Empirical prediction limit estimation methods for feed-forward neural networks

To link to this article: DOI: 10.1080/03081070600782170
URL: http://dx.doi.org/10.1080/03081070600782170
Empirical prediction limit estimation methods for feed-forward neural networks

RATNA BABU CHINNAM†* and PUNDARIKAKSHA BARUAH‡†

†Department of Industrial & Manufacturing Engineering, Wayne State University, 4815 Fourth Street, Detroit, MI 48202, USA
‡Department of Industrial & Manufacturing Engineering, Wayne State University, 4815 Fourth Street, Detroit, MI 48202, USA

(Received 23 February 2003; in final form 19 January 2006)

Feed-forward neural networks (FFNs) have gained a lot of interest in the last decade as empirical models for their powerful representational capacity, non-parametric nature and multivariate characteristics. While these neural network models focus primarily on accurate prediction of output values, often outperforming their statistical counterparts in dealing with sparse date sets, they usually do not provide any information regarding the confidence with which they make these predictions. Since prediction limits (PLs) indicate the extent to which one can rely on predictions for making future decisions, it is of paramount importance to estimate these limits. Two empirical PL estimation methods for FFNs are presented. The two methods differ in one fundamental aspect: the method employed for modeling the properties of the neural network model residuals. While one method uses a local approximation scheme, the other utilizes a global approximation scheme. Simulation results reveal that both methods have their relative strengths and weaknesses.

Keywords: Prediction limits; Feed-forward neural networks; Self-organizing maps; Radial-basis functions; Estimation

1. Introduction

Feed-forward neural networks (FFNs) have gained a lot of interest as empirical models for their powerful representational capacity, non-parametric nature and multivariate characteristics. As expected, most FFNs with non-linear nodal functions have been proved to be universal approximators (Haykin 1999). While the application for FFNs discussed in literature are significantly abound, they have shown most promise for classification, regression, function approximation, time-series forecasting and dynamic systems modelling problems. Literally thousands of papers have been authored in the literature regarding the theory and application of FFNs for such applications. One of the most common applications is the regression problem. While FFN models focus primarily on accurate prediction of output values, often outperforming their statistical counterparts in dealing with sparse date sets, they usually do
not provide any information regarding the confidence with which they make these predictions. Since prediction limits (PL) indicate the extent to which one can rely on predictions for making future decisions, it is of paramount importance to estimate these limits.

Locally generalizing FFNs such as radial-basis function (RBF) networks and cerebellar model arithmetic computer (CMAC) networks have a naturally well-defined concept of "local neighbourhoods". Training data and testing data are considered "local" to a test point if they are within a limited region around the test point. Such networks have been extended in the literature to include statistical PLs on network predictions. For example, the validity index (VI) network derived from RBF networks performs multivariate regression and calculates statistical error bounds for its predictions (Leonard et al. 1992). In contrast, globally generalizing FFNs such as the very popular multi-layer perceptron (MLP) network with sigmoidal nonlinearities, proven to be very effective for regression and classification, are ill-defined with regard to this concept of a local neighbourhood. Hence, globally generalizing FFNs do not naturally lend themselves for estimation of PLs. Techniques proposed in the literature, including the confidence interval prediction technique proposed by Chryssolouris et al. (1996), typically make the strong assumption of constant variance of data in the output space. What are needed are robust PL estimation methods for both global approximation FFNs as well as local approximation FFNs. This paper presents two empirical methods for estimation of PLs for FFNs employed for regression. Actually, the proposed methods are compatible with any nonparametric regression method (neural network based or otherwise).

The paper is organized as follows: Section 2.1 and Section 2.2 provides a brief overview of MLP and RBF, respectively. Section 2.3 discusses the properties of self-organizing maps (SOMs) and their inherent ability to partition input spaces, thereby, leading to the definition of a local-neighbourhood. Section 3 briefly discusses the difference between local and global function approximation. Section 4 outlines a PL estimation method proposed by Chinnam and Ding (1998) that employs SOMs. The method in essence constitutes a local approximation scheme to construct PLs. Section 5 introduces a new global approximation method for PL estimation. Section 6 evaluates the performance of the proposed PL estimation methods using test data sets. Finally, Section 7 offers some concluding remarks.

