Chapter 3
Soft Computing Techniques in Solar PV

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

- Importance of soft computing techniques such as neural networks, fuzzy logic and genetic algorithms in solar PV system.
- Soft computing techniques used in MPPT of Solar PV system and its MATLAB/SIMULINK model.
- Prediction of solar irradiance using soft computing techniques.
- Parameter estimation of Solar PV module using Genetic Algorithms.

3.1 Introduction

Problems that are computationally difficult to solve are known as NP-Complete/NP-Hard problems. Traditional algorithms take a very long time to determine solutions for these problems. The basic example of such kind of a problem is the Travelling Salesman Problem (TSP). Such problem cannot be solved using conventional deterministic process and hence stochastic process is required to handle them encompassing uncertainties. Soft computing techniques comprises of those paradigms in AI that relate to some kind of biological or naturally occurring system. General consensus suggests that these paradigms are neural networks, evolutionary computing, and fuzzy systems. Neural networks are based on their biological counterparts in the human nervous system. Similarly, evolutionary computing draws heavily on the principles of Darwinian evolution observed in nature. Genetic Algorithms are part of evolutionary computing algorithms. Finally, human reasoning using imprecise, or fuzzy, linguistic terms is approximated by fuzzy systems.

During the recent years, these soft computing techniques are commonly applied for designing, modeling, predicting, optimizing, and forecasting of renewable

S. Sumathi et al., *Solar PV and Wind Energy Conversion Systems*,
Green Energy and Technology, DOI 10.1007/978-3-319-14941-7_3

energy systems. In addition, innovative models, environmentally benign processes, etc. used in renewable energy systems are computationally-intensive, non-linear and complex as well as involve a high degree of uncertainty. Soft computing techniques are most suitable in handling the noise, imprecision, and uncertainty in the data, and in turn achieve robust, low-cost solutions. Hence, soft computing paradigms and intelligent algorithms are increasingly applied in the study of renewable energy systems. This chapter describes the application of fuzzy logic and neural networks for MPP tracking in solar PV arrays, fuzzy based solar powered energy, genetic algorithms for parameter identification in solar cells, and neural networks for forecasting of solar irradiance.

3.2 MPPT Using Fuzzy Logic

The application of fuzzy logic for MPPT has become more popular with the utilization of microcontrollers. Fuzzy logic controllers have the benefits of operating with general inputs, not needing associate correct mathematical model and handling nonlinearities. mathematical logic management typically consists of three stages: fuzzification, rule base table operation, and defuzzification. Throughout fuzzification, numerical input variables are regenerated into linguistic variables supported by a membership operate almost like Fig. 3.1. Fuzzy levels used are of the type: NB (Negative Big), NS (Negative Small), ZE (Zero), PS (Positive Small), and lead (Positive Big). In Fig. 3.1a, b are the membership values of the numerical variable.

The inputs to a MPPT fuzzy logic controller (FLC) are usually an error E and a change in error ΔE. The user has the flexibility of choosing how to compute E and ΔE. Since dP/dV vanishes at the MPP, the following approximation is used.

$$E(n) = P(n) - P(n-1)/V(n) - V(n-1) \tag{3.1}$$

$$\Delta E(n) = E(n) - E(n-1) \tag{3.2}$$

Once E and ΔE are the calculated and deviated change applied to the linguistic variables, the symbolic logic controller output, that is often an amendment in duty quantitative relation ΔD of the ability convertor, is researched in an exceedingly rule base table like Table 3.1. The linguistic variables assigned to ΔD for the various combos of E and ΔE based on the data of the user. Consider an example, the in operation purpose is much to the left of the MPP, that's E is metal, and

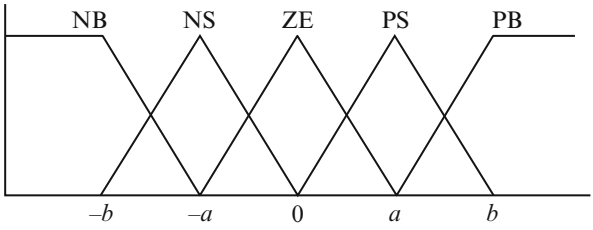


Fig. 3.1 Membership function for input and output of FLC

Table 3.1 Fuzzy rule base table

	ΔE				
E	NB	NS	ZE	PS	PB
NB	ZE	ZE	NB	NB	NB
NS	ZE	ZE	NS	NS	NS
ZE	NS	ZE	ZE	ZE	PS
PS	PS	PS	PS	ZE	ZE
PB	PB	PB	PB	ZE	ZE

ΔE is ZE, then the duty quantitative relation is basically accumulated, that's ΔE is metal to achieve the MPP.

In the defuzzification stage, the fuzzy logic controller output is regenerated from a linguistic variable to a numerical variable still employing a membership operation as in Fig. 3.1. This provides an analog signal which will manage the facility device to the MPP. MPPT fuzzy logic controllers have shown to perform well below variable atmospherical conditions. However, their effectiveness depends on the data of the user or management engineer in selecting the correct error computation and bobbing up with the rule base table.

3.2.1 Implementation

To beat a percentage of the disservices mentioned in past MPPT strategies, fuzzy logic controller is utilized for most extreme force following the PV Panel. The primary distinction from the past routines is that the correct depiction of the framework to be controlled is not needed. Fuzzy logic permits the determination of the principle base by etymological terms and consequently, the tuning of the controller is an extremely basic way which is qualitatively not quite the same as accepted configuration systems. Moreover, fuzzy control is nonlinear and versatile in nature, which provides strong execution under parameter variety, load and supply voltage. Fuzzy logic is progressively utilized within present times as a helpful device to model control frameworks which are nonlinear in nature, in the same way as the solar based PV exhibit to track the greatest force. The control inputs to the fuzzy logic controller are lapse and change of failure, while the yield is the change of control indicator for the band width adjustment generator. A band width modulator is utilized to supply a band train to the exchanging MOSFET. The MOSFET is inclined into a directing state when the beat voltage is high and into a non-leading state when the beat voltage is low.

The obligation cycle of the beat decides the powerful impedance seen by the sun oriented cell. Therefore by basically modifying the obligation cycle of the switch, the current stream to the battery may be controlled. A suitable charge controller to attain the control portrayed above is a DC-to-DC controller with beat width balance control. A buck sort converter, which steps down cell voltage to an easier quality, is indicated in Fig. 3.2.

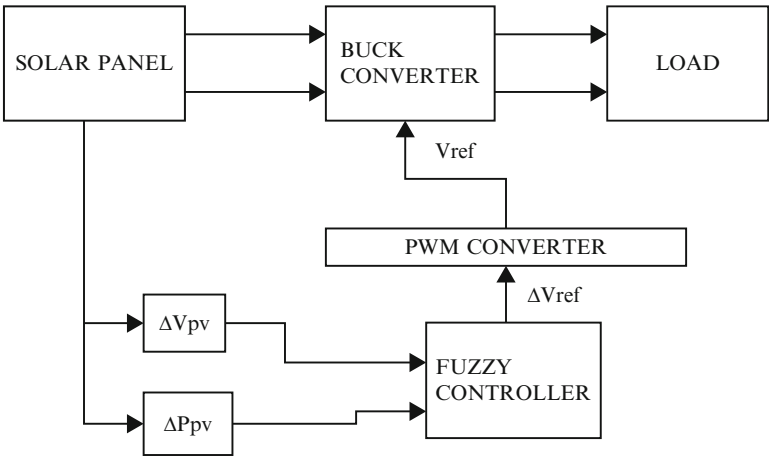


Fig. 3.2 Block diagram of the scheme

The inputs to the fuzzy controller are change in PV power (ΔPpv) and change in PV show voltage (ΔVpv) relating to the two inspecting time moments. The two inputs are prepared by the fuzzy controller and the yield of the fuzzy controller is the incremental reference voltage (ΔVref), which shifts in size and extremity relying upon which region of the Ipv × Vpv curve, the framework is working on. This yield is given to PWM generator, which yields the reference voltage to the buck converter. The fuzzy based plan utilized yields an incremental reference voltage of fitting extremity and variable greatness. In this way throughout transient conditions the fuzzy logic controller yields a bigger incremental reference voltage to accelerate the transient reaction however yields practically zero incremental reference voltage close to the top power locale to diminish motions about the MPP.

3.2.2 Description and Design of FLC

Fuzzy model of the framework is planned based on master information of the fuzzy framework. The fuzzy logic controller is partitioned into four segments: Fuzzification, rule base, inference and defuzzification. The inputs to the fuzzy logic controller are change in PV array power (ΔPpv) and change in PV array voltage (ΔVpv) and the output is the change in reference voltage (ΔVref) (Fig. 3.3).

3.2.2.1 Fuzzification

The fuzzy model is developed on a trial-and-error basis to meet the desired performance criteria.

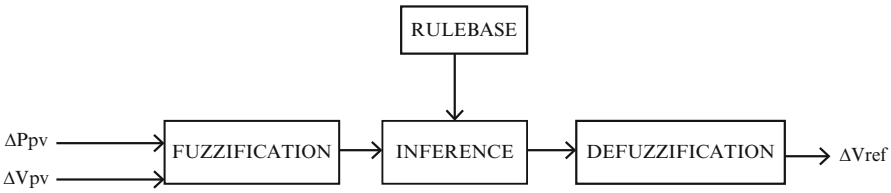


Fig. 3.3 Block diagram of the FLC

The universe of discourse for input variable 1 (ΔPpv) is divided into seven Fuzzy sets: PL (Positive Large), PM (Positive Medium), PS (Positive Small), Z (Zero), NS (Negative Small), NM (Negative Medium) and NL (Negative Large). Here, the Fuzzy set PS assumes a membership value greater than zero beginning at the origin, in order to speed up the start-up process and at the same time prevent variation of the reference voltage at the MPP. Additional Fuzzy sets PM and NM has been added to improve the control surface.

The universe of discourse for input variable 2 (ΔVpv) is divided into seven fuzzy sets: PL (Positive Large), PM (Positive Medium), PS (Positive Small), Z (Zero), NS (Negative Small), NM (Negative Medium) and NL (Negative Large). The universe of discourse for the output variable (ΔVref) is divided into seven fuzzy sets: PL (Positive Large), PM (Positive Medium), PS (Positive Small), Z (Zero), NL (Negative Large), NM (Negative Medium) and NS (Negative Small). The membership functions for the input and output variables are shown in Fig. 3.4. The membership functions for the input and output variables are designed to model the unsymmetrical nature of the PV panel Ipv × Vpv curve. The membership functions are denser at the center to provide greater sensitivity in the region near the MPP. Input membership functions are normalized and suitable tuning gains are used to match the inputs to the respective universe of discourse.

3.2.2.2 Rule Base

The Fuzzy algorithm tracks the maximum power based on the rule-base: If the last change in the reference voltage (Vref) has caused the power to increase keep changing the reference voltage in the same direction; else if it has caused the power to drop, move it in the opposite direction. A rule base consisting of 49 rules is designed as shown in Table 3.2.

3.2.2.3 Inference Method

The Inference technique decides the yield of the fuzzy controller. Mamdani's inference system is utilized within the recognized framework alongside the max-min creation strategy. This is on the grounds that this system is computationally more proficient and has preferable interpolative properties over techniques dependent upon other suggestion capacities. Henceforth, Mamdani's deduction system is normally prominent for most control building provisions.

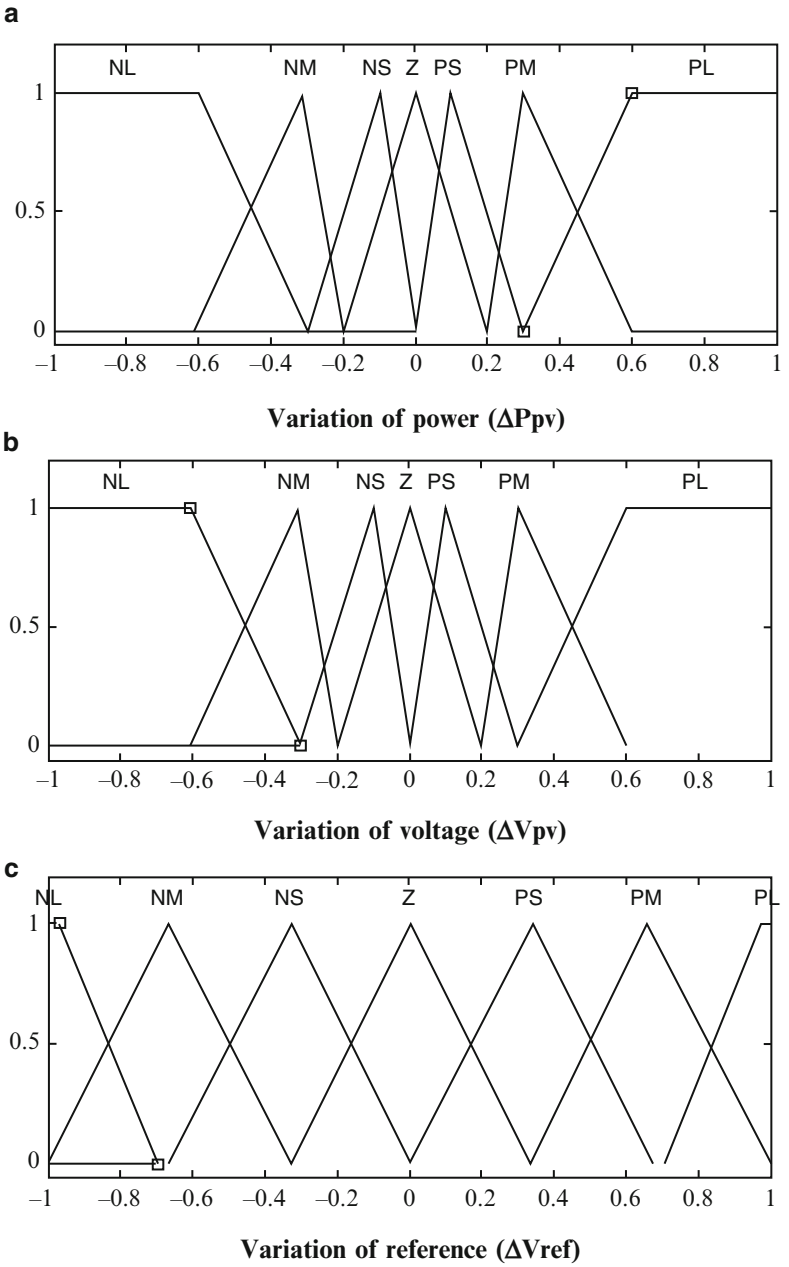


Fig. 3.4 Membership functions for the fuzzy model (**a**) input ΔPpv, (**b**) Input ΔVpv (**c**) output ΔVref

Table 3.2 Rule base for the fuzzy model

ΔV_{pv}/	ΔP_{pv}						
	NL	NM	NS	Z	PS	PM	PL
NL	PL	PL	PL	PL	NM	Z	Z
NM	PL	PL	PL	PM	PS	Z	Z
NS	PL	PM	PS	PS	PS	Z	Z
Z	PL	PM	PS	Z	NS	NM	NL
PS	Z	Z	NM	NS	NS	NM	NL
PM	Z	Z	NS	NM	NL	NL	NL
PL	Z	Z	NM	NL	NL	NL	NL

3.2.2.4 Defuzzification

The output of the fuzzy controller is a fuzzy variable. However a crisp quantity is needed. Thus the output of the fuzzy controller ought to be defuzzified. The centroid method of defuzzification is one of the normally utilized defuzzification routines and is the one being utilized for the framework. This system has great averaging properties and recreation effects have demonstrated that it gives the best outcome.

3.2.3 Simulation and Results

The SIMULINK model of fuzzy logic based maximum power tracking for a solar panel with 60 W is shown in Fig. 3.5. The performance of the SIMULINK model is validated with a PV panel,, buck converter and a resistive load. The objective of the modeled FLC using SIMULINK is to track maximum power irrespective of panel voltage variations. Accordingly FLC uses two input variables: change in PV array Power (ΔPpv) and change in PV array voltage (ΔVpv) corresponding to the two sampling time instants. Equations (3.3, 3.4, and 3.5) determine the required power, change in array voltage and reference variable. The change in control variable (ΔVref) is considered as the output variable. (ΔVref) is integrated to achieve desired Vref value. The duty cycle of the DC-DC converter is determined using the reference voltage Vref.

$$\Delta P_{pv} = \left[P_{pv}(k) - P_{pv}(k-1)\right] * K1 \tag{3.3}$$

$$\Delta V_{pv} = \left[V_{pv}(k) - V_{pv}(k-1)\right] * K2 \tag{3.4}$$

$$\Delta V_{ref} = \left[V_{ref}(k) - V_{ref}(k-1)\right] * K3 \tag{3.5}$$

where K1, K2, K3 denote the gain coefficients and k is a time index. The crisp inputs and output are converted to linguistic variables to determine FLC output value. In order to perform this conversion fuzzy membership functions are used. Here, all membership functions are defined between −1 and 1 interval by means of input scaling factors K1 and K2 and the output scaling factor K3. Thus, since simple

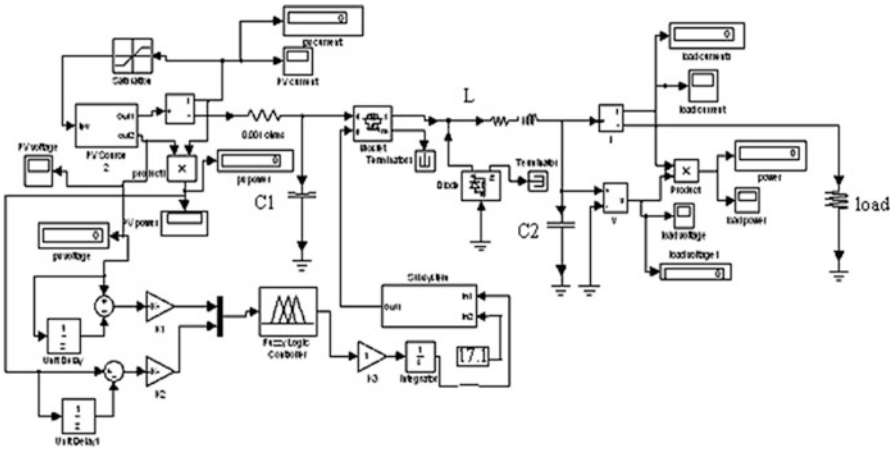


Fig. 3.5 FLC SIMULINK model

numbers are now processed in controller after scaling, fuzzy computation is performed in a shorter time. The linguistic terms for input and output values are represented by seven membership functions as shown in Fig. 3.4. The two inputs (ΔPpv) and (ΔVpv) are processed by the FLC and outputs a control variable (ΔVref) based on the control rules as shown in Table 3.2. The fuzzy output is given to the PWM generator which outputs the reference voltage to the buck converter. Fundamentally, the operating principle of PWM generator is based on the comparison of two signals. One of the signals is a triangular waveform and the other one is fixed linear signal, which represents time equivalent of triggering voltage.

Consequently, reference voltage time signal and triangular signal are U1 and U2 variables of 'IF' block used in simulation model as shown in Fig. 3.5. The output of the pulse width modulator is used to supply a pulse train to the switching MOSFET. The MOSFET is biased into a conducting state when the pulse voltage is high and into a non-conducting state when the pulse voltage is low. Thus the fuzzy algorithm tracks the maximum power based on the defined rules. The PV source block and the sub-system block are shown in Figs. 3.6 and 3.7 respectively.

Fuzzy logic controller model is simulated in MATLAB/SIMULINK environment to track the maximum power point and the value of the maximum power tracked is 59.9 W as shown in Fig. 3.8.

3.3 Neural Networks for MPP Tracking

The human brain mainly inspires artificial neural networks. This doesn't mean that Artificial Neural Networks are exact simulations of the biological neural networks inside our brain because the actual working of human brain is still a mystery. Neural

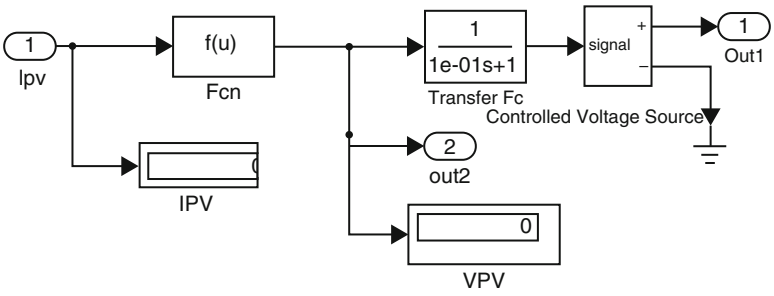


Fig. 3.6 PV source block

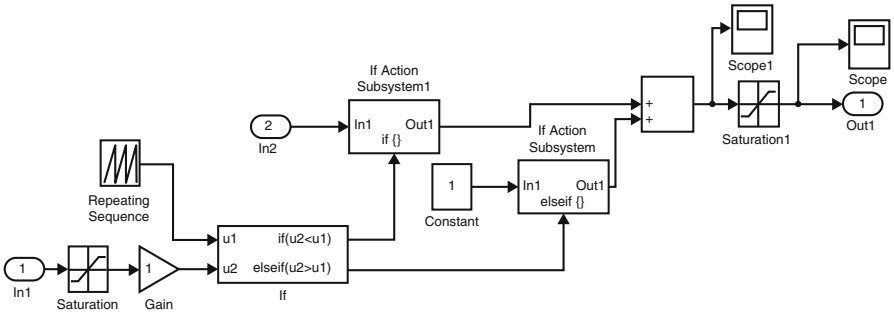


Fig. 3.7 Subsystem block

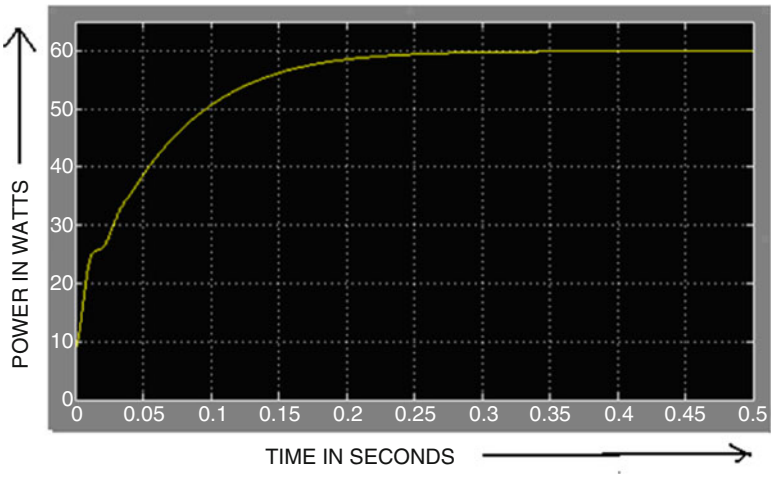


Fig. 3.8 Power characteristic curve with FLC

Network is a machine that is designed to model the way to which the brain performs a particular task. The network is implemented using electronic components or is simulated in software on digital computer. Neural networks perform usual computations through process of learning.

3.3.1 Background of Neural Networks

Neural network is a massively parallel distributed processor made up of simple processing units, which has a natural property of storing experimental knowledge and making it available for use. It resembles the brain in two respects:

– Knowledge is acquired by the network from its environment through learning process.
– The interneuron connection strengths, known as synaptic weights are used to store the acquired knowledge.

The primary significance of the neural network is the ability of the network to learn from its environments and to improve its performance through learning. It learns about its environment through an interactive process of adjustments applied to its synaptic weights and biases. The network becomes more knowledgeable about its environment after each iteration of learning process. The definition of learning process implies the sequence of events:

- The neural network is stimulated by an environment.
- Neural network undergoes changes in its parameters.
- Neural network responds in a new way to the environment because of the changes that have occurred in its internal structure.

The fundamental processing element of a neural network is a neuron. This building block of human awareness encompasses a few general capabilities. Basically, a biological neuron receives inputs from other sources, combines them in some way, performs a generally nonlinear operation on the result, and then outputs the final result. Figure 3.9 shows the relationship of these four parts.

Within humans there are many variations on this basic type of neuron, further complicating man's attempts at electrically replicating the process of thinking. Yet, all natural neurons have the same four basic components. These components are known by their biological names – dendrites, soma, axon, and synapses. Dendrites are hair-like extensions of the soma, which act like input channels. These input channels receive their input through the synapses of other neurons. The soma then processes these incoming signals over time. The soma then turns that processed value into an output, which is sent out to other neurons through the axon and the synapses.

But currently, the goal of artificial neural networks is not the grandiose recreation of the brain. On the contrary, neural network researchers are seeking an understanding of nature's capabilities for which people can engineer solutions to

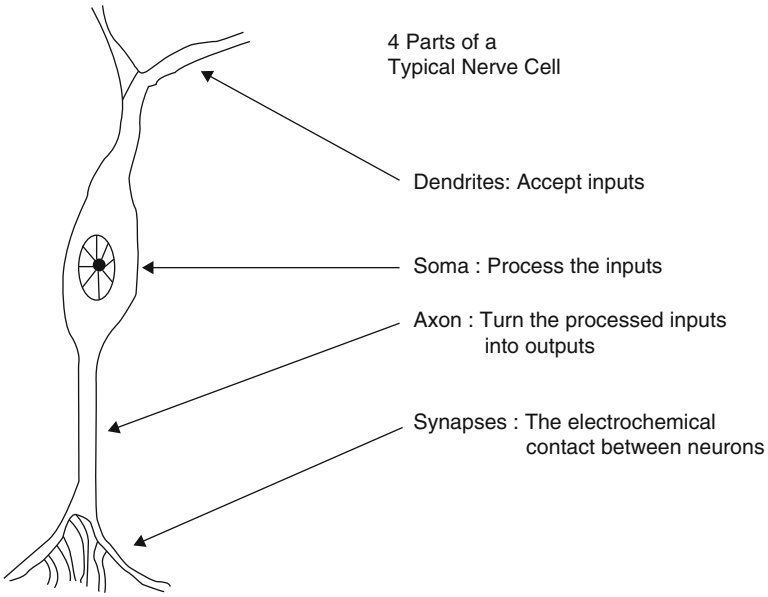


Fig. 3.9 A simple neuron

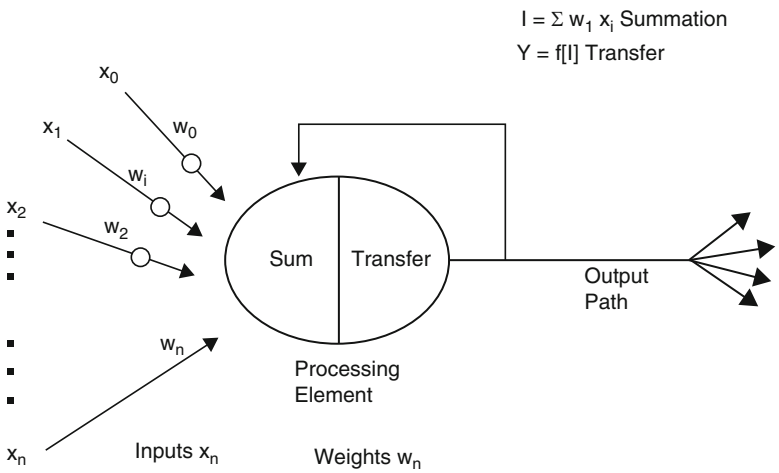


Fig. 3.10 A basic artificial neuron

problems that have not been solved by traditional computing. To do this, the basic unit of neural networks, the artificial neurons, simulate the four basic functions of natural neurons. Figure 3.10 shows a fundamental representation of an artificial neuron.

In Fig. 3.10, various inputs to the network are represented by the mathematical symbol, x(n). Each of these inputs are multiplied by a connection weight. These weights are represented by w(n). In the simplest case, these products are simply summed, fed through a transfer function to generate a result, and then output. This process lends itself to physical implementation on a large scale in a small package. This electronic implementation is still possible with other network structures, which utilize different summing functions as well as different transfer functions.

3.3.2 Implementation

A three layered feed forward network is created using newff command in MATLAB. The network is created with two neurons in the input layer, 50 neurons in the hidden layer and one neuron in the output layer. The two inputs to the network are the insolation and the temperature. Totally 23 sets of input data samples are given to the network for training. The targets of the network are specified to be the values of voltages that are obtained from modeling for a particular input set among training sets. The activation functions used at the input and output layers are tansig and purelin functions respectively. The tansig activation function calculates its output according to:

$$\text{tansig}(\text{n}) = 2/(1 + \exp(-2 * \text{n})) - 1 \tag{3.6}$$

The purelin activation function calculates its output according to:

$$\text{purelin}(\text{n}) = \text{n} \tag{3.7}$$

The basic structure of the neural network is shown below in Fig. 3.11.

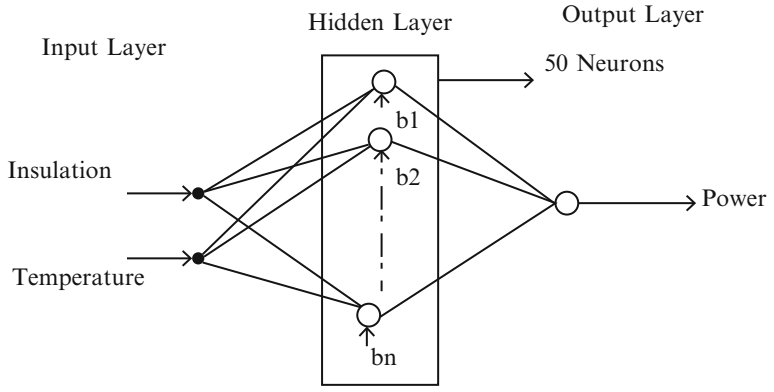


Fig. 3.11 Basic structure of neural network

The error criterion that is considered for training is Mean Square Error. The learning function is taken to be trainlm. "Trainlm" is a network training function that updates weight and bias values according to Levenberg-Marquardt (LM) optimization. This is one of the most popularly used algorithms which are a type of numerical optimization technique that has many advantages over its counterparts. The main advantage of this algorithm is that it requires less number of data for training the network and achieves accurate results. The other advantage of using LM optimization method is that it produces accurate results even if the system is not completely controllable and observable. Also it need not compute the Hessian matrix. When the performance function has the form of a sum of squares, then the Hessian matrix is approximated as

$$\mathrm{H} = \mathrm{J}^{\mathrm{T}}\mathrm{J} \ldots . \tag{3.8}$$

and the gradient is computed as

$$\mathrm{g} = \mathrm{J}^{\mathrm{T}}\mathrm{e} \ldots . \tag{3.9}$$

The Jacobian matrix(J) is computed through a standard back-propagation (BP) that is much less complex than computing the Hessian matrix. The LM method uses this approximation to the Hessian matrix in the following Newton-like update as follows.

$$x_{k+1} = x_k - \left[J^T J + \mu I\right]^{-1} J^T e \tag{3.10}$$

When the scalar μ is zero, this is just Newton's method, using the approximate Hessian matrix. When μ is large, this becomes gradient descent with a small step size. Newton's method is faster and more accurate near an error minimum, so the aim is to shift towards Newton's method as quickly as possible. Thus, μ is decreased after each successful step (reduction in performance function) and is increased only when a tentative step would increase the performance function. In this way, the performance function will always be reduced at each iteration of the algorithm. The error BP learning is implemented for updating the weights of the network in order to minimize the mean square error. The BP algorithm consists of two passes.

