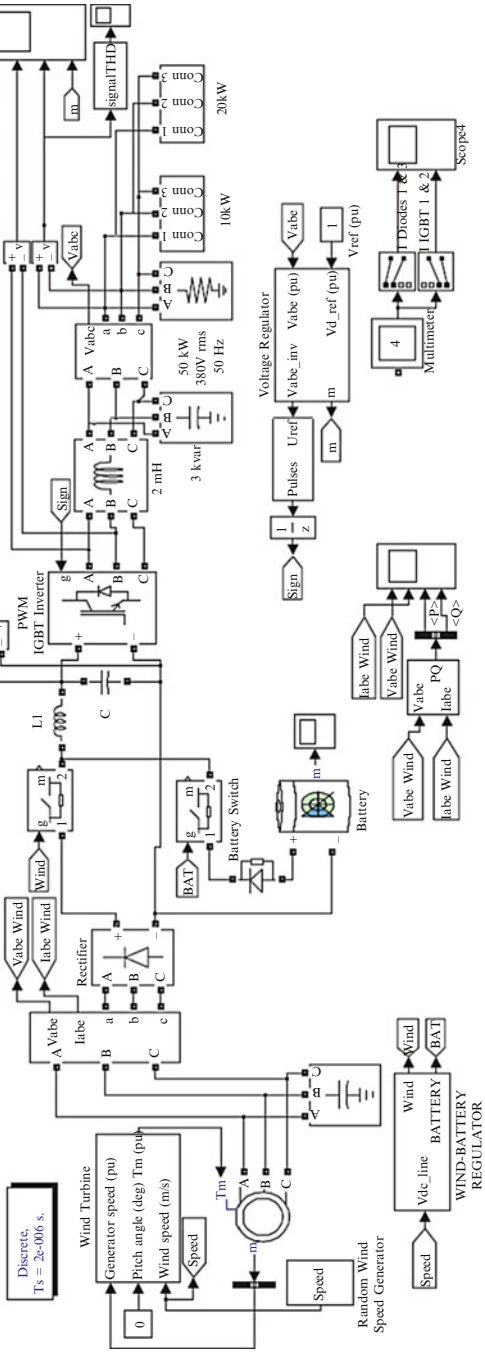


Fig. 5.44 Main system

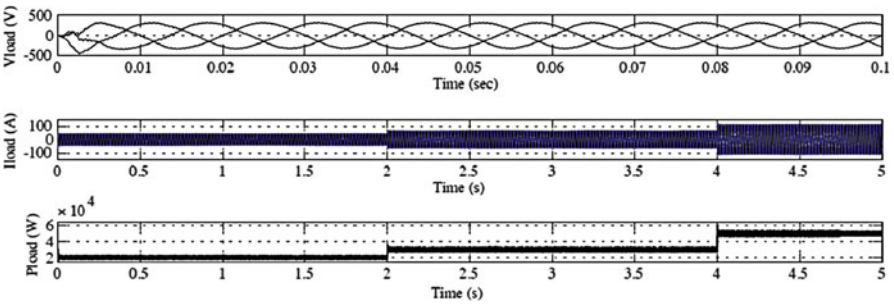


Fig. 5.45 Power on the load with a PI controller

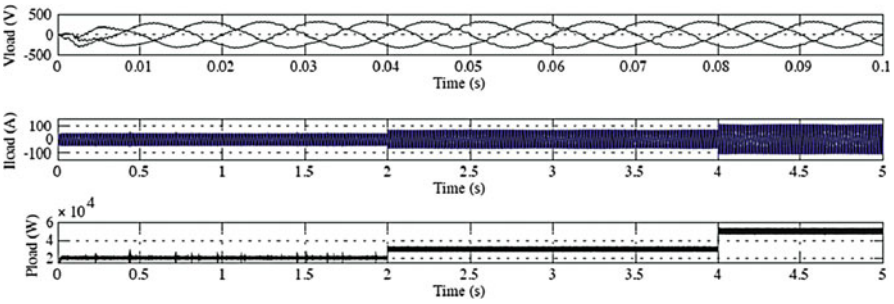


Fig. 5.46 Power on the load with FLC

on and off the wind turbine and the battery unit. This process is conducted in the wind-battery block.

The voltage regulator block is the place where the control process occurs. Here, as explained before, the transfer process is done from the a-b-c axes to the d-q axes. The voltage is divided into its d and q components, which are controlled separately, and, in the output of the controller, these data are converted to the a-b-c axes from the d-q axes. The PWM is driven by that voltage.

By taking some measurements from every point, data about the system's current, voltage, and power can be examined. By changing the wind speed between 0 and 12 m/s, a simulation close to reality is undertaken.

In Figs. 5.45 and 5.46, the active power, current, and voltage on the load with both a PI controller and a FLC are shown. The increase of the loads can be seen clearly. First a 20-kW, then a 10-kW, and finally a 20-kW load were added to the system. As a result, a total load of 50 kW is supplied by the system.

In Fig. 5.47, the voltage on the load reaches a value of 380 V at 2 s with the PI controller. The maximum overshoot voltage that can be reached is 392 V. The three-phase THD value is less than 10 % at the steady situation. In Fig. 5.48, the results derived from the control done by using the FLC are shown, along with the DC output voltage, the inverter's output voltage between the a and b phases, the