2. Background on feed-forward networks

This section provides a concise description of the two most popular types of FFNs often employed for regression and functional approximation problems, namely, MLP networks and RBF networks. The section also briefly outlines SOMs.

2.1 Multi-layer perceptron networks

A typical multi-layer perceptron neural network with an input layer, an output layer and one hidden layer is shown in figure 1 (referred to normally as a two-layer network; input layer is not counted). The most popular non-linear nodal function for MLP networks is the sigmoid [unipolar \( \gamma(x) = \frac{1}{1 + e^{-x}} \) where \( 0 \leq \gamma(x) \leq 1 \) for \( -\infty < x < \infty \) and bipolar \( \gamma(x) = \frac{(1 - e^{-x})}{(1 + e^{-x})} \) where \( -1 \leq \gamma(x) \leq 1 \) for \( -\infty < x < \infty \)]. The illustrated network carries a single output in the output layer and can be extended to multiple outputs without loss of generality. Each layer of the network can be represented by the operator

\[ N_l[x] = \Gamma[w_l^0 x]. \]
and the input–output mapping of this two-layer the MLP network can be represented by

\[
y = N[x] = \Gamma[\bf{w}^{(2)}]\Gamma[\bf{w}^{(1)}x] = N_2N_1[x]. \tag{2}\]

The objective is to determine an adaptive algorithm or rule which adjusts the weights of the network (i.e. the model parameters \(\bf{w}^{(1)}\) and \(\bf{w}^{(2)}\)) based on a given set of input–output data pairs. Readers can see (Haykin 1999) for information regarding training algorithms.

It has been shown in Hornik et al. (1989), that even an MLP network with just one hidden layer and arbitrarily large number of nodes can approximate any continuous function \(f \in (R^N, R^M)\) over a compact subset of \(R^N\) to arbitrary precision. This provides the motivation to use MLP networks for regression and function approximation problems. Numerous studies have revealed that these models routinely outperform their statistical counterparts for such tasks as classification, regression and function approximation.

### 2.2 Radial-basis function networks

As stated earlier, FNNs have been widely studied and used for approximating arbitrary functions of a finite number of real variables. Most of these FFNs deal with cases in which the hidden layer non-linearity is a sigmoid (i.e. MLP networks). Traditional training algorithms for MLP networks such as the back-propagation algorithm require long computation times for training and the incremental adaptation approach of back-propagation can be shown to be susceptible to false minima (Specht 1990). It has been proven that FFNs that incorporate RBFs as nodal functions (i.e. RBF networks) are also capable of universal approximation with one hidden layer (Park and Sandberg 1991). In addition, RBF networks can often be trained more rapidly than MLP networks (Moody and Darken 1989). However, MLP networks tend to be more compact in nature and generally demonstrate superior generalization capability over RBF networks.

A typical RBF network consists of three layers of nodes with successive layers fully connected by feed-forward arcs, as shown in figure 2. The connections between the input and the hidden layers are unweighted and the transfer functions at the hidden layer nodes are RBFs. The illustrated network carries a single output in the output layer and can be extended to multiple outputs without loss of generality. In what we present, we use the particular architecture and training scheme described in Moody and Darken (1989) along with a Gaussian transfer function

\[
a_h = e^{-||x-\mu_h||^2/\sigma_h^2}, \quad h = 1, \ldots H \tag{3}\]
where \(a_h\) is the output of node \(h\) in the hidden layer given the input vector \(x\). Each RBF node has \(k + 1\) internal parameters: \(\mu_h\), the position of the centre of the radial unit in the input space, and \(\sigma_h\), the unit width. Each RBF unit has a significant activation over a specific region determined by \(\mu_h\) and \(\sigma_h\). The value of the output, \(y\), is given by

\[
y = \sum_{h=1}^{H+1} w_{1h} a_h
\]

A bias node is represented by \(a_{H+1} = 1\). A three-step polynomial time method presented in Moody and Darken (1989) is used to determine the parameters \(\mu_h\), \(\sigma_h\) and \(w_{1h}\). First, unit centres \(\mu_h\) are determined by \(k\)-means clustering (MacQueen 1967). Then, a \(P\)-nearest neighbour heuristic is used to determine the unit widths \(\sigma_h\). Finally, second-layer weights are determined by linear least squares. For a more in depth treatment of the topic of RBF networks, see Haykin (1999).

2.3 Background on self-organizing maps

Unlike MLP and RBF networks that fall into the class of supervised learning algorithms where a “teacher” is present to correct the model behaviour, SOMs employ unsupervised learning algorithms. In general, SOMs adaptively transform an incoming signal pattern of arbitrary dimension into a one- or two-dimensional discrete map in a topological ordered fashion (Kohonen 1982). A brief description of the SOM from Haykin (1999) follows to illustrate the ability of these maps to create the necessary concept of a local neighbourhood for estimation of PLs using local approximation schemes.

Let, \(X\) denote a spatially continuous input space, the topology of which is defined by the metric relationship of the vectors \(x \in X\). Let \(A\) denote a spatially discrete output space, the topology of which is endowed by arranging a set of neurons \((N)\) as the computation nodes of a lattice. A non-linear transformation \(\Phi\) can be defined, which maps the input space \(A\) onto the output space \(A\), as shown by \(\Phi\): \(X \rightarrow A\). This may be viewed as an abstraction that defines the location of a winning neuron \(i(x)\) developed in response to an input vector \(x\). Given an input vector \(x\), the SOM algorithm proceeds by first identifying a best-matching or winning neuron \(i(x)\) in the output space \(A\), in accordance with the feature map \(\Phi\). The synaptic weight vector \(w_i\) of neuron \(i(x)\) may then be viewed as a “pointer” for that neuron into the input space \(X\).

The ability of a self-organizing map \(\Phi\) to: (1) provide a good approximation of the input space; (2) exhibit topological ordering (spatial location of a neuron in the lattice corresponds
to a particular domain or feature of input patterns); and (3) provide density matching† (regions in the input space \(X\) that have high probability density, \(pdf(x)\), are mapped onto larger domains of output space \(A\)), makes it an excellent candidate to introduce the definition of a local neighbourhood for estimation of PLs for FFNs using local approximation schemes. This does not preclude the use of other clustering algorithms. However, traditional clustering algorithms such as those driven by the standard \(k\)-nearest-neighbor rule (Moody and Darken 1989) do not exhibit density matching property and may lead to neighbourhoods with significantly different populations (i.e. number of data points). This will influence the accuracy of estimation of covariance matrices for residuals in these neighbourhoods, critical for accurate PL estimation, as discussed later.

3. Function approximation: local and global methods

In the context of function approximation, there are occasions when a single model (often termed “global”) is not appropriate for estimating the underlying model \(f\) over the entire input space. To qualify for a good approximation, a representation must be able to conform to validation. However, if \(f\) is highly nonlinear, there is no guarantee that any single global model will approximate \(f\) efficiently. Within a local approximation approach, the training dataset is partitioned into several subsets (that may have non-empty intersections). A local model is estimated from each subset and the final model results from aggregation of the local models. Among local and global approximation techniques, global method is convenient, but unless the representation is well matched to the map \(f\), it may not produce a good representation, especially since many of the standard nonlinear representations undergo an explosion of parameters as dimensionality increases (Farmer and Sidorowich 1988). Preference of a local method over global method would depend on the problem domain, degree of nonlinearity in the underlying function, size the dataset, and so on.

4. Estimation of PLs using SOMs—a local approximation method

What follows is a concise description of the PL estimation method proposed by Chinnam and Ding (1998) that employs SOMs. Let \(M\) represent the total number of training patterns spanning the entire input space \(X\) and \(N\) denote the number of neurons in the SOM. Let \(M_j\), the “membership” of neuron \(j\) in the discrete output space \(A\), represent the subset of training patterns from input space \(X\) that activate it. This is shown by:

\[
 i(x) = j \quad \text{for all} \quad x \in M_j, \quad j = 1, 2, \ldots, N. \quad (5)
\]

It is also true that the sum of the memberships of the SOM neurons in the lattice output space must equal the total number of training patterns for the SOM, as given by:

\[
\sum_{j=1}^{N} M_j = M. \quad (6)
\]

†To enhance the density matching property of the SOM algorithm, we advocate the utilization of conscience into the SOM algorithm, as is proposed by DeSieno (1988).
The three properties exhibited by SOMs (discussed earlier) provide the motivation to utilize the SOM to “break” the input space $X$ into $N$ distinct regions (denoted by $X_j$) that are mutually exclusive and hence satisfy the following relationship:

$$\sum_{j=1}^{N} X_j = X.$$  

(7)

All the patterns from any given distinct region $X_j$, when provided as input to the feature map $\Phi$, will activate the same output neuron $j$. This is shown by:

$$i(x) = j \quad \text{for all } x \in X_j, \quad j = 1, 2, \ldots, N$$  

(8)

Thus, using SOMs, one can introduce the concept of a “local neighbourhood,” the resolution depending on the number of neurons ($N$) in the discrete output space. Utilizing this definition of a local neighbourhood, input signal patterns can be associated unambiguously with one of the distinct regions $X_j$. Assuming that a FFN is being used for regression or function approximation, an estimate of the covariance matrix for the FFN model residuals within the domain of region $X_j$ is given by Chryssolouris et al. (1996):

$$\text{Cov}_j = \begin{bmatrix} S_{11} & S_{12} & \cdots & S_{1O} \\ S_{21} & S_{22} & \cdots & S_{2O} \\ \vdots & \vdots & \ddots & \vdots \\ S_{O1} & S_{O2} & \cdots & S_{OO} \end{bmatrix}$$  

(9)

where:

$$S_{pq} = \frac{1}{(M_j - 1)} \sum_{k=1}^{M_j} (E_{kp})(E_{kq}),$$

and denotes the covariance between output variables $p$, $q$; $E_{kp}$ denotes the FFN model residual for output variable $p$ for pattern $k$; $O$ denotes the number of output variables predicted by the FFN.

Assuming that the residuals are independent and Gaussian distributed with a constant covariance matrix over the domain of any region but varying from domain to domain, the $(1 - \alpha)$ quantile is given by the point $y$ satisfying the following condition Chryssolouris et al. (1996):

$$(y - \mu_j)^T\text{Cov}_j^{-1}(y - \mu_j) \leq \chi^2_O(\alpha)$$  

(10)

where $\chi^2_O(\alpha)$ denotes the $(1 - \alpha)$ quantile of the Chi-Square distribution with $O$ degrees of freedom, $\mu_j$ denotes the mean residual vector for domain $X_j$ and $\text{Cov}_j^{-1}$ is the inverse of the matrix $\text{Cov}_j$.

Simulation experiments have revealed that the residuals in distinct neighbourhoods do tend to exhibit a Gaussian distribution as long as the true noise in the overall data set is Gaussian (Chinnam and Ding 1998). In fact, if the output variables are assumed to be independent, one can even relax the Gaussian residual assumption for distinct neighbourhoods and arrive at an upper limit on the true PLs for each of the output variables.
based on the Chebyshev’s theorem (Moody and Darken) as follows (Chinnam and Ding 1998):

\[ P(|y - \mu| \geq k\sigma) \leq \frac{1}{k^2} \]  

(11)

where \( \mu \) and \( \sigma \) denote the mean and standard deviation, respectively of variable \( y \). The inequality holds true for any value \( k > 0 \).

If the FFN has adequate representational capacity, the fit should not be significantly biased and the mean residual vector \( (\mu_j) \) can be a null vector. The above PL estimation method can also be easily extended to determine the limits of the dispersion of the mean, i.e. the range of possible values for the mean predicted value (rather than the value for a single sample).

However, there exists a strong limitation with the above approach, in that, the ability of the method to accurately estimate the PLs in distinct regions of the input space would depend on the resolution of \( N \). In addition, the method exhibits the following detrimental characteristics: (1) within any given distinct input space neighbourhood, the estimated PLs are at constant width from the FFN model prediction; and (2) PLs are disjoint at boundaries of neighbourhoods due to the abrupt transition from one neighbourhood to the adjacent.

One approach to address these limitations is to introduce the concept of a “truth function” to evaluate the strength of the membership of any given data point within any distinct “local neighborhood.” Such a truth function will facilitate a weighted approach to estimation of PLs, where the weights are the data point’s strengths in different neighbourhoods. One such truth function that has showed promise is given below (Chinnam and Ding 1998):

\[ \text{Truth}_j(x) = \left( \frac{1/Y_j(x)}{\sum_{j=1}^{N} (1/Y_j(x))} \right) \]  

(12)

where \( Y_j(x) \) is the output of neuron \( j \) in the self-organizing feature map†.

Essentially, this truth function determines the strength for a data point in a particular neighbourhood as a function that is inversely proportional to the Euclidean distance of the data point from the location (or center) of the SOM neuron representing the neighbourhood. Since the PLs are being estimated using a weighted approach, it is essential that the total membership for any data point among all neighbourhoods be equal to unity. In other words, the following constraint has to be satisfied by any credible truth function:

\[ \sum_{j=1}^{N} \text{Truth}_j(x) = 1.0 \]  

(13)

The proposed truth function (12) certainly meets this constraint. Simulation studies have demonstrated that this approach is effective in estimating the PLs for neural network models (Chinnam and Ding 1998).

While this local approximation scheme for estimation of PLs offers a lot of promise, a fundamental concern with this method is that the neighbourhoods are created strictly based on the input data densities using an unsupervised approach. The consequence is that

---

†Here, the output is the Euclidean distance between the neuron and the data point in the input space. In the SOM algorithm proposed by Johnson (1994), these outputs are further transformed onto a binary scale using the concept of similarity matching (for finding the best-matching or winning neuron, using the minimum-Euclidean distance criterion).
the partitioning scheme does not “directly” account for any “patterns” that might exist in the dispersion properties of the residuals in the output space. For example, suppose that the output response variability is significantly high in certain parts of the input space. Under these circumstances, given that the input space is being partitioned using an unsupervised method, the partitioning scheme does not recognize the above phenomenon and might generate neighbourhoods that “divvy” up the neighbourhood, potentially leading to biased PL estimates.

5. Estimation of PLs using a residual variance FFN—a global approximation method

As was stated earlier, the most fundamental limitation of the above local approximation method (see Section 4) for estimation of PLs is its inability to “directly” account for any patterns that might exist in the dispersion properties of the FFN model residuals. In an attempt to address this limitation, here we present a global approximation method for estimation of PLs. Rather than dividing the FFN model input space into distinct neighbourhoods and estimate the mean residuals vector and covariance matrices for each of the neighbourhoods, this global approximation method attempts to directly model the dispersion properties of the residuals of the main FFN model, denoted $NN_M$, using another FFN model, denoted $NN_{PL}$. For the case where $NN_M$ is modelling a multiple-input single-output (MISO) process, this process is illustrated in figure 3.

Here are the specifics. Let us suppose that $NN_M$ is modelling a MISO process with a “target” function $f(x)$. Let the “actual” function captured by $NN_M$ after the training process be denoted by $F_M(x)$ where $w$ denotes the parameters of the model. The residual from the $NN_M$ model, denoted by $e_M(x)$, can be broken down into its components as follows:

$$e_M(x) = (y_d - F_M(x)) = (y_d - f(x)) + (f(x) - F_M(x)) = e_M(x) + (f(x) - F_M(x)).$$

(14)

If one were to assume that there is no bias in the $NN_M$ model or that the bias is not significant, then, it follows from equation (14) that the model residuals for any given $x$ are representative of true error residual (or unpredictability of the response).

Now, if we were to take the squares of the residuals of the $NN_M$ model (i.e. $e_M^2(x)$) and feed them as desired responses into $NN_{PL}$, the resulting predictions are representative of the variance of the residuals of the $NN_M$ model. This can be demonstrated as follows. Let $3$ and $w$ denote the training data and the parameter vector, respectively for $NN_{PL}$. Typically, $w$ is
obtained by minimizing the following:

$$\xi(\mathbf{w}) = E_3[(e_M^2(x) - F_{PL}(x, \mathbf{w}))^2]$$  \hspace{1cm} (15)$$

where $E_3$ denotes the average operator taken over the entire training sample $3$.

One can rearrange the terms of the $NN_{PL}$ model cost function $\xi(\mathbf{w})$, and lead to the following:

$$\xi(\mathbf{w}) = E_3[(e_M^2(x) - F_{PL}(x, \mathbf{w}))^2] = E_3[(\epsilon_M(x) + \text{Bias}_M)^2 - F_{PL}(x, \mathbf{w}))^2].$$ \hspace{1cm} (16)$$

If the bias in the $NN_M$ model is zero, it can be shown rather easily that equation (16) leads us to the following expression:

$$\xi(\mathbf{w}) = E_3[(e_M^2(x) - \sigma_M^2(x))^2] + E_3[(\sigma_M^2(x) - F_{PL}(x, \mathbf{w}))^2].$$ \hspace{1cm} (17)$$

Here $\sigma_M^2(x) = E[\epsilon_M^2(x)|\mathbf{X} = \mathbf{x}]$. Given that the intrinsic error cannot be minimized, the training process of the $NN_{PL}$ model results in the minimization of the “natural measure of effectiveness”. Hence, $\lim_{\text{Bias}_M \rightarrow 0} F_{PL}(x, \mathbf{w}) \rightarrow \sigma_M^2(x)$. This implies that if care is exercised in building the $NN_M$ such that its bias becomes zero, $NN_{PL}$ theoretically has the ability to directly estimate $\sigma_M^2(x)$.

In the event $NN_M$ is modelling a MIMO process, in order to estimate the PLs, one has to directly estimate the covariance matrix $\Sigma_M(x)$. By definition, the diagonal elements of the covariance matrix $\Sigma_M(x)$ correspond to variances of the residuals for any given output variable and the non-diagonal elements correspond to the covariance of the residuals between a pair of output variables. Given that $\Sigma_M(x)$ is by definition symmetric, one has to estimate $q(q + 1)$ elements of the $q \times q$ dimensional matrix $\Sigma_M(x)$, where $q$ denotes the dimensionality of the $NN_M$ model output space.

Hence, if $NN_M$ is modelling a MIMO process with output dimension $q$, the needed output dimensionality for the $NN_{PL}$ model grows at the rate of $q(q + 1)$, leading to the curse of dimensionality. On the contrary, if one can assume that there exists no multi-collinearity among the output variables (i.e. covariances are zero), $\Sigma_M(x)$ is a diagonal matrix and hence, the dimensionality for the output space of the $NN_{PL}$ model is simply $q$.

There are two limitations with this global approximation method for estimation of PLs. The first limitation is that one has to exercise great care in building $NN_M$ and $NN_{PL}$ models to avoid any potential bias. Simulation evaluation revealed that the overall performance of this approach is sensitive to the training process of the $NN_{PL}$ model. This is particularly true when the model output space is multi-dimensional and the data density is sparse. The second limitation is that the dimensionality of the $NN_{PL}$ model grows at the rate of $q(q + 1)$ when there exists multi-collinearity.

### 6. Evaluation of proposed PL estimation methods

To evaluate the performance of the proposed global approximation method for estimation of PLs with respect to the local approximation method proposed by Chinnam and Ding (1998), we repeat the same evaluation experiments carried out by Chinnam and Ding (1998).

Data sets were generated with one- and two-dimensional input spaces as shown in Table 1. In the case of the one-dimensional input space, the data sets were generated from the following
Table 1. Cases used for evaluation of the proposed PL estimation methods and their results.

<table>
<thead>
<tr>
<th>Case name</th>
<th>Input space</th>
<th>PDF of input space</th>
<th>Gaussian noise parameters</th>
<th>Average estimation error for 90% PLs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case I-A</td>
<td>1D</td>
<td>Truncated exponential: ( f(x) = \frac{1}{2} (e^{-\frac{x}{2}} + 1) )</td>
<td>( \mu_x = 1.0 ), ( \sigma_x = 1.0 )</td>
<td>( y_1: 13.17% ), ( y_2: 11.86% )</td>
</tr>
<tr>
<td>Case I-B</td>
<td>2D</td>
<td>Uniform: ( f(x_1, x_2) = 1.0 )</td>
<td>( \mu_x = [0, 0], \sigma_x = [1, 1] )</td>
<td>( y_1: 15.34% ), ( y_2: 6.27% )</td>
</tr>
<tr>
<td>Case II</td>
<td>2D</td>
<td>Uniform: ( f(x_1, x_2) = 1.0 )</td>
<td>( \mu_x = [0, 0], \sigma_x = [1, 1] )</td>
<td>( y_1: 17.23% ), ( y_2: 11.86% )</td>
</tr>
</tbody>
</table>
function: \( y(x) = 0.5 \sin(1.5\pi x + \pi/2) + 2.0 + \nu(x) \), where \( \nu(x) \) is Gaussian noise. The mean of the Gaussian noise was chosen to be zero and the standard deviation a decreasing function of \( x \). In the case of the two-dimensional input space, the data set was generated from the following functions: 

\[
y_1(x_1, x_2) = 35 + 2x_1 - 3x_2 + 6 \sin(2\pi x_1) - 6 \sin(1.5\pi x_2) - \nu_1(x_1, x_2)
\]

and

\[
y_2(x_1, x_2) = 15 + 2x_1 - 3x_2 + 6 \sin(2\pi x_1) + 6 \sin(1.5\pi x_2) + \nu_2(x_1, x_2),
\]

for the two output variables. Again, the mean of the Gaussian noise was chosen to be zero and standard deviations are functions of \( x_1 \) and \( x_2 \), as shown in Table 1. Note that multi-collinearity is intentionally avoided to facilitate better graphical evaluation of the proposed PL estimation methods.

For Case I, the data sets were generated with 1500 data points (labeled Case I-A) and 150 data points (labelled Case I-B) to study the degradation in the accuracy of calculation of PLs as a function of the size of the data set. For Case II, with two-dimensional input space and two dimensional output space, the data set was generated with 400 data points. In all cases, MLP networks, proved to be universal approximators in the literature (Hornik et al. 1989), were used for building \( N_{NM} \) models. Chinnam and Ding (1998) reported the configurations of the MLP networks that proved to be effective in offering good generalization. These configurations are reported in Table 1 as well. Unless stated otherwise, all the MLP networks studied here have a hyper-tangent function as the sigmoid nonlinearity except for the neurons in the output layers that were linear.

With respect to selecting the configuration for the SOMs, two factors should be taken into consideration (Chinnam and Ding 1998). First, the partition scheme should ensure that all the neighbourhoods carry enough data points as members (i.e. \( M_j \)) to accurately estimate the residual covariance matrices (in general, for statistical analysis, 10–20 data points are considered adequate for accurate estimation of any given covariance matrix; we recommend the same: \( 10 \leq M_j \leq 20 \) for all \( j \)). Second, the partition resolution sought by the SOM should parallel the complexity of the associated function approximating neural network. Experimental investigation has revealed that this guideline approximately translates to stating that the number of nodes in the SOM should be no more than twice the number of nodes in the hidden layers in the function approximation neural network. This guideline partially ensures that the estimated residual covariance matrices reflect the true dispersion characteristics of the output variables and not the biases associated with the function approximation neural network predictions. Simulation studies conducted for the test cases strongly revealed the appropriateness of these two guidelines. The configurations of the SOMs used to partition the input spaces for the test cases are shown in Table 1 and certainly meet the above guidelines. The SOM training scheme suggested by Haykin (1999) was used, involving 20,000 epochs, a time-varying learning rate parameter and a time-varying neighbourhoods function. The configurations of the SOMs used by Chinnam and Ding (1998) to optimally partition the input spaces are reported in Table 1 as well. The configurations of the \( NN_{PL} \) models developed for estimating the PLs using the global approximation method proposed are also reported in Table 1.

Cases I-A and I-B involve a single output variable and hence, the covariance matrix (9) is reduced to a scalar estimate of the variance of the residuals in each of the neighbourhoods \( (\hat{\sigma}_j^2) \). In the single-output case, with the weighted truth function approach applied to equation (10), the \( 1 - \alpha \) PLs are:

\[
\hat{y}(x) \pm \sum_{j=1}^{N} t_{1-\alpha/2,M_j-1} \cdot \hat{\sigma}_j \cdot \left( 1 + \frac{1}{M_j} \right)^{0.5} \cdot \text{Truth}_j(x) \quad (18)
\]
where $t_{1 - \alpha/2, M - 1}$ denotes the $(1 - \alpha/2)$ quantile of Student’s $t$ statistic with $(M - 1)$ degrees of freedom.

Figure 4 presents the estimated 90% PLs ($\alpha = 0.1$) for Case-I simulation study for both the methods, along with the true 90% PLs directly calculated using the actual residual standard deviation (i.e. $\sigma(v(x)) = -0.12x + 0.13$). In spite of a 90% reduction in the size of the data set (from Case I-A to Case I-B), the results obtained with data sets containing only 150 data points were reasonably good (compare figure 4(a) with figure 4(b)). Careful examination of figure 4(b) by the reader will reveal that there are discrepancies in some areas between the true and estimated PLs for the “local” method in some areas.

![Figure 4(a)](image1)

(a) Case I-B data set with 1500 data points.

![Figure 4(b)](image2)

(b) Case I-A data set with 150 data points.

Figure 4. Results from Case-I study; truncated exponential pdf in input space; 90% PLs. (a) Case I-B data set with 1500 data points. (b) Case I-A data set with 150 data points.
Figure 5 presents the results for the Case-II simulation study. Great care has to be exercised in evaluating the PL estimation error percentage graphs. While these error percentages seem large in certain areas of the input space, note that the actual magnitudes of these errors are very small (Table 1 for noise standard deviation expressions). It is evident from these figures that the proposed empirical PL estimation methods are reasonably effective even with a relatively sparse two-dimensional input/output space. The PL estimation accuracy for all these cases was also quantified and shown in Table 1 for both the methods. Note that the data points shown on all PL graphs are testing data sets and hence, the fraction of the number of data points falling outside the limits (should be close to 10% given that $\alpha$ was chosen to be 0.1 in estimating the PLs) is representative of the accuracy of the proposed PL estimation methods.
The global method outperformed the local method for Case-I. It is the other way around for Case-II. We strongly believe that this is attributed to the significant drop in data density. The local method seems to outperform the global method when the data density is very low and the output dimensionality is high.

7. Conclusion

In contrast to locally generalizing networks such as RBF networks and CMAC networks that have a naturally well defined concept of local neighbourhoods, globally generalizing FFNs,
such as MLP networks, are not inherently capable of providing prediction intervals. While Chinnam and Ding (Farmer and Sidorowich) proposed a local approximation method for estimation of PLs, there is a fundamental limitation with that method in that it has no ability to “directly” account for any patterns that might exist in the dispersion properties of the residuals in the output space. This paper presents a global approximation method for estimation of PLs that overcomes the above limitation. Both these methods compute prediction intervals (error bounds) for any regression model (neural network based or otherwise).

As stated earlier, while the global approximation method proposed here overcomes the limitations of the local approximation method proposed by Chinnam and Ding (Farmer and Sidorowich), it has its own limitations as well. The quality of the PLs estimated using this procedure is quite sensitive to any bias present in the NNPL model. Overall, both these empirical methods seem to be effective in estimating prediction intervals for FFN models. They certainly overcome the standard assumption of constant residual variance often invoked by other PL estimation methods proposed in the literature.

References


Ratna Babu Chinnam is an Associate Professor in the Department of Industrial & Manufacturing Engineering at Wayne State University (Detroit, USA). He is the author of over 50 technical publications in the areas of Smart Engineering Systems, Intelligent Manufacturing, Autonomous Diagnostics/Prognostics, and Supply Chain Management. Most of his research is funded by NSF and he maintains strong collaborations with Ford Motor Company and DaimlerChrysler. He is a member of Alpha Pi Mu, INFORMS, the International Neural Network Society, and the North American Manufacturing Research Institute.
Pundarikaksha Baruah is a doctoral candidate in the Department of Industrial & Manufacturing Engineering at Wayne State University (Detroit, USA). His research work has been published in International Journal of Production Research, International Journal of Material and Product Technology and in seven conference proceedings. His research interests are in the areas of Management Science, Smart Engineering Systems, Supply Chain Management, and Quality. He is a member of American Society of Quality, APICS and INFORMS.