CHAPTER 2

WIND SPEED FORECASTING AND CHARACTERIZATION

2.1 INTRODUCTION

A number of wind speed forecasting techniques are available in order to predict the uncertainty of the wind, which is a key to estimate available wind power generation. It is gaining more attention with the recent evolution of smart grids, which throws a challenge of integrating wind power into the grid. A few strategies have been suggested to give wind speed forecast. In the recent years there are lots of research happening to predict wind speed with several mathematical methods and biologically inspired computing techniques to reduce the prediction error.

Most of the wind prediction methods accessible in the papers give deterministic expectation, yet given the variability and instability of twist, such forecasts confine the utilization of the current methods for choice making under unsure conditions. Therefore, probabilistic prediction, which gives data on vulnerability connected with wind power estimating, is increasing expanded consideration. This chapter presents different artificial intelligence techniques and proposes a novel hybrid algorithm for wind speed forecasting. The contribution of this thesis is to develop an accurate, efficient,
and robust Wavelet Neural Network (WNN) model based on wavelet transform (WT) and artificial neural network.

2.2 ARTIFICIAL INTELLIGENCE TECHNIQUES

2.2.1 Feed Forward Backpropagation Algorithm (FFBP)

A FFBP (Deshmukh 2010) is a artificial neural system where associations between the units don't frame a coordinated cycle. In this system, the data moves in one direction, forward, from the information nodes, through the hidden nodes and to the output nodes. There are no cycles or circles in the system. This system can be utilized as a general function approximator. A FFBP comprises of two layers. The primary layer, or hidden layer, has a tan-sigmoid (tan-sig) actuation capacity, and the second layer, or output layer, has a linear activation function. Accordingly, the main layer restrains the output to a thin range, from which the linear layer can create all qualities. The output of each layer can be represented as in Equation (2.1)

\[ Y_{N \times 1} = f \left( W_{N \times M} X_{M \times 1} + b_{N \times 1} \right) \]  \hspace{1cm} (2.1)

where \( Y \) is a output vector from each of the \( N \) neurons in a given layer, \( W \) is a weight matrix that has the weights for each of the \( M \) inputs for all \( N \) neurons, \( X \) is a input vector, \( b \) is a bias vector and \( f(\bullet) \) is the activation function. The network is shown in Figure 2.1.
Cascade Forward Backpropagation Algorithm (CFBP)

Cascade Forward Backpropagation algorithms are like food forward systems, however incorporate a weight association from the input to every layer and from every layer to the progressive layers. For instance, a three-layer system appeared in Figure 2.2 has associations from layer 1 to layer 2, layer 2 to layer 3, and layer 1 to layer 3. The three-layer system likewise has associations from the input to each of the three layers. The extra associations may enhance the velocity at which the system takes in the coveted relationship.
Probabilistic Neural Networks (PNN) are utilized for classification problems as a result of simplicity of preparing and a sound statistical establishment in Bayesian estimation hypothesis. At the point when an input is given, the primary layer registers separations from the data vector to the training of input vectors and produces a vector whose components demonstrate how close the input is to training input. The second layer adds these values for every class of inputs to create as its net yield a vector of probabilities. At long last, a complete transfer function on the output of the second layer picks the most extreme of these probabilities, and produces a 1 for that class and a 0 for alternate classes. PNN predicts the estimation of one or more reliant variables, given the estimation of one or more autonomous variables. PNN can take an input vector $X$ of length $n$ ($X=[x_1, x_2, \ldots, x_i, \ldots, x_n]$) and gives the output vector $Y'$ of length $K$ ($Y'=[y_1, y_2, \ldots, y_k, \ldots, y_k]$), where $Y'$ represents the estimate of the actual $Y$. PNN does this by equating a new input pattern $X$ with a set of $K$ stored patterns $X_k$ ($K$ pattern units) for which the output $y_k$ is known. In every example unit, a standardized Gaussian function is connected to the distance measure (Euclidian standard) between the obscure data design $X$ and the training design $X_k$, which gives a measure of the separation or difference between two examples. The property of $h_k$ is that, its magnitudes for a stored pattern $X_k$ can be conversely in respect to its distance from the input design $X$, if the distance is zero the $h_k$ is a most extreme of solidarity. In the summation units, one unit must ascertain $N$, the aggregates of the results of $h_k$ and related known output $y_k$. It additionally figures $D$, the total of all $h_k$. Finally, the output unit partitions $N$ by $D$ to deliver the output $Y$. A simple probabilistic neural network is shown in Figure 2.3.
2.2.4 Generalized Regression Neural Network (GRNN)