- Forward pass.
- Backward pass.

In the forward pass, 23 pairs of insolation and temperature are given to the network. As these are given all at a time, this type of training is called Batch training. These inputs propagate through the network layer-by-layer and the output voltage is generated. These inputs are used for computing the gradient and updating the network weights and biases. Now the output voltage is compared with the corresponding target value and then the error which is the difference between these

two values is propagated through the network in the backward pass. During this process, the weights of the network are updated recursively. The BP algorithm uses the following rule for updating the weights of the network.

$$w(k+1) = w(k) - \eta g(k) \tag{3.11}$$

where η------learning rate

g(k)-----gradient vector

The gradient vector is computed in the backward pass by applying the chain rule. The training parameters for trainlm are epochs, show, goal, time, min_grad, max_fail, mu, mu_dec, mu_inc, mu_max. Training occurs according to the trainlm's training parameters as shown below.

net.trainParam.epochs ---------- Maximum number of epochs to train
net.trainParam.goal ---------- Performance goal
net.trainParam.max_fail -------- Maximum validation failures
net.trainParam.mem_reduc------ Factor used for memory/speed purpose
net.trainParam.min_grad ------- Minimum performance gradient
net.trainParam.mu ---------- Initial Mu
net.trainParam.mu_dec---------- Mu decrease factor
net.trainParam.mu_inc---------- Mu increase factor
net.trainParam.mu_max --------- Maximum Mu
net.trainParam.show ---------- Epochs between showing progress
net.trainParam.time ---------- Maximum time to train in seconds.

The performance goal is specified as 0.15. The maximum number of epochs are set to 30. This is done on trial basis. When the epoch number is mentioned to be low than this number, training is stopped without reaching the performance. Maximum validation failures is kept as 5. Mu decrease factor is mentioned as 0.01. The time interval for showing the Epoch's progress is set to 50. This shows the training curve for every 50 epochs. The other parameters are set to their default values. The default values for these parameters are shown below.

net.trainParam.mu_inc ---------10
net.trainParam.mu_max ------- 1e-10
net.trainParam.mem_reduc---- 1
net.trainParam.min_grad ------ 1e-25

The parameter mu is the initial value for μ. This value is multiplied by mu_dec whenever the performance function is reduced by a step. It is multiplied by mu_inc whenever a step would increase the performance function. If mu becomes larger than mu_max, the algorithm is stopped. The trained network thus obtained is tested using nine sets of data.

3.3.3 Algorithm for ANN Based MPPT

Step-1: Construct the network and initialize the synaptic weights with random values.

Step-2: Apply the input sets to the network.

Step-3: Set the parameters of the network and calculate the corresponding output values by training the network.

Step-4: Compare the actual outputs with the desired outputs and determine a measure of the error.

Step-5: Determine the amount by which each weight is to be changed and make corrections to each weight.

Step-6: Repeat Step-3 to Step-4 until the error for the sets in the training set is reduced to an acceptable value.

Step-7: Validate the so formed network using testing sets.

A neural network SIMULINK block is generated using gensim (net) command. After the network is generated, nine sets of insolation and temperature are given to the network for validating the network. Each data set is given to the network at different instances of time with an interval of 1 s. The network is simulated. The curves for voltage, current and power are obtained. The overall SIMULINK model is shown in Fig. 3.12.

The structure of neural network is shown below in Fig. 3.13. It consists of two layers. The structures of the first and the second layers are shown in Figs. 3.14 and 3.15 respectively.

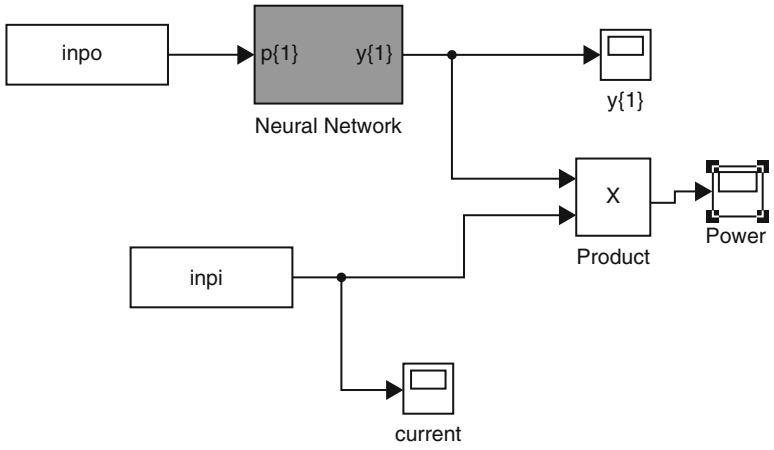


Fig. 3.12 Overall SIMULINK model

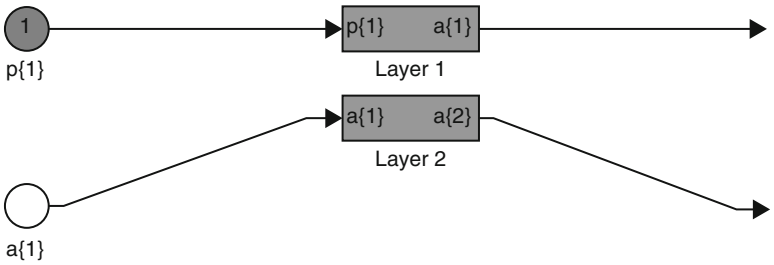


Fig. 3.13 Structure of neural network

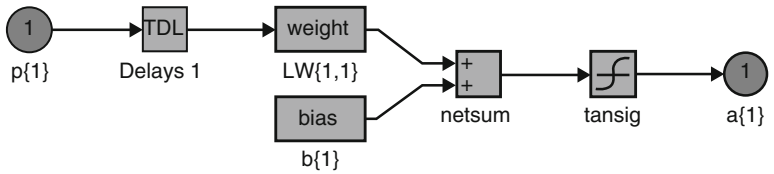


Fig. 3.14 Structure of first layer of neural network

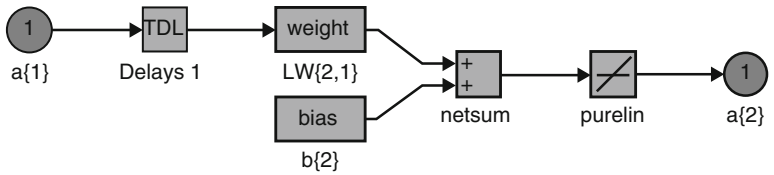


Fig. 3.15 Structure of second layer of neural network

3.3.4 Simulation Results

The training data used is shown below. These training sets were chosen to cover all the typical input space in order to get good performance where temperature ranges from −40 °C to 52 °C and solar irradiation ranged from 50 to 1,000 W/m^2. The first column shows the various values of insolation, second column corresponds to various temperatures and the third column corresponds to the target voltages.

Training data:		
50	50	16.15
100	−40	7.95
100	0	11.82
180	30	15.69
200	10	13.75
220	40	17.04

(continued)

Training data:		
600	52	19.45
700	42	18.58
770	47	19.15
830	35	18
850	50	19.68
900	10	15.41
910	12	15.6
920	13	18.17
950	23.75	16.96
955	23.87	16.97
960	24	16.96
965	24.12	16.96
970	24.25	17.02
975	24.37	17.02
980	24.5	17.08
985	24.62	17.13
995	24.87	17.14

The results are shown in Table 3.3. Training is completed for 25 epochs and the performance goal is met. The Mean Square error after training is 0.146852. The training curve is shown in Fig. 3.16.

The trained network is validated with the following nine sets of insolation and temperature conditions. The corresponding voltages and powers are tabulated in Table 3.3.

The curves for voltage, current and Power are shown below in Figs. 3.17, 3.18, and 3.19 respectively.

Table 3.3 Results of ANN based MPPT method

Insolation (W/m^2)	Temperature (°C)	Target voltage obtained from modeling (V)	Power obtained from modeling (W)	Voltage obtained from ANN based MPPT method (V)	Power obtained from ANN based MPPT method (W)
960	24	16.96	56.84	16.9698	56.8380
175	5	13.1	7.892	12.04	7.6394
1,000	25	17.1	59.9159	17.0862	59.8885
700	42	18.52	44.82	19.2	44.8192
965	24.125	16.96	57.21	16.98	57.2030
930	12	16.6	48.64	16.5	48.6104
995	24.87	17.14	59.41	17.0674	59.4013
820	40	18.46	53.14	18.05	52.9013
980	24.5	17.08	58.31	17.02	58.3007

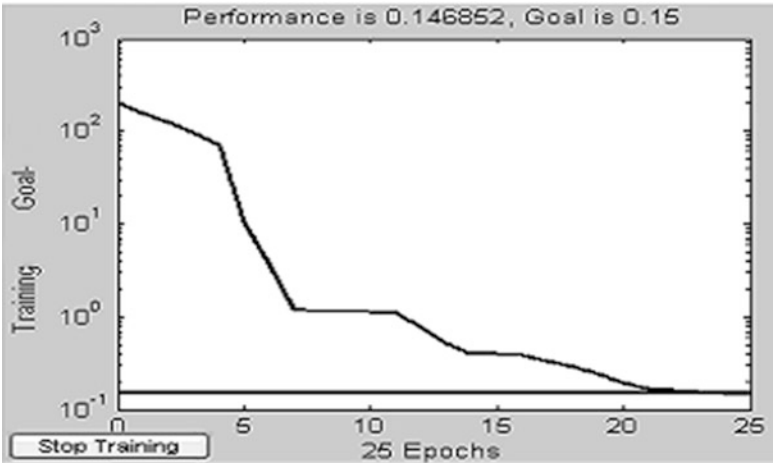


Fig. 3.16 Training curve

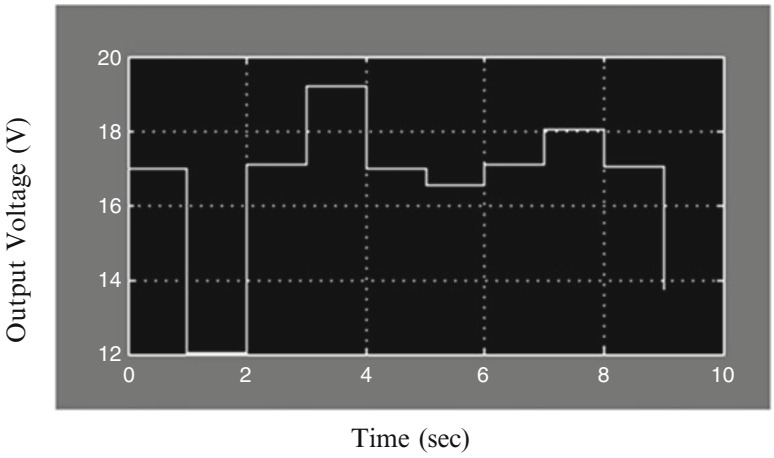


Fig. 3.17 Voltage curve for neural network based MPPT method

3.4 Neuro-Fuzzy Based MPPT Method

Over the previous decades there has been a solid resurgence in the field of information based nonlinear framework demonstrating and distinguishing proof including specialists from different orders. This present reality power framework issues might not, one or the other fit the presumptions of a solitary system nor be successfully explained by the qualities and competencies of a solitary procedure. One methodology to manage the complex true issues is to incorporate the utilization of two or more systems with a specific end goal to consolidate their distinctive

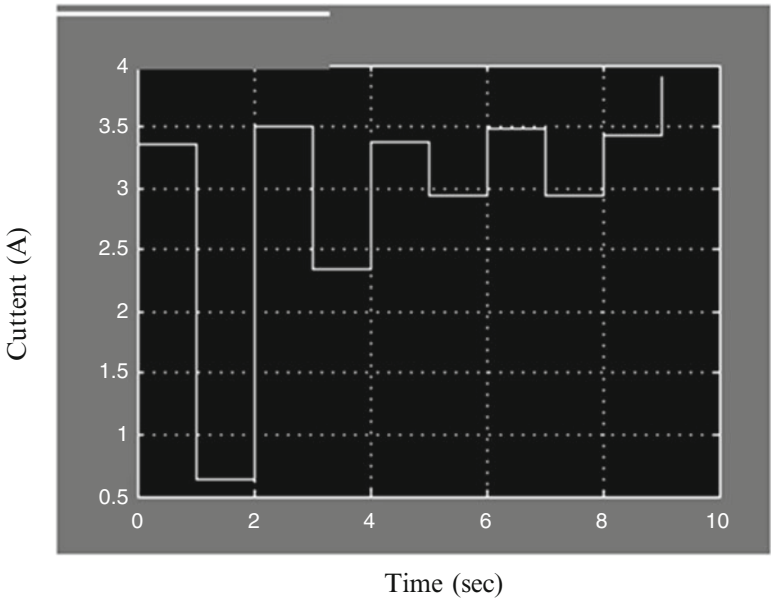


Fig. 3.18 Current curve for neural network based MPPT method

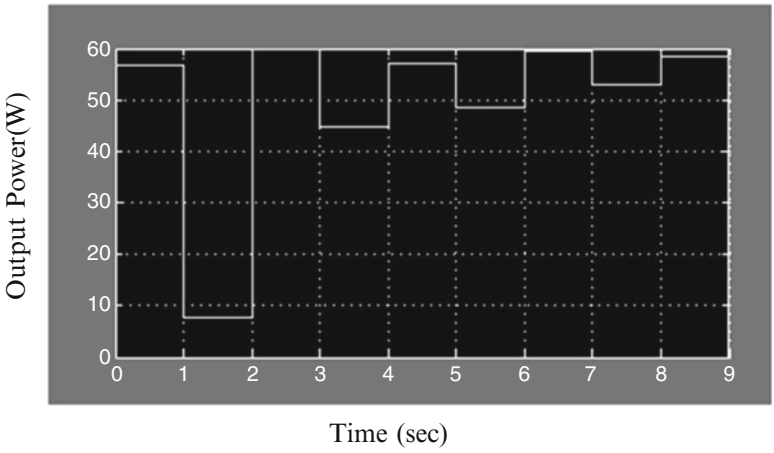


Fig. 3.19 Power curve for neural network based MPPT method

qualities and overcome one another shortcoming to produce cross breed results. Fuzzy Logic is the compelling apparatus for displaying uncertainty and approximate reasoning; however the presence of an expert is the most important part of the design. Master less outline obliges information which must blanket all the potential outcomes of operation. Neural systems are the best learning machines in the field. Adaptive Neuro Fuzzy Inference Systems joins the point of interest of fuzzy logic

and neural systems in one bundle and gives an enhanced fuzzy derivation framework installing the entire learning of the framework conduct (following operation). PV cells carry on as non-straight sources relying upon climatic parameters. Insolation and Temperature are the most paramount variables affecting the greatest force conveyed from the PV exhibit. Here, a cross breed system that joins fuzzy logic and neural systems is connected to recognize the greatest force point.

3.4.1 Fuzzy Neural Network Hybrids

In Engineering, a class of neural systems is ended up being equipped for speaking to a class of obscure nonlinear input–output mappings with self-assertive little rough guess lapse competence. The preference of the FL methodology is its logicality and transparency, where it is not difficult to consolidate from the earlier learning around a framework into an express fuzzy guideline base. A Neuro-Fuzzy hypothesis brings the plans of ANN and FL together in a binding schema, such that the resultant model has structure for learning properties and is connected with a fuzzy tenet base about the produced information. Neuro-fuzzy frameworks join together the regular etymological/typical transparency of fuzzy frameworks with the provable learning and representation ability of the ANN weights. The combo of qualitative based thinking through fuzzy logic and quantitative versatile numeric/information handling by means of ANN, is a possibly compelling idea, since it permits inside a solitary schema clever qualitative and quantitative thinking is attained. Really savvy frameworks must make utilization of all accessible learning: numerical, master or heuristic standards and referred to practical connections, for example, physical laws. Neuro-fuzzy frameworks permit all these. There are two methodologies to fuzzy neural structural planning which are alluded to as neuro-driven and fuzzy driven. The neuro-driven methodology offers power to the neural side of the depiction. Fuzzy math is utilized as a method for enhancing neuronal and/or whole system conduct. Then again, in the fuzzy driven methodology, neural systems are utilized as auxiliary apparatuses either to focus participation capacities, or to change the set of standards adaptively.

3.4.2 Theoretical Background of ANFIS

If X is a superset of n multi-class predictor maps X_i (i = 1 to n), each containing m patterns, then the strength of x_{ij}, the j^{th} (j = 1 to m) pattern on the i^{th} predictor map X_i, as an indicator of a target mineral deposit-type can be estimated in terms of class score (cs_{ij}), which is defined as:

$$cs_{ij} = w_i \;\; x \;\; w_{ij} \quad \left(\forall x_{ij} \in X_i\right) \tag{3.12}$$

where w_i is the map weight of the i^{th} predictor map and w_{ij} is the class weight of the j^{th} pattern on the i^{th} predictor map. The procedure for assigning class weights and map weights is based on the class scores of predictor patterns, n fuzzy sets $\tilde{A}_i^1$ (i = 1 to n) in X, containing 'favorable indicators of target mineral deposit-type,' can be defined as:

$$\tilde{A}_i^1 = \left\{ \left(x_{ij}, \mu_{\tilde{A}_i^1}(x_{ij}) \right) \ \forall x_{ij} \in X_i \right\} \tag{3.13}$$

where the fuzzy membership function $\mu_{\tilde{A}} i^1$ for estimating the fuzzy membership value of x_{ij} in $\tilde{A}_i^1$ is defined as:

$$\mu_{\tilde{A}_i^1}(x_{ij}) = e^{\frac{-\left(cs_{ij}-c_{i1}\right)^2}{2\sigma_{i1}}} \quad (\forall xij \in X_i) \tag{3.14}$$

where c_{i1} (i = 1 to n) and σ_{i1} (i = 1 to n) are the parameters that, respectively, define the center and the spread of the Gaussian function and cs_{ij} is the class score of x_{ij}.

Similarly, n fuzzy sets $\tilde{A}_i^2$(i = 1 to n) in X, containing 'unfavorable indicators of target mineral deposit-type,' can be defined as follows:

$$\tilde{A}_i^2 = \left\{ \left(x_{ij}, \mu_{\tilde{A}_i^2}(x_{ij}) \right) \ \forall x_{ij} \in X_i \right\} \tag{3.15}$$

where the fuzzy membership function $\mu_{\tilde{A}_i^2}$ for estimating the fuzzy membership value of x_{ij} in $\tilde{A}_i^2$ is defined as:

$$\mu_{\tilde{A}_i^2}(x_{ij}) = e^{\frac{-\left(cs_{ij}-c_{i2}\right)^2}{2\sigma_{i2}}} \quad (\forall xij \in X_i) \tag{3.16}$$

where c_{i2} (i = 1 to n) and σ_{i2} (i = 1 to n) are the parameters that, respectively, define the center and the spread of the Gaussian function and cs_{ij} is the class score of x_{ij}.

In the context of a hybrid neuro-fuzzy model, each unique combination of spatially-coincident predictor patterns is considered a vector of predictor features (or a feature vector). Because each predictor map is represented by one, and only one, pattern in a feature vector, the number of components (or dimensions) of the feature vector is equal to the number of predictor maps. The favorability of a feature vector with respect to target mineral deposit-type is estimated as follows. Consider, for simplicity, a two-dimensional feature vector $T = [x_{1j}, x_{2j}]$, where x_{1j} and x_{2j} are the patterns representing, respectively, the predictor maps X_1 and X_2 in the feature vector. The membership values of x_{1j} in the fuzzy sets $\tilde{A}_i^1$ and $\tilde{A}_i^2$ are estimated using the fuzzy membership functions $\mu_{\tilde{A}_i^1}$ (3.14) and $\mu_{\tilde{A}_i^2}$ (3.16), respectively.

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 2^n 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 2^n 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:

$$\begin{aligned} s_1 &= \mu_{\tilde{A}_1^1}\left(x_{1j}\right) \times \mu_{\tilde{A}_2^1}\left(x_{2j}\right) \\ s_2 &= \mu_{\tilde{A}_1^1}\left(x_{1j}\right) \times \mu_{\tilde{A}_2^2}\left(x_{2j}\right) \\ s_3 &= \mu_{\tilde{A}_1^2}\left(x_{1j}\right) \times \mu_{\tilde{A}_2^1}\left(x_{2j}\right) \\ s_4 &= \mu_{\tilde{A}_2^1}\left(x_{1j}\right) \times \mu_{\tilde{A}_2^2}\left(x_{2j}\right) \end{aligned} \tag{3.17}$$

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:

$$O_k = s_k.F_k \tag{3.18}$$

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:

$$Overall\ output = \frac{\sum_{k=1}^{4} O_k}{\sum_{k=1}^{4} s_k} \tag{3.19}$$

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.

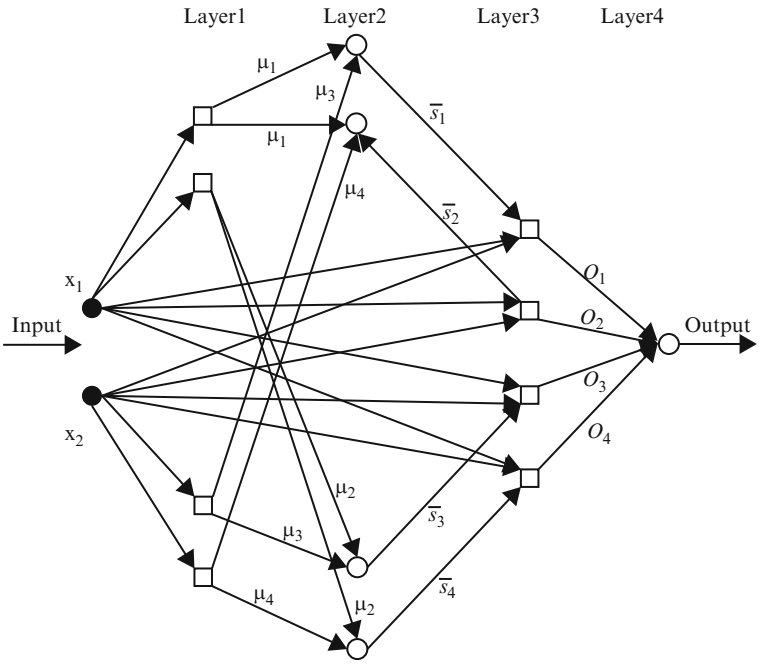


Fig. 3.20 Simplified ANFIS for mineral potential mapping. Square and circular nodes contain, respectively, adaptive and fixed functions

3.4.3 Architecture of Adaptive Neuro-Fuzzy Inference System

The fundamental construction modeling of an ANFIS embodies a Takagi-Sugeno sort fuzzy deduction framework in a five-layer food forward system. Figure 3.20 demonstrates the rearranged four-layer ANFIS structural engineering utilized within a crossover neuro fuzzy model for mineral potential mapping. The essential usefulness of each one layer is outlined in the accompanying sections.

3.4.3.1 Layer 1

There are 2^n (=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 (3.14), 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 defined in (3.16), 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.

3.4.3.2 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:

$$s_k = \mu_{\tilde{A}_1^q}(x_{1j}) \;\; x \;\; \mu_{\tilde{A}_2^q}(x_{2j}) \;\; x.....x \;\; \mu_{\tilde{A}_n^q}(x_{nj}) \tag{3.20}$$

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:

$$\overline{s}_k = \frac{s_k}{\sum_{k=1}^{2^n} s_k} \tag{3.21}$$

3.4.3.3 Layer 3

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

$$O_k = s_k.F_k = s_k\left(P_{k0} + P_{k1}x_{1j} + P_{k2}x_{2j} + ... + P_{kn}x_{nj}\right) \tag{3.22}$$

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, $F_k = P_{k0}$.

3.4.3.4 Layer 4

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

$$Overall\ Output = \sum_{k=1}^{2^n} O_k \tag{3.23}$$

3.4.4 Hybrid Learning Algorithm

Assuming that an ANFIS has two input variables, x_1 and x_2, the overall output is (based on 3.22 and 3.23):

$$\begin{aligned} O_1 + O_2 &= \overline{s}_1(P_{10} + P_{11}x_1 + P_{12}x_2) + \overline{s}_2(P_{20} + P_{21}x_1 + P_{22}x_2) \\ &= \overline{s}_1 P_{10} + \overline{s}_1 x_1 P_{11} + \overline{s}_1 x_2 P_{12} + \overline{s}_2 P_{20} + \overline{s}_2 x_2 P_{21} + \overline{s}_2 x_2 P_{22} \end{aligned} \tag{3.24}$$

which is linear in the consequent parameters P_{ki}, if the premise parameters and, therefore, the firing strengths s_k of the fuzzy if-then rules are fixed. ANFIS uses a hybrid learning procedure for estimation of the premise and consequent parameters. The hybrid learning procedure estimates the consequent parameters (keeping the premise parameters fixed) in a forward pass and the premise parameters (keeping the consequent parameters fixed) in a backward pass. In the forward pass, the information propagates forward until Layer 3 where the consequent parameters are estimated by the least square estimator method. In the backward pass, the error signals propagate backwards and the premise parameters are updated by a gradient descent method. The following description of the hybrid learning procedure is drawn from Jang and Sun.

3.4.4.1 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), (3.24) can be expressed as:

$$\mathrm{B} = AX \tag{3.25}$$

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 $\|AX\text{-}B\|^2$. It can be computed as below:

$$X* = \left(A^T A\right)^{-1} A^T B \tag{3.26}$$

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, moreover, 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 qth row vector of A and b_q^T is the qth element of B, then X can be calculated iteratively as follows:

$$X_{q+1} = X_q + \sum\nolimits_{q+1} a_{q+1}\left(b_{q+1}^T - a_{q+1}^T X_q\right) \tag{3.27}$$

$$\sum\nolimits_{q+1} = \sum_q - \frac{\sum_q a_{q+1} a_{q+1}^T \sum_q}{1 + a_{q+1}^T \sum_q a_{q+1}} \left(for\ q = 0, 1, \ldots\ldots\ldots\ldots.., Q-1\right) \tag{3.28}$$

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.

3.4.4.2 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 (3.25), 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:

$$E_q = \left(b_q - o_q\right)^2 \tag{3.29}$$

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

$$\varepsilon_{l,i} = \frac{\partial + E_q}{\partial o_{l,i}} \tag{3.30}$$

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:

$$\varepsilon_4 = \frac{\partial + E_p}{\partial o_{(4,1)}} = \frac{\partial E_p}{\partial o_{(4,1)}} = -2\left(b_q - o_q\right) \tag{3.31}$$

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

$$\varepsilon_{l,i} = \frac{\partial + E}{\partial o_{(l,i)}} = \sum_{m=1}^{N_{(l+1)}} \frac{\partial + E}{\partial o_{(l+1,m)}} \frac{\partial f_{(l+1,m)}}{\partial o_{(l,i)}} = \sum_{m=1}^{N_{(l+1)}} \varepsilon_{(l+1,m)} \frac{\partial f_{(l+1,m)}}{\partial o_{(l,i)}} \tag{3.32}$$

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 $\varepsilon(l+1,m)$ 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 (3.31) once to get error signals at the Layer 4 and then applying (3.32) 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:

$$\frac{\partial^+ E_p}{\partial c_i} = \frac{\partial^+ E_p}{\partial o_{(1,i)}} \frac{\partial^+ \mu_{1,i}}{\partial c_i} = \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial c_i} \tag{3.33}$$

and

$$\frac{\partial^+ E_p}{\partial \sigma_i} = \frac{\partial^+ E_p}{\partial o_{(1,i)}} \frac{\partial^+ \mu_{1,i}}{\partial \sigma_i} = \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial \sigma_i} \tag{3.34}$$

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:

$$\frac{\partial^+ E}{\partial c_i} = \sum_{q=1}^{Q} \frac{\partial^+ E_q}{\partial c_i} = \sum_{q=1}^{Q} \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial c_i} \tag{3.35}$$

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

$$\Delta c_i = -\eta \frac{\partial^+ E}{\partial c_i} \tag{3.36}$$

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

$$\eta = \frac{\kappa}{\sqrt{\sum_{i=1}^{N(1)} \left(\frac{\partial E}{\partial c}\right)^2}} \tag{3.37}$$

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:

$$\frac{\partial^+ E}{\partial \sigma_i} = \sum_{q=1}^{Q} \frac{\partial^+ E_q}{\partial \sigma_i} = \sum_{q=1}^{Q} \varepsilon_{(1,i)} \frac{\partial^+ \mu_{(1,i)}}{\partial \sigma_i} \tag{3.38}$$

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

$$\Delta \sigma_i = -\eta \frac{\partial^+ E}{\partial \sigma_i} \tag{3.39}$$

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

$$\eta = \frac{\kappa}{\sqrt{\sum_{i=1}^{N(1)} \left(\frac{\partial E}{\partial \sigma}\right)^2}} \tag{3.40}$$

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.

The above learning procedures were implemented using the software and procedure described by Jang and Gulley.

3.4.5 *Neuro-Fuzzy Network Model and Calculation Algorithm*

Fuzzy logic methodology is utilized to figure out the greatest force purpose of PV cells. Fitting decision of the parameters and thorough acceptance of decision are

involved to fulfill a faultless representation. In the event of critical levels of missed perceptions, the choice of the sort and estimations of the parameters of participation is not simple. In these circumstances, determination is focused around the qualities of information/yield information. By using neuro-fuzzy methods it is conceivable to learn data about the information set so that the parameters of the enrollment capacities are processed in such a path, to the point that the related fuzzy inference framework maps the given info/yield information legitimately. ANFIS frameworks join together the coherent impact force of fuzzy frameworks and the numerical force of neural systems. The connected ANFIS display for this situation is a first request Takagi-Sugeno model. The first stage is a direct synthesis of data variables in addition to a steady term and the last yield is the weighted normal of each one guideline's yield.

ANFIS utilizes a half breed learning calculation to distinguish subsequent parameters of Sugeno sort fuzzy induction frameworks. It applies a synthesis of the minimum squares technique and Back-propagation system for preparing fuzzy surmising framework enrollment capacities parameters to imitate a given set.

This method of Neuro-fuzzy system owes much from the feed forward neural network with supervised learning capability. Figure 3.21 shows the fuzzy inference system while Fig. 3.22 shows the equivalent ANFIS system.

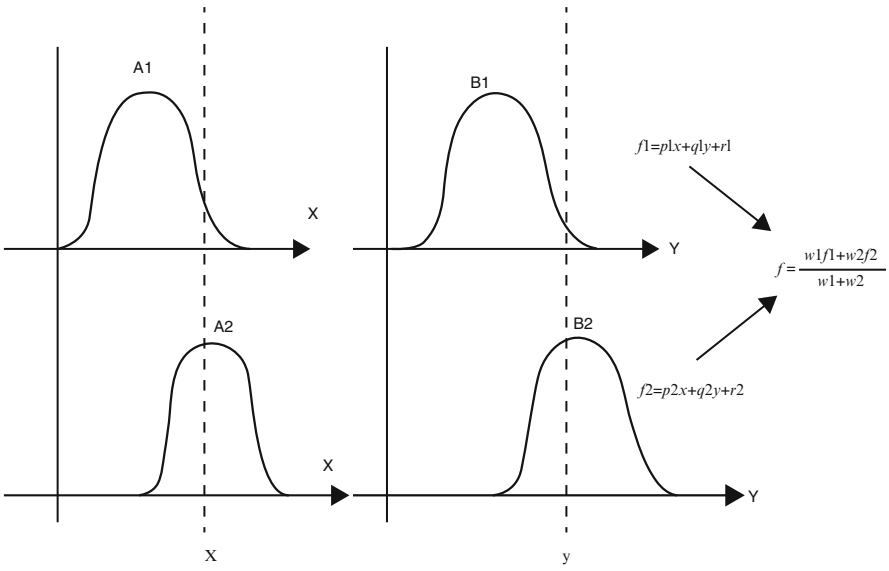


Fig. 3.21 Fuzzy inference system

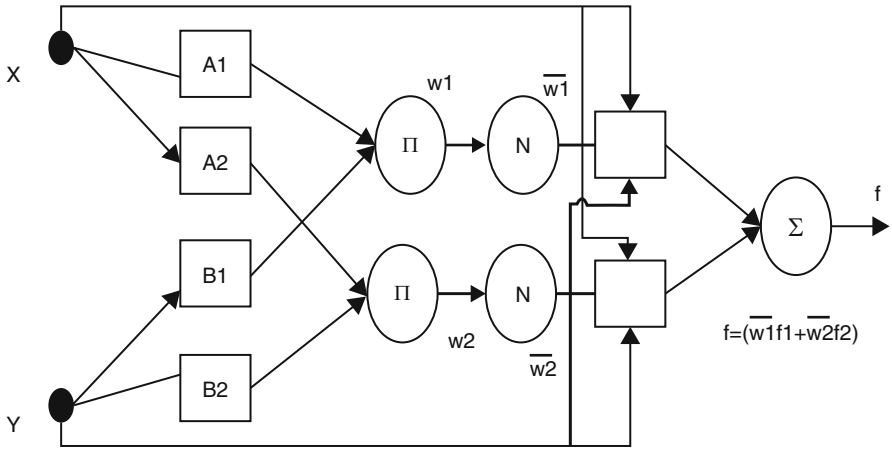


Fig. 3.22 ANFIS equivalent system

3.4.6 ANFIS Network Specifications

- Inputs: Insolation (G), Temperature (T).
- Output: Maximum power point Voltage.
- Number of Membership functions:

  Three (3) for Insolation
  Three (3) for Temperature

- Number of Rules: 9 (Sugeno FIS).

3.4.7 Algorithm for Neuro-Fuzzy Based MPPT

Step 1: Fuzzification Step (Input Step) in the first layer:

The ANFIS architecture is shown in Fig. 3.23. This layer is a fundamental fuzzification layer where the crisp inputs are apportioned to relative fuzzy qualities. The membership functions are bell shaped. The output of the first layer ith membership function is computed as:

$$O_i^1 = \mu A_i(G) = e^{\left\{-\left(\frac{x-c_i}{a_i}\right)^2\right\}^{b_i}} \tag{3.41}$$

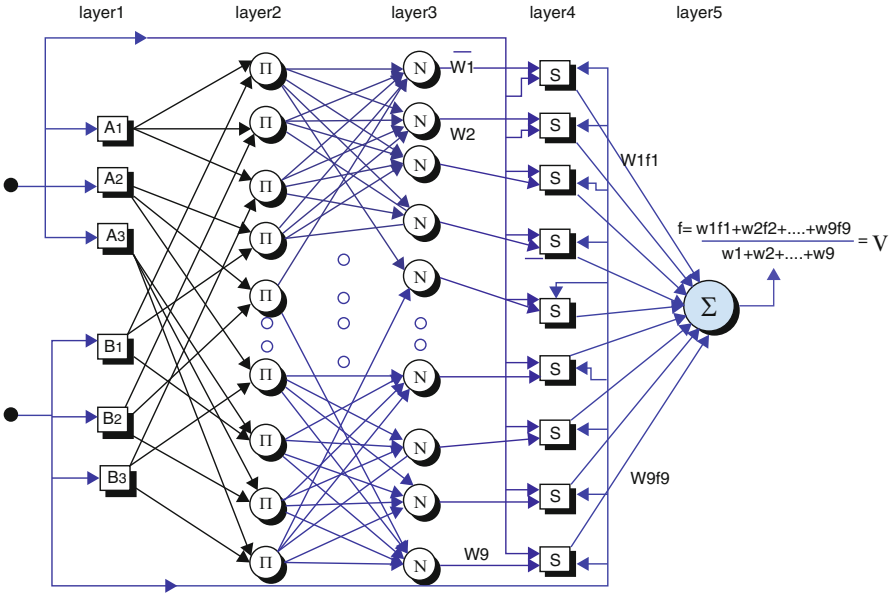


Fig. 3.23 ANFIS architecture for the system implemented

$$O_i^1 = \mu B_i(T) = e^{\left\{-\left(\frac{y-c_i}{a_i}\right)^2\right\}^{b_i}} \tag{3.42}$$

where O_i^1 is the output of the first layer, G is the insolation and T is the temperature (crisp values) for this application. A_i and B_i are linguistic variables characterized by appropriate membership functions, $\{a_i, b_i, c_i\}$ are the premise parameters described from the characteristics of bell shaped function. These premise parameters are fixed in the forward pass and are updated in the adaptation process by gradient descent method in the backward pass.

Step 2: Calculation of rules (or) generalization of firing strengths in the second layer:

The degree to which a piece of fuzzy rule is fulfilled and it shapes the respective output function is known as the firing strength. The intelligent administrator is the logical AND operator which is connected to acquire the output for every defined rule. The product inference principle is utilized at fuzzification level to yield values at this level. The yields at this hub are named as a consequence of increase of inputs from the layer one hubs.

$$O_i^2 = W_i = \mu A_i(G) \times \mu B_i(T) \tag{3.43}$$

where i = 1,2,. . .9

The value of node output represents the strength of the rule.

Step 3: Calculation of the ratio of firing strengths in the third layer.

The nodes in this layer are represented by circular nodes labeled N. The main objective of this step is to calculate the ratio of each ith rule's firing strength to the sum of all rule's firing strength, which is also called as normalized firing strength with the output O_i^3 at this step.

$$O_i^3 = \overline{W_i} = \left\{ \frac{W_i}{\sum_i W_i} \right\} \text{ where } i = 1, 2, \ldots., 9 \tag{3.44}$$

Step 4: Contribution of each rule: The fourth layer

Calculation of the contribution of each ith rule towards the total output O_i^4 is formed at this step.

$$O_i^4 = \overline{W_i}(p_i T + q_i G + r_i) \text{ where } i = 1, 2, \ldots 9 \tag{3.45}$$

where W_i is the output of layer 3 and $\{p_i,q_i,r_i\}$ are consequent parameters which are updated on forward pass by least square estimates and are fixed in backward pass. This layer establishes the relation between weight based premise parameters and linear summation based consequent part of the adapted ANFIS structure.

Step 5: Summation Step (Defuzzification Step) in the fifth layer:

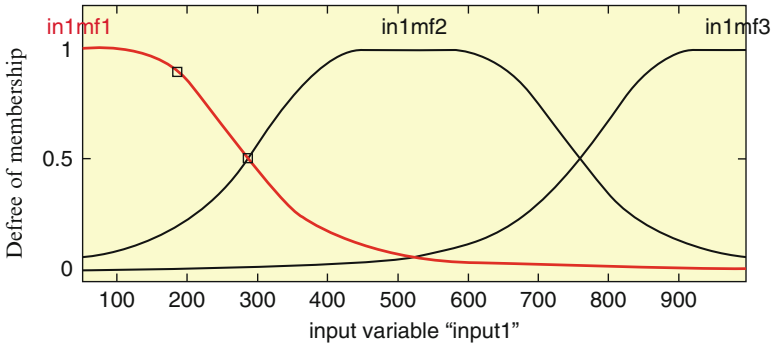
Each rule's fuzzy results are transformed into a crisp output O_i^5 at this step. The single output node in this layer sums up all the outputs of the previous layer as the output of ANFIS in this case is the maximum power point voltage of the PV cells.

$$K_p = \sum_i \overline{W_i} f_i = \sum_i \left\{ \frac{W_i f_i}{W_i} \right\} \text{ where } i = 1, 2, \ldots 9 \tag{3.46}$$

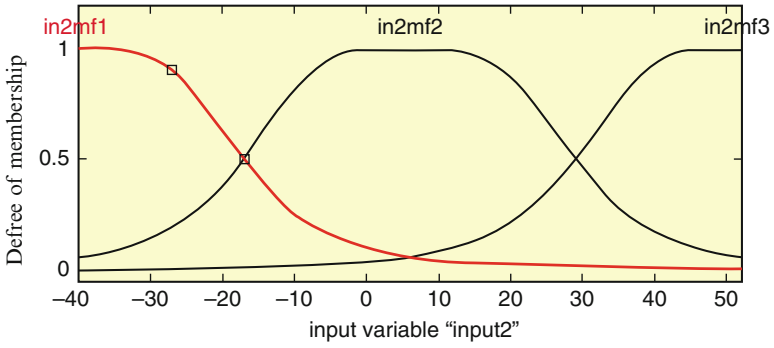
3.4.8 *Results for Neuro-Fuzzy Based MPPT*

3.4.8.1 Network Training

A set of 25 training patterns was presented to the network, while another set of nine points were randomly chosen as the testing data points which were not utilized for training . These training patterns were uniformly distributed to cover all the typical input space in order to get good performance where temperature ranges from −40 °C to 50 °C and solar irradiation ranged from 0 to 1,000 W/m^2. The implementation of Neuro-Fuzzy based MPPT for maximum power point tracking using MATLAB is done. Initial Memberships of the temperature T and Irradiation G are shown in Fig. 3.24 while Fig. 3.25 presents the final membership functions after ANFIS training. Figure 3.27 presents the ANFIS architecture obtained.



Original membership function for irradiation



Original membership function of temperature

Fig. 3.24 Original memberships of T and G before training

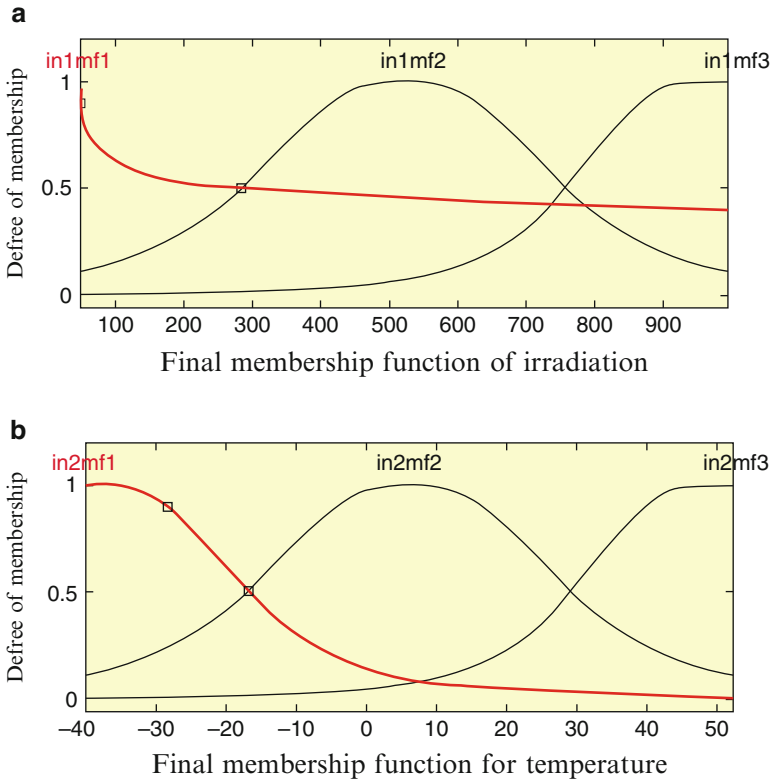


Fig. 3.25 Final memberships of T and G after training

3.4.8.2 Training Data for Neuro-Fuzzy Controller

1)	900	10	16.16
2)	830	35	8.75
3)	910	12	16.39
4)	850	50	20.36
5)	920	13	16.57
6)	600	52	20
7)	770	47	19.9
8)	700	42	19.9
9)	950	23.75	17.7
10)	980	24.5	17.89
11)	985	24.62	17.8
12)	995	24.87	18
13)	180	30	13.51

(continued)

14)	220	40	16.77
15)	825	20.6	17.1
16)	925	18	17.4
17)	940	20	17.68
18)	930	23	17.7
19)	650	16	16
20)	955	23.87	17.7
21)	960	24	17.8
22)	970	24.25	17.9
23)	975	24.37	17.8
24)	100	−40	7.8995
25)	100	0	11.74

3.4.8.3 Root Mean Square Error

Root mean square error evaluates the magnitude of deviations between the measured values and actual predictions.

$$RMSE = \left[\sum_{i=1}^{n} \frac{(m_i - z_i)^2}{n}\right]^{0.5} \tag{3.47}$$

where m_i – measurements, z_i – model outputs and N – number of observations.

3.4.8.4 Validation of the Model

The accuracy of the model is tested using a set of data different from training and the ANFIS controller gives accurate results and is able to closely mimic the actual values. The original and final membership functions of T and G are shown in Figs. 3.24 and 3.25 respectively. The RMSE of the training data with respect to training epoch is shown in Fig. 3.26. The ANFIS model with insolation and temperature as inputs and voltage as output is shown in Fig. 3.27. The accuracy of the obtained model is validated using testing data. Figure 3.28 show the variations of *VPP* (*V*) for various insolation and temperature conditions. The obtained model designed and trained has shown a very fast convergence with little number of memberships.

Table 3.4 summarizes the Neuro-Fuzzy controller output values for various insolation and temperature conditions.

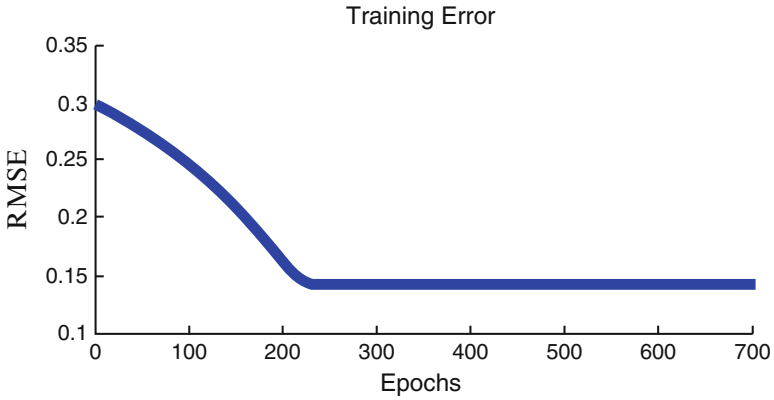


Fig. 3.26 RMSE of the training data with respect to training epoch

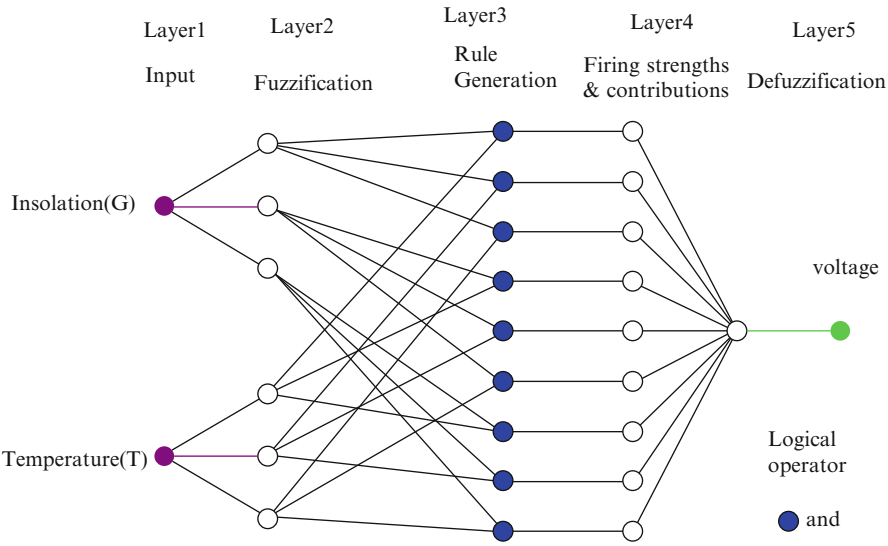


Fig. 3.27 ANFIS architecture

3.5 Fuzzy Based Solar Tracking

A solar tracking system is configured to optimize the functioning of solar energy receiving system. If solar receivers are installed appropriately with a tracking system, then additional solar energy can be accumulated throughout the day. In this section, a solar tracking system is simulated by applying MATLAB/SIMULINK and a fuzzy logic control is developed for the control of this system. The combination of the generated controller with the solar tracking system is realized with the fuzzy logic controller in the MATLAB/SIMULINK environment.

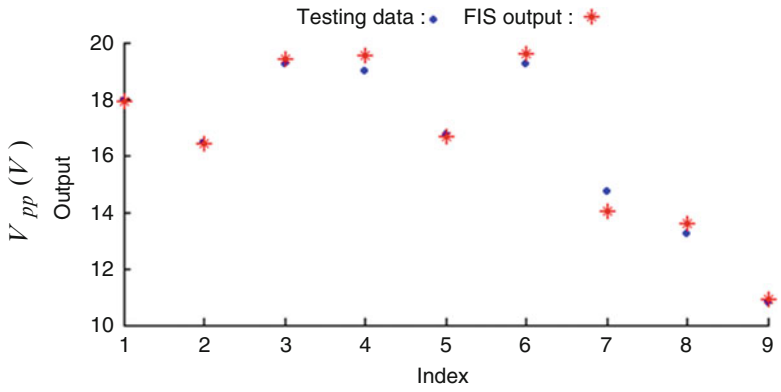


Fig. 3.28 Vpp plotted for various values of insolation and temperature

Table 3.4 Output of neuro-fuzzy controller

Insolation (W/m^2)	Temperature (°C)	Maximum power tracked by neuro-fuzzy controller
1,000	25	59.91
930	12	51.45
700	40	45.45
500	46	32.091
950	15	53.7
820	40	53.46
200	20	9.74
100	−10	3.48

Concurrently, PI control is implemented to the system and the solutions found using the PI control are compared with those obtained using fuzzy logic.

In order to obtain optimal efficiency, solar panels ought to be orthogonal according to the sunlight where the illumination is heaviest. Nevertheless, as the focus of sunlight shifts naturally throughout the day, and from season to season, a superior solar tracking system that performs better can maximize utilization of the panels. This scheme is comprised of two permanent magnet DC (PMDC) motors, two directional light detecting circuits and two amplifiers to drive the motors. The two drive motors are uncoupled, i.e. the rotation angle of one motor does not influence that of the other motor, therefore reducing control problems. Such kind of implementation minimizes the power consumption of the system during its operation thus increasing the efficiency and the total quantity of electricity yielded. The light detecting circuit of design comprises of four light sensitive devices, namely LDR, photodiodes or phototransistors, mounted on the solar panel and placed in an enclosure. The quadruplet light detectors are blocked out form one another by opaque surfaces. The sensing elements are designed in a sense that LDR1 and LDR2 are accustomed to track the sun horizontally, meantime LDR3 and LDR4 allow tracking the sun vertically. As one experiences more light than the

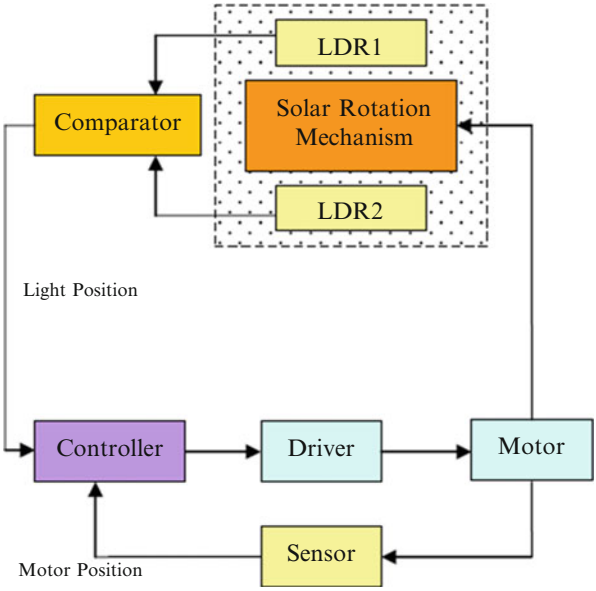


Fig. 3.29 Solar tracking control architecture for one direction (vertical or horizontal)

other, the panel is not coordinated properly and therefore an error voltage occurs. The error voltage is used as a command to an amplifier circuit to drive the motor and align the panel to be orthogonal to the light beam. In Fig. 3.29, the architecture of unidirectional solar tracking control is shown.

Photo Detecting Circuit and Amplifier

The photo detecting circuit and amplifier supply an electrical thrust to the motor which is proportional to the rotational misalignment of the panels to the light source. The photo detecting circuit and amplifier can be considered as a single variable K, where K is proportionality constant with units of volts per radian. The value of K constitutes the gain of the "photo detector" circuit and the "gain adjust" circuit.

Permanent Magnet DC Motor

Two permanent magnet DC (PMDC motors) are used to actuate the solar panel. The design of a PMDC motor for MATLAB/SIMULINK GUI Environment is discussed in this section. The PMDC motors are applicable in a wide range of low-power applications in which the field winding is a permanent magnet. Some of the benefits of PMDC are: they don't require external excitation, they require less space and they are cheaper. The equivalent circuit of PMDC motor is shown in Fig. 3.30 and the equations governing the model are given by (3.48, 3.49, 3.50, and 3.51). The MATLAB/SIMULINK model for PMDC motor is shown in Fig. 3.31.

The motor winding is a resistor connected in series with an inductor. When the motor has a nonzero speed, there is back emf voltage generated. The back emf voltage is proportional to the speed. According to Kirchoff's Voltage Law:

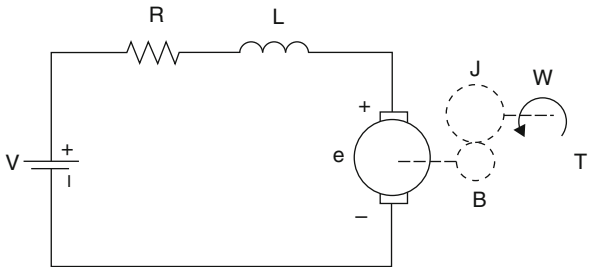


Fig. 3.30 Equivalent circuit of a PMDC motor

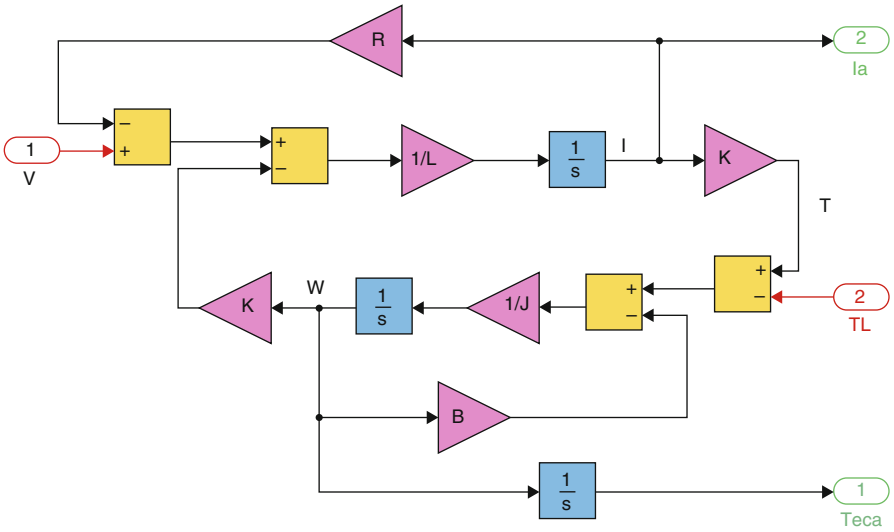


Fig. 3.31 MATLAB/SIMULINK model for PMDC motor

$$V(t) = R \cdot i(t) + L \cdot \frac{di(t)}{dt} + E(t) \tag{3.48}$$

where V is the voltage applied to the motor and

$$E(t) = K \cdot \omega(t) \tag{3.49}$$

The torque generated by the motor is proportional to the current. According to Newton's second law (rotational version):

$$T(t) - T_f(t) - T_L(t) = J \cdot \frac{d\omega(t)}{dt} \tag{3.50}$$

where T_L is the load torque and

$$T(t) = K \cdot i(t) \; and \; T_f(t) = B \cdot \omega(t) \tag{3.51}$$

Equations 3.48 and 3.50 describe the dynamic behaviour of the motor. Equations 3.48, 3.49, 3.50, and 3.51 can be re-arranged as in (3.52) and (3.53) for construction of the block diagram.

$$\frac{di(t)}{dt} = \frac{1}{L} \cdot (V(t) - R \cdot i(t) - K \cdot \omega(t)) \tag{3.52}$$

$$\frac{d\omega(t)}{dt} = \frac{1}{J} \cdot (K \cdot i(t) - T_L(t) - B \cdot \omega(t)) \tag{3.53}$$

Differential equations (3.52) and (3.53) for armature current and angular speed can be arranged as state space equations in (3.54).

$$\begin{bmatrix} \dot{i}(t) \\ \dot{\omega}(t) \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} & -\frac{K}{L} \\ \frac{K}{J} & -\frac{B}{J} \end{bmatrix} \cdot \begin{bmatrix} i(t) \\ \omega(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L} & 0 \\ 0 & -\frac{1}{J} \end{bmatrix} \cdot \begin{bmatrix} V(t) \\ T_L(t) \end{bmatrix} \tag{3.54}$$

3.5.1 *Design Process of the Fuzzy Controller*

For most investigations performed so far, a fuzzy controller system is frequently thought of which emulates an individual pro. However, the feeling of the individual driver can be put in the type of a set of fuzzed lingual policies. These kinds of principles would green goods an approximate decision the identical way a human could complete. The structural plan of the fuzzy logic technique is shown in Fig. 3.32 with the design mechanism.

The fuzzified control comprises the subsequent four aspects. These include fuzzification, rule base, inference system along with defuzzification. Some sort of fuzzification program changes the particular highly detailed crisp inputs into the fuzzified values which have been framed in the rule base so that the user can execute these associated principles making sure that an output may be generated. Any principle base is made up of rule base including details related to the device. The inference process emulates throughout interpretation as well as making use of know-how about computers and how to manage the levels. A defuzzification interface converts the findings with the inference procedure into curt advices with the practice. The particular ideals regarding problem (e(k)) as well as alter (Δe(k)) transpiring through the surgery on the program form the fresh inputs of the method.

$$e(k) = r(k) - y(k) \tag{3.55}$$

$$\Delta e(k) = e(k) - e(k-1) \tag{3.56}$$

r(k), y(k) and k are expressed as the reference input, the actual output of the system and the sampling step respectively. The design of FLC in the MATLAB/

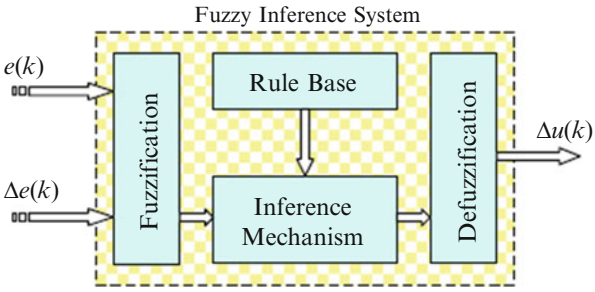


Fig. 3.32 The basic structure of fuzzy logic based controller

SIMULINK environment is discussed following the expressions. The basic elements of the FLC are the fuzzy membership functions which are available in the modeling of the controller. In several applications, different types of membership functions such as triangle, trapezoid, bell, Gaussian, sigmoid and sinusoid are used. The approach used to design FLC in this section uses triangular membership function given the following equation (10).

$$\mu(x) = \max\left[\min\left(\frac{x - x_1}{x_2 - x_1}, \frac{x_3 - x}{x_3 - x_2}\right), 0\right] \tag{3.57}$$

3.5.2 SIMULINK Model

Figure 3.33 shows the SIMULINK model of the triangular fuzzy membership function where x_1, x_2, and x_3 are the crisp values which in turn define the location and shape of the triangle. The input x is the crisp variable whose membership value on this triangle fuzzy subset is the output mu(x).

The crisp inputs e(k) and Δe(k) are converted to fuzzy membership values on the fuzzy subsets NB, NS, ZZ, PS and PB. The fuzzy inference mechanism has the structure as in verbal expression (3.58).

$$\textit{If}\ \text{e is A}\ \textit{and}\ \Delta\text{e is B}\ \textit{then}\ \Delta\text{u is}\ C. \tag{3.58}$$

In the general rule, parameters A, B and C in (3.58) denote any one of the fuzzy subsets NB, NS, ZZ, PS and PB. The input space in (3.58) is the part that represented by the expression (e is A ***and*** Δe is B). The connector "and" represents the Boolean operator "min". The weight coefficient required for each rule is determined by using the Boolean operator "min" (min(μA(x), μB(x))). Therefore, the membership values of each variable in the output space are obtained. The process of fuzzification with 9 rules with respect to the input is shown in Fig. 3.34.

These weight coefficients which are shown as mu1, mu2, . . ., mu9 are multiplied by the crisp values of each corresponding fuzzy subset in the output space Δu.

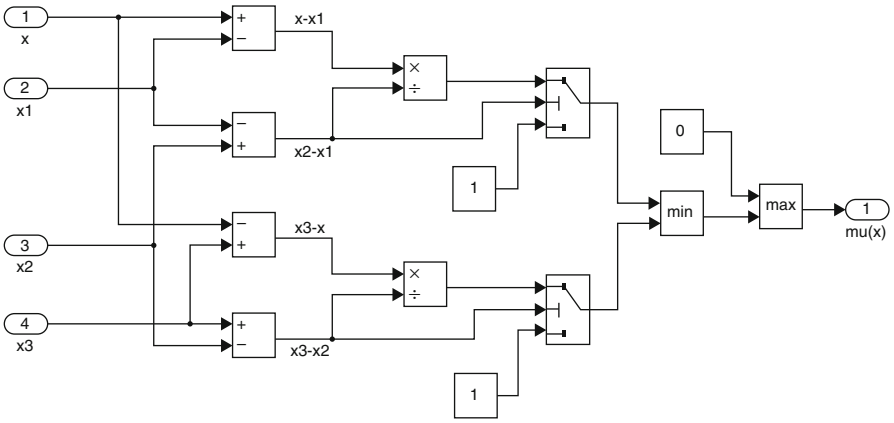


Fig. 3.33 SIMULINK model of the triangular fuzzy membership function

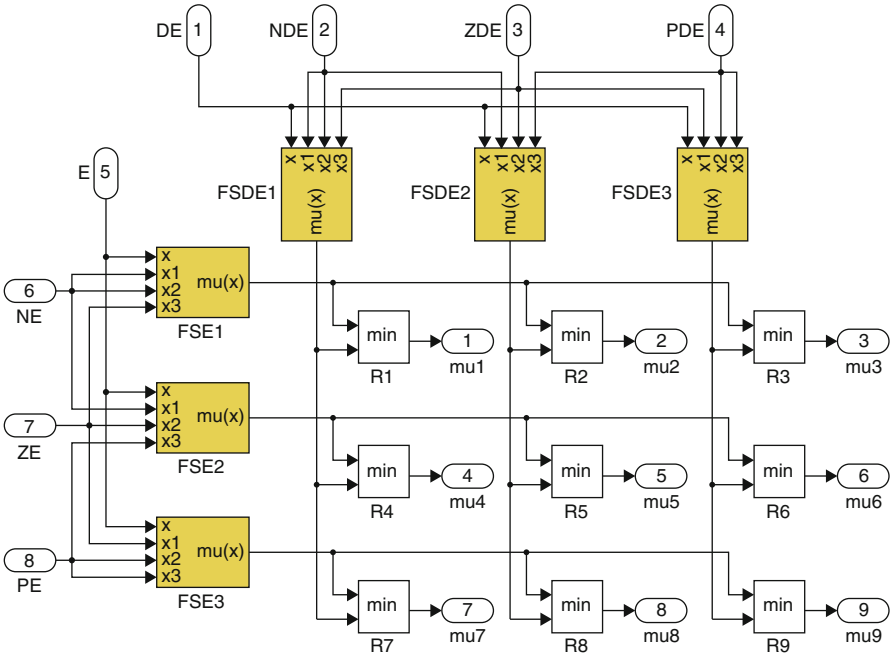


Fig. 3.34 The SIMULINK model of the verbal expression

These crisp values indicate the peak locations of the triangular fuzzy subsets. This multiplication process represents the products in the denominator of defuzzification method called the centre of area. Then the sum of these products is divided by the sum of membership values obtained as in Fig. 3.35.

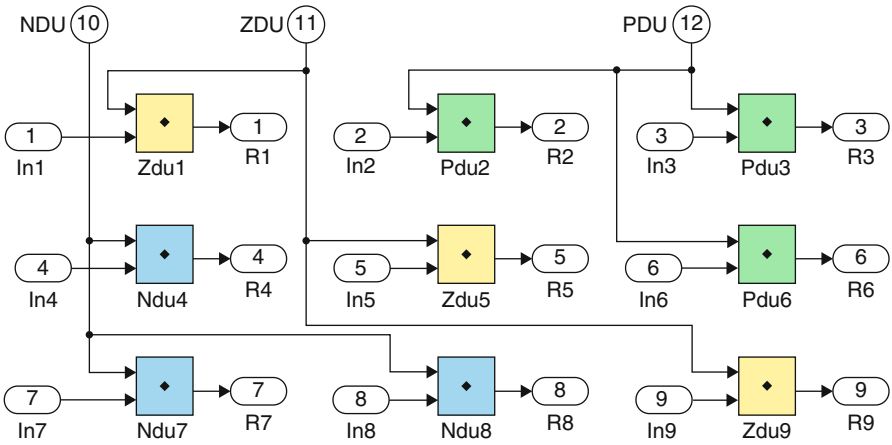


Fig. 3.35 The SIMULINK model of the fuzzy rules

Though different types of defuzzification methods are applied, the most commonly used in fuzzy logic control applications is the center of area method. The center of area method is given the following equation (3.59).

$$\Delta U(k) = \frac{\sum_{i=1}^{n} \mu_i(\Delta u_i) \cdot \Delta u_i}{\sum_{i=1}^{n} \mu_i(\Delta u_i)} \tag{3.59}$$

ΔU(k) is the final crisp output. The center of area method as shown in (3.59) is implemented in MATLAB/SIMULINK. A general overlooked view of the FLC is given in Fig. 3.36 where the processes from inputs e and Δe to output Δu are shown. The input data blocks to represent fuzzy membership functions for the error e, error change Δe and the controlled output change Δu are shown in Fig. 3.37. The user is able to edit and change the parameters of the membership functions on this stage without going into the detail of the FLC.

3.5.3 Simulation Results of Solar Tracking System

The model of solar tracking system is simulated using both FLC and PI controllers for comparison and validation purposes using MATLAB/SIMULINK models. The proportional gain (K_P) set to 1.55 and the integral gain (K_I) set to 0.02. Figure 3.38 shows the MATLAB/SIMULINK model of the developed system.

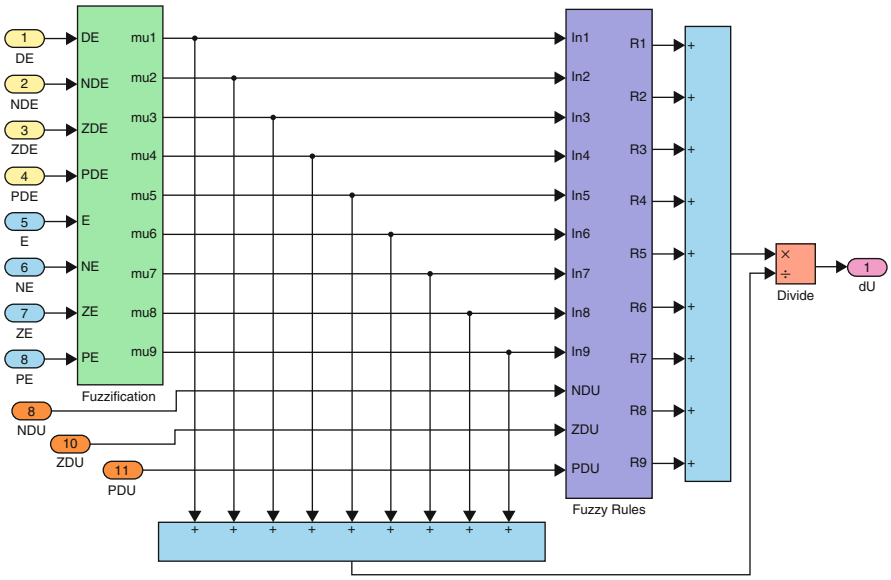


Fig. 3.36 Fuzzy reasoning representing the process from fuzzification to defuzzification

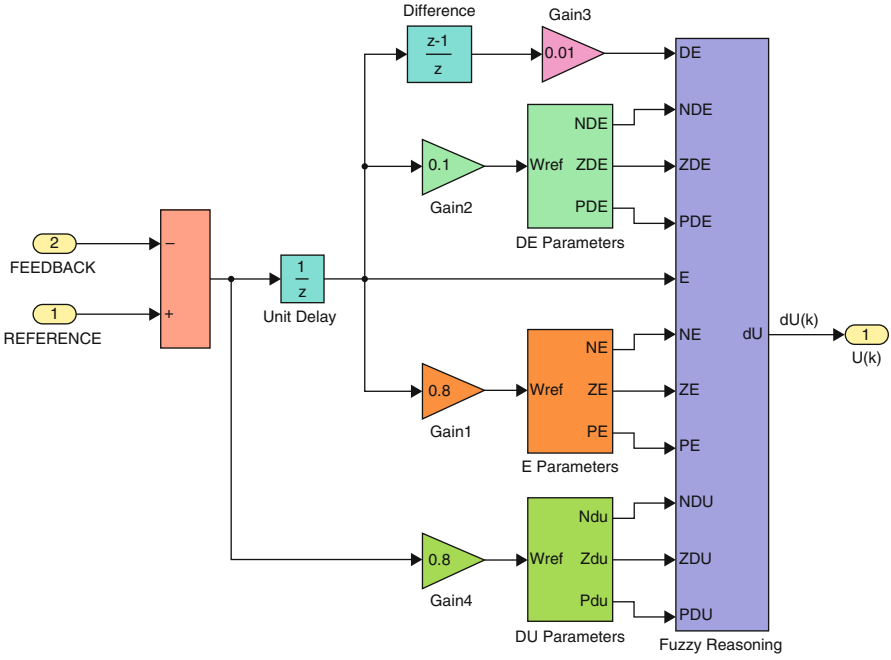


Fig. 3.37 The input and output units of the FLC

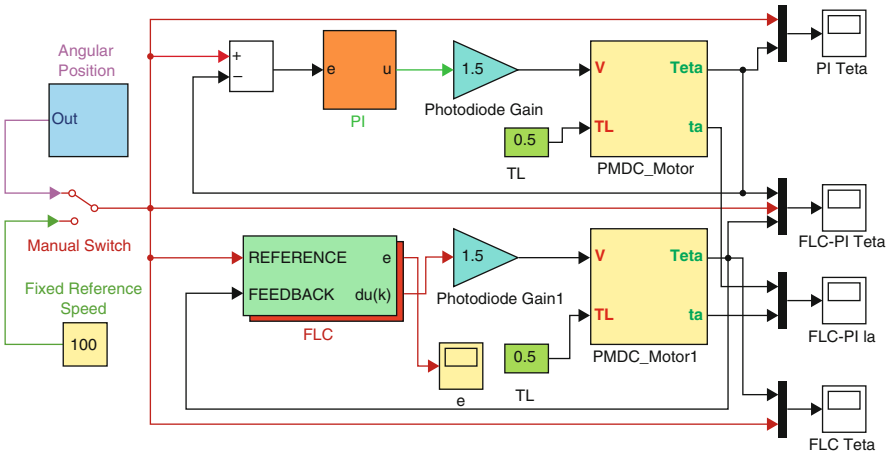


Fig. 3.38 MATLAB/SIMULINK model for the solar tracking system

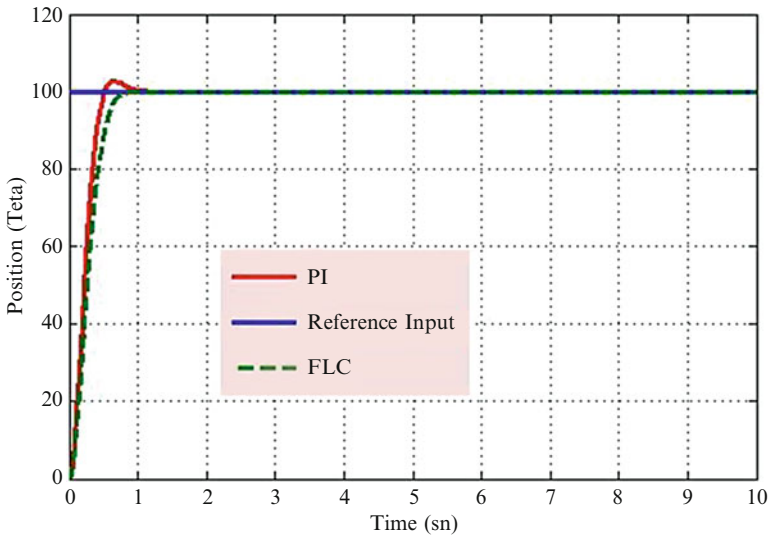


Fig. 3.39 The output of the system for FL and PI controllers

Figure 3.39 shows the system responses for FL and PI controllers. Comparatively, it is noted that the FLC gives faster response and less overshoot than PI controller.

Figure 3.40 shows the change of the error in fixed reference input for FL controller. It is observed that the error falls to zero quickly when compared with the reference input.

According to variable angular position applied to the system, the system responses from both FL and PI controllers are shown in Fig. 3.41.

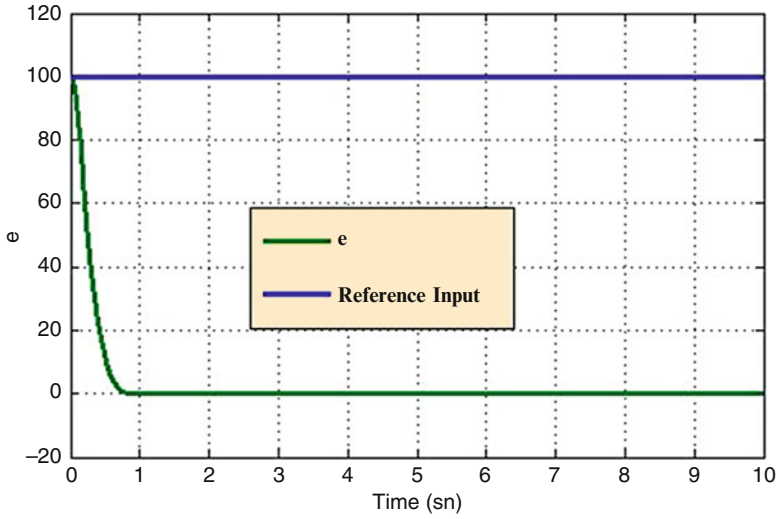


Fig. 3.40 The change of the error in fixed reference input for FL controller

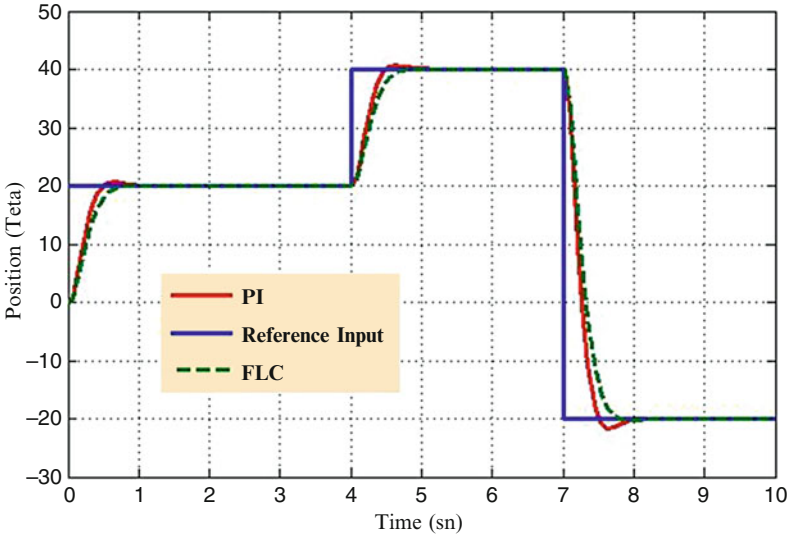


Fig. 3.41 The outputs of the system from both FL and PI controller for variable angular position

Although there are some differences during the transient period of position, both controllers follow variable reference input. The position response from FLC seems to be slower than that of obtained with PI when step changes occur in reference position input level. If a load of 0.5 N is applied to the motor shaft, the system responses from both FL and PI controllers for variable reference input are shown in Fig. 3.42.

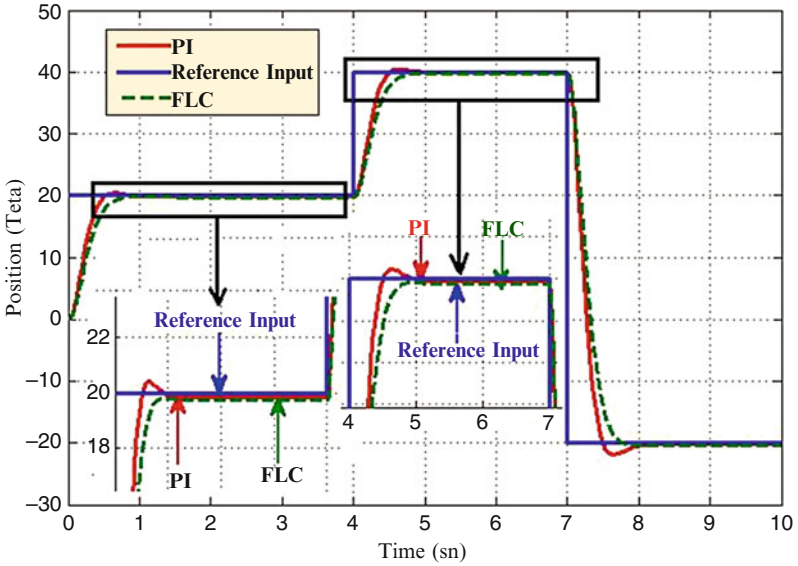


Fig. 3.42 The output of the system from both controllers for a load of 5 N

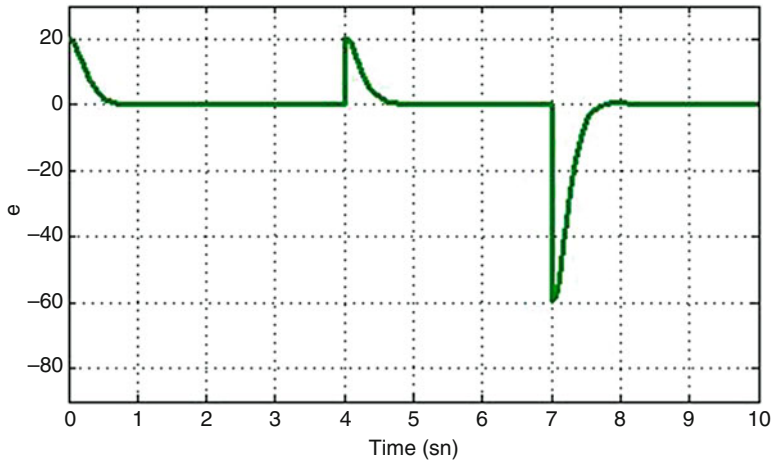


Fig. 3.43 The change of the error in variable reference input for FL controller

It is seen from Fig. 3.42 that when a load is applied to the motor shaft, FLC gives better results than PI controller. Because, in Fig. 3.42, PI controller is occurred overshoot more than FLC but PI controller has less steady-state error. As the load applied to the motor shaft increases, the error of the system also increases. The change of the error in variable reference input for FL controller is shown in Fig. 3.43.

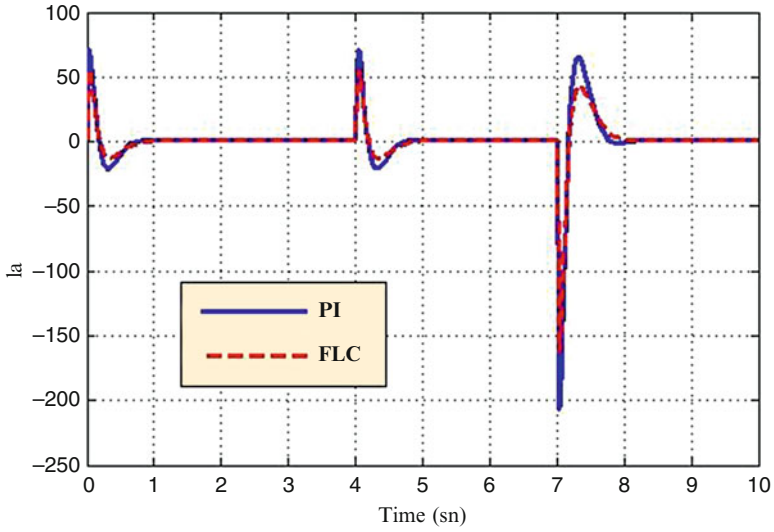


Fig. 3.44 The motor current from both controllers for a load of 5 N

However, each controller gives different current responses during the transients. As depicted in Fig. 3.44, the current has less ripple magnitudes during the transients with FLC than it has with the PI controller. In this section, a solar tracking system was designed for MATLAB/SIMULINK environment and a control method was developed for this system. The results from the FLC are compared with those obtained using PI controller. The results from simulations showed good and acceptable performances for the FLC. It was observed that both FLC and PI have almost the same settling time but the FLC show less overshoot. The most important advantage of this control has less ripple magnitudes during the transient. Practically obtained results show that using such controller increases speed of the time response.

3.6 MATLAB/SIMULINK Model of Two Axis Sun Tracker Using Fuzzy Logic

In this section, a two axis solar tracking system which uses two stepper motors as the drive source to rotate the solar panel is delineated. The basic structural model is shown in Fig. 3.45. The positioning of the sunlight is ascertained by applying a tracking sensor, the sensor interpretation is changed from analog to digital signal, and then it is passed to a fuzzy logic controller implemented on PIC microcontroller. The output of the FLC is linked to the driver of the stepper motor to rotate PV panel in one axis until it faces the sun.

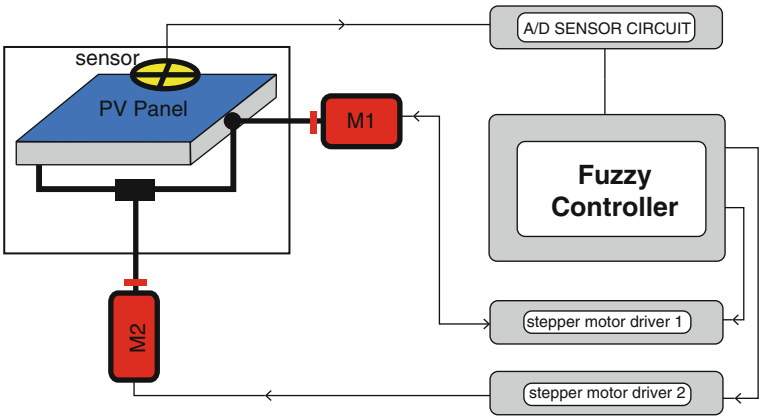


Fig. 3.45 Block diagram for the sun tracker system

3.6.1 Sensors

The solar tracking system discussed in this section uses two sensors namely the photo sensor and the position sensor.

3.6.1.1 Photo Sensor

Light dependent resistors (LDR) are applied to fabricate the sensor, since they are the most authentic sensor that can be exploited for light detection. LDR is fundamentally a resistor whose electric resistance changes based on the intensity of light, hence more intensity leads to less resistance. Several LDR sensors can be obtained in the market. The largest size is applied to fabricate the sensor since larger the area of the sensor leads to larger sensitivity or lesser the time taken for output to change according to the changes in the input.

3.6.1.2 Tracking Sensor Design

The tracking sensor is composed of four LDR sensors, which are located based on directions such as east, west, south, and north to detect the light source intensity. The LDR sensor casts a 45° angle based on the light source. In the LDR sensor positions, brackets are available to insulate the brightness from extra orientations to attain a wide-angle search and promptly decide the sun's positioning.

To sense the positioning of sun in one axis say east/west, two LDR sensors are mounted on the solar panel and placed in an enclosing. The response of this approach is similar to that of a human eye. The LDR sensors in east and west directions compare the intensity of received light. Based on the changes in the sun's

position, here the light source intensity level accepted by the sensors is dissimilar; the system gets signals from the sensors' output voltage in the two orientations. The system then ascertains the level of intensity received by the sensor based on the sensor yield voltage measure represented by voltage type A/D converter. Thus the system drives the stepper motor towards the orientation of this sensor. When the output measures of the two sensors are equal, the difference yielded at the output is zero and the motor's drive voltage is zero, which implies that the system has tracked the current position of the sun.

3.6.1.3 Position Sensor

Position sensors are accustomed to find out the location of the PV panel, thus preventing the panel from the shock while it arrives at the edges, and to get the PV panel to the beginning point at the night. This sensor employed a varying resistor (potentiometer) placed with the rotor of the motor and rotates with it, and the value of the resistor (R) changes with the rotary motion as shown in Fig. 3.46. While the position sensor achieves the measures at the PV at the edges, the controller blocked off the motor and exempt it from rotating in this direction. During night the LDRs sensors' measures are very large. In such a position the controller goes to night procedure to rotate the PV panel until the position sensor attains the starting point value.

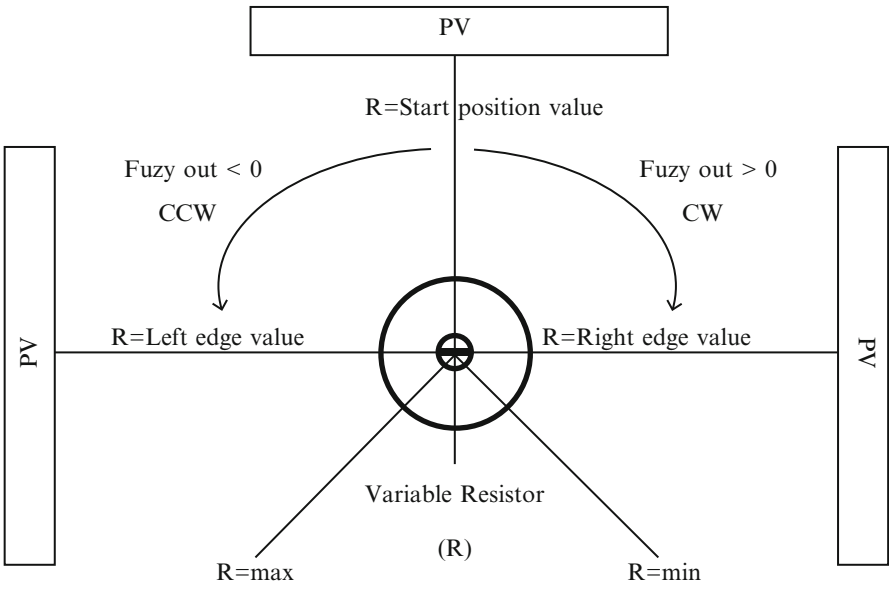


Fig. 3.46 Position sensor

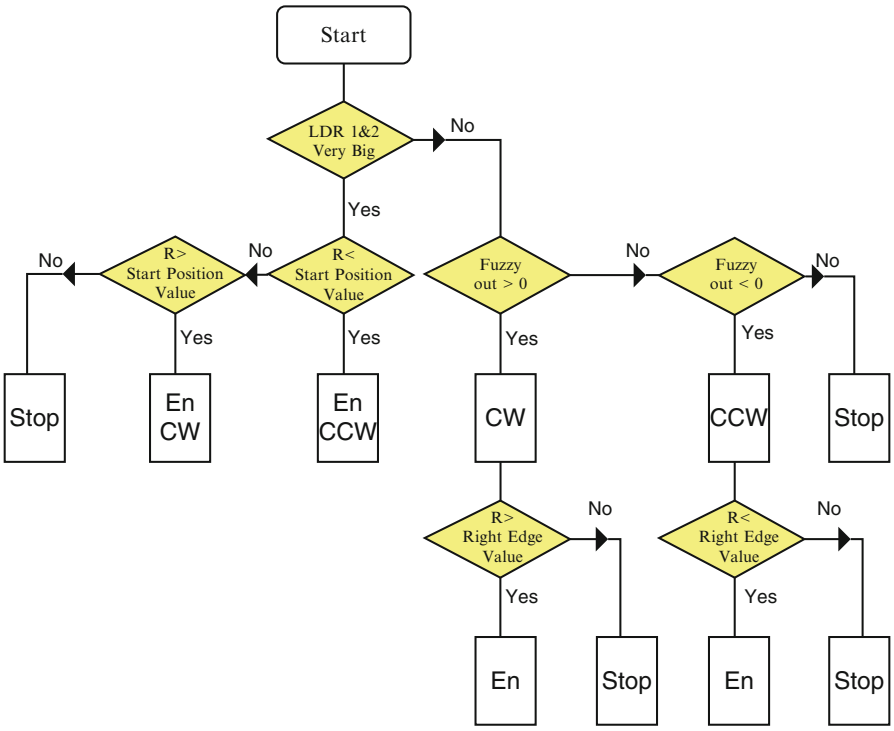


Fig. 3.47 Motor control signals algorithm

The algorithm for extracting the motor control signals based on the reading obtained from the sensors and the output of the controller is shown in Fig. 3.47. Here, R is the value of the position sensor, and En is the enable signal to rotate the motor.

3.6.2 Design of FLC for Sun Tracking System

Figure 3.48 shows the block diagram of the FLC for the sun tracker system.

3.6.2.1 FLC Design

The two inputs to the FLC are: error and the change in error. Only one output is used which feed the stepper motor driver. Though Mamdani and Sugeno approaches are applied for FLCs, in this section Mamdani based model is used to construct the fuzzy controller. The FLC contains three basic parts: Fuzzification, Base rule, and Defuzzification.

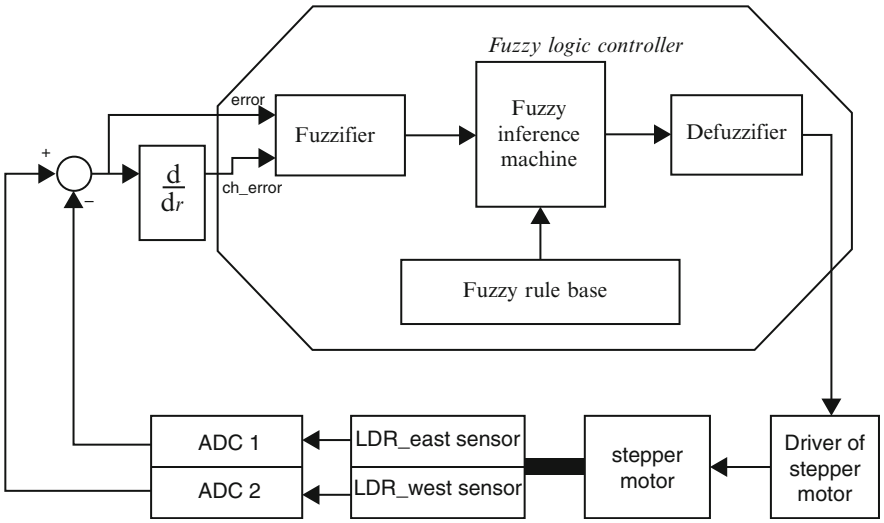


Fig. 3.48 FLC controller for the sun tracker system

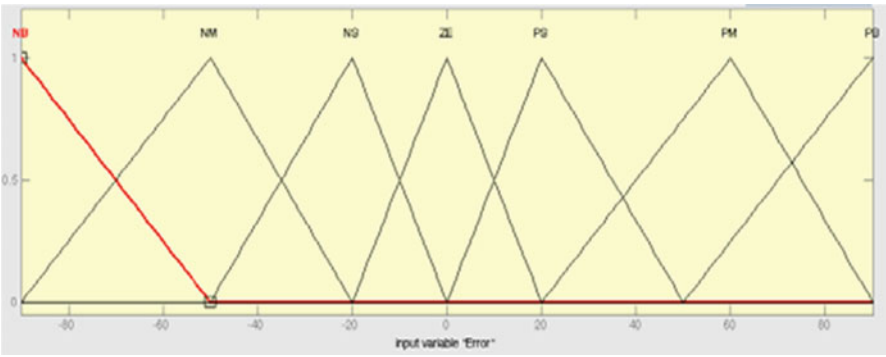


Fig. 3.49 Error fuzzy set of FLC

3.6.2.2 Fuzzification

Figure 3.49 illustrates the fuzzy set of the Error input which contains seven Triangular memberships.

Figure 3.50 illustrates the fuzzy set of the Change of Error input which contains seven Triangular memberships.

Figure 3.51 illustrates the fuzzy set of the output which contains seven Triangular memberships.

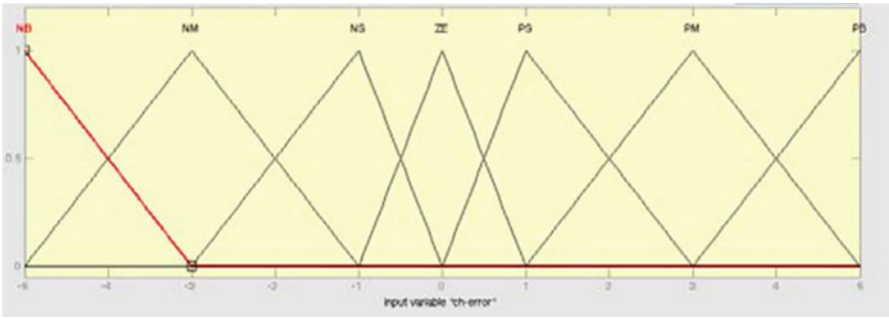


Fig. 3.50 Change in error fuzzy set of FLC

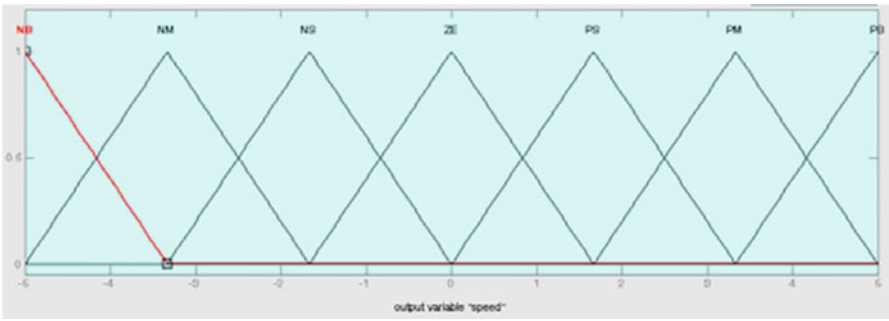


Fig. 3.51 Fuzzy set of FLC output entering to stepper motor driver

Table 3.5 Control rule base for fuzzy controller

	CE						
Er	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NB	NM	NS	ZE
NM	NB	NB	NM	NM	NS	ZE	PS
NS	NB	NM	NS	NS	ZE	PS	PM
ZE	NB	NM	NS	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PS	PM	PB
PM	NS	ZE	PS	PM	PM	PB	PB
PB	ZE	PS	PM	PB	PB	PB	PB

3.6.2.3 Control Rule Base

The knowledge base is defined by the rules for the desired relationship between the input and output variables in terms of the membership functions illustrated in Table 3.5. The control rules are evaluated by an inference mechanism, and represented as a set of:

IF Error is . . . and Change of Error is . . . THEN the output will . . ..

For example: Rule 1: IF Error is NS and Change of Error is ZE THEN the output is NS.

The linguistic variables used are:

NB: Negative Big.
NM: Negative Medium.
NS: Negative Small.
ZE: Zero.
PS: Positive Small.
PM: Positive Medium.
PB: Positive Big.

3.6.2.4 Defuzzification

The centre of gravity method is widely used in Mamdani approach which has been selected in this section to compute the output of the FLC, which is the motor speed as:

$$Speed = \frac{\sum_{i=1}^{n} S_i \cdot \mu(S_i)}{\sum_{i=1}^{n} \mu(S_i)} \tag{3.60}$$

3.6.3 SIMULINK Model and Results of FLC Based Sun Tracker System

Figure 3.52 illustrates the SIMULINK block diagram for the Fuzzy controller for sun tracker system.

The controller has been tested using SIMULINK motor module in MATLAB, by applying the step input and initial degree of the rotor is -10°. The output step response is shown in Fig. 3.53. The range from -10 to 0° takes five steps since each step in our motor is 1.8°, so $(10/1.8) = 5$ steps.

3.7 FLC for Solar Powered Energy

Solar powered electric schemes comprise of several components to be controlled individually. As the generated power is a function based on varying environmental factors, it demands additional care to design controllers that deal with such unpredictable issues and sustain effective load matching power. Here, a photovoltaic (PV) solar array model is designed using MATLAB/SIMULINK GUI environment and controlled using a fuzzy logic controller (FLC), which is also designed in the GUI environment. Additionally, the FLC is applied to maintain the DC load bus voltage at a constant value and to control the speed of a PMDC motor as one of the loads being

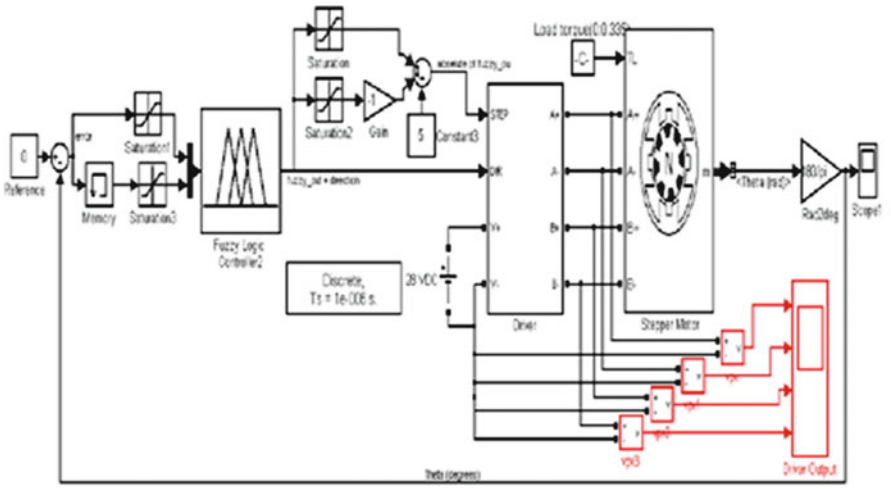


Fig. 3.52 Testing the FLC in the sun tracker system using MATLAB/SIMULINK

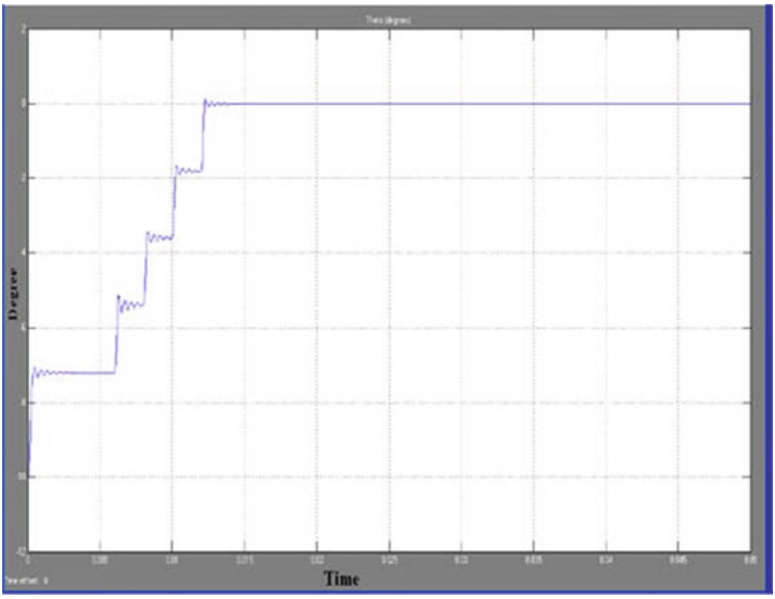


Fig. 3.53 Output degree

fed. This FLC controller has common design criteria's therefore it can easily be altered and expanded for controlling several systems. The modeled FLC is applicable for three different parts of the PVA stand alone utilization scheme: (i) the speed control of the PMDC load (ii) controlling the DC load bus voltage, (iii) maximum power point (MPP) tracking control, which is used to operate the PVA at its available maximum power as the solar insulation and ambient temperature change.

3.7.1 Methodology

The application of modern effective photovoltaic solar cells (PVSCs) has risen out as a crucial solution in energy conservation and demand-side direction during the past few decades. Due to their initial high costs, PVSCs have not as yet been an attractive mutually exclusive for electricity users who are able to buy less expensive electrical energy from the utility grid. All the same, they have been applied extensively for water pumping and air conditioning in distant and detached areas wherever substitute power is not available or is excessively unaffordable to transport. Though solar cell (SC) costs have reduced substantially during the last years owed to modern growths in the film engineering and fabricating technique, PV arrays are even believed quite expensive compared with the utility fossil fuel generated electrical energy costs. Subsequently building up such an expensive renewable energy system, the user by nature prefers to control the PV array at its maximum conversion efficiency by endlessly utilizing the utmost acquirable output power of the array. The electrical system powered by solar cells necessitates exceptional design circumstances due to the changing nature of the solar power generated resulting from irregular changes in weather conditions which affect the solar radiation level as well as the cell operating temperature. Different schemes to solve the problems on maximum power point tracking (MPPT) have been an all time active topic for PVA employment systems.

PVA operational strategies comprise of primarily four different controlled parts as MPP tracking controller, backup battery charge regulator and controller, load bus voltage controller, and special load controllers. A PVA system must be weighed as a complete unit of measurement because each of these parts are attached up collectively and need to be controlled together. Separate circumstance commonly leads to the failure of required process and affect the efficiency of the system. Hence the PVA system here consists of three controllers for MPP tracking, load bus voltage and the special load controller such as PMDC motor speed control. The battery backup unit controller is combined with the MPP tracking controller and is not treated separately.

3.7.2 Theoretical Explanation

3.7.2.1 PVA Utility System

Based on Fig. 3.54, the PVA utility system comprises of a PVA, a PMDC motor driving a fan type load, a constant R-L load, a switch able R-L load, a MPPT unit consisting of a switch capable of back-up battery unit, and a filter circuit. A PMDC motor speed controller unit, a voltage controller unit, and a MPP tracking controller unit are accumulated to the system. A DC load bus is made using the electric voltage kept constant by using a controlled DC chopper. Basically more DC buses

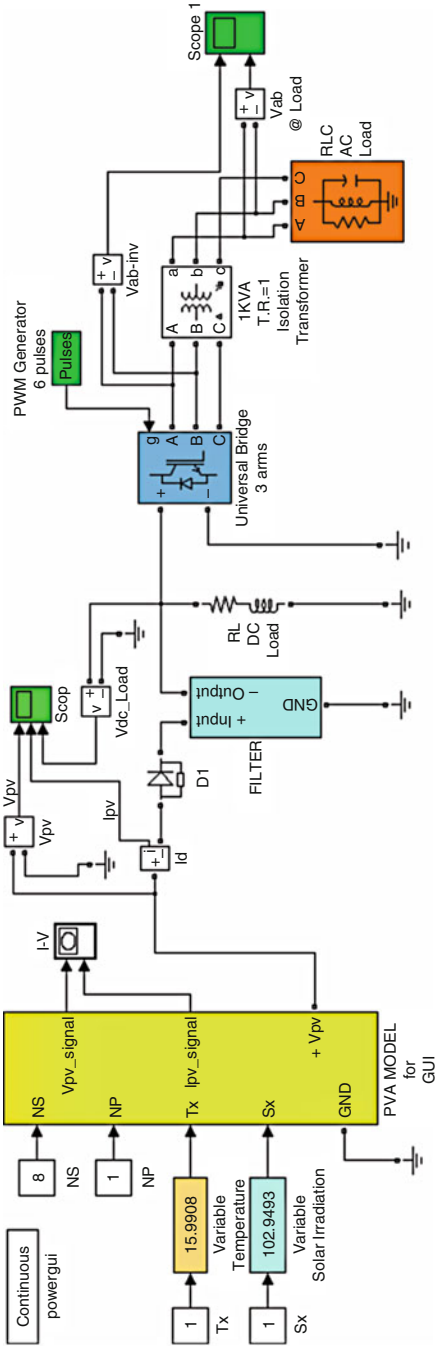


Fig. 3.54 Operational functional block diagram of the PVA model

can be produced with assorted voltage values which unbroken continuous using a DC chopper for every single DC bus given that the total load power stays between upper and lower power limitations plus the PVA MPPT operating voltage is the same as or more as opposed to voltages with the load buses. The particular potential drop in weight jalopy can be retained frequent considering that the lots are often built to end up being controlled from the user definite ranked voltage pertaining to more secure procedure. Additional operating conditions are required such as the PMDC motor as shown in Fig. 3.54. Here, the PMDC motor is actually operated on various speeds. As a result yet another controller is utilized for speed control of the PMDC generator. The load is characterized by a R-L load connected to the load bus which in turn is handled on constant voltage without any control strategy. The PMDC motor and the R-L loads are believed to be the loads fed at all times. Additional loads can be characterised by switching R-L loads connected either in series or parallel. The entire load power has to match the PVA output power.

3.7.2.2 GUI Modeling of the PVA

An over-all block diagram of the PVA model for GUI environment using SIMULINK is presented in Fig. 3.54. The PVA block in the GUI is the last stage on the model. The other stages of PVA are masked as functional subsystem blocks in the last stage shown in Fig. 3.54. The first stage in the PVA modeling is depicted in Fig. 3.55 in which the mathematical model of a single PV cell is represented with the block known as Equation 1. Another block representing the impacts of solar

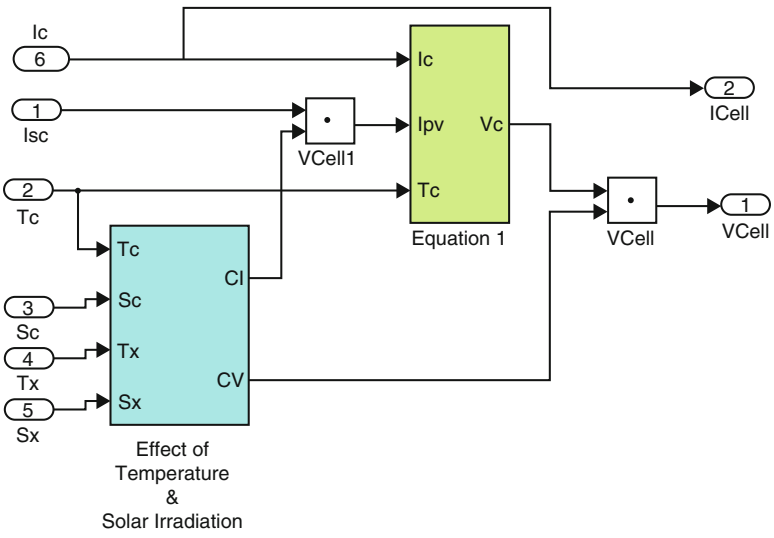


Fig. 3.55 Modeling stage 1

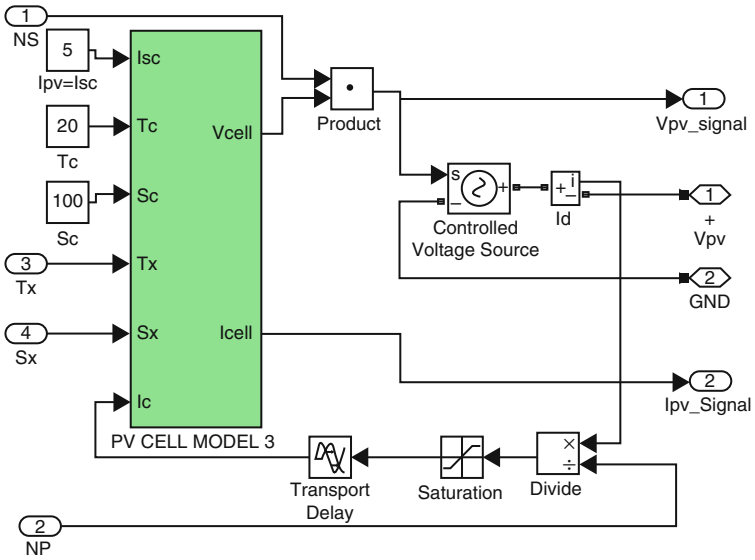


Fig. 3.56 Modeling stage 2

insulation and operating temperatures are also built into Fig. 3.56, a sub-mask of the stage 2, given in Fig. 3.56.

3.7.3 SIMULINK Model of FLC Blocks

The blocks of an FLC such as: fuzzification, rule base inference, and defuzzification are explained in this section.

The crisp inputs e(k) and Δe(k) are converted to fuzzy membership values on the fuzzy subsets Negative Big (NB), Negative Small (NS), Zero (Z), Positive Small (PS), and Positive Big(PB). Each fuzzy subset (FS) is represented by a triangular membership function. The letters E and DE are used to indicate whether the elements belong to error or its change DE, respectively.

The Boolean operator "*min*" is used with the verbal connector "*and*" to simulate the *input space of the rules* that have the structure as in expression;

If e is PB ***and*** Δe is NB ***then*** Δu1 is NB.

The input space in above expression is the part that represented by the expression (e is PB ***and*** Δe is NB). Therefore the min operator in SIMULINK Block Library is used to model the input spaces of 25 rules used by FLC. The outputs of the "*min*" operators indicate the strength (membership degree) of the rules in the output space Δu1. The implementation of the rule input space by the expression (e is PB ***and*** Δe is NB) is nothing but the fuzzification of the two crisp inputs e and Δe for all the

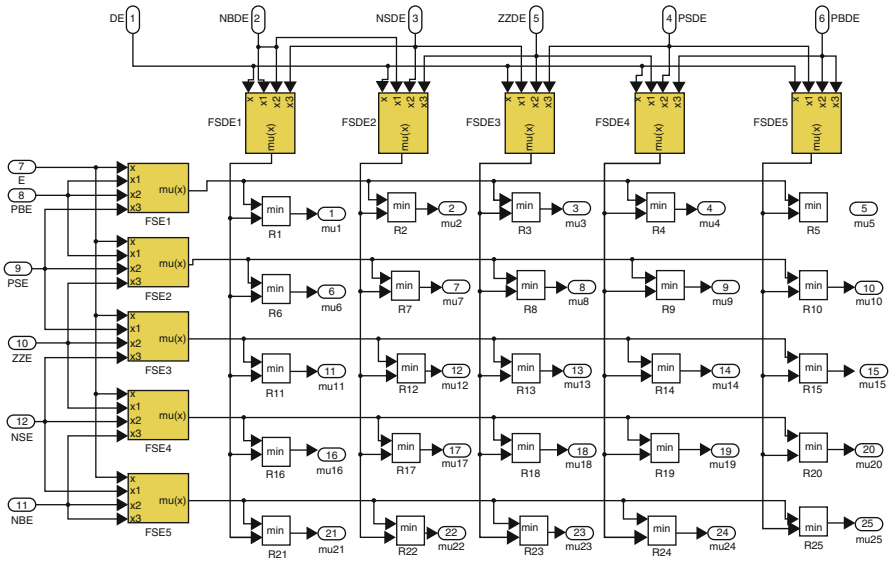


Fig. 3.57 The SIMULINK model of the fuzzification process

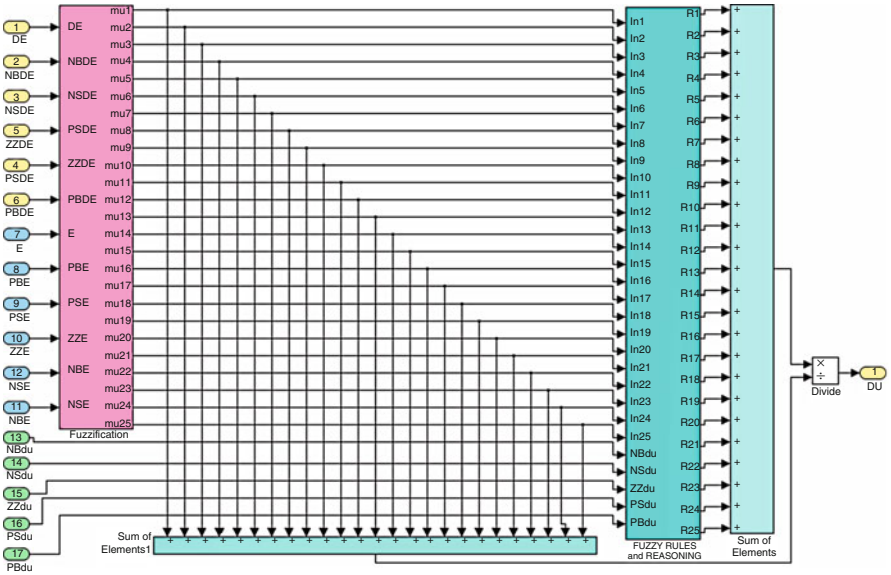


Fig. 3.58 The process from fuzzification to defuzzification in the FLC block

rules. The process of fuzzification of the input space with 25 rules is shown in Fig. 3.57. Simulation model of the *centre of area* method for defuzzification and reasoning is depicted in Fig. 3.58. This is the final stage of the FLC to generate the required change in control signal for the current k*th* sampling.

3.7.4 Simulation Results

The MPP tracking controller is tested by simulating the machine given in Fig. 3.54. The pace in the PMDC motor along with the load bus voltage are also controlled using a fuzzy logic controller just like the one explained above. Since rated voltage in the PMDC motor is 36 V, the strain bus voltage is kept constant at 40 V. The average importance of the output voltage on the chopper is weighed against 40 V reference voltages along with the difference utilized from the controller to get the desired chopper duty cycle in order that the DC bus voltage is kept at 40 V. The duty cycle in the chopper for PMDC motor speed control depends on final results controller, which is used to maintain the motor speed at 200 rad/s. The total current drawn by each of the loads and backup batteries being charged would be the current drawn from the PVA. This total current is divided by the number of cells in parallel to get the current of any single cell so that it enables to simulate PV cell model to build a voltage, which is then multiplied by the number of cells in series to discover the PVA output voltage.

A novel MATLAB/SIMULINK based unified Functional Block Model is presented above and a novel but simple FLC model for use in MATLAB/SIMULINK GUI environment. The nature of maximum power point MPP-Search and Detection algorithm is dynamic and operates without any required measurement or forecasted PV array information about the irradiation and temperature. The FLC GUI model can be used in different stages of the PVA scheme to control both MPP trackers, load bus voltage and load motor speed. Simulation results in Figs. 3.59, 3.60, 3.61, and 3.62 show acceptable MPP tracking as well as voltage and speed control responses.

3.8 Fuzzy Optimization for Solar Array System

During the recent years there has been an ever rising energy demand, owing to industrial development and population growth, motivating research and technological investments related to improved energy efficiency and generation. The reduced life cycle and upkeep of alternative resources can complement hydropower, fossil fuel and nuclear power generation. Solar photovoltaic power can be a prime candidate in many countries, where the solar power density with direct overhead incidence is 1 kW/m^2. Regardless of the high initial solar installation costs plus a low cell conversion efficiency (from 12 % on ordinary units up to and including maximum of 29 % on very special ones) the costs with the solar cells and power electronics are sharply decreasing, and that is supposed to aid the acceptance of such systems. The solar array is usually associated with an electricity storage system, that may consist of a method of batteries, fuel cells, flywheels or perhaps the distribution mains. The solar power is subjected to varying solar intensity and alterations in environmental temperature which, in turn, get a new maximum power

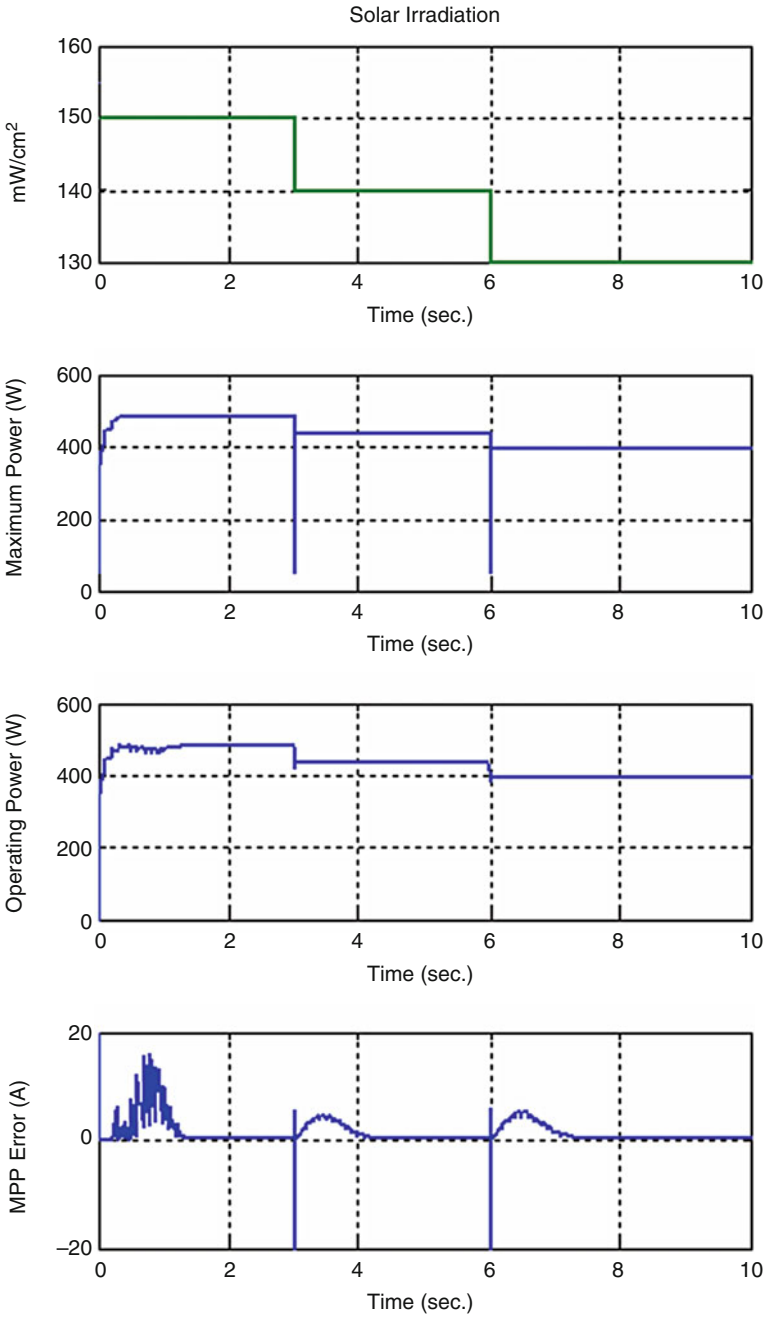


Fig. 3.59 Variations of the PVA power quantities as the solar irradiation level decreases

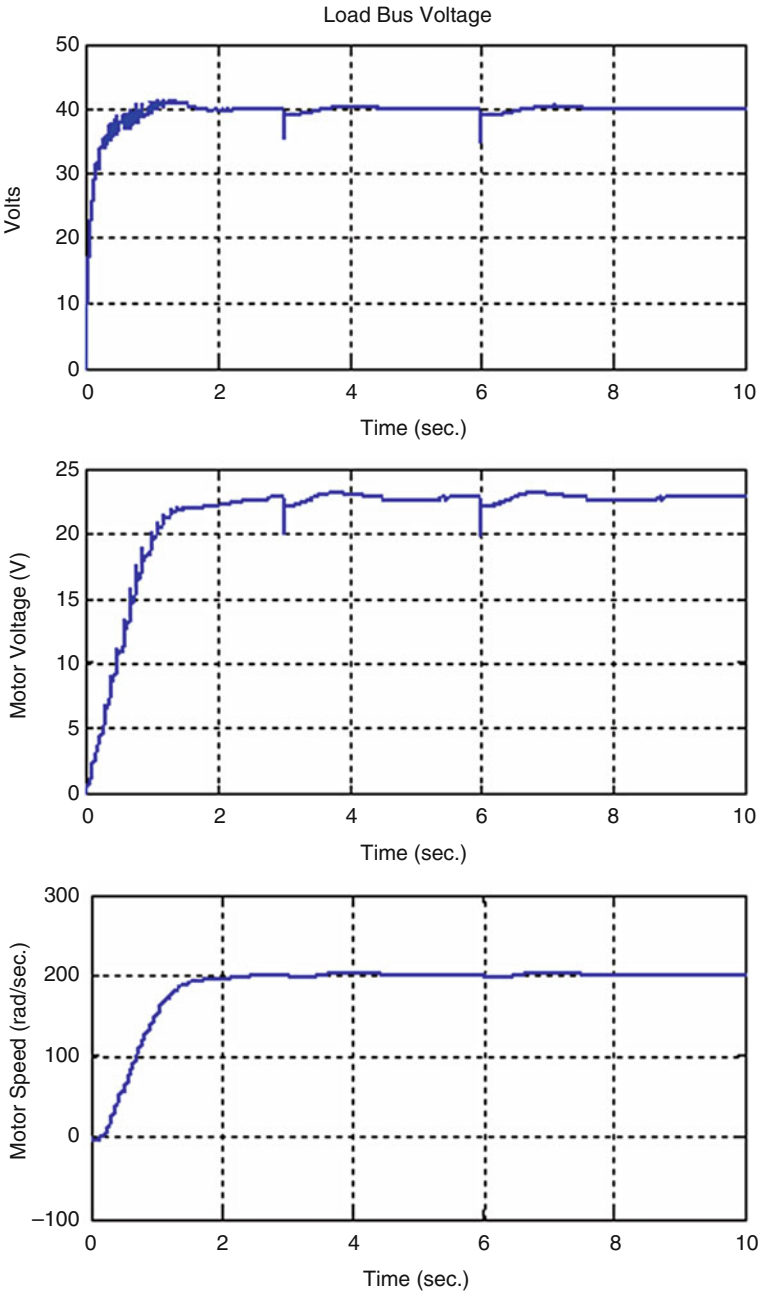


Fig. 3.60 Variations of the DC load bus voltage, PMDC motor voltage, and the motor speed as the solar irradiation level decreases

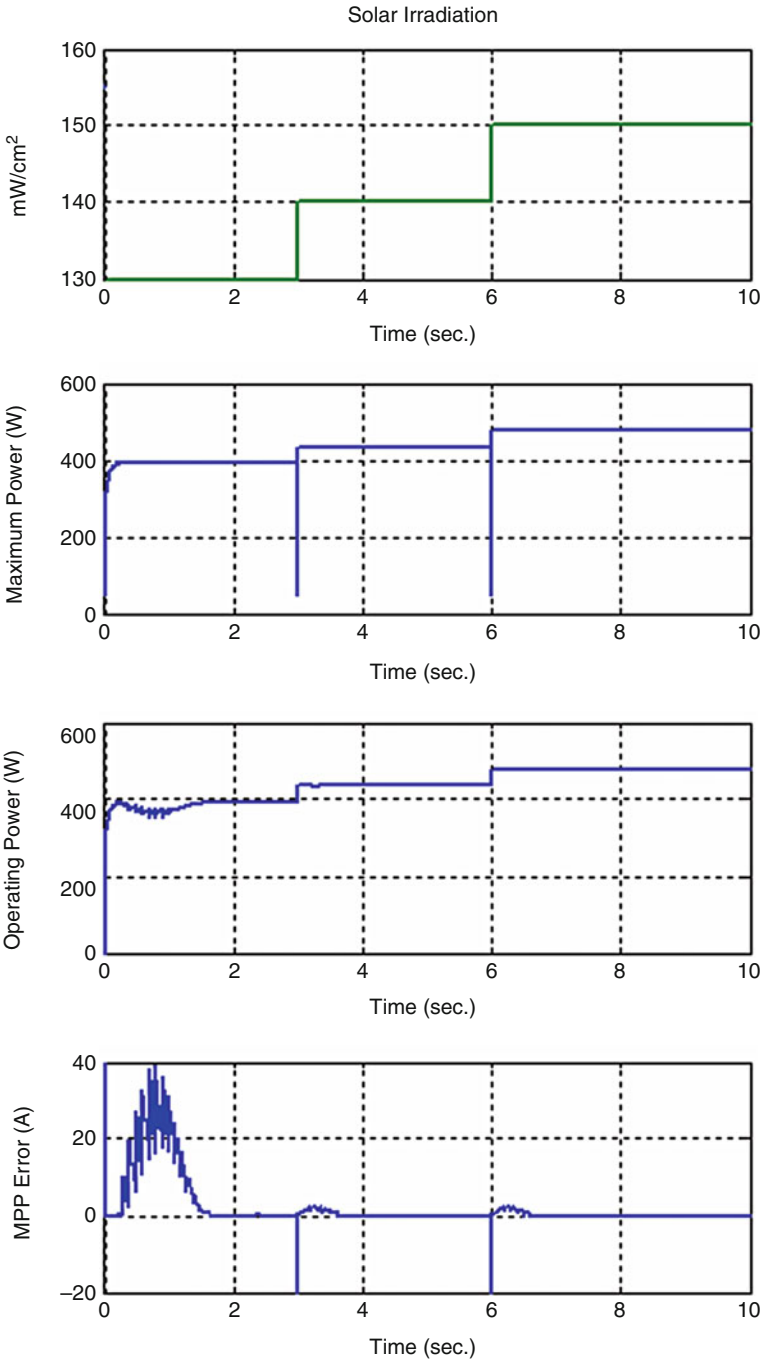


Fig. 3.61 Variations of the PVA power quantities as the solar irradiation level increases

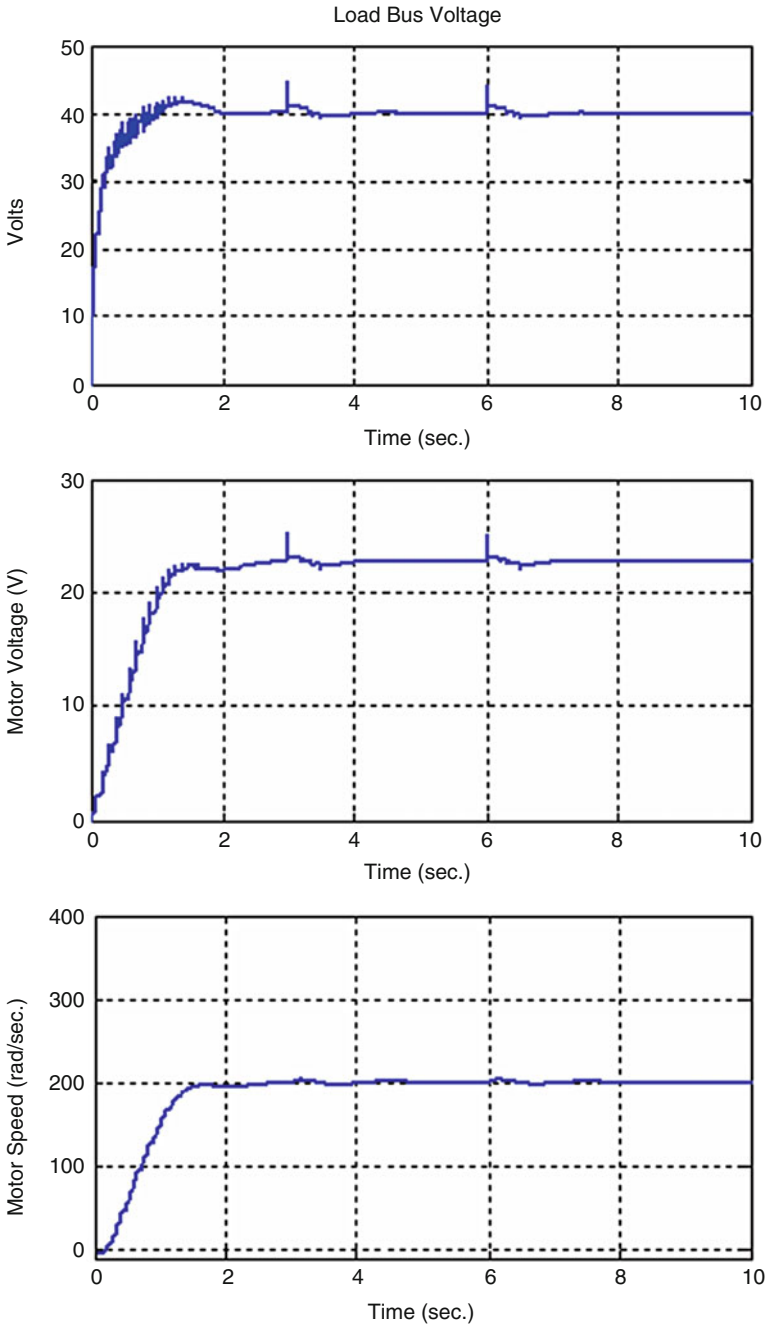


Fig. 3.62 Variations of the dc load bus voltage, PMDC motor voltage, and the motor speed as the solar irradiation level decreases

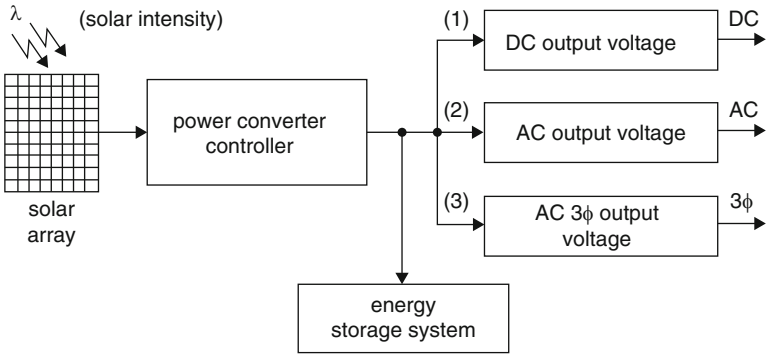


Fig. 3.63 Photovoltaic system block diagram

operating point. So as to increase the amortization of installation costs, it is essential to extract the utmost power available on continuous basis.

A few prominent abilities of fuzzy system are: online adaptive search of maximum power, robustness to solar intensity and temperature variation, does not require external sensors for solar intensity and temperature measurements. In this section, the system have been analyzed simulated and implemented by a low cost RISC microcontroller. The rig is modular, that permits paralleling for higher power installations.

3.8.1 Photovoltaic Systems

A typical photovoltaic system is shown in Fig. 3.63. The solar panel receives the sunlight and converts it to electricity by means of a power converter which possesses a peak-power tracking controller. A power storage system and possible connections with a single phase or three phase utility may also be provided. Usually the storage system is connected by way of a DC-link, but very recently, AC links using lossless resonant power converters have demonstrated promise. Several schemes are available for photovoltaic systems for residential applications and solar powered vehicles and various different control strategies for instance online conductance optimizers that keep the power factor near to unity, dynamic modeling for power maximization and fuzzy PI controllers to interchange conventional PI controllers inside feedback circuit. Special power electronic inverter circuits have also been suggested for photovoltaic systems. Solar arrays are already modeled for mathematical optimization, with emphasis on their physical properties.

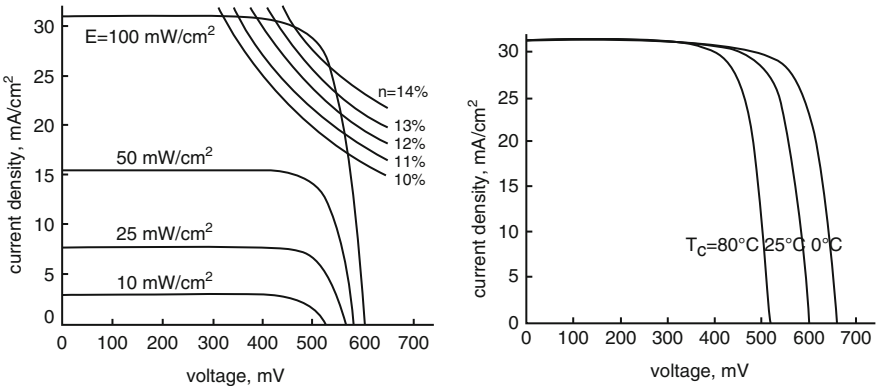


Fig. 3.64 Operating curves

3.8.1.1 Photovoltaic Array System Modeling

Solar panels are devices that convert photons into electrical potential in a p-n silicon junction. Due to the complex physical structure in the photovoltaic cell, manufacturers generally present a family of operating curves (I-V) shown in Fig. 3.64, with solar incidence and temperature as parameters. For each condition there is a unique point located in the knee with the I-V curve, where the solar cell will generate maximum power. Figure 3.64 also demonstrates the cell current density which is a function of voltage, and that the curves could be displaced vertically by solar intensity and horizontally by temperature variation. The model equations are listed below:

$$J = k1 \cdot \lambda - k2 \cdot e^{\alpha V - \beta T - \gamma} \tag{3.61}$$

$$\upsilon = \frac{1}{\alpha}\left(\ln\left(\lambda\frac{k1}{k2} - \frac{i_p}{k2 \cdot S}\right) + \beta T + \gamma\right) \tag{3.62}$$

$$\upsilon_p = \sum_{j=1}^{N_{cel}} \upsilon_j - N_{cel} \cdot R_s \cdot i_p \tag{3.63}$$

The parameters k1 (non-dimensional), k2 (m A/cm^2), α (voltage coefficient, v^{-1}), β (temperature coefficient, °C^{-1}) and γ (non-dimensional) are adjusted on the manufacturer's curves having a multi-rectilinear regression algorithm, to be able to minimize a quadratic error function. Figure 3.65 shows a PWM power-boost converter that steps in the array voltage to a higher DC bus voltage operating in continuous conduction mode (CCM). Equations 3.64, 3.65, and 3.66 provide dynamic equations, which in turn govern the boost converter operation

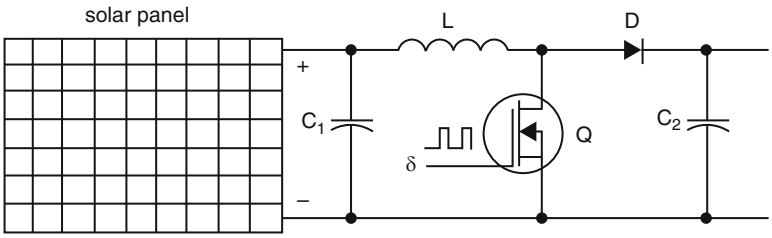


Fig. 3.65 Solar array driven by a boost converter

$$\frac{di}{dt} = -\frac{R_L + (1-\delta)(R_L/R_C)}{L} i - \frac{(1-\delta)R}{L(R+R_C)} v + \frac{v_p}{L} \tag{3.64}$$

$$\frac{dv}{dt} = -\frac{(1-\delta)(R)}{(R+R_C)C} i - \frac{v}{(R+R_C)C} \tag{3.65}$$

$$v_0 = (1-\delta)(R_L/R_C) + \frac{R}{(R+R_C)} v \tag{3.66}$$

3.8.1.2 Open Loop Simulation Studies

The system depicted in Fig. 3.65 is implemented in MATLAB/SIMULINK using the above equations. A thirty-cell solar array supplied voltage to the boost converter which, in turn, delivered power to an equivalent output load. Initially the system will operate in open-loop mode, in order to determine the behavior for various combinations of input variables and conditions. Figure 3.66, 3.67, 3.68, 3.69, 3.70, 3.71, and 3.72 show the system static characteristics for three temperatures (0 °C, 25 °C, 60 °C) with a fixed load resistance of 100 Ω and the solar intensity λ, varying gradually, from 10 mW/cm^2 up to the maximum of 100 mW/cm^2.

The shift of the peak power operating point is evident in such figures, as seen by the different duty-cycle, δ, values for the peak power. Figure 3.73 shows the power for three different load resistance values, at a fixed temperature (T = 25 °C) and solar intensity showing that the duty cycle, where the power is maximum also changes with the equivalent load resistance.

3.8.2 Peak-Power-Transfer Search

Figure 3.74 shows the solar array current/voltage curves superimposed with a set of constant power curves, indicating the region of maximum power delivery for nominal solar intensity and temperature, °C. The produced instantaneous power emerged with the product from the DC-link voltage with the DC-link current plus

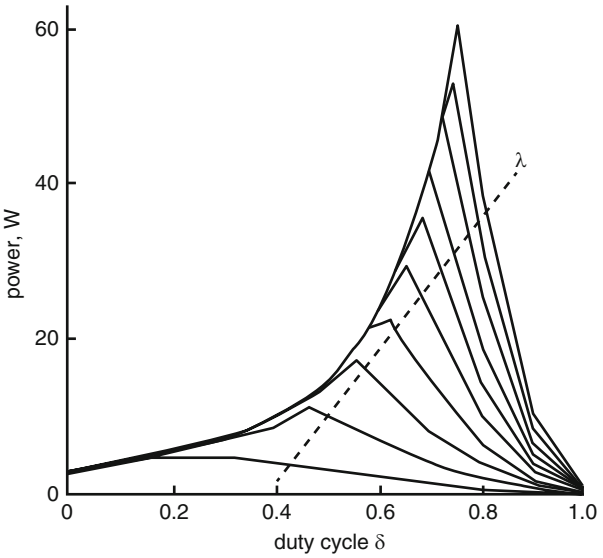


Fig. 3.66 Photovoltaic system output for λ from 10 to 100 mW/m^2 at T = 0 °C

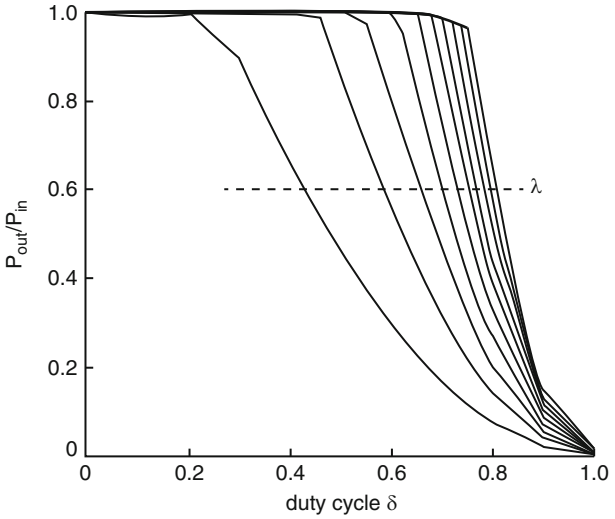


Fig. 3.67 Photovoltaic system efficiency P_{out}/P_{in} for λ from 10 to 100 mW/m^2 at T = 0 °C

the power used in the DC-link equals the solar-array power (assuming a stable-state lossless system). This means that, for that specific parameters, the obligation cycle from the boost power converter might be varied to vary the load-line slope, i.e. the solar-array impedance.

During the steady-state operation, it can be desirable to modify Z^{-1} so the peak-array power is available. This algorithm may be based on a look-up table, which

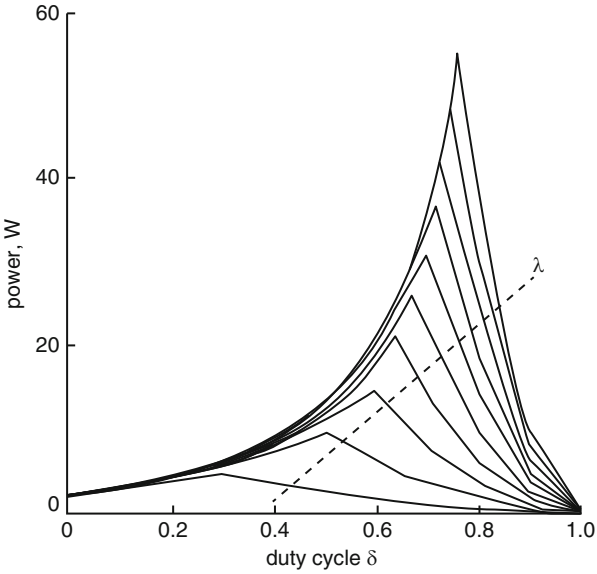


Fig. 3.68 Photovoltaic system output for λ from 10 to 100 mW/m^2 at T = 25 °C

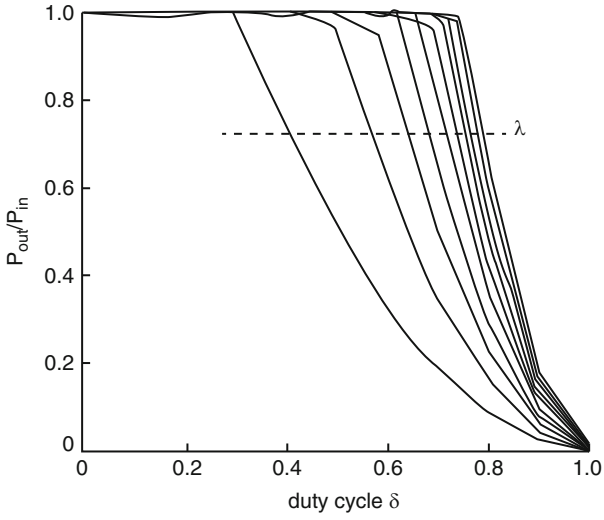


Fig. 3.69 Photovoltaic system efficiency P_{out}/P_{in} for λ from 10 to 100 mW/m^2 at T = 25 °C

would either need an exact solar-array parameter estimation, or a step-by-step duty-cycle alteration. The step-by-step duty cycle alteration is less favorable, since it may create a time-consuming search and a low-frequency oscillation towards the optimum point.

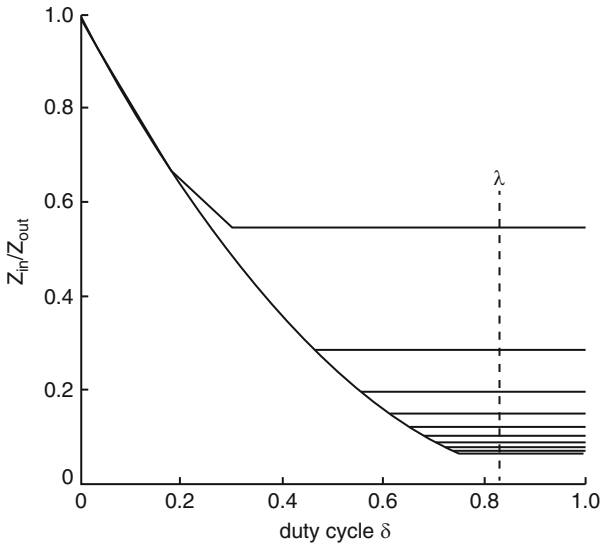


Fig. 3.70 Photovoltaic system normalized array impedance for λ from 10 to 100 mW/m^2 at T = 0 °C

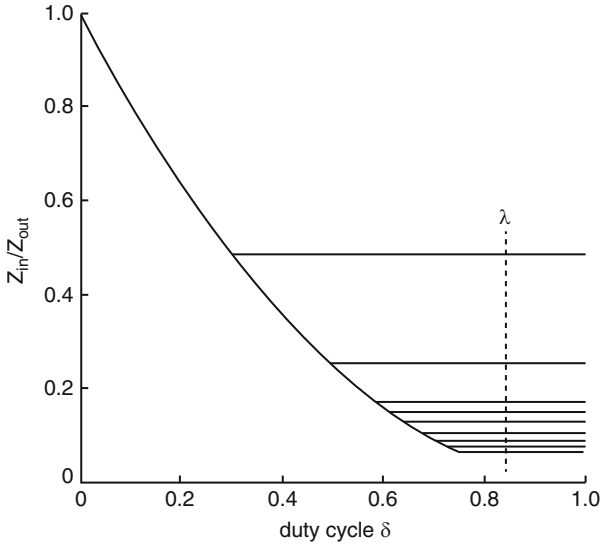


Fig. 3.71 Photovoltaic system normalized array impedance for λ from 10 to 100 mW/m^2 at T = 25 °C

3.8.3 Fuzzy Logic Based Solar Array Controller

The block diagram of the complete system is shown in Fig. 3.75. The solar array and the boost-power converter can be modeled according to (3.61, 3.61, 3.62, 3.63, 3.64, and 3.66), while the fuzzy control algorithm is based on rules written and fine tuned by simulations in the SIMULINK/MATLAB environment. The fuzzy algorithm searches for the maximum power heuristically, using the meta-rule:

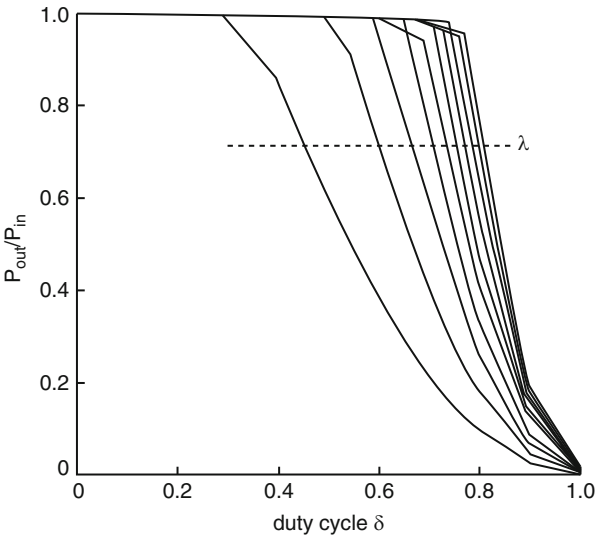


Fig. 3.72 Photovoltaic system efficiency P_{out}/P_{in} for λ from 10 to 100 mW/m^2 at T = 60 °C

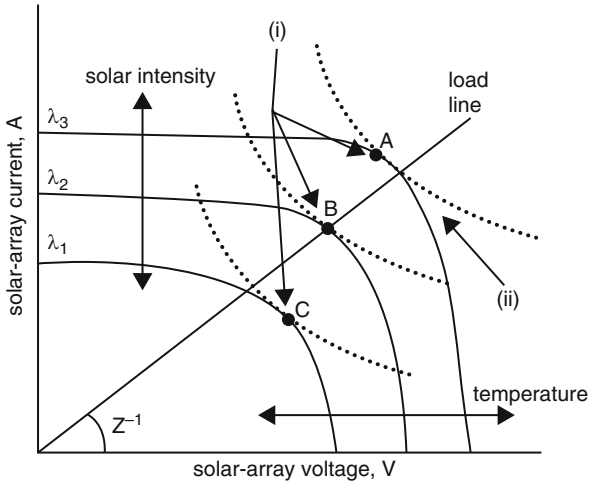


Fig. 3.73 Curves for solar intensity and temperature variation (i) Maximum peak power points, (ii) Maximum power hyperbola

If the last change in the duty-cycle (DC) has caused the power to go up, then exercise the duty (DC) in the same direction; otherwise, when it is mainly responsible for the power dropping, move it in the opposite direction.

The control logic, described above being a simple sentence, is translated using the following fuzzy control rules:

- IF ΔDC_k is P AND ΔP_k is PB THEN ΔDC_{k+1} is PB.
- IF ΔDC_k is P AND ΔP_k is PS THEN ΔDC_{k+1} is PM.
- IF ΔDC_k is N AND ΔP_k is PB THEN ΔDC_{k+1} is NB.

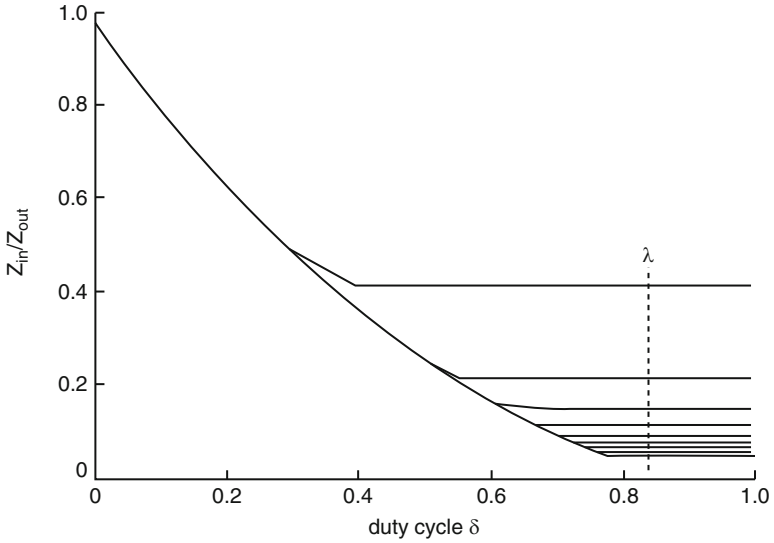


Fig. 3.74 Photovoltaic system normalized array impedance for λ from 10 to 100 mW/m^2 at T = 60 °C

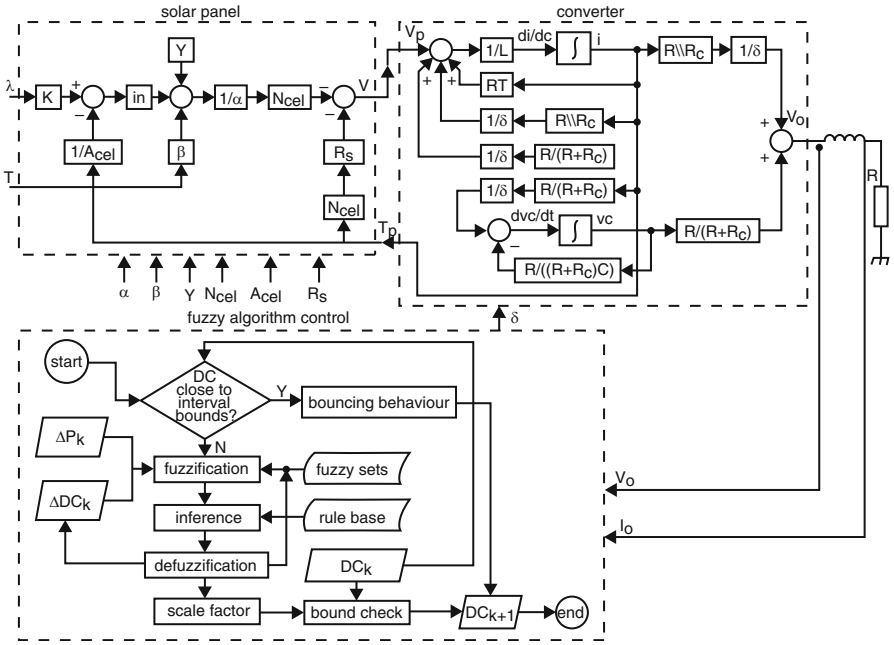


Fig. 3.75 Fuzzy logic based solar array controller

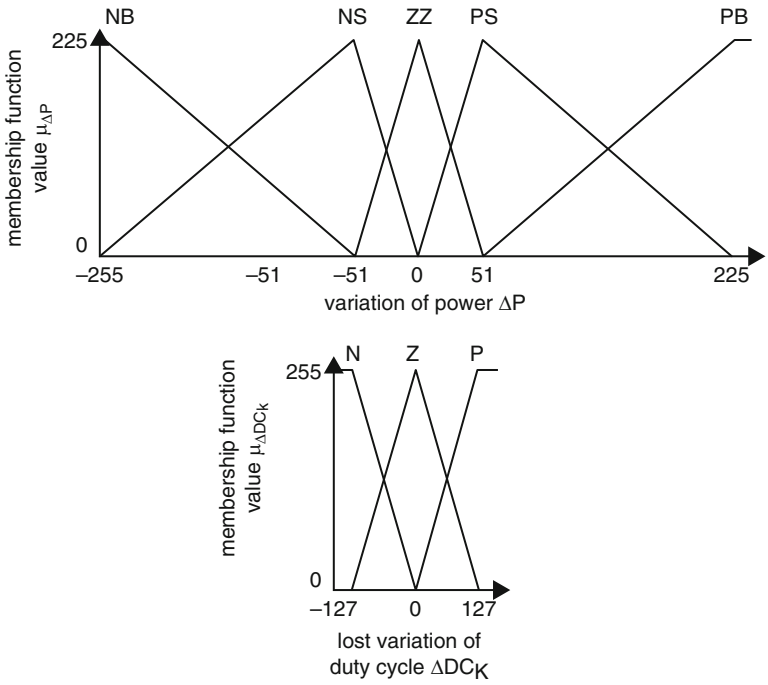


Fig. 3.76 Membership functions for fuzzy search controller

- IF ΔDC_k is N AND ΔP_k is PS THEN ΔDC_{k+1} is AM.
- IF ΔDC_k is P AND ΔP_k is NB THEN ΔDC_{k+1} is NB.
- IF ΔDC_k is P AND ΔP_k is NS THEN ΔDC_{k+1} is NM.
- IF ΔDC_k is N AND ΔP_k is NB THEN ΔDC_{k+1} is PB.
- IF ΔDC_k is N AND ΔP_k is NS THEN ΔDC_{k+1} is PM.

The two inputs used in this approach namely the variation-of-duty-cycle (DC) and variation–of-power (P) are represented by fuzzy sets as shown in Fig. 3.76. Additionally, other rules are also required, due to the fact that the characteristic curves might change with temperature and sunlight level, leading to an overall shift of the optimum point. In order to keep track of this situation, the system must have the following rules:

- IF ΔDC_k is Z AND ΔP_k is PB THEN ΔDC_{k+1} is PM.
- IF ΔDC_k is Z AND ΔP_k is PS THEN ΔDC_{k+1} is PS.
- IF ΔDC_k is Z AND ΔP_k is NB THEN ΔDC_{k+1} is AM.
- IF ΔDC_k is Z AND ΔP_k is NS THEN ΔDC_{k+1} is NS.

From the simulation studies it is observed that a vital feature to provide an artificial inertia is required, in order to keep it from stopping whenever a zero-crossing of the power is detected, resulting in a curve for ongoing motion for some

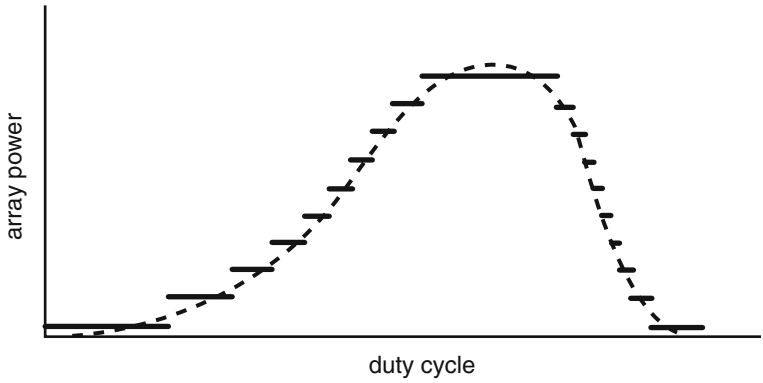


Fig. 3.77 Quantization effect on power search

cycles. The usefulness in this group of rules becomes clear when one considers the quantization effect shown in Fig. 3.77. Considering that the input signals are digitized, the continuous curve is broken down into a series of plateaus. Clearly, the steeper the curve, the shorter the plateau. Since the optimum point tends to match the condition, it might recognize any large plateau to be a maximum power region. The subsequent rules are actually identified for avoiding the stabilizing effect inside a region other than that of true peak power:

- IF ΔDC_k is P AND ΔP_k is ZZ THEN ΔDC_{k+1} is PS.
- IF ΔDC_k is N AND ΔP_k is ZZ THEN ΔDC_{k+1} is NS.

At last, it is required to provide the system with a rule that stabilizes the point of operation at a peak power point:

- IF ΔDC_k is Z AND ΔP_k is ZZ THEN ΔDC_{k+1} is ZZ.

According to Fig. 3.76, the membership functions for ΔP and ΔDC are more dense towards the centre, thus providing more sensitivity as the changes in power approaches zero. The implementation of fuzzy-algorithm is shown in the block diagram of Fig. 3.78. The duty-cycle parameter (ΔDC) is internal to the PWM routine, which is stored as a floating point number in order to increase control resolution; ΔDC_k is a feedback parameter, stored as an 8-bit value (1 sign bit, 7-bit absolute value); ΔP_k is a calling parameter, stored as a 9-bit value (1 sign bit, 8-bit absolute value); DC is obtained as a floating point value, in order to allow precise scaling at the end of the process; however, it is converted to an integer at the end of the sampling-time routine.

The duty cycle is bound by upper and lower limits (90 % and 10 %) as a safety measure; if DC is close enough to one of the ends of the interval, the fuzzy algorithm is overridden and the DC is 'bounced' back to the centre of the interval. The same safety reasons apply to the bound check that is performed at the end of the

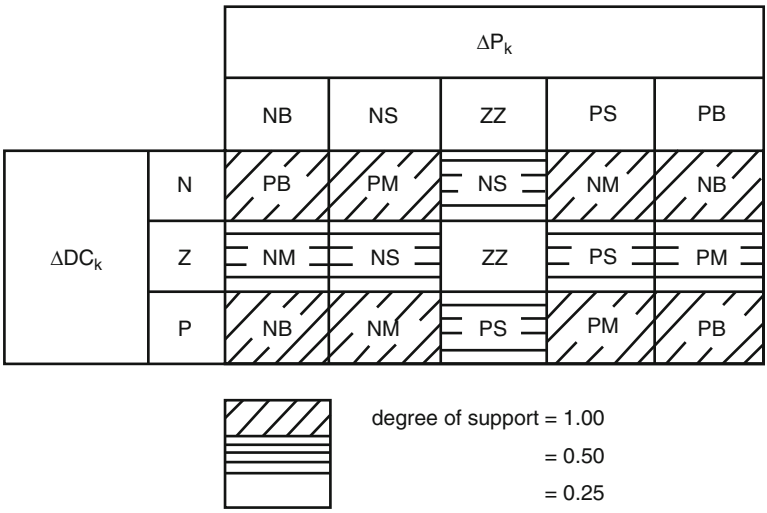


Fig. 3.78 Fuzzy rule base with different degree of support

control cycle; if the new duty cycle ($\Delta DC_k + S.\ \Delta DC_{k+1}$, where S is a scaling factor) ends up being outside the specified interval, it is reduced to the closest bound. Once these exceptions have been accounted for, the fuzzy algorithm is run. The first task is to perform the fuzzification of the input parameters, indicated by the fuzzy sets shown in Fig. 3.76. Following this, the fuzzy inference engine applies the control rule base described above.

The user can find several rule weights to satisfy different conditions. The strongest condition needs to be the one corresponding towards the first band of rules, since the task represented by them describes the conventional system operation. The other and third sets deal with special conditions thereby have a lower strength. They are active only when certain specific conditions apply. A final list of rules, activated whenever the device reaches the optimum point, only need an easy weight. The quantization effect on power search is illustrated in Fig. 3.77. These varying strengths are implemented with different degrees of support on the corresponding rules, as shown by Fig. 3.78.

Once all the rules have been evaluated, then the final stage of FLC, defuzzification is performed. Here, the centre of gravity method of defuzzification is employed, thus presenting the result in floating-point format. The usefulness of utilizing the time and memory consuming floating-point format becomes clear when the user considers the precision that the scaling factor S needs to be tuned. After the effect can result in defuzzified and scaled, it can be placed on the present DC and fed returning to the following control cycle, by making $\Delta DC_k = \Delta DC_{k+1}$.

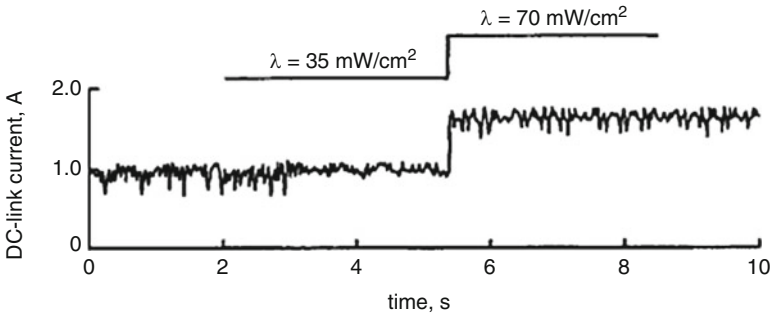


Fig. 3.79 Solar intensity variation and the corresponding DC-link current change with fuzzy duty cycle search

3.8.4 Experimental Results

The hardware model of the system shown in Fig. 3.75 was implemented in the laboratory with a RISC microcontroller type PIC16C74. A few notable features in this chip include, inbuilt PWM, input compare capture functions, serial and parallel interfaces and can be programmed with only 35 instructions. The simulation study was carried out in MATLAB/SIMULINK and the software was written in an assembler, on the basis of the block diagram in Fig. 3.75. The complete peak-power tracking system with the solar array was connected in the laboratory to charge the batteries, using variable illumination supplied by tungsten halogen lamps.

Experiments also included the application of sunlight, with a protective screen to produce a variable shade to the solar array. Figure 3.79 shows a step response for illumination level variation, showing that the DC-link current flowing to the batteries quickly establishes the new operating condition after the solar intensity has been varied from $\lambda = 35$ mW/cm^2 to $\lambda = 70$ W/cm^2.

An entire fuzzy logic solar array peak power tracking controller have been simulated, designed and implemented from the laboratory. The power circuit based on a boost converter, and the controller, which has a fuzzy algorithm that results in optimum duty cycle and transfers the peak power from the solar array into a charger. The major advantages of the fuzzy controller are that the control algorithm gives fast convergence and robust performance against parameter variation, and can accept noisy and inaccurate signals. It was found to reliably stabilize the utmost power transfer in all types of operating conditions.

3.9 Forecasting of Solar Irradiance Using ANN

Solar irradiance forecasting could be the foundation of power prediction of PV generation. While using the fast development of photovoltaic generation in recent times, there's an increasing desire for more precise and applicable modeling,

forecasting and prediction of solar irradiance. The possible computational models include linear regression models, satellite data based models and ANN models using meteorological parameters. A regressive model to the estimation from the hourly diffuse solar irradiation under all environmental conditions will depend on the sigmoid function and applies the clearness index and also the relative optical mass as predictors. Several solar irradiance forecasting models were applied in the past. Such models could be classified into two main groups: physical models and statistical models. The physical models depend on mathematical equations which define the physical state and dynamic motion of the atmosphere. These physical models are extremely complex non-linear equations which require strong computing power to solve them. Numerical methods were applied to obtain the approximate solutions of the equations, and they are also often known as numerical weather predictions (NWP) models. NWP can be used in forecasting but it's not available commonly.

The difficulties of solar irradiance forecast based on NWP vary significantly and rely on the different climate and dynamic motion of the atmosphere in the study location. The statistical models include time series models, satellite data based models, sky images based models, ANN and wavelet analysis based models. The statistical models depending on satellite data and sky images detect the motion of cloud structures using motion vector fields. The motion of cloud structures is determined from two consecutive cloud index images from satellite data or sky camera images. The errors of satellite data and sky images based forecasts proposed in the literature increase drastically under low sun elevations, high spatial variability and low irradiance conditions.

Usually, the statistical models are less complex than physical models simply because they require less input information and shorter computation time for the forecast. Two basic features must be looked into in order to evaluate the models: the foremost is the supply of information to be applied as input by the models and the second would be the accuracy and complexity on the forecasting model. Satellite data, sky images and NWP aren't always intended for various occasions. However, common meteorological parameters including ambient temperature, relative humidity, wind speed, wind direction, sunshine duration and clearness index are instantly available in most instances, hence the conventional solar irradiance forecasting model is created with all the input on the historical measured data of solar irradiance as well as other meteorological parameters.

Based on the technical specifications in the State Grid Corporation of China, the time scale of solar irradiance forecasting includes ultra short-term (around 4–6 h ahead) and short-term (up to 24–72 h ahead) forecasting. The forecast as much as 24 h ahead or maybe more should be used for the power dispatching plans along with the optimization operations of grid-connected PV plants and also the coordination charge of energy storage devices. The most prevalent short-term forecast horizon is 24 h overnight. Solar irradiance forecasting models on different time scales according to ANN using meteorological parameters without NWP and satellite data show good

performance and could be classified into three differing types. The first estimates the solar irradiance based on some meteorological parameters, the other one allows forecasting the future solar irradiance in line with the past observed data of solar irradiance, a final one combines both the previous models. These approaches need some meteorological parameters as input from the ANN model, but the meteorological data aren't always available. So as to overcome this challenge, a fairly easy method for forecasting 24 h ahead solar irradiance using multilayer perceptron may be applied. It shows good performance in forecasting solar irradiance except under unstable conditions including fast variations or violent fluctuations.

The constituents of the input vector determine the adequacy and sufficiency on the input information. Based on the input vector dimensions, excessive components could make the model structure become too complex, which enhances the difficulty of modeling, but not enough components will result in the lack of info that cause the model not to ever reflect truth variation. Merely a reasonable balance between model complexity and information absence can settle the condition of the determination with the dimensions of input vector. Through the mindset from the introduced forms with the pieces of input vector, most ANN models directly utilize original data number of meteorological parameters as being the input vector on the model. This may import much information redundancy minimizing the generalization ability in the ANN forecasting model. To overcome these shortcomings and improve forecast accuracy, in this particular section the ANN model is improved upon by 50 % aspects: firstly, effective input components are constructed while using the few available meteorological parameters under the input dimension limits to extract more features on the existing data, secondly, CV is applied to the ANN model structure to optimize the amount of neurons along with other model parameters.

3.9.1 Relation Between Solar Irradiance and Weather Variations

The definition for solar irradiance is follows: The amount electromagnetic energy incident on a surface per unit time and per unit area is defined as solar irradiance. The energy emitted from the Sun passes through space until it can be intercepted by planets, other celestial objects, or interstellar gas and dust. The concentration of solar radiation striking these objects is dependent upon an actual law known as the inverse square law. Just about 40 % (general level) from the solar panel technology intercepted near the top of Earth's atmosphere passes to the symptoms. The proportion takes a different approach under different conditions: this proportion is greater than 40 % in sunny days, approximately equal to 40–50 % in cloudy days and fewer than 40 % in overcast days. The solar constant GSC is the sunshine, per unit time received over a unit division of surface perpendicular for the direction in

the Sun's rays, at mean Earth-Sun distance, away from atmosphere. The planet Metrological Organization suggests a value of 1,367 W/m^2.

The parameters of solar irradiance are extraterrestrial irradiance G_0 and surface irradiance G_s. Extraterrestrial irradiance refers to the upper bound irradiance and that is not affected by the aerosphere and weather conditions but is dependent upon planet earth's rotation and revolution. Surface irradiance is related to the latitude, the sun's rays elevation angle, date plus the time of the day. Ignoring some minor factors like polar motion, the precession of the planet's axis and leap years, extraterrestrial irradiance changes yearly. For the certain location, with the connection between different day's 12 months and different time of just 1 day, the corresponding extraterrestrial irradiance G_0 might be calculated as,

$$G_0 = G_{SC}\left(1 + 0.33\cos\frac{360n}{365}\right)(\cos\delta\cos\varphi\cos\omega + \sin\delta\sin\varphi)$$

where G_{SC} is solar constant, n is the date sequence number in 1 year, $n \in [1,365]$, δ is solar declination, φ is latitude, ω is solar hour angle. While the solar radiation transmits from the upper bound to the Earth's surface, a certain amount of attenuation occurs due to the absorption, scattering and reflection of the atmosphere. This degree of attenuation is related to various factors such as the amount of clouds, cloud form, cloud thickness, water vapor, and aerosols. Hence the surface irradiance at few locations is less than its corresponding extraterrestrial irradiance. The changes in surface irradiance occur randomly which in turn are closely related to the atmospheric motion. The difference between G_0 extraterrestrial irradiance and G_s surface irradiance G_d (hereinafter referred to as solar irradiance difference $G_d = G_0 - G_s$) can reflect the weather conditions. The measured data of surface solar irradiance G_s (from 2010.03.09 to 2010.03.12, Kunming, Yunnan, China), corresponding extraterrestrial solar irradiance G_0 and solar irradiance difference G_d are shown in Fig. 3.80.

From Fig. 3.80, it can be observed that the solar irradiance difference G_d is small and has similar shape under sunny days (20100311, 20100312). As opposed, G_d is larger and contains more inflections on cloudy days (20100309, 20100310) than sunny days. Quite simply, it means the change of G_d is closely related to weather conditions. The analysis of G_d related to different weather conditions often leads and allow us to extract more information from the measured data of surface solar irradiance G_s plus the calculated data of corresponding extraterrestrial solar irradiance G_0 by finding and selecting suitable mathematic parameters which can reflect these correlative variation characteristics. Due to the strong correlation between a few meteorological parameters and irradiance, these meteorological parameters are directly influenced by irradiance. Thus these parameters can reflect the alterations of irradiance which enable it to consider due to the input in the ANN forecasting model. Such parameters include ambient temperature, relative humidity, etc.

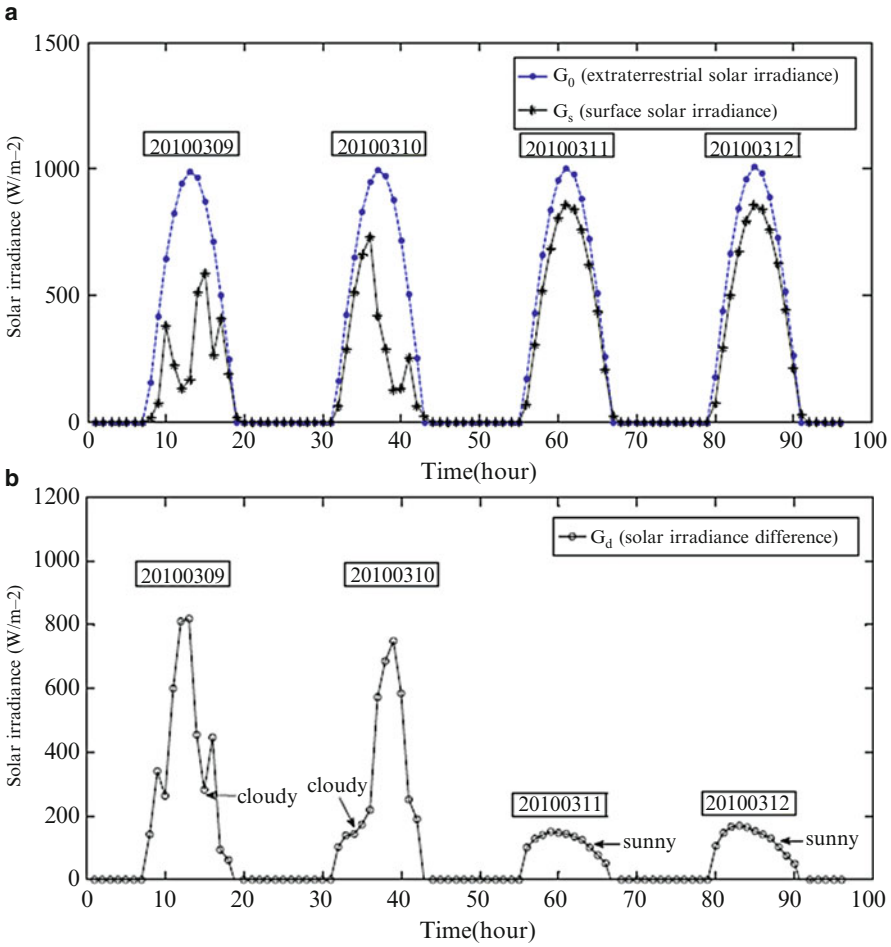


Fig. 3.80 Extraterrestrial irradiance, surface irradiance and solar irradiance difference; (**a**) Extraterrestrial irradiance and surface irradiance; (**b**) Solar irradiance difference

3.9.2 Reconstruction for the Input Vector of the Forecasting Model

The components of forecasting model input vector are classified as forecasting factors, whose selection is significant towards performance in the model. Generally, forecasting factors with the ANN model are selected from two classes of historical data: solar irradiance itself along with the meteorological parameters associated with solar irradiance. Two aspects should be considered in the selection: exactly what data must be selected and what form these data can be imported in. Generally in most from the current ANN solar irradiance forecasting models, multi-

dimensional historical data combination of solar irradiance in just a stretch of time is employed as forecasting factors, it can be shown as:

$$I_G = [G_{t-1}, G_{t-2}, \ldots, G_{t-m}]$$

where I_G is the input vector of the ANN forecasting model, G_{t-i} ($i = 1,\ldots,m$) is the historical data of solar irradiance, m is embedding dimension. The time choice of the historical results are dependant on m and sampling interval. Usually, m is more than 8.

Under varying conditions, the symptoms of solar irradiance changes randomly. The outer lining solar irradiance is zero during the period of time between sunset and sunrise even as it varies constantly between sunrise and sunset, utilizing the unprocessed multi-dimensional historical data compilation of solar irradiance as forecasting factors will import a lot of repeated information and produce high dimensions with the input vector with the model which can make the modeling tougher and complicated. Therefore, as opposed to the historical data series of solar irradiance, the statistical indicators of the historical data inside a period of time can be taken as the new forecasting factors, so effective information in the historical data will probably be fully extracted and utilized. Meanwhile, the size and style of input vector will likely be decreased. Moreover, because solar irradiance difference G_d reflect varying weather conditions, it can be viewed as forecasting factors in appropriate forms, so that accurate weather condition information can be obtained to the model along with the forecasting. The symptoms irradiance G_s is discrete sampling data and extraterrestrial irradiance G_0 is corresponding calculated value, thus G_d is a discrete multi-dimensional data series. Taking into consideration the restriction on the input vector dimensions, we could take some mathematical parameters in the data combination of G_d as forecasting factors.

Usually, the derivative is a proper index of the variation tendency of the corresponding variable. The measured data of surface solar irradiance G_s (from 2010.02.18 to 2010.02.23, Kunming, Yunnan, China), corresponding extraterrestrial solar irradiance G_0, the 1st-order, second-order and third-order derivative of solar irradiance difference are calculated and shown in Fig. 3.81. It follows through the mathematical definition that derivatives of solar irradiance difference are related to the fluctuation of climate. In Fig. 3.81, it could be seen that each one order derivatives (1st, 2nd and 3rd order) of solar irradiance difference are positively correlated towards fluctuations power of surface solar irradiance. Further research indicated that third-order derivative of solar irradiance difference (TOD = d3Gd/dt3) is obviously in excess of additional two derivatives (1st and 2nd order) while you will discover mutations from the curve of G_s meaning fast variation and violent fluctuation were took place the weather.

The comparisons of the derivatives shown in Fig. 3.81a have been illustrated here. Therefore, third-order derivative of solar irradiance difference is usually selected just as one appropriate measure to explain the variation and fluctuation of weather conditions. To achieve more significant, clear and description for several climate conditions of merely 1 day, the ideal price of TOD denoted by TODmax is

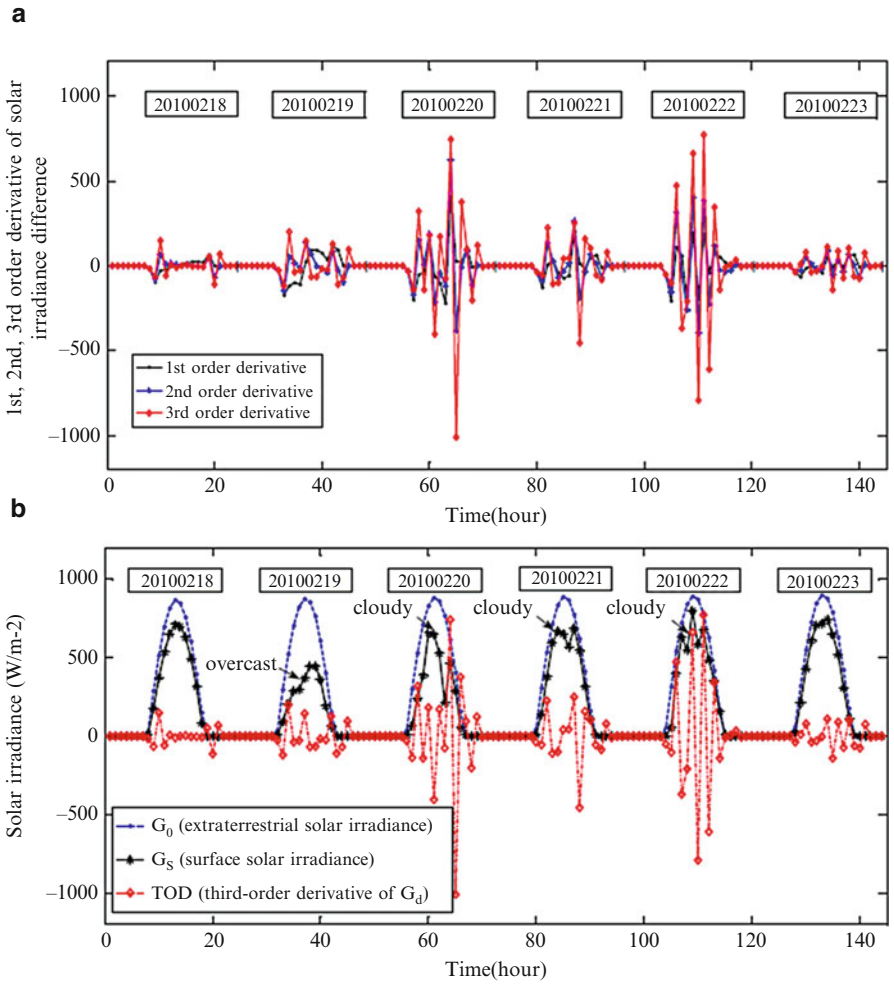


Fig. 3.81 Solar irradiance and the derivatives of solar irradiance difference; (**a**) First-order, second-order and third-order derivative of solar irradiance difference; (**b**) Extraterrestrial solar irradiance, surface solar irradiance and *TOD*

selected being an index to the weather changing characteristics. From another angle, we can notice that is very important distinction between surface irradiance G_s and extraterrestrial irradiance G_0 is closely related to the alteration inside the conditions, which is relatively smaller in sunny days when compared to cloudy or overcast days. To make clear comparison in the difference between surface irradiance and extraterrestrial irradiance, the irradiance value are normalized using (3.67) to eliminate the impact of numerous amplitude of the actual irradiance data:

$$G_{0N,i} = \frac{G_{0,i}}{\max\limits_{i=1,2,\ldots,k} G_{0,i}} \times 100 \qquad G_{sN,i} = \frac{G_{s,i}}{\max\limits_{i=1,2,\ldots,k} G_{s,i}} \times 100 \tag{3.67}$$

where $G_{0N,i}$ is normalized extraterrestrial irradiance, $G_{sN,i}$ is normalized surface irradiance ($i = 1,2,\ldots,k$), k is the number of the irradiance data. Later, the normalized discrete difference (NDD) of solar irradiance is defined in (3.68) as a specific index to describe the difference between surface irradiance and extraterrestrial irradiance:

$$NDD = \sqrt{\frac{1}{k}\sum_{i=1}^{k} (G_{0N,i} - G_{sN,i})^2} \tag{3.68}$$

Based on the definition, the first and normalized value of solar irradiance, the NDD index for each day (k = 24) towards the same irradiance data in Fig. 3.81 are calculated and shown in Fig. 3.82.

In Fig. 3.82, NDD2 = 9.00 (overcast day), NDD3 = 14.40 (cloudy day), NDD4 = 5.59 (cloudy day), NDD5 = 9.76 (cloudy day), it may be seen that NDD index during these 4 days are significantly above from the other sunny days. Out of this, the NDD index is used to measure different conditions. Together with TODmax and NDD, another component that is highly recommended is the date sequence number n a single year, n ∈ [1, 365]. Determined by these three factors, the typical surface irradiance Gsavg and the average ambient temperature Tavg of 1 day may also be given to the input vector. The last input vector in the new ANN forecasting model is composed of five components and shown as,

$$I_{ANN} = \left[G_{savg}, TOD_{\max}, NDD, T_{avg}, n\right] \tag{3.68a}$$

where I_{ANN} is the new input vector of the ANN forecasting model.

3.9.3 ANN Forecasting Model Using Statistical Feature Parameters

ANN is used as the modeling of solar irradiance forecast in line with the input vector consisting of statistical feature parameters. The time scale of short-term forecast is 24–72 h ahead. Some time interval of input and output solar irradiance series is depended on the actual situation. In line with the reconstruction for that input vector simply 3 above, a brief-term forecasting model depending on ANN using statistical feature parameters (ANN-SFP) is presented. This ANN model consists of four layers i.e., input layer, hidden layer 1, hidden layer 2 and output layer, the model is shown in Fig. 3.83.

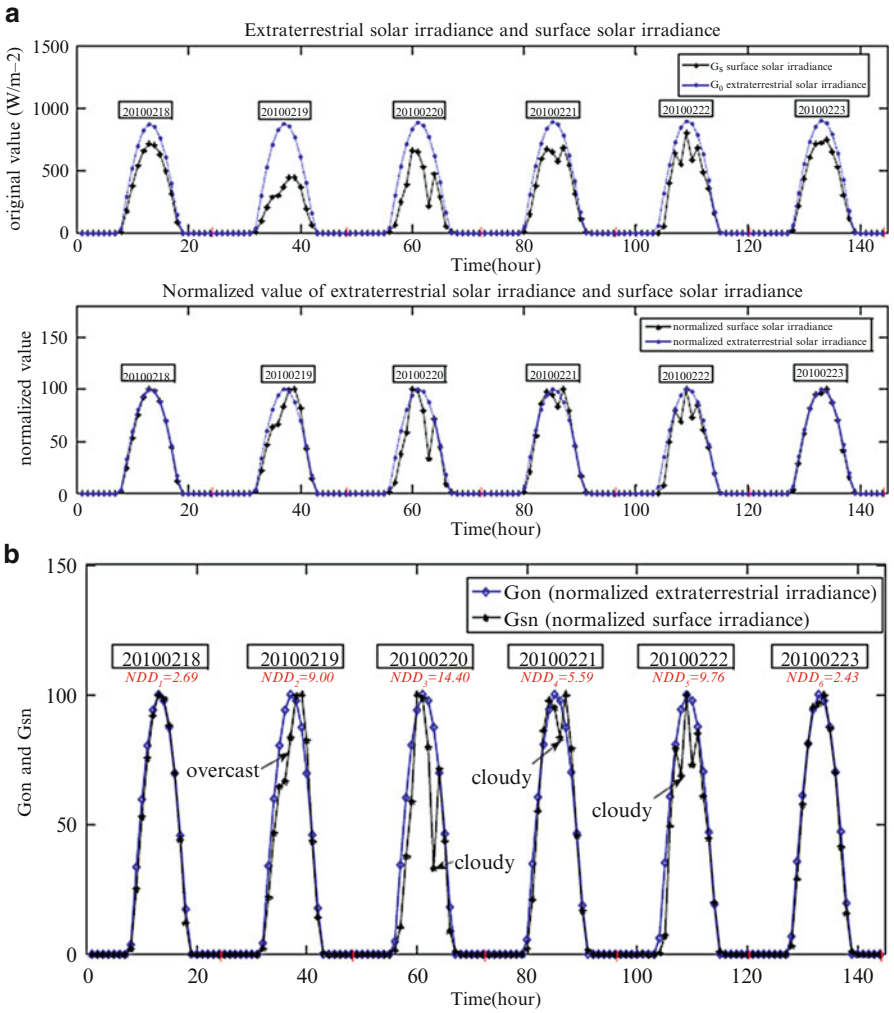


Fig. 3.82 Original value, normalized value and *NDD* of solar irradiance; (**a**) Original and normalized value of solar irradiance; (**b**) Normalized discrete difference of solar irradiance

The input vector relies on 3.68a, including three statistical feature parameters of irradiance Gsavg (n), TODmax(n) and NDD(n), one statistical feature parameter of ambient temperature Tavg(n), and also the date sequence number n. The output vector includes 24 components represent the surface irradiance from the next day (i.e., the date sequence number is n + 1). Any time resolution is 1 h for the components of output vector. The two hidden layers in the ANN model, and also the volume of neurons in hidden layer 1 and hidden layer 2 are p, q respectively. Levenberg-Marquardt Algorithm (LMA) is used rather than conventional BP algorithm. LMA combines the feature of Gauss-Newton algorithm along with the

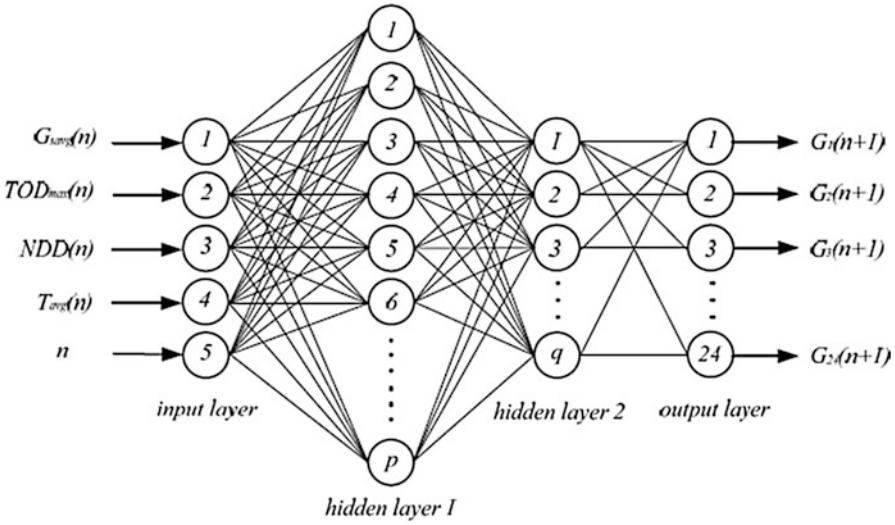


Fig. 3.83 ANN forecast model using statistical feature parameters

gradient descent algorithm. LMA might be more robust than Gauss-Newton algorithm since it finds an alternative whether or not it starts very far from the last minimum. In a large distance from the function minimum, the steepest descent strategy is utilized to provide steady and convergent progress toward the solution. As the solution approaches the minimum, damping parameter is adaptively decreased, the LMA approaches the Gauss-Newton algorithm, and the solution typically converges rapidly to the minimum.

The connection weights are adjusted by the LMA according to (3.69):

$$\Delta\omega = -\left[\mu I + \sum_{p=1}^{P} J^P(\omega)^T J^p(\omega)\right]^{-1} \nabla E(\omega) \tag{3.69}$$

where ω is connection weights, $\Delta\omega$ is weight correction, $E(\omega)$ is learning error, $J(\omega)$ is the Jacobian matrix of error vector, μ is damping parameter, I is identity matrix, P is the number of sample data. Three different error statistical indicators including mean absolute percentage error (MAPE), root mean square error (RMSE) and mean absolute bias error (MABE) are selected to evaluate the forecast accuracy of the models, these error statistical indicators can be computed according to the following formulas:

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left(\left|\frac{G_{f,i} - G_{m,i}}{G_{m,i}}\right|\right) \times 100 \tag{3.70}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(G_{f,i} - G_{m,i}\right)^2} \tag{3.71}$$

$$MABE = \frac{1}{N}\sum_{i=1}^{N}\left(\left|G_{f,i} - G_{m,i}\right|\right) \tag{3.72}$$

where $G_{f,i}$ is forecasted value, $G_{m,i}$ is measured value, ($i = 1,2,\ldots,N$), N is the number of the solar irradiance data. The benefit of the traditional ANN-HDS model is that it might be simply built only through the use of historical data compilation of solar irradiance. One main problem with ANN-HDS model is usually that the high dimension with the input vector, which will bring about more complex model that is hard to build and train. A result of the absence of the opposite related parameters except solar irradiance, another disadvantage to ANN-HDS model is the information inside the input vector is incomplete this also can lead to the losses of precision.

The solar irradiance of adjacent sunny days shows strong similarity and regularity and as a consequence perfect prediction can be performed through the common ANN-HDS forecasting model. Under changeable climate conditions, solar irradiance show irregular complex changes caused by a volume of meteorological factors, and therefore the ANN-HDS forecast model has stopped being applicable. The input vector in the ANN forecast model is reconstructed, and few statistical feature parameters are selected to change the multi-dimensional historical data group of solar irradiance. The input vector dimension is reduced plus the main information about the variation of climate might be effectively extracted through such improvement. The info redundancy and multiple coupling between input components are reduced inside the ANN-SFP forecast model.

3.10 Parameter Identification of Solar Cell Using Genetic Algorithm

The comparison of the voltage–current characteristic I(V) of a solar PV cell is dead set utilizing hereditary calculations focused around segregated purposes of the trademark got tentatively. The mathematical expressions involve a PV model with seven parameters. The basic model consists of a current source in parallel with a diode, as shown in Fig. 3.84. The sunlight makes the cell dynamic, delivering an electrical current denoted as photocurrent Iph. The output current I relies on the estimation of diode current I_D.

The basic scheme in Fig. 3.84 is extended with two resistors as shown in Fig. 3.85: RD is connected in parallel while RS is connected in series. RP shapes the crystal defects leading to less currents passing through the p-n junction. For well constructed solar cells, these defects are minor, such that RP is moderately high. The safety resistance RS is of a very small value.

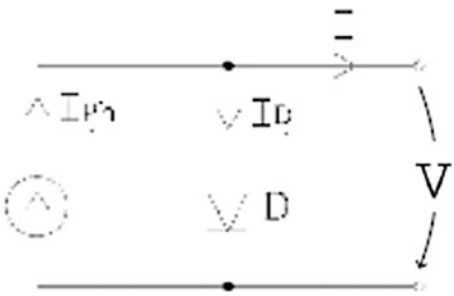


Fig. 3.84 Simplified conceptual model, with one diode, for a photovoltaic/solar cell

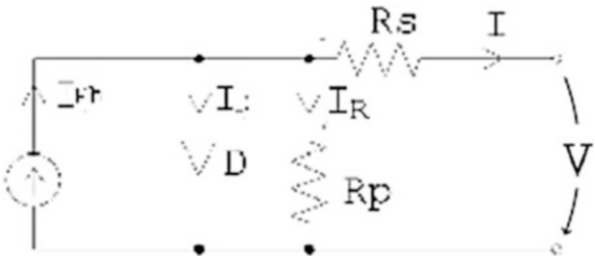


Fig. 3.85 Extended conceptual model, with one diode, for a photovoltaic/solar cell

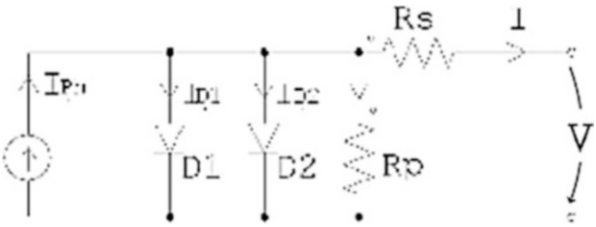


Fig. 3.86 Conceptual model with a two diodes for a photovoltaic/solar cell

An additional diode D2 is used in Fig. 3.86. The real diode effect is provided by this diode which is visible in the I(V) characteristics especially for $V < 0$ and partially for $V > 0$.

The model shown in Fig. 3.86 holds default *I(V)* characteristics and it works as follows.

$$I = I_{ph} - I_{D1} - I_{D2} - \frac{V + IR_s}{R_p} \tag{3.73}$$

Considering for I_{D1} and I_{D2} the same expressions as for Shockley diodes, the equality (3.73) becomes:

$$I = I_{ph} - I_{01}\left(e^{\frac{V+IR_s}{V_{T1}}} - 1\right) - I_{02}\left(e^{\frac{V+IR_s}{V_{T2}}} - 1\right) - \frac{V + IR_s}{R_p} \tag{3.74}$$

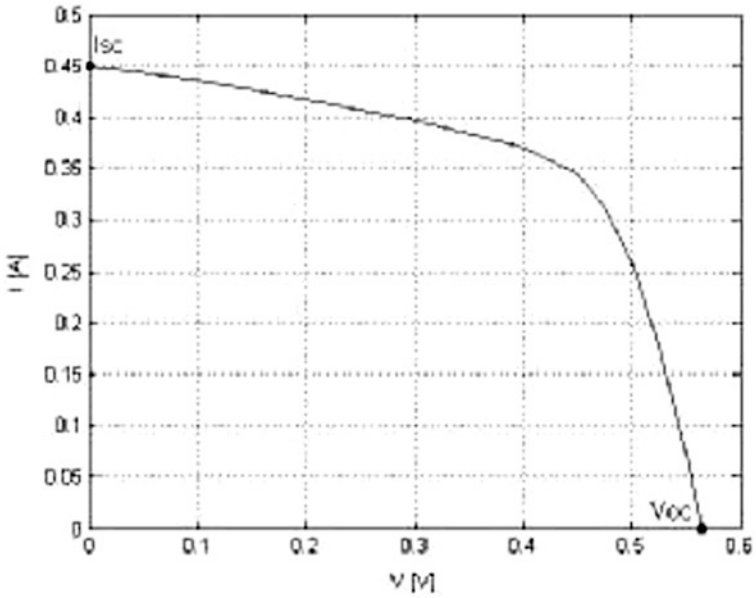


Fig. 3.87 Solar cell *I(V)* characteristic for constant solar radiation

The parameters such as I_{ph}, I_{01}, I_{02}, R_p, R_s are known in (3.74). Apart from these, two more parameters namely V_{T1} and V_{T2}. serve to model the non-linear influence of the temperature on the diode effect. V_{T1} and V_{T2} can be approximated by the relationship:

$$V_{T1} = \frac{n_1 \cdot k \cdot T_1}{q} \qquad V_{T2} = \frac{n_2 \cdot k \cdot T_2}{q} \tag{3.75}$$

where n_1 and n_2 are diode ideality factors for diode D_1 and D_2 respectively, k is the Boltzmann constant, T_1 and T_2 are the temperature of diode D_1 and D_2 respectively expressed in °K, and q is the elementary electrical charge. The functional part of the I(V) characteristic of the solar cell characteristic curve cold also "direct voltage" area is shown in Fig. 3.87. This in turn is equivalent to the values $V \geq 0, I \geq 0$ of the I-V curve and is bounded by the short circuit points ($V = 0$, $I = I_{SC}$), and open circuit point ($V = V_{OC}$, $I = 0$). The short circuit and open circuit points can be determined experimentally. With the assumption that V_{OC} and I_{SC} are known from measurements, the open circuit point and short circuit point imposed two links between the seven parameters.

$$0 = I_{ph} - I_{01}\left(e^{\frac{V_{OC}}{V_{T1}}} - 1\right) - I_{02}\left(e^{\frac{V_{OC}}{V_{T2}}} - 1\right) - \frac{V_{OC}}{R_p} \tag{3.76}$$

$$I_{SC} = I_{ph} - I_{01}\left(e^{\frac{I_{SC}R_S}{V_{T1}}} - 1\right) - I_{02}\left(e^{\frac{I_{SC}R_S}{V_{T2}}} - 1\right) - \frac{I_{SC}R_S}{R_p} \tag{3.77}$$

Assuming known the values of I_{01}, I_{02}, R_S, V_{T1} and V_{T2}, (3.76) and (3.77) make possible to obtain the parameters R_P and I_{ph} with the formulas

$$R_p = \frac{V_{OC} - I_{SC}R_S}{I_{SC} - I_{01}\left(e^{\frac{V_{OC}}{V_{T1}}} - e^{\frac{I_{SC}R_S}{V_{T1}}}\right) - I_{02}\left(e^{\frac{V_{OC}}{V_{T2}}} - e^{\frac{I_{SC}R_S}{V_{T2}}}\right)} \tag{3.78}$$

$$I_{ph} = I_{01}\left(e^{\frac{V_{OC}}{V_{T1}}} - 1\right) - I_{02}\left(e^{\frac{V_{OC}}{V_{T2}}} - 1\right) - \frac{V_{OC}}{R_p} \tag{3.79}$$

Hence, identification of expression given in (3.74), implies the determination of the following:

- seven parameters if the relations (3.75, 3.76, and 3.77) are not used;
- five parameters if only relation (3.75) is used;
- three parameters if all relations (3.75, 3.76, 3.77, 3.78, and 3.79) are used.

3.10.1 *Method of Determining the Parameters of Solar Cell Using Genetic Algorithms*

Identification of the expression in the I(V) characteristic curve leads to determination of seven parameters namely, I_{Ph}, I_{01}, I_{02}, R_S, R_P, V_{T1} and V_{T2}. Let us assume that R_P and I_{ph} can be calculated with formulas given by (3.78) and (3.79), while V_{T1} and V_{T2} are estimated with the formula given in (3.75). This leads to determination of remaining three parameters I_{01}, I_{02} and R_S. Let us refer this as Case I.

If relation (3.75) is not used, then five parameters such as V_{T1}, V_{T2}, I_{01}, I_{02} and R_S are determined. Let us refer this situation as *case II*.

Here Genetic Algorithms are applied to identify the parameters in both cases I and II. In Case I, an individual is defined by the vector (I_{01}, I_{02}, R_S), and in case II by the vector (V_{T1}, V_{T2}, I_{01}, I_{02}, R_S). The fitness function is framed as follows:

$$fitness = \sqrt{\sum_{k=1}^{12} c_k^2 \left(I_{meas}(V_k) - I_{calc}(V_k)\right)^2} \tag{3.80}$$

where $I_{meas}(V_k)$ is the measured current when $V = V_k$ and $I_{calc}(V_k)$ is the value calculated for voltage $V = V_k$.

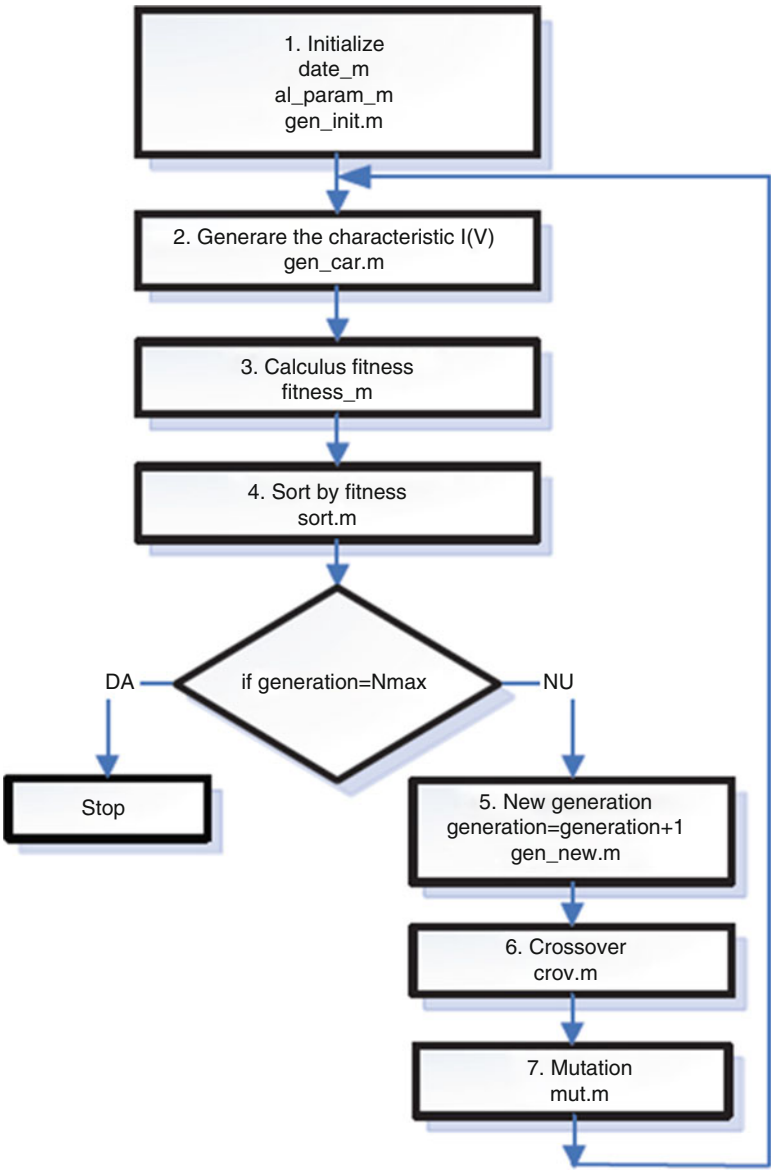


Fig. 3.88 Flowchart for parameter identification based on GA

The approach used in *case I* is depicted in the form of a flow chart as shown in Fig. 3.88. The MATLAB model of the program uses the following m files:

- *date.m* for the measured data {*Vk, Imeas(Vk)*};
- *al_param.m* for parameters as the cell surface or cell temperature;
- *gen_init.m* for the initial generation of genetic algorithms;

- *gen_car.m* for computing of the characteristic *Icalc(Vk)* of an individual;
- *fitness.m* for fitness computing of an chromosome / individual; descending order of fitness values;
- *gen_new.m* for a new generation;
- *crov.m* and *mut.m* for crossover, respectively mutation operations.

The steps illustrated in the flow chart are explained below:

Step 1: The parameters such as (V_{k_meas}, I_{k_meas}) are introduced; V_{T1} and V_{T2} are calculated using (3.75); With these inputs, the starting value for the parameters I_{01}, I_{02}, R_S are estimated. The initial population is formed in a random manner with 20 chromosomes in the form of vectors (I_{01}, I_{02}, R_S). The values are formed within a range of 20 % of the starting values. The parameters R_P and I_{Ph} of each chromosome in the population are computed using (3.78) and (3.79). The initial population is configured as the current generation.

Step 2: For each individual in the population, the characteristics $I_{calc}(V_{k\_meas})$ of current generation is calculated.

Step 3: The fitness function is evaluated for all the individuals in the current generation according to (3.80).

Step 4: Based on the fitness values, the individuals of the current population are ordered in descending order.

Step 5: Stopping condition – If the number of iterations does not exceed N_{max} on proceed to step 6, otherwise Stop.

Step 6: Based on the ordering with respect to the fitness function, a new population is formed with the best 10 individuals of the current population.

Step 7: The crossover operator is applied. The new population expands and continues to grow based on crossover operation. The growing of the new generation is continued using the crossover operation. The type of crossover operator is based on roulette wheel followed by a intermediary crossover to generate two offspring. The descendent value is generated randomly within the zone Z in Fig. 3.89. After applying crossover, R_P and I_{Ph} are calculated with formulas (3.78) and (3.79).

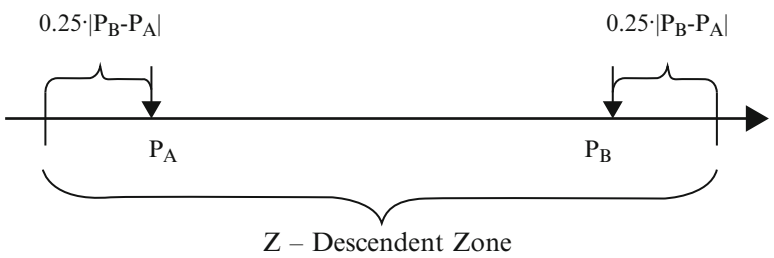


Fig. 3.89 Intermediary crossover

Step 8: The mutation operator is applied on a randomly chosen individual, by modifying one or more parameters. For each modified parameter the value of the descendent maintained in range of 20 % of the choice parameter. R_P and I_{Ph} are calculated as in step 7.

The same flowchart is followed for case II with changes in Step 1 of the algorithm. In Step 1, the starting value is estimated for the parameters such as V_{T1}, V_{T2}, I_{01}, I_{02}, R_S.

A maximum of 120-200 generations is required for the fitness function to stabilize. The stabilization of the fitness function is based on the following:

- the impact of intervention in panel because of the availability of several cells;
- the positive influence of higher surface of the cells from panel on the starting value estimations of parameters.

A method to identify the parameters of the external characteristics I(V) of the solar PV cells is discussed in this section. The methodology is based on genetic algorithms.

3.11 Application of Neuro-Fuzzy Technique for Prediction of Solar Radiation

The prediction is incredibly beneficial in solar powered energy applications since it permits to estimate solar data for locations where measurements aren't available. The developed artificial intelligence models predict the solar radiation time series better compared to the conventional procedures based on the clearness index. The forecasting ability of some models may very well be further enhanced with the aid of additional meteorological parameters. After having simulated a variety of structures of neural networks and trained using measurements as training data, the most beneficial structures were selected in order to evaluate their performance with regards using the performance of a neuro-fuzzy system. Since the alternative system, ANFIS neuro-fuzzy system was considered, because doing so combines fuzzy logic and neural network techniques which might be found in order to get more efficiency.

3.11.1 Neuro-Fuzzy Predictor (NFP)

The method for prediction system design based on fuzzy logic presents three characteristics: simplicity, a one-pass operation on the numerical input–output pairs to extract the rules and fast computational time. Suppose we are given N input output samples:

$$\left(x_1^{(p)}, x_2^{(p)}, \ldots\ldots, x_M^{(p)}; y^{(p)}\right) y^{(p)} \in R \qquad p = 1, 2, \ldots, N$$

where x_i = inputs, M = the number of inputs, y = the output.

This kind of is made up of the subsequent five steps as shown in Fig. 3.90. ANFIS is really a way of automatically tuning (Back propagation algorithm) TS-type fuzzy inference system based on some bunch of input–output data. The ANFIS predictor uses first-order TS-type systems, single output derived by weighted defuzzification and membership function type this is a generalized bell curve (MATLAB). Another method for solving forecasting tasks could be the combination of neural net predictors called committee machine. The committee machine is usually a static structure. In this particular technique the predictors are combined using a mechanism that does not involve the input. The outputs of four different predictors are nonlinearly combined to create an optimum prediction as shown in Fig. 3.91. Here, the nonlinear mechanism could be the neuro-fuzzy system ANFIS that is trained because of the output data from predictors. For simplicity, the NNP will likely be known as IL-HL-OL network, corresponding to the variety of nodes in each layer (IL: Input Layer, HL: Hidden layer and OL: Output layer). In iteration (time step) n, the nth training example is presented towards the network. The symbol Zi(n) refers back to the prediction signal appearing in the output of NNPi. The symbol Y (n) refers to the final prediction at iteration n.

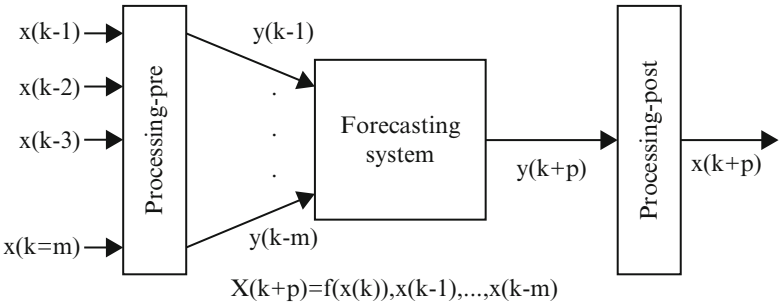


Fig. 3.90 Basic structure of the forecasting system

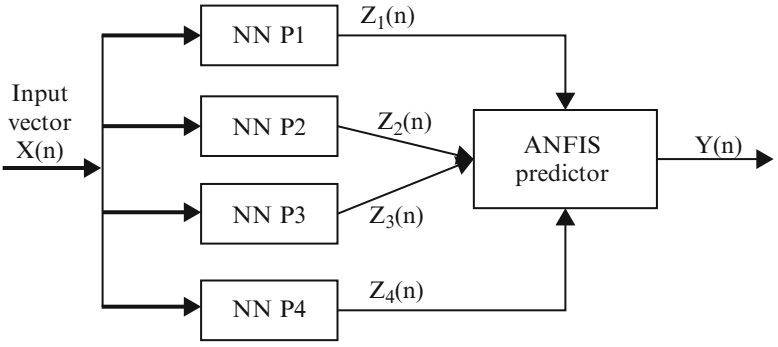


Fig. 3.91 A typical architecture of the non-linear mechanism of ANFIS model

Step 1: Divide the input and output spaces into fuzzy regions: Assume that the input xi and the output y lie in the domain intervals [xilmin, xilmax] and [ymin, ymax,] respectively. Each interval is then divided into 2z + 1 fuzzy regions and each region is assigned a symmetrical triangular fuzzy set.

Step 2: Data-generated fuzzy rules: The mth numerical data pair ((m) (m) (m) (m)) 1 2 M x,x,. . .,x;y is chosen from the training set. The membership grades are calculated for each data pair in the attributed fuzzy sets. The highest membership degree from the respective grades is chosen for each variable. A rule is formed based on the m training pair as follows:

Rm : IF x_1^m is A_1^m AND. . ..AND x_M^m is A_M^m THEN y^m is C^m

where A_i^m and C^m are fuzzy sets. These sets are attributed in the condition and conclusion parts of the rule and m is the index of the rule. Especially, li(i = 1,. . ., M) fuzzy sets are defined A_i^q, q = 1,. . .li for each input xi where lo represents the number of membership functions in the output space.

The fuzzy set A_i^m is one of the A_i^q's. Generally, in real applications we give, in the fuzzy sets linguistic names, like "big", "very positive".

Step 3: Assign a degree to each rule: Because there are usually many data pairs and as a consequence many rules generated, there exists high odds of conflict. That is certainly, rules who have the same IF part as well as a different then part. To settle this issue would be to assign a Truth Degree (TD) to each rule and accept merely the rule that has the biggest truth degree. We utilize following product strategy:

$$TD(m) = \mu_{A_1^m}\left(x_1^{(m)}\right).....\mu_{A_M^m}\left(x_M^{(m)}\right).\mu_{C^m}\left(y^{(m)}\right) \tag{3.81}$$

Step 4: Create a combined fuzzy rule base: The maximum number of rules that can be generated is l1 · l2 · . . . · lM. Through the 3rd step the reduction of the amount of rules is achieved. The generated rules determine a combined fuzzy rule base.

Step 5: Determine a mapping based on the combined fuzzy rule base: Determine the entire continuous fuzzy predictive model. Using the combined rule base with K fuzzy rules, the goods inference engine, the singleton fuzzifier as well as the center-average defuzzifier, the subsequent fuzzy technique is obtained:

$$y = f(x) = \frac{\sum_{j=1}^{K} y_c^j \left(\prod_{i=1}^{M} \mu_{A_i^j}(x_i)\right)}{\sum_{j=1}^{K} \left(\prod_{i=1}^{M} \mu_{A_i^j}(x_i)\right)} \tag{3.82}$$

where y_c^j is the centre of Cj. The output variable y is based on the inputs (x1, x2,. . ., xM). Methods to train object is always to minimize the mean square error between the predictions plus the real values. To find the most appropriate training algorithm, there is developed a small neural network 3-10-1, with sigmoid

activation function from the nodes in the hidden layer plus it was simulated for normalized data in the community 0.1–0.9 for 13 different algorithms using the neural network toolbox of MATLAB. Five error metrics were used in order to find the best algorithm: MSE, RMSE, AME, NDEI, ρ. The algorithms based on metrics used are as follows:

3.11.2 Error Metrics

$$MSE = \frac{1}{n}\sum_{k=1}^{n}\left(x(k) - \hat{x}\,(k)\right)^2 \tag{3.83}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(x(k) - \hat{x}\,(k)\right)^2} \tag{3.84}$$

$$AME = \frac{1}{n}\sum_{k=1}^{n}\left|x(k) - \hat{x}\,(k)\right| \tag{3.85}$$

$$NDEI = \frac{RMSE}{\sigma} = \sqrt{\frac{\sum_{k=1}^{n}\left(x(k) - \hat{x}\,(k)\right)^2}{\sum_{k=1}^{n}x^2(k)}} \tag{3.86}$$

$$\rho = \frac{\sum_{k=1}^{n}\left(x(k) - \overline{x}\right).\left(\hat{x}\,(k) - \overline{x}\right)}{\sqrt{\sum_{k=1}^{n}(x(k) - \overline{x})^2.\sum_{k=1}^{n}(\hat{x}\,(k) - \overline{x})^2}} \tag{3.87}$$

where, x(k) is the real value in time instance k, $\hat{x}\,(k)$ is the prediction of the model, and N is the number of test data used for prediction. It must be mentioned which the most characteristic error criterion showing the caliber of prediction was proved really do the correlation coefficient criterion (ρ). As the prediction improves, ρ is getting close to 1.

3.11.3 Neural Networks Training

After having trained and tested all of the different cases and structures of neural networks while using the different in normalization and type meteorological time-

series, they are compared good above error criteria in order to result in the most suitable neural predictor's structure for every single different time-series:

- To start with there was clearly chosen the best four neural networks for every different form of normalization.
- Next there are chosen the most beneficial four neural network predictors for every single different type of meteorological time-series.
- Finally there were made choice of the most effective neural network predictor for each and every different time-series just to be capable to compare its results with ANFIS or any other system created.

The best neural network predictors for any different time-series using Sigmoid Activation Function (SAF) are introduced.

Mean daily solar radiation: 5-15-1, normalized inside the range 0.1–0.9, using SAF.

Mean daily temperature: 5-10-1, normalized within the range 0.1–0.9, using SAF.

'Best' neural network predictors Vs ANFIS: To the training and testing of knowledge there seemed to be made a first TS type system composed of seven inputs. A combination of fuzzy logic with neural networks turned out to have good results about the daily solar radiation and temperature forecasting. Below there is presented an assessment is presented relating to the 'best' N.N. predictors and ANFIS, with the four different time series, using as criterion the metric that turned out to be probably the most accurate, which is the correlation coefficient (ρ). With the global prediction scheme we split the collected data into two categories. The set includes the temperature on the first 5 years (1991–1995), while the test set includes the rest of the 5-years (1996–2000). Choosing four inputs to make the TS method case-dependent. For that numerical fuzzy approach 125 rules were obtained. In the table of comparison the computational time has already been recorded.

3.11.4 Prediction Results with NNP

The key object would be to create many different structures of neural network predictors and train and test them out while using meteorological data available in order to summarize the top plus more efficient topology for forecasting solar radiation and temperature. For the meteorological time series utilized in these studies, one hidden layer is proper and sufficient. These cases of structures of neural networks are created and simulated:

Inputs: 2,3,5,7 previous daily measurements for the 12 different time series created (real and normalized data),
Number of hidden layers: 1,
Number of nodes of hidden layer: 2,3,5,10,15,
Output: 1 (One day prediction),

Activation Functions used in neurons:

Hidden layer: sigmoid, linear,
Output layer: linear.

After having trained and tested all the various cases and structures of neural networks with the different in normalization and type meteorological time-series, they've been compared good above error criteria so that the most appropriate neural predictors' structure for different time-series. The optimum neural network predictors for different time-series are introduced.

Daily mean solar radiation: 5-15-1, normalized in the range 0.1–0.9, using SAF.
Daily mean temperature: 5-10-1, normalized in the range 0.1–0.9, using SAF.

Concerning neuron-fuzzy predictors plus more specifically ANFIS it can be proved which the combined linguistic rules of fuzzy logic while using training algorithm utilised in neural networks, contribute in very qualitative prediction results, which approach the 'best' neural predictor's results. The structure from the TS fuzzy model is identified having a method which permits to determine the optimal structure on automatic manner while being based on the calculation in the incredible importance of the policies in the calculation on the model output. However, the linear parameters are estimated with the WLS algorithm along with the non-linear parameters are estimated while using the TS method. Additionally, an assessment is completed between identified TS fuzzy model. The obtained results show that this identified TS fuzzy model provides satisfactory performances. So, we conclude which the fuzzy systems can be used as an alternative approach to generate solar data for locations where measurements will not be available.

3.12 Summary

In this chapter, the basic concepts of soft computing techniques such as neural networks, fuzzy logic, genetic algorithms, hybrid model of neural networks and fuzzy logic are discussed. The importance of soft computing techniques such as neural networks, fuzzy logic and genetic algorithms in solar PV system is delineated. MATLAB/SIMULINK models have been developed for ANN, fuzzy and ANFIS based MPPT of Solar PV system, fuzzy based solar powered energy and optimization in solar PV array. The approach to predict solar irradiance using soft computing techniques and parameter estimation of Solar PV module using Genetic Algorithms is explained in detail.

Review Questions

1. State the different types of Artificial Neural Networks.
2. Mention the differences between biological and artificial neurons.
3. What are considerations to take into account while choosing a neural network?

4. Define uncertainty and imprecision with respect to fuzzy logic.
5. What do you mean by fuzzification?
6. Discuss the different methods available for defuzzification.
7. State the step by step procedure for Genetic Algorithms.
8. Define crossover and mutation probability.
9. What is the role of soft computing techniques in Solar PV systems?
10. Develop a MATLAB/SIMULINK model for MPPT control using Genetic Algorithms.
11. Discuss optimization in Solar PV systems using soft computing techniques.
12. Describe solar cell parameter extraction using Artificial Neural NetwoSIMULINK mode/SIMULINK model.
13. How is ANFIS applied to solar irradiance forecasting?
14. State and explain applications of solar PV using Fuzzy logic control.

Bibliography

Ahmed MM, Sulaiman M (2003) Design and proper sizing of solar energy schemes for electricity production in Malaysia. National Power and Energy Conference 2003, pp 268–271

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

Altas IH, Sharaf AM (2007) A novel GUI modeled fuzzy logic controller for a solar powered energy utilization scheme. The 13th international conference on emerging nuclear energy systems (ICENES2007), Istanbul, 3–8 June 2007

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

Banerjee S, Jeevananathan (2010) Modeling of PV array and performance enhancement by MPPT algorithm. Int J Comput Appl 7:35–39 (0975–8887)

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

El-Moghany MS, Hamed BM (2012) Two axes sun tracker using fuzzy controller via PIC16F877A. The 4th international engineering conference –towards engineering of 21st century, 2012

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

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

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

Karthika S, Rathika P, Devaraj D (2013) Fuzzy logic based maximum power point tracking designed for 10 kW solar photovoltaic system. Int J Comput Sci Manag Res 2(2):1421–1427

Koutroulis E et al (2001) Development of a microcontroller-based photovoltaic maximum power tracking control system. IEEE Trans Power Electron 16(1):46–54

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

Petcuţ F-M, Dragomir TL (2010) Solar cell parameter identification using genetic algorithms. Control Eng Appl Inf 12(1):30–37

Rahoma WA, Ali Rahoma U, Hassan AH (2011) Application of neuro-fuzzy techniques for solar radiation. J Comput Sci 7(10):1605–1611

Simoes MG, Franceschetti NN (1999) Fuzzy optimization based control of a solar array system. IEE Proc Electr Power Appl 146(5)

Usta MA, Akyaz Ö, Altaş İH (2011) Design and performance of solar tracking system with fuzzy logic controller. 6th international advanced technologies symposium (IATS'11), Elazığ, 16–18 May 2011

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

Wang F, Mi Z, Su S, Zhao H (2012) Short-term solar irradiance forecasting model based on artificial neural network using statistical feature parameters. Energies 5:1355–1370