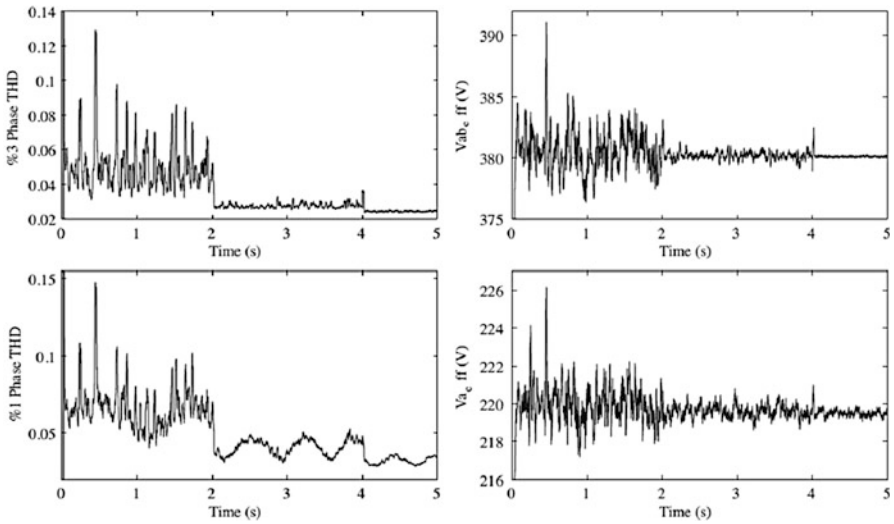


Fig. 5.47 PI controller results

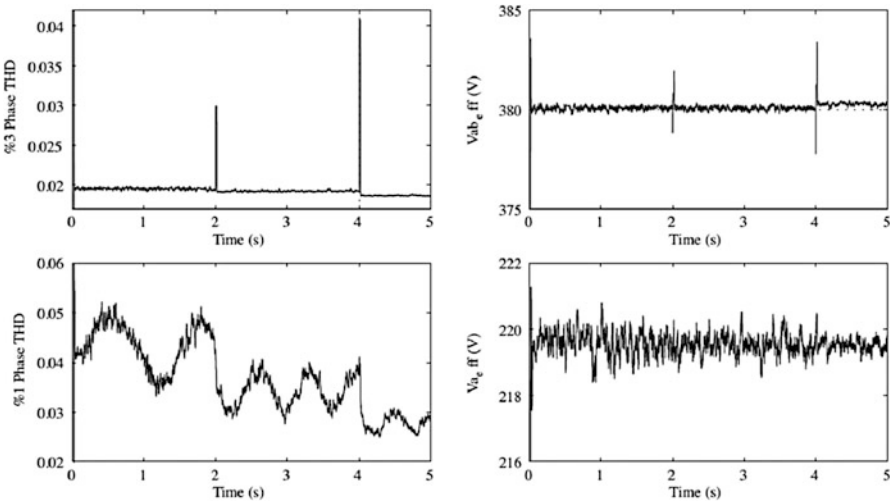


Fig. 5.48 FLC results

load voltage between the a and b phases, and the variation of the modulation indexes. The voltage on the load reaches a value of 380 V at 0.05 s with the FLC. The maximum overshoot value is 384 V. The three-phase THD value is less than 2 % at the steady situation.

In Figs. 5.49 and 5.50, the variation of the voltage between the a and b phases done in simulations with the PI controller and FLC in sequence are shown. As can be seen from the THD values, the FLC has a less harmonic and a clean sinus wave.

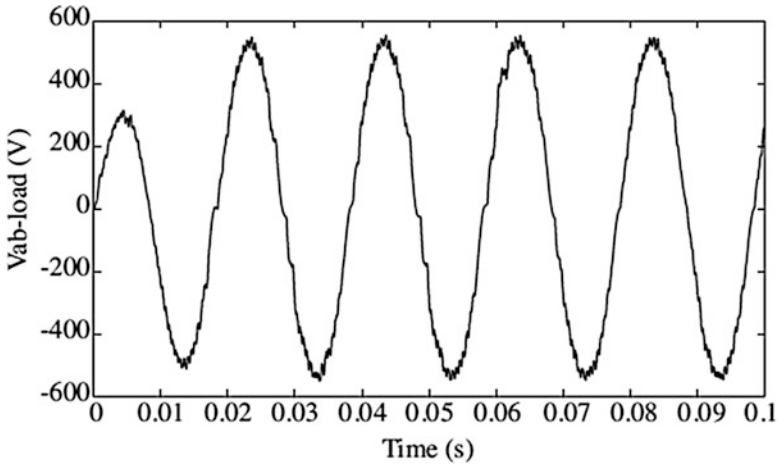


Fig. 5.49 V*ab* – *load* variation for PI controller

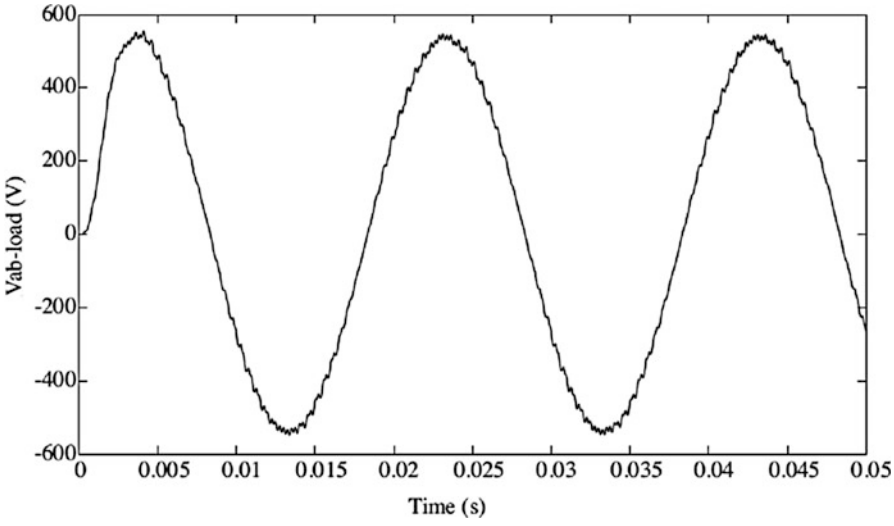


Fig. 5.50 V*ab* – *load* variation for the FLC

The waveform of the loads was very similar to the sinus wave form using both the PI and fuzzy logic controllers. The system can fix the voltage on the loads at a constant value of 380 V regardless of effects from the variations of the wind speed. The system frequency value is steady at 50 Hz. According to the value of the wind speed, the used regulator works effectively by turning on and off the batteries. The maximum overshoot and settling time values of the FLC were much better than those of the PI controller. The PI controller maximum overshoot voltage that can be

reached is 392 V. This value is 384 V with the FLC system. The PI controller's setting time was 2 s; this value was 0.05 s for the FLC. When the THD values are compared, it is seen that both controllers had values in the standard ranges. However, the FLC's value was better than the PI's, as was its sinus wave form. When the produced energy is greater and the loads are low, the wind turbine must be arranged to recharge the batteries. This can be done by the management of the energy. When there is no wind, the loads are supplied only with batteries. When the batteries are empty, the loads will have no energy supply. To prevent this situation, a diesel generator can be added to the system or the system can be supplied with energy by the main network.

5.8 Prediction of Wind Speed Based on FLC

Fuzzy logic controller (FLC) is used in order to achieve maximum power delivery for each wind speed and to have more robustness. Maximum output power operating point deviates from the maximum torque point. The torque follows the square-law characteristics and the output power follows the cube law. This means that under light load conditions, additional generator magnetic flux controller is needed in order to achieve more power efficiency, by reducing the iron losses. The optimum fuzzy logic control of an induction generator require controller which will track wind speed in order to achieve λopt and thus extract maximum power.

The product of torque and speed gives turbine power, and by assuming a steady state lossless system it is equal to the grid power. The curves in Fig. 5.51 show the grid power pO in dependence of generator speed ωr in terms of wind velocities vw.

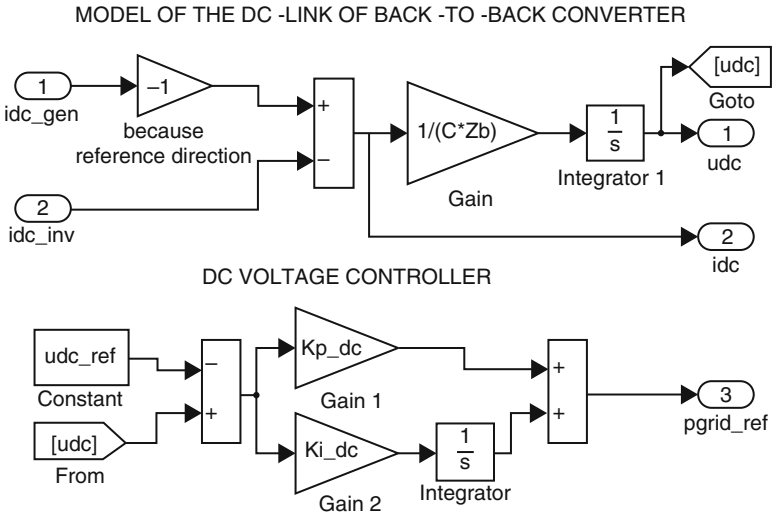


Fig. 5.51 MATLAB/SIMULINK model of the DC-link and DC voltage controller

What is the exact task of fuzzy controller it could be best seen from these set of power curves.

For a particular value of wind velocity, the function of the fuzzy controller FLC is to seek the generator speed in steps until the system settles down at the maximum output power operating point. For example, at a wind velocity of *Vw4*, the output power will be at point A if the generator speed is *ωr1*. The FLC will change the speed in steps until it reaches the speed *ωr2*, where the output power is maximized at point B. If the wind velocity increases to *Vw2*, the output power will jump to point C, and then FLC will bring the operating point to D by searching the generator speed to *ωr4*. The strategy for decrease of wind velocity is similar. If wind velocity decreases to *Vw3*, output power is altered and settled at operating point E. FLC will now decrease generator speed to the optimum value *ωr3* (point F) where output power is at maximum.

Therefore, the principle of the fuzzy controller is to increment or decrement the generator speed in accordance with the corresponding increment or decrement of the estimated output power *pO*. If *ΔpO* is positive with the last positive *Δωr*, the search is continued in the same direction. If, on the other hand, positive *Δωr* causes negative change *ΔpO*, the direction of search is reversed. So, during these step changes of generator speed, controller observes changes in turbine output power and due to this it keeps searching generator speed for which this change of output power would be zero. That would be the maximum power delivery point. If *ωr1* is far from *ωr2*, controller could give greater value of generator speed increments, *Δωr*, for faster convergence to the maximum delivery point. Similar, if current generator speed is close to the *ωr2* controller must give smaller value of speed increments, *Δωr*, to avoid oscillations and ensure stable system. Descriptive rules like this could be applied converting the system into the fuzzy system. Actual values that indicates input variables, here change of output power, *ΔpO,* and last change of generator speed, *LΔωr*, are initially converted into the correspondent fuzzy sets with human descriptive and intuitive values such are terms BIG, MEDIUM, SMALL, ZERO. This is done in the "fuzzification block", where the variables *ΔpO* (variation of output power), *Δωr* (variation of generator speed) and *LΔωr* (last variation of generator speed) are described by membership functions. Afterwards, it is possible to apply descriptive rules of reasoning like "if the last change of output power *ΔpO* during maximum power searching was POSITIVE and BIG and the last change of desired generator speed *LΔωr* was POSITIVE then keep tracking the maximum power in the same POSITIVE direction with BIG increment *Δωr*". Rules like this are involved in block "rules table", and they are given in Table 5.7. Finally, the fuzzy set of output reference change of generator speed *Δωr* is back "deffuzified" to convert it to the actual value. That means the output values such are BIG, MEDIUM, SMALL are translated to numbers which indicates a measurable (but normalized) value of the generator speed. It could be also noticed the output of controller *Δωr* is added by some amount of *LΔωr* in order to avoid local minima in characteristics *Cp(λ)* due to the changes of wind speed. value. That means the output values such are BIG, MEDIUM, SMALL are translated to numbers which indicates a measurable (but normalized) value of the

Table 5.7 Fuzzy rules

	$\Delta\omega_r$		
ΔP_o	*P*	*ZE*	*N*
PVB	*PVB*	*PVB*	*NVB*
PBIG	*PBIG*	*PVB*	*NBIG*
PMED	*PMED*	*PBIG*	*NMED*
PSMA	*PSMA*	*PMED*	*NSMA*
ZE	*ZE*	*ZE*	*ZE*
NSMA	*NSMA*	*NMED*	*PSMA*
NMED	*NMED*	*NBIG*	*PMED*
NBIG	*NBIG*	*NVB*	*PBIG*
NVB	*NVB*	*NVB*	*PVB*

generator speed. It could be also noticed the output of controller $\Delta\omega r$ is added by some amount of $L\Delta\omega r$ in order to avoid local minima in characteristics $Cp(\lambda)$ due to the changes of wind speed.

5.8.1 Controller Model

The controller operates on a per unit basis so that the response is insensitive to system variables and the algorithm is universal to any system. Membership functions of FLC controller are implemented by using MATLAB/SIMULINK fuzzy control toolbox. In Fig. 5.52 realized model of fuzzy logic controller is shown. Heart of the controller is block of fuzzy logic controller from MATLAB/SIMULINK fuzzy control toolbox. With this powerful tool it is possible to define the fuzzy controller in relatively easy way using graphical interface and intuitive dialogs, which is obvious from Figs. 13 and 14. The scale factors *KPO* and *KWR* are functions of the generator speed, so that control becomes somehow insensitive to speed variation. The scale factors are generated by fuzzy computation realized in *MATLAB*/*SIMULINK* as shown in Fig. 5.53. Their values are generated partly empirically, and partly taking into account dependence of wind turbine power related to wind speed in terms of different generator speed. The membership functions are scaled using MATLAB editor shown in Fig. 5.54.

In order to verify control principle, detailed model of the system in MATLAB/SIMULINK has been developed. The system data are shown in Table 5.8.

5.8.2 Experimental Results

Simulation results shown in Fig. 5.55 illustrate performance of wind turbine with fuzzy controller which tracks the maximum power delivery operating point. They are given for the case of wind velocity decrease from nominal to 80 % of nominal

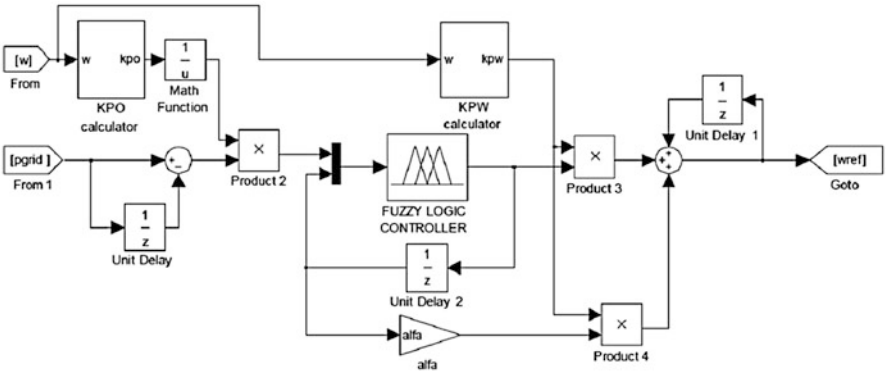


Fig. 5.52 MATLAB SIMULINK model of FLC

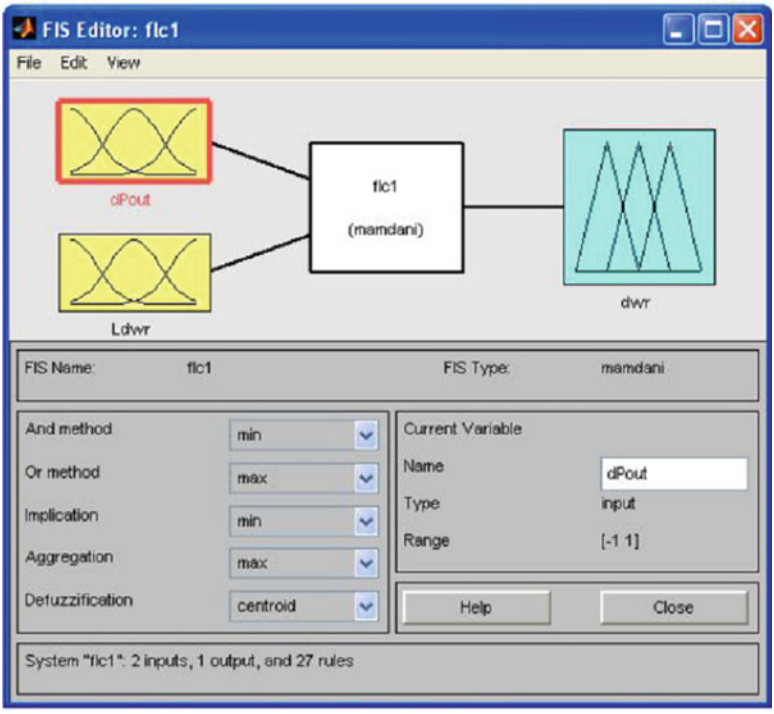


Fig. 5.53 MATLAB graphical editor for defining of fuzzy logic controller

value. It could be noticed that before disturbance ($t < 5$ *s*) all variables are settled at nominal values. After wind velocities decrease power fed to the grid gradually decrease to value around 50 % of nominal, which is expected because turbine power follows the cube law. Reference value of q current component i_{sq} also drops, and

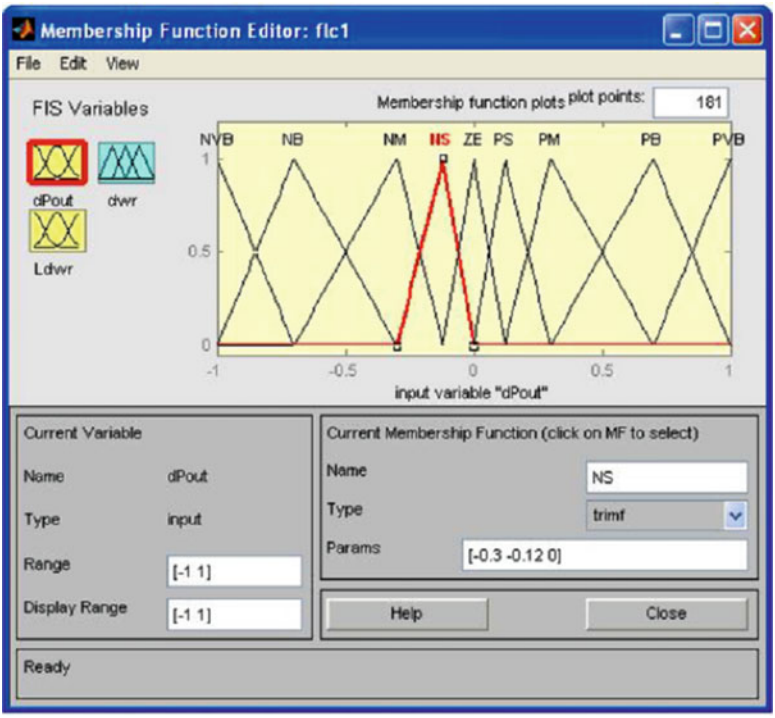


Fig. 5.54 MATLAB graphical editor for defining membership functions

Table 5.8 The system parameters used in simulation

Stator resistance, *rS* 7.7 m (p.u.)
Rotor resistance, *rR* 9.0 m (p.u.)
Stator reactance, *xS* 3.56 (p.u.)
Rotor reactance, *xR* 3.53 (p.u.)
Magnetization reactance, *xM* 3.47 (p.u.)
Leakage coefficient, σ 41 m (p.u.)
Grid resistance, *rGRID* 6.4 m (p.u.)
Grid reactance, *xGRID* 40 m (p.u.)
Fuzzy controller sampling per., *Tflc* 0.014 s
Capacitance of DC-link, *C* 1,000 μF
Based impendance, *ZB* 16 Ω
Turbine power coefficient, *CpMAX* 0.4639
Based angular speed, ωB $100 \cdot \pi$ rad/s

thus the value of electromagnetic torque. *D* current component *isd* remain the same, so the magnetic flux is unchanged. DC-link voltage is maintained constant which means that all generated power is transferred to the grid. Generator rotational speed is changing consistent with changes of wind velocity. Speed response is without overshoot and abrupt transients, as a consequence of fuzzy controller usage.

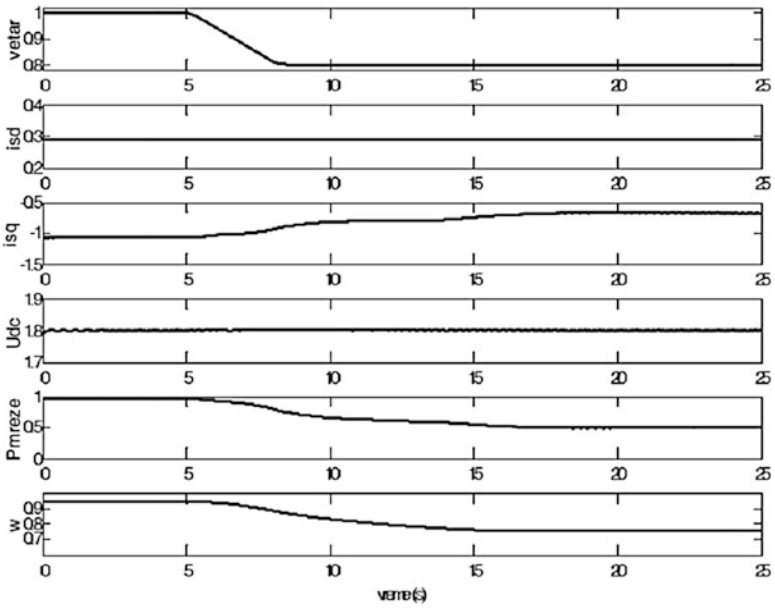


Fig. 5.55 Simulation results. From up to down: wind velocity *Vw*, *d* grid current component *isd*, *q* grid current component *isq*, DC-link voltage *UDC*, grid power *PO*, and generator speed *ωr*

In order to verify that maximum power is extracted from the available wind, power coefficient *Cp(λ)* has to be observed. In Fig. 5.56, *Cp(λ)* is shown during the change of wind velocity. It could be noticed that during the change of wind velocity, power coefficient *Cp(λ)* differs from optimum value. Deviation is relatively small, but after the transient period which lasts around 10 s, it settles at the optimum value.

In this section optimum fuzzy control of wind turbine in order to extract maximum power is described and verified through the simulation. The main goal of implemented fuzzy controller is to continuously adapt the rotational speed of the generator to the wind speed in a way that the turbine operates at its optimum level of aerodynamic efficiency. The advantages of using fuzzy controller are universal control algorithm, fast response, and parameter insensitivity. Implemented system has satisfactory dynamic and static performances.

5.9 Fuzzy Logic Controlled SPWM Converter for WECS

A novel sinusoidal PWM switched AC/DC/AC converter interface scheme using real novel tri-loop voltage error tracking fuzzy logic controller (FLC), to stabilize the stand alone Wind Energy Conversion Scheme (WECS) using an induction generator is presented in this section. The novel Sinusoidal Pulse Width Modulator

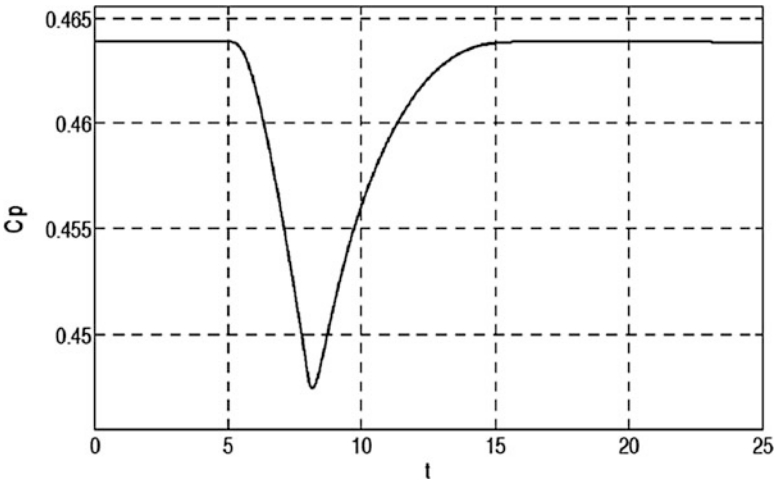


Fig. 5.56 Power coefficient *Cp(λ)* during simulation

(SPWM) switched (AC/DC/AC) converter (diode rectifier and 6-pulse IGBT inverter) interface scheme serves as a combined voltage stabilization regulator and maximum wind energy utilization and enhancement compensator.

5.9.1 Components of Standalone WECS

The standalone WECS connected to the local load bus over a radial transmission line comprising the following main components, as shown in Fig. 5.57.

- Wind turbine.
- Gear box.
- Step up and step down transformers.
- Distribution power lines.
- Induction generator.
- Stabilization interface scheme and stabilization controller.
- The hybrid electric load.

Each component of the proposed WECS shown in Fig. 5.57 is modeled in MATLAB/SIMULINK environment. The hybrid composite linear, nonlinear and motorized loads are shown in Fig. 5.58.

The parameters of the proposed controller are selected by a guided trial and error off-line simulation to ensure the minimum induction load and generator voltage excursion for any large wind and load variation. The standalone WECS sample study system unified AC system model parameters, comprising the induction

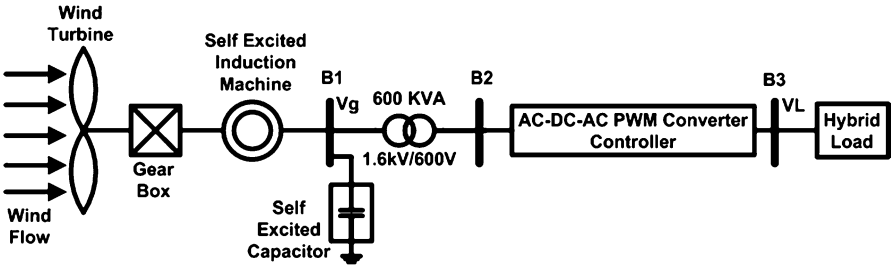


Fig. 5.57 Sample study 600 kVA wind energy conversion scheme

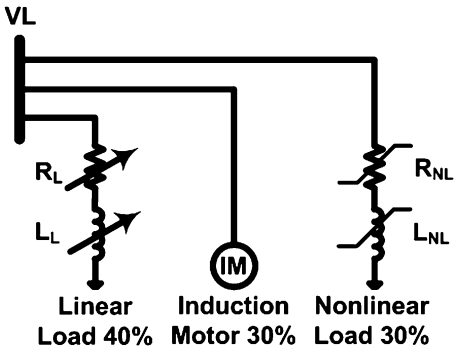


Fig. 5.58 Hybrid composite linear, nonlinear and motorized load model

generator, wind turbine, combined hybrid load, and controller parameters are all given below.

5.9.1.1 Simple Wind Turbine Model (Quasi-static Model)

$$T_W = \frac{1}{2\lambda}\rho A R C_P V_W^2 = \frac{1}{2\omega_W}\rho A C_P V_W^3 = k\frac{V_W^3}{\omega_W}$$

where

ρ is the specified density of air (1.25 kg/m^2)
A is the area swept by the blades
R is the radius of the rotor blades
C_P is power conversion coefficient
λ is the tip speed ratio
ω_W is the wind turbine velocity in rpm
k is equivalent coefficient in per unit (0.745)

Induction Generator

Three phase, two pairs of poles, $V_g = 1.6$ kV (L-L), Sg = 600 KVA, Cself = 150 uF/Phase

$$R_s = 0.016, L_{ls} = 0.06, R_r^{'} = 0.015, L_{lr}^{'} = 0.06$$
$$L_m = 3.5, H = 2, F = 0, p = 2$$

Combined Hybrid AC Load Model (@ V = 1.0 pu)

Linear PQ Load (40 %)

$$P_L = 0.4pu, Q_L = 0.4pu$$

Nonlinear (Voltage-Dependent Type) PQ Load (30 %)

$$P = P_o\left(\frac{V_g}{V_{go}}\right)^{\alpha},\ Q = Q_o\left(\frac{V_g}{V_{go}}\right)^{\alpha}$$
$$P_o = 0.3pu, Q_o = 0.3pu, V_{go} = 1.0pu,$$
$$\alpha = 2 - 3, \beta = 2 - 3\,(Nonlinearity\ \ order)$$

Three Phase Squirrel Cage Induction Motor Inrush Type PQ Load (30 %)

Power: $S_M = 0.3pu$
Stator resistance and leakage inductance: $R_s = 0.0201pu, L_{ls} = 0.0349pu$
Rotor resistance and leakage inductance:$R_r = 0.0377pu, L_{lr} = 0.0349pu$
Magnetizing inductance: $L_m = 1.2082pu$
Pole pairs: 2

Per Unit Base Values Used

S_{base} = 600KVA, V_{base} = 1.6 kV (L-L)

DC-Link RLC Filter Parameters

$$R_f = 0.1\Omega,\ \ L_f = 0.2mH,\ \ C_f = 5000uF$$

Weight Factors

$$\gamma_{vg} = 10, \ \gamma_{vdc} = 10, \ \gamma_{vload} = 0.1$$

SPWM Switching Frequency

$$f_{sw} = 2kHz, \ 6 pulse IGBT \ operation$$

5.9.2 MATLAB/SIMULINK Model

The MATLAB/SIMULINK functional model of full AC study system is given in Fig. 5.59.

The sample WECS standalone scheme was subjected to severe combined sequence of load switching/load variation/load excursion and wind speed variation and gusting. The novel tri-loop dynamic voltage tracking and FLC control scheme is shown in Fig. 5.60. The control signal adjusts the sinusoidal pulse width modulated IGBT six pulse inverter. The system real time dynamic response for a combined load/wind excursion time sequence as follows:

t = 0.03 s Linear Load excursion applied, +30 %
t = 0.04 s Linear Load excursion removed, +30 %
t = 0.05 s Linear Load excursion applied, −30 %
t = 0.06 s Linear Load excursion removed, −30 %
t = 0.07 s Wind Speed excursion applied, −30 %
t = 0.08 s Wind Speed excursion removed, −30 %
t = 0.09 s Wind Speed excursion applied, +30 %
t = 0.10s Wind Speed excursion removed, +30 %

5.9.3 Simulation Results

The WECS dynamic performance is compared for the two cases, with and without the AC/DC/AC PWM converter based on the novel tri-loop dynamic voltage tracking tri-loop FLC. Fig. 5.61 shows the system dynamic response including generator voltage (RMS), load voltage (RMS), DC-link voltage, generator current (RMS), average generator power, the voltage-current two-dimensional phase portraits for the wind energy system controller interface system as shown in Fig. 5.61.

This section discussed a novel low cost dynamic voltage error tri-loop fuzzy logic controller and sinusoidal PWM driven IGBT inverter scheme for stand alone wind energy system. This PWM power converter scheme is extremely effective in ensuring voltage stabilization and enhancing power/energy utilization under severe

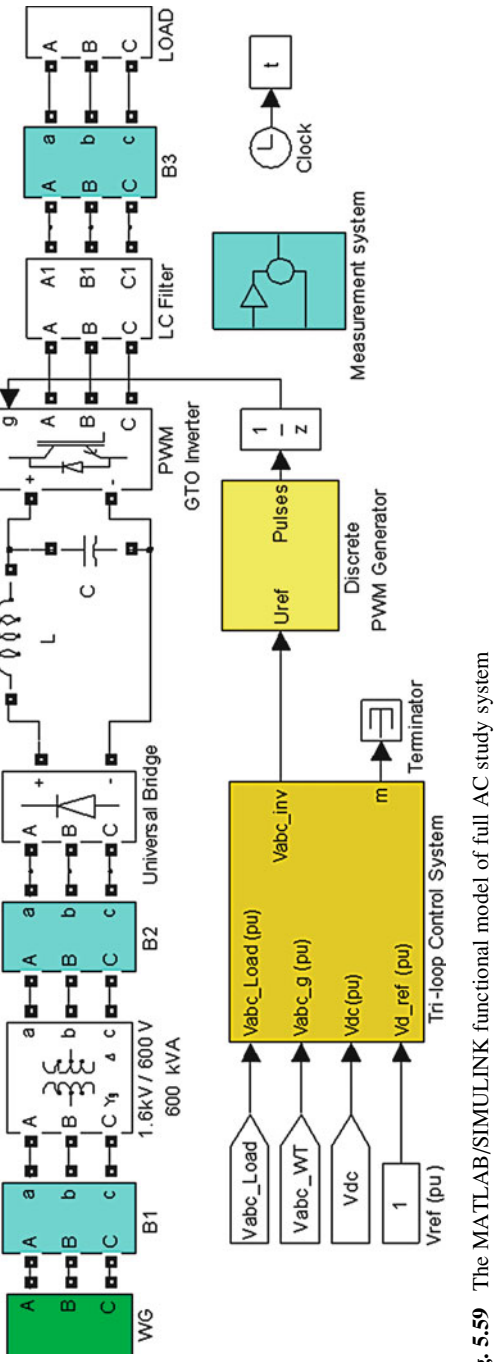


Fig. 5.59 The MATLAB/SIMULINK functional model of full AC study system

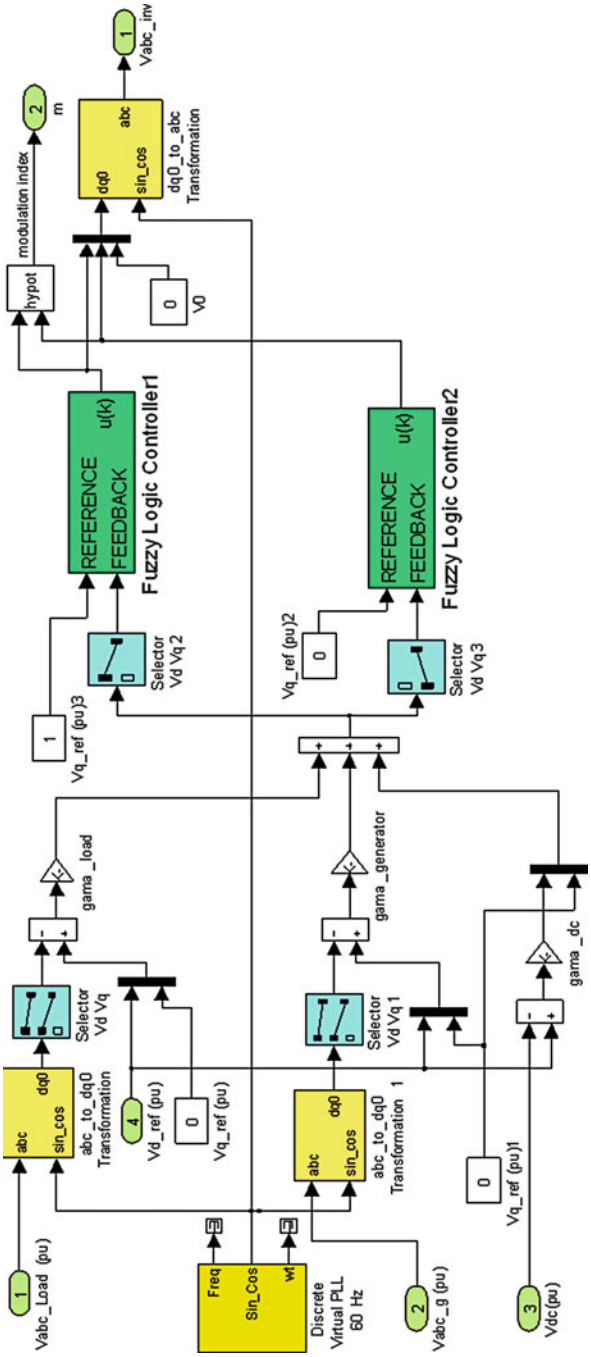


Fig. 5.60 Dynamic tri-loop global voltage error tracking FL controlled scheme for voltage stabilization at generator, load and inverter buses

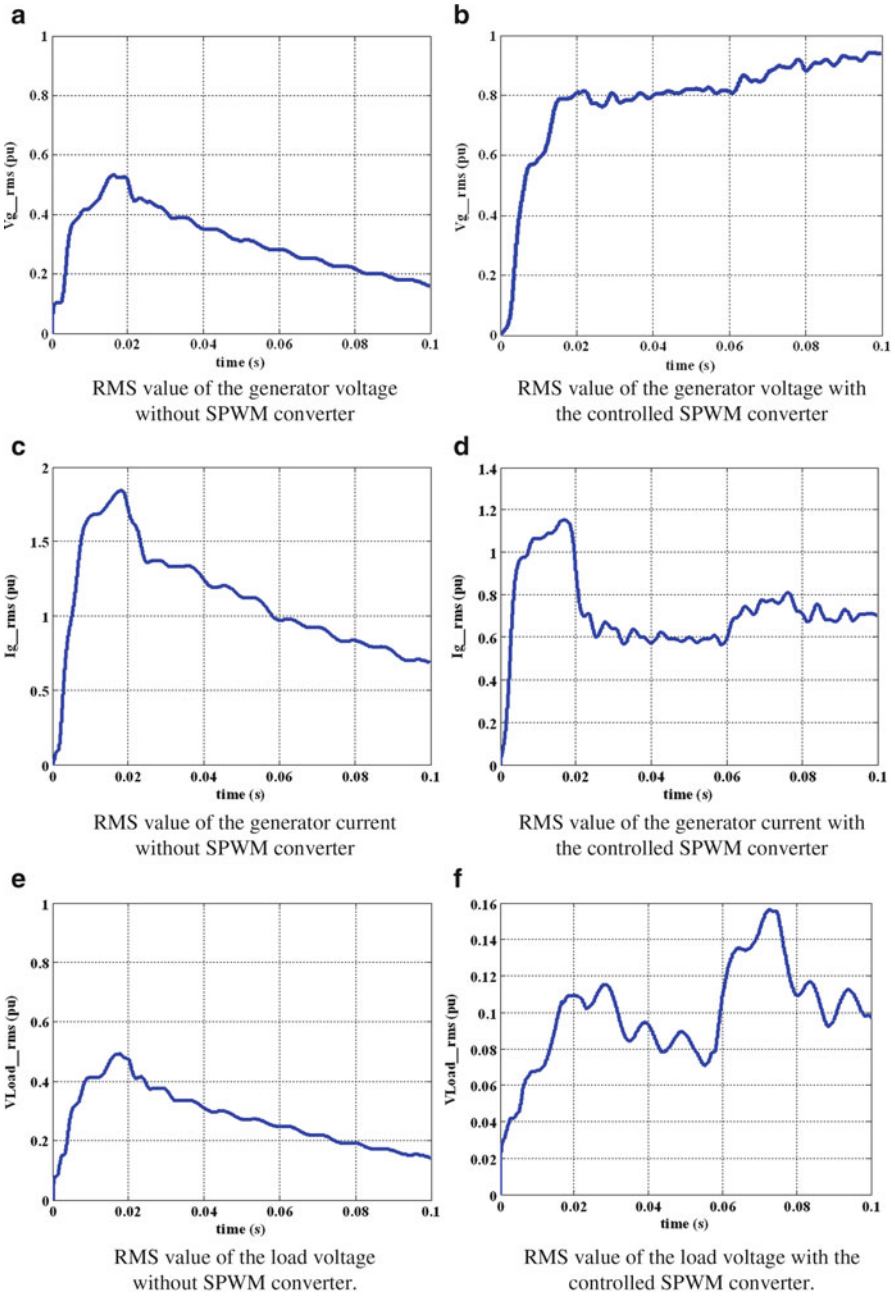
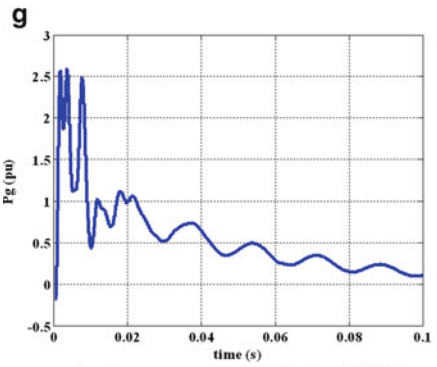
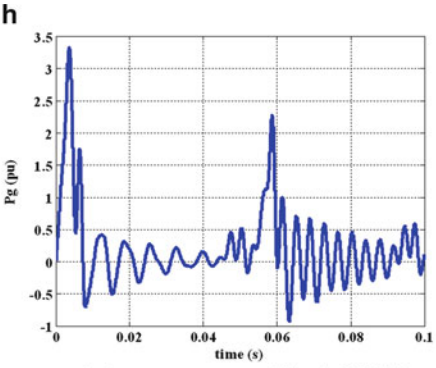


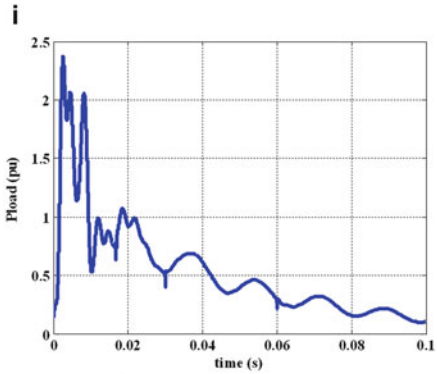
Fig. 5.61 WECS performance without and with the SPWM converter for a combined excursion sequence



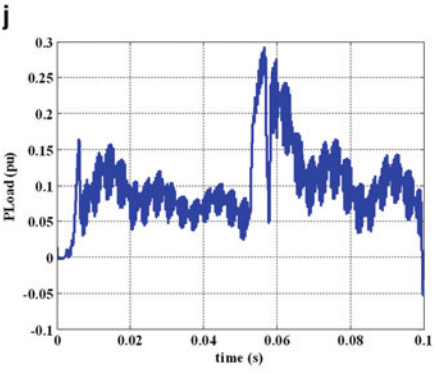
Active power generated by the WECS without SPWM converter.



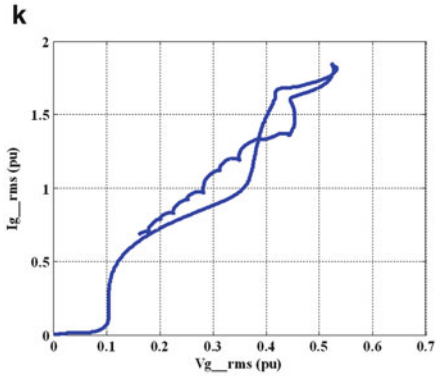
Active power generated by the WECS with the controlled SPWM converter.



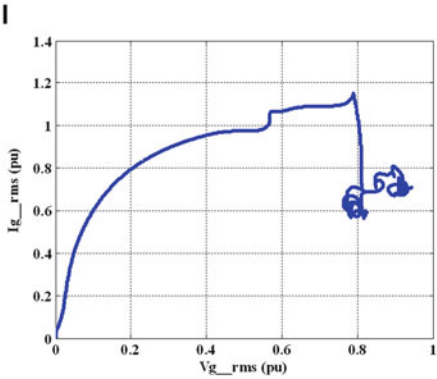
Active power dissipated by the load without SPWM converter.



Active power dissipated by the load with the controlled SPWM converter.



RMS current-voltage trajectory at generator terminals without SPWM converter.



RMS current-voltage trajectory at generator terminals with SPWM converter.

Fig. 5.61 (continued)

load and wind prime mover/wind velocity excursion. A tri-loop dynamic error driven Fuzzy Logic Controller is designed and employed in this application. The use of tri-loop dynamic error signal to the controller ensures that all changes, excursions or variations in WECS, in Loads, and in DC interface system are included in the controller.

5.10 Summary

This chapter reviewed and discussed the recent research, development and trends in the field of wind energy conversion systems based on soft computing tools. A theoretical knowledge based on the methodology involved in applications to wind energy power factor, pitch angle control, MPPT, economic dispatch for wind energy, and SEIG driven WECS based on artificial neural networks, ANFIS, fuzzy logic and genetic algorithms were summarized. MATLAB/SIMULINK models for several practical applications were provided with the simulation results for easy understanding and development. It has been observed that soft computing tools can enhance the efficiency and effectiveness of the operation and maintenance of wind power systems. In addition, the state of the art of soft computing techniques in wind energy conversion systems were provided to serve as guidance for future research work.

Review Questions

1. Develop a MATLAB/SIMULINK model for optimizing the wind blade chord and twist angle using Genetic Algorithm.
2. Explain the procedure involved in Fuzzy logic based Pitch Angle Control. Develop a suitable SIMULINK model and discuss the results.
3. How is wind turbine flicker calculated? Briefly discuss the application of Artificial Neural Network in wind turbine flicker computation.
4. How are ANNs applied to detect and identify faults in Wind Turbine Systems?
5. Develop a SIMULINK model for tuning of wind turbine using PI, and PID controllers using Genetic Algorithms.
6. Develop a DSTATCOM model using SIMULINK and compare the results with a fuzzy based design.

Bibliography

Adzic E, Ivanovic Z, Adzic M, Katic V (2009) Maximum power search in wind turbine based on fuzzy logic control. Acta Polytech Hung 6(1):131–149

AL-Ismail FS, Abido MA (2011) The impact of statcom based stabilizers on power system stability, using intelligent computational optimization approach. Innovative Smart Grid Technologies Asia (ISGT), 2011 IEEE PES, pp 1–13

Amei K, Takayasu Y, Ohji T, Sakui M (2002) Amaximum power control of wind generator system using a permanent magnet synchronous generator and a boost chopper circuit. Proc Power Conv 3:1447–1452

Attia A-F, Soliman H, Sabry M (2006) Genetic algorithm based control system design of a self-excited induction generator. Acta Polytech 46(2):11–22

Belghazi O, Cherkaoui M (2012) Pitch angle control for variable speed wind turbines using genetic algorithm controller. J Theor Appl Inf Technol 39(1):6–10

Brini S, Abdallah HH, Ouali A, Economic dispatch for power system included wind and solar thermal energy. Leonardo J Sci (14):204–220

Chang HC, Liaw CM (2009) Development of compact switched reluctance motor drive for EV propulsion with voltage-boosting and PFC charging capabilities. IEEE Trans Veh Technol 58:3198–3215

Chen Y-M, Liu Y-C, Lin S-H (2003) Double-input PWM DC/DC converter for high/low voltage sources. In: Proceedings of the IEEE international telecommunications energy conference, pp 27–32

Dhal PK, Christober Asir Rajan C (2012) Transient stability improvement using neuro-fuzzy controller design for STATCOM. IEEE – international conference on advances in engineering, science and management (ICAESM – 2012), March 2012, pp 510–515

Goel N, Patel RN, Chacko ST (2010) Genetically tuned STATCOM for voltage control and reactive power compensation. Int J Comput Theor Eng 2(3)

Houck CR, Joins JA.A genetic algorithm for function optimization: a matlab implementation

http://www.ren21.net. Accessed June 2008

Hui J (2008) An adaptive control algorithm for maximum power point tracking for wind energy conversion systems. Department of Electrical and Computer Engineering, Queen's University

Hussein KH, Muta I, Hoshino T, Osakada M (1995) Maximum photovoltaic power tracking: an algorithm for rapidly changing atmospheric conditions. IEE Proc Gener Trans Distrib 142 (1):59

Jianzhong Zhang, Ming Cheng, Zhe Chen, Xiaofan Fu (2008) Pitch angle control for variable speed wind turbines. DRPT2008 6-9. Nanjing China

Lai Y-S, Lin J-C (2003) New hybrid fuzzy controller for direct torque control induction motor drives. IEEE Trans Power Electron 18(5):1211–1219

Lee J-C, Lin W-M, Liao G-C, Tsao T-P (2011) Quantum genetic algorithm for dynamic economic dispatch with valve-point effects and including wind power system. Int J Electr Power Energy Syst 33:189–197

Lin C-T, Lee CSG (1996) Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems. Prentice-Hall, Upper Saddle River

Luo A, Tang C, Shuai Z (2009) Fuzzy-PI-based direct-output-voltage control strategy for the STATCOM used in utility distribution systems. IEEE Trans Ind Electron 56(7):2401–2411

Mengi OO, Altas IH (2012) Fuzzy logic control for a wind/battery renewable energy production system. Turk J Electr Eng Comput Sci 20(2):187–206

Mienski R, Pawelek R (2004) Shunt compensation for power quality improvement using a STATCOM controller: modeling and simulation. IEEE Proc Gener Trans Distrib 151(2):274

Rasit ATA (2007) Neural prediction of power factor in wind turbines. J Electr Electron Eng 7 (2):431–438

Rasit ATA (2009) An adaptive neuro-fuzzy inference system approach for prediction of power factor in wind turbines. J Electr Electron Eng 1(9):905–912

Saetieo S, Torrey DA (1998) Logic control of a space-vector PWM current regulator for three-phase power converters fuzzy. IEEE Trans Power Electron 13(3):419–426

Shathiand B, Natatajan SP (2009) Comparative study on various unipolar PWM strategies for single phase five level cascaded inverter. Int J Power Electron 2(1):36–50

Soliman HF, Attia AA, Mokhymar SM, Badr MAL. Fuzzy algorithm for supervisory control of self-excited induction generator. JKAU: Eng Sci 17(2):19–40

Valarmathi R, Palaniswami S, Devarajan N (2012) Simulation and analysis of wind energy and photo voltaic hybrid system. Int J Soft Comput Eng 2(2):193–200, ISSN: 2231-2307

Xing-Peng Li, Wen-Lu Fu, Qing-Jun Shi, Jian-Bing Xu, Quan-Yuan Jiang (2013) A fuzzy logical MPPT control strategy for PMSG wind generation systems. J Electron Sci Technol 11(1)