A generalized regression neural network (Donald Specht 1991) is regularly utilized for function estimate and relapse problems. It has an outspread premise layer and an uncommon straight layer. The likelihood density function utilized as a part of GRNN is the Normal Distribution. Each training sample, \(X_j\), is used as the mean of a Normal Distribution.

\[
D_j^2 = (X - X_j)^T (X - X_j) \tag{2.2}
\]

\[
Y(X) = \frac{\sum_{i=1}^{n} Y_i \exp\left[\frac{-D_i^2}{2\sigma^2}\right]}{\sum_{i=1}^{n} \exp\left[\frac{-D_i^2}{2\sigma^2}\right]} \tag{2.3}
\]

The parameters \(X\) and \(Y\) are vectors. In system identification, the dependent variable, \(Y\), is the system output and the independent variable, \(X\), is the system input where \(\sigma\) is the smoothing parameter. The distance, \(D_j\), between the training illustration and the point of estimate, is used as a degree how well each training sample can characterize the position of prediction, \(X\).
If the distance, $D_j$, between the training example and the point of estimate is small, $\exp\left(-\frac{D_j^2}{2\sigma^2}\right)$ becomes big. For $D_j=0$, $\exp\left(-\frac{D_j^2}{2\sigma^2}\right)$ turn into unity and the point of assessment is characterized best by this training example. The Euclidian distance to all the other training samples is larger. A larger distance, $D_j$, causes the term $\exp\left(-\frac{D_j^2}{2\sigma^2}\right)$ to become lesser and consequently the impact of the remaining training samples to the estimate is comparatively lesser. The term $Y_j \exp\left(-\frac{D_j^2}{2\sigma^2}\right)$ for the $j^{th}$ training sample is the major one and provides very much to the forecast.

The spread value plays a main consideration in the contribution of the neurons in the GRNN system. A bigger spread prompts an extensive zone around the input vector where the outspread basis layer neurons will react with critical outputs. Accordingly, if spread is little the radial basis function is exceptionally steep, so that the neuron with the weight vector nearest to the input will have a much bigger yield than remaining neurons. The system has a tendency to react with the objective vector connected with the closest design input vector. As spread gets to be bigger the radial basis function’s incline gets to be smoother and a few neurons can react to an input vector. As spread turns out to be exceptionally bigger, more neurons add to the normal, with the outcome that the system function gets to be smoother.

2.2.5 $K^{th}$ Nearest Neighbour Network (KNN)

The KNN (Dudani 1976) is a prevalent decision for some genuine applications. In the KNN calculation, the Euclidean separation is calculated between the new element vector and every component vector from the preparation set. K-nearest neighbors (K being the quantity of neighbors) are then establish by dissecting the distance matrix. Euclidean separation metric
can be utilized to compute closeness. In the event that the output variable is a
downright variable, then the KNN calculation takes a vote among the K-
closest neighbors and picks the class voted in favor of by most of the
neighbors. On the off chance that the output variable is a constant variable,
the value of the output is the normal of the K-closest examples. KNN ascribes
missing qualities by the normal estimation of the K closest examples, as in
Equation (2.4)

\[ x_{ij} = \frac{1}{k} \sum_{k=1}^{K} x_{kj} \]  

(2.4)

where \( x_{ij} \) represents a missing value in the \( j \)th variable of the \( i \)th instance. \( K \) is
the number of nearest neighbours and \( x_{kj} \) is the value of the \( j \)th variable of the
\( k \)th nearest neighbour.

2.3 ANN BASED WIND SPEED PREDICTION MODEL

A model is created with the information, for example, temperature,
dampness, dew point, pressure and wind direction as inputs and the wind
speed as the objective utilizing the artificial neural system methods. To assess
the proposed model for wind velocity forecast, information sets are gathered
from a automated weather station and the study is done for 24 hours ahead.
The tested time arrangement utilized as a part of the model comprises of 1000
data altogether, comparing to 30 minutes mean information. The time series is
split into two, one is the training set with 952 samples utilized for the model's
training and the other is the test set that contains the remaining to be specific
36 tests which are utilized to confirm the accuracy of the created model. Some
of the data are purposely multiplied by a constant (*10, *100 or *1000) to
avoid storage of floating point numbers.
2.3.1 Wind Speed Prediction results using ANN Techniques

2.3.1.1 Training data

The training data set comprises temperature, humidity, dew point, pressure, wind direction and wind speed as shown in Figures 2.4 to 2.9 respectively.

Figure 2.4 Temperature vs Time (Training data)
Figure 2.5 Humidity vs Time  (Training data)

Figure 2.6 Dew point vs Time  (Training data)

Figure 2.7 Pressure vs Time  (Training data)
2.3.1.2 Test Data

The test data set comprises temperature, humidity, dew point, pressure, wind direction and wind speed as shown in Figures 2.10 to 2.15 respectively.
Figure 2.10 Temperature vs Time (Test data)

Figure 2.11 Humidity vs Time (Test data)

Figure 2.12 Dew point vs Time (Test data)
Figure 2.13 Pressure vs Time  (Test data)

Figure 2.14 Wind direction vs Time  (Test data)

Figure 2.15 Wind speed vs Time  (Test data)
2.3.2 Simulation Results

Figure 2.16 present the wind speed predicted by different ANN models and the actual wind speed.

![Wind speed prediction using different ANN techniques](image)

Figure 2.16 Wind speed prediction using different ANN techniques

The performance indices considered are the Mean Square Error (MSE), Mean Absolute Percentage Error (MAPE) and linear regression. These indices are found using the Equation (2.5) to Equation (2.7)

\[
MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \tag{2.5}
\]

\[
MAPE = \frac{1}{n} \left[ \sum_{i=1}^{n} \left| \frac{\hat{y}_i - y_i}{y_i} \right| \right] \times 100\% \tag{2.6}
\]

\[
\hat{y}_i = my_i + c \tag{2.7}
\]
where $y_i$ and $\hat{y}_i$ are the actual and predicted wind speeds at $i^{th}$ time interval, $m$ and $c$ are the scaling factor and the $y$-axis (prediction-axis) intercept of the linear relation, respectively. The performance measures are calculated for various ANN techniques and presented in Table 2.1.

From Table 2.1, it can be observed that MAPE for the five algorithms varies from 2.30 to 5.109, the MSE varies from 2.856 to 6.748 and the regression varies from 0.512 to 0.898. According to the forecasted errors, it is clearly seen that GRNN model is the optimal model because of the lowest MSE (2.856) and also better $R^2$ (0.898); Secondly, the PNN model; and then comes the KNN.

<table>
<thead>
<tr>
<th>ANN methods</th>
<th>Performance measures</th>
<th>FFBP</th>
<th>CFBP</th>
<th>PNN</th>
<th>GRNN</th>
<th>KNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>6.748</td>
<td>6.817</td>
<td>3.795</td>
<td>2.856</td>
<td>5.882</td>
<td></td>
</tr>
<tr>
<td>MAPE (%)</td>
<td>5.109</td>
<td>5.267</td>
<td>2.972</td>
<td>2.30</td>
<td>2.910</td>
<td></td>
</tr>
<tr>
<td>$R^2$</td>
<td>0.512</td>
<td>0.531</td>
<td>0.849</td>
<td>0.898</td>
<td>0.874</td>
<td></td>
</tr>
</tbody>
</table>

All these methods suffer from obtaining monolithic (rigid) global models for a time-series (Ulagammai et al 2007). To enhance the ANNs ability in learning the signals, the hidden patterns i.e. all the random variations in the data should be extracted. Hence, to address this a multi-resolution decomposition technique such as wavelet transform is introduced.

### 2.4 WAVELET NEURAL NETWORK

To have a reliable prediction of speed, it is necessary to process the original data to avoid the random variations problem. Hence, for the analysis
of highly non-linear and random load signal, wavelet transform approach is introduced. Wavelet change (WT) is a versatile windowing procedure. The flexible window size permits the utilization of long time interims when more exact low-frequency data is wanted and brief time intervals while seeking high-frequency data. Wavelet analysis adopts the concept of scale and relation between scale and frequency. It breaks the signals into shifted scaled variants of the first wavelet when contrasted with Fourier examination, where signals are broken into sinusoids of various frequencies. Notwithstanding that, the strategy utilizes a time scale locale rather than a time-frequency region.

### 2.4.1 Discrete Wavelet Transform

The discrete wavelet transform (DWT) calculation is fit for delivering coefficients of fine scales for catching high frequency data, and coefficients of coarse scales for catching low frequency information. If we consider a mother wavelet function \( \Psi \) and for a given signal, a scientific representation for a DWT can be expressed as,

\[
f(t) = \sum_j C_{j0} \phi_{j0} + \sum_k \sum_j W_{jk} 2^{j/2} \psi(2^j t - k)
\]

where \( j \) is the dilation or level index, \( k \) is the translation or scaling index, \( \phi_{j0} \) is a scaling function of coarse scale coefficients \( C_{j0} \), \( W_{jk} \) is the scaling function of detail (fine scale) coefficients and all functions of \( \Psi(2^j t-k) \) are orthonormal, the additive components of the above equation signifies the approximation and the detail respectively.

DWT is a commonly used technique that is applied for signal compression. The reason for the popularity is its ability to set a large portion of the coefficients to zero without any substantial loss of information. In
addition, if additional properties other than the stationary properties of a signal are desired, DWT would definitely be a superior option as compared with the conventional method of Fourier transform.

### 2.4.2 Decomposition of Signal

For many signals, the low-frequency content is the most important part. This gives the identity of the signal. The high-frequency content, on the other hand, imparts flavour or nuance. In wavelet analysis, there are approximations (A) and detail (D) coefficients. The approximation coefficients are the high-scale, low-frequency components of the signal. The detail coefficients are the low-scale, high-frequency components. Fig 2.17 illustrates the filtering process, at its most basic level.

![Wavelet Decomposition Diagram](image)

**Figure 2.17 Wavelet Decomposition**

The original signal, S, passes through two complementary filters and emerges as two signals. The decomposition process can be iterated, with
successive approximations being decomposed in turn, so that one signal is broken down into many lower resolution components. This is called the wavelet decomposition tree.

2.4.3 Wavelet Reconstruction

The process of building the original signal from low and high frequency components without loss of information is called as reconstruction, or synthesis as shown in Figure 2.18. The mathematical manipulation that affects synthesis is called the inverse discrete wavelet transform (IDWT).

![Three-level wavelet reconstruction tree](image)

**Figure 2.18 Wavelet Reconstruction**

Wavelet decomposition involves filtering and down sampling, whereas the wavelet reconstruction process consists of up sampling and filtering. Up sampling is the process of lengthening a signal component by inserting zeros between samples. Figure 2.19 shows the procedure involved in wavelet analysis.
2.5 WNN BASED FORECASTING MODEL

The wind velocity data is first decomposed into several sub-serials using wavelet, which show the different frequency characteristics of the wind speed. To forecast each sub-serial, each neural network is constructed. The final wind speed forecasting result can be obtained by summing up all the sub-serial forecasting results. A given signal $s(t)$ is decomposed into numerous other signals with different levels of resolution by Dyadic Wavelet Transform. To compute dyadic wavelet transform, a mother wavelet is first chosen and then dilated by the powers of 2. The dyadic wavelet transform of $s(t)$ is defined as follows:

$$s(m,n) = 2^{m/2} \int \psi(t) \varphi^\ast\left(\frac{t-n2^m}{2^m}\right) dt$$  \hspace{1cm} (2.9)$$

where the * denotes a complex conjugate, $m$ and $n$ are scale and time-shift parameters, respectively, and $\psi\ (t)$ is a given basis function (mother wavelet). The dyadic wavelet transform is implemented using a multi-resolution pyramidal decomposition technique.

The WNN forecasting procedure shown in Figure 2.20 comprises a development of a preliminary forecast model followed by pre-signal

![Figure 2.19 Wavelet decomposition - reconstruction](image-url)
Stage 1: Pre-signal Processing

In pre-signal processing, authentic wind speed information are given to the proposed model as time-series signals. The Non-decimated Wavelet Transform (NWT) is utilized as the pre-signal processor and relying upon the chosen reduction level, the individual time-series signals are decomposed into various wavelet coefficients. These disintegrated coefficients are then standardized and given as inputs to the signal predictor (Neural Networks) for either training or estimation.

Stage 2: Signal Prediction

ANNs are utilized for signal prediction as a part of the estimate model. The number of ANNs required for the model is dictated by the number of wavelet coefficient signals at the output of the pre-processor. For every wavelet coefficient signal (counting the guess part), one ANN is required to perform the related forecasting.
Stage 3: Post-signal Processing

In post-signal processing, the similar wavelet procedure and resolution level as mentioned in pre-signal processing are used. In this stage, the outputs from the signal predictor (ANNs) are joined to form the final predicted output. This is achieved by adding all the forecasted wavelet coefficients.

2.6 SIMULATION RESULTS

The proposed WNN model is utilized for simulating the similar set of data as in section 2.3 and the results are compared with the methods explained in the section 2.3. The number of ANNs depends on the wavelet family and the resolution level. The wavelet family considered is db2 with resolution level of 2 and hence three ANNs are used. The individual ANN is constructed based on the wavelet coefficients. In general, db2 wavelet family produces 4 filter coefficients for single decomposition. By linear convolution number of approximation (A) and detail (D) coefficients are given by \((m+n-1)/2\) where \(m\) is the input data size and \(n\) is the filter coefficient size.

In this case, the input data size for one sample is 15 and db2 family is chosen. Hence the number of A and D in the first level decomposition are,

\[
e.g \quad A = D = (15+4-1)/2 = 9
\]

For second level decomposition, the number of coefficients are :

\[
(9+4-1)/2 = 6;
\]

```
15 (input)
/  \  
9   9
/  \  
6   6
```
The decomposition levels are represented in suffixes. Thus the input neurons are 6, 6 and 9 for ANN1, ANN2 and ANN3 respectively.

Ten hidden neurons and one output neuron is selected for all neural networks. The ANN and WNN parameters are listed in Table 2.2.

**Table 2.2 ANN and WNN parameters**

<table>
<thead>
<tr>
<th>ANN Parameters</th>
<th>WNN Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Input neurons</td>
<td>15 Wavelet family db2</td>
</tr>
<tr>
<td>Number of Hidden neurons</td>
<td>10 Resolution level 2</td>
</tr>
<tr>
<td>Number of output neurons</td>
<td>1 Number of neural networks constructed(for WNN) 3</td>
</tr>
<tr>
<td>Accelerating factor</td>
<td>0.1 Number of input neurons for ANN1 6</td>
</tr>
<tr>
<td>Momentum bias coefficient</td>
<td>0.25 Number of input neurons for ANN2 6</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Number of input neurons for ANN3 9</td>
</tr>
<tr>
<td></td>
<td>Number of output neurons for all ANNs 1</td>
</tr>
</tbody>
</table>

Figure 2.21 presents the wind speed predicted by WNN. The performance measures considered are MSE, MAPE and $R^2$. These measures are calculated using the Equation (2.5) to Equation (2.7) and the results are presented in Table 2.3 and shown in Figure 2.22.
Figure 2.21 Wind speed Forecasting using WNN

Figure 2.22 Comparison of WNN and ANN techniques
Table 2.3 Comparison of performance measures for different ANN techniques and WNN

<table>
<thead>
<tr>
<th></th>
<th>FFBP</th>
<th>CFBP</th>
<th>PNN</th>
<th>GRNN</th>
<th>KNN</th>
<th>WNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>6.748</td>
<td>6.817</td>
<td>3.795</td>
<td>2.856</td>
<td>5.882</td>
<td><strong>2.778</strong></td>
</tr>
<tr>
<td>MAPE (%)</td>
<td>5.109</td>
<td>5.267</td>
<td>2.972</td>
<td>2.30</td>
<td>2.910</td>
<td><strong>1.5075</strong></td>
</tr>
<tr>
<td>$R^2$</td>
<td>0.512</td>
<td>0.531</td>
<td>0.849</td>
<td>0.898</td>
<td>0.874</td>
<td><strong>0.8034</strong></td>
</tr>
</tbody>
</table>

From Table 2.3 it can be noted that WNN is the optimal model because of its lowest MSE, MAPE and $R^2$ and then comes GRNN followed by PNN and KNN respectively. Thus WNN is observed to be the better technique for forecasting the wind speed compared to the other ANN techniques. The results confirm that the performance of the proposed hybrid model is highly efficient for probabilistic forecasting. Short-term forecasting of wind power with a higher rate of accuracy is extremely important for the power system operators as they face challenges associated with varying wind power production with the increasing installed wind power penetration. Based on the presented simulation results, the proposed forecasting framework demonstrates a significant improvement over the other ANN models. MATLAB R2008 is used for developing the forecasting models. The average computation time required by the proposed hybrid model for short-term (hour-ahead) wind speed forecast is around 3 minutes on a Pentium IV 2.4 GHz processor.

2.7 WIND SPEED CHARACTERIZATION

In general, wind power prediction methods are categorized into two groups: physical and statistical. The first one implies physical considerations such as topography, terrains, local temperature and pressure to estimate the wind field more accurately and, subsequently, the energy potential. The later one, on the other hand, use statistical models in order to establish the
relationship between power and other variables as well as their historical and forecasted values (Ramnez and carta 2005) (Zaharim et al 2009). Weibull distribution provides better fit to probability distributions compared to Rayleigh model and analyzes the wind speed data by using statistical distributions (Xiao et al 2006) (Akpinar EK and Akpinar S 2004). The Weibull distribution (named after the Swedish physicist W. Weibull, who applied it when studying material strength in tension and fatigue in the 1930s) provides a close approximation to the probability laws of many natural phenomena. It has been widely used in reliability and survival analysis due to its flexible shape and ability to model a wide range of failure rates (Akpinar EK and Akpinar S 2005) (Azad et al 2010). For more than half a century the Weibull distribution has attracted the attention of statisticians working on theory and methods as well as various fields of statistics (Celik 2004).

The Weibull distribution function with a shape factor of 2 is otherwise called the Rayleigh distribution. The benefits of the Weibull distribution are noted as: 1) it is a two parameter circulation, which is more broad than the single parameter Rayleigh distribution, however less muddled than the five-parameter bi-variate typical normal distribution. 2) it is already proven that the observed data of wind speeds follows a Weibull distribution; and 3) if the \( k \) and \( c \) parameters are known at one height, a methodology exists to find the corresponding parameters at another height. The probability density function for a Weibull distribution is given by,

\[
f_v (v) = \left( \frac{k}{c} \right) \left( \frac{v}{c} \right)^{k-1} e^{-(v/c)^k}
\]  

(2.10)

For example, the Weibull distribution curve of a random variable for different scale factors with a given shape factor of 3 is shown in Figure 2.23.
The forecasted wind speed fits the Weibull distribution curve and is given in Figure 2.24. The cut-in speed, cut-out speed and rated speed in the Weibull distribution are considered as 2 m/s, 25 m/s and 15 m/s respectively. The shape factor and scale factor of Weibull distribution for the predicted wind speed is obtained as 2.24 and 8.83 respectively.
2.8 WIND POWER CALCULATION

Once the uncertain nature of the wind is characterized as a random variable, the output power of the Wind Energy Conversion Systems (WECS) may also be characterized as a random variable through a transformation from wind speed to wind power. Ignoring minor non-linearities, the output of the WECS with a given wind speed input may be stated as (Hetzer et al 2008),

\[ W = 0 \text{ for } v < v_1 \text{ and } v > v_o \]  \hspace{1cm} (2.11)

\[ W = W_r \frac{(v - v_l)}{(v_r - v_l)} \text{ for } v_l \leq v \leq v_r \]  \hspace{1cm} (2.12)

\[ W = W_r \text{ for } v_r \leq v \leq v_o \]  \hspace{1cm} (2.13)

The wind speed has the Weibull distribution and it has to be converted as wind power distribution. This is achieved by linear transformation given below (Peebles and Probab 2001):

\[ W = T(V) = aV + b \]  \hspace{1cm} (2.14)

where T is the transformation, a and b are the linear transformation parameters, W is wind power random variable, V is wind speed random variable. After the transformation, the Weibull probability density function (PDF) of wind power (W) takes the following form,

\[ f_w(w) = \left( \frac{k l v}{c} \right) \left( \frac{(1+\rho) v_l}{c} \right)^{k-1} e^{-\left( \frac{(1+\rho) v_l}{c} \right)^k} \]  \hspace{1cm} (2.15)

Where \( k \) shape factor in Weibull distribution; \( c \) Scale factor in Weibull distribution; \( v \) is the wind speed;
\( \rho = w/w_r \) ratio of wind power output to rated wind power; and
\( l = (v_r - v_i)/v \) ratio of linear range of wind speed to cut-in wind speed.
The wind speed distribution is converted into wind power distribution using Equations (2.14) and (2.15). Figure 2.25 depicts the wind power PDF \( f_w(w) \) curve and the wind power is predicted for each hour by approximating the area under the wind power PDF curve. It is assumed that the area for each hour in the distribution is trapezoidal and hence area of the trapezoid is used.

![Wind Power pdf curve](image)

**Figure 2.25 Wind Power pdf curve**

### 2.9 CONCLUSION

Because of the variable nature of wind power generation, wind speed forecasting has become one of the challenging assignments to the experts. 60 minutes ahead forecast is suitable for small power system operations and one hour power markets. Be that as it may, one day expectation is fitting for interconnected power system operations, for
example, unit commitments, thermal generators planning, and additionally one day power markets. Therefore, 24 hours prediction model is created for wind speed. New improved ANN prediction tools such as CFBP, PNN, KNN, GRNN and WNN are proposed for 24 hours ahead prediction of average wind speed. The test results from the models reveal the performance and the precision of the used neural network algorithms. The hybrid WNN model has proven to be an effective way for wind speed forecasting. From the forecasted wind speed, wind power prediction is carried out using linear transformation. The predicted wind power is further used for further operations of power system such as economic scheduling and optimal power flow and they are discussed in detail in the forthcoming chapters.