EURASIP Journal on Applied Signal Processing 2004:17, 2696–2704
© 2004 Hindawi Publishing Corporation

Surface Approximation Using the 2D FFENN Architecture

S. Panagopoulos
Institute for Communications & Signal Processing, University of Strathclyde, Royal College Building, Glasgow G1 1XW, UK
Email: spyros@spd.eee.strath.ac.uk

J. J. Soraghan
Institute for Communications & Signal Processing, University of Strathclyde, Royal College Building, Glasgow G1 1XW, UK
Email: j.soraghan@eee.strath.ac.uk

Received 27 August 2003; Revised 10 March 2004; Recommended for Publication by Bernard Mulgrew

A new two-dimensional feed-forward functionally expanded neural network (2D FFENN) used to produce surface models in two dimensions is presented. New nonlinear multilevel surface basis functions are proposed for the network's functional expansion. A network optimization technique based on an iterative function selection strategy is also described. Comparative simulation results for surface mappings generated by the 2D FFENN, multilevel 2D FFENN, multilayered perceptron (MLP), and radial basis function (RBF) architectures are presented.

Keywords and phrases: neural networks, sea clutter, surface modeling.

1. INTRODUCTION

One of the main properties of feed-forward neural networks is that of learning an input-output mapping from a set of examples characterizing a real system. The network is trained with some examples comprising an input signal and the desired response. The network weights are then modified, using an adaptive optimization technique to minimize the difference between the desired response and actual response. Two well-known feed-forward artificial neural networks are the multilayered perceptron (MLP) and radial basis function (RBF). Both networks have been termed as *universal approximators* [1, 2]. Their performance has been demonstrated in various application areas such as linear and nonlinear adaptive filtering [3], time series prediction [4], dynamic reconstruction [5], and black-box modeling [6]. However, these networks suffer from a number of drawbacks, such as convergence characteristics and network topology selection [7].

MLP networks traditionally employ sigmoidal activation functions that cannot model local nonlinearity optimally. Also, their nonlinear in-the-parameters structure requires complex and computationally intense learning algorithms, such as the backpropagation algorithm. Furthermore, there is no way to say whether a single hidden layer is optimum to support the MLP network learning or a way to specify the exact number of hidden neurons required in order for a system to be generalizable.

On the other hand, RBF networks that traditionally employ radial symmetric functions cover only small localized regions and therefore they cannot model global nonlinearity well. Moreover, dealing with RBF networks' great difficulty is experienced in selecting the appropriate centers for the radial basis functional expansion. Additionally, a large number of basis functions is usually required in order to cover high-dimensional input spaces. Nonetheless, simple learning algorithms may be used for training, as the RBF structure is linear in the parameters.

In this paper, the design of a new single hidden layer, linear in the parameters, two-dimensional feed-forward functionally expanded neural network surface modeler (2D FFENN) is presented. Previously, a 1D FFENN has been successfully applied to time series prediction [8, 9] and cochannel interference [10]. The aim of this new design is to explore the modeling capabilities of such a feed-forward network in two dimensions. The main objective is to approximate any nonlinear continuous surface, from a 2D data set, to an arbitrary degree of accuracy. Like its predecessor 1D FFENN design, the design of 2D FFENN can be considered as a hybrid neural network. In essence, it is an extended model that incorporates the modeling capabilities of the existing architectures of MLP, RBF, and Volterra neural networks (VNN).

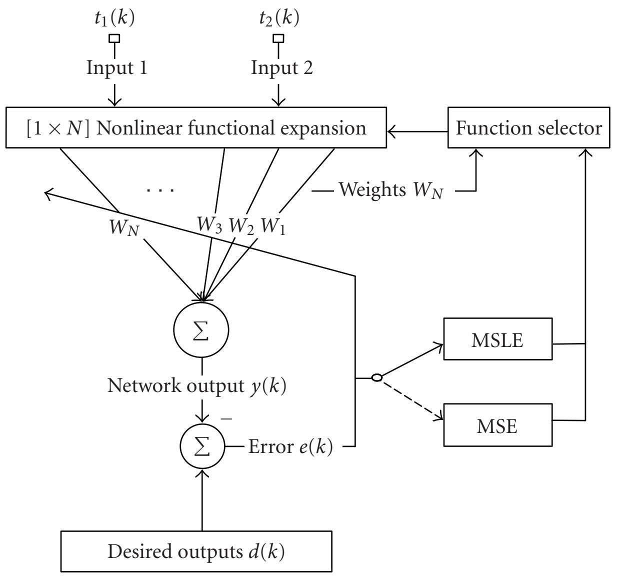

FIGURE 1: The 2D FFENN structure.

The remainder of the paper is organized as follows. In Section 2, an overview of the 2D FFENN and multilevel 2D FFENN architectures is given. Section 3 provides a brief description of the network weight adaptation algorithm. Section 4 describes the characteristics of a function pruning technique, which is developed to optimize the network's functional expansion. In Section 5, a number of representative simulation results are presented that illustrate the surface modeling capabilities of both the 2D FFENN and multilevel 2D FFENN designs in comparison with the MLP and RBF networks. Section 6 concludes the paper.

2. THE 2D FFENN ARCHITECTURE

2.1. *The 2D FFENN structure*

The architecture of the 2D FFENN structure is depicted in Figure 1. It consists of two layers: a single hidden layer and an output layer. The hidden layer acts like a feature detection layer. As the learning process progresses, the hidden neurons begin to gradually discover the salient features that characterize the training data. This is achieved by the functional expansion unit, which performs a nonlinear transformation of the input data into a new space called the feature space. The output layer of the network comprises a set of linear combiners that join together all the weighted functionally expanded inputs to form a single output.

The functional expansion unit of 2D FFENN takes two inputs $\bar{t}_1$ and $\bar{t}_2$ which are the grid indices that specify the 2D data set to be modeled. Both inputs are normalized to within the range $(+1, -1)$.

The entire functional expansion is described by $F(k)$ as follows:

$$F(k) = \text{sum of } N (\text{linear \& nonlinear}) \text{ basis functions}. \tag{1}$$

In a similar fashion to the functions described in [11], the linear terms of the expansion are the original input terms, whilst the nonlinear terms are a combination of *trigonometric* and *polynomial* 2D functions of the input. The modeling efficiency of the 2D FFENN is the result of this hybrid functional expansion [12]. These functions have been chosen in such a way to combine the global approximation capability of the MLP network' the local approximation capability of the RBF network and also to emulate the modeling capability of the VNN.

In general, a multiple-input multiple-output (MIMO) FFENN (n, N, m) will completely be specified for a given number of n inputs and m outputs by a similar $F(k)$ expansion.

The output of the two-layered, two-input, and N-term FFENN (2, N, 1) is defined as follows.

(a) Hidden layer functional expansion vector at time k:

$$F(k) = [f_1(k), f_2(k), \ldots, f_N(k)]^T \tag{2}$$

and the associated weight vector as

$$W(k) = [w_1(k), w_2(k), \ldots, w_N(k)]^T. \tag{3}$$

(b) Single 2D FFENN output:

$$y(k) = F^T(k) \cdot W(k). \tag{4}$$

(c) Prediction error:

$$e(k) = d(k) - y(k), \tag{5}$$

where $d(k)$ is the reference response.

Network weight adaptation is achieved using the exponentially recursive least squares (RLS) algorithm. Complex training algorithms are not required because of the linear in-the-parameters network structure.

The function selection unit shown in Figure 1 is used in order to reduce the size of the functionally expanded network. The scope of the functional expander is to introduce to the network new functional terms in order to enhance its nonlinear approximation ability. However, this process can lead the expansion to very large and highly redundant networks. For this reason, a pruning or function selection scheme is occupied to choose only the most significant functions.

2D FFENN functional expansion

Analytically, the functional expansion of 2D FFENN, for 2 inputs (x_1 and x_2) normalized within the range $(+1, -1)$, is described by the following set of terms.

(1) Zero-order (dc) term (1 term).
(2) Original input terms (2 terms). Linear system modeling.
(3) Sine expansion of the 2 inputs, comprising $\sin(x_i)$, $\sin(2x_i)$, and $\sin(3x_i)$ terms for $i = 1, 2$ (6 terms).
(4) Cosine expansion of the 2 inputs, comprising $\cos(x_i)$, $\cos(2x_i)$, and $\cos(3x_i)$ terms for $i = 1, 2$ (6 terms).

(5) Cross input and sine expansion of the 2 inputs, comprising $x_i \sin(x_j)$, $x_i \sin(2x_j)$, and $x_i \sin(3x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (6 terms).
(6) Cross input and cosine expansion of the 2 inputs, comprising $x_i \cos(x_j)$, $x_i \cos(2x_j)$, and $x_i \cos(3x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (6 terms).
(7) Cross sine expansion of the 2 inputs, comprising $\sin(x_i)\sin(x_j)$, $\sin(x_i)\sin(2x_j)$, and $\sin(x_i)\sin(3x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (5 terms).
(8) Cross cosine expansion of the 2 inputs, comprising $\cos(x_i)\cos(x_j)$, $\cos(x_i)\cos(2x_j)$, and $\cos(x_i)\cos(3x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (5 terms).
(9) Cross cosine and sine expansion of the 2 inputs, comprising the terms $\cos(x_i)\sin(x_j)$, $\cos(x_i)\sin(2x_j)$, $\cos(x_i)\sin(3x_j)$, $\cos(2x_i)\sin(x_j)$, $\cos(3x_i)\sin(x_j)$, $\cos(2x_i)\sin(3x_j)$, $\cos(3x_i)\sin(2x_j)$, $\cos(2x_i)\sin(2x_j)$, and $\cos(3x_i)\sin(3x_j)$ for $i \neq j$, $i, j = 1, 2$ (18 terms).
(10) Cross input and cosine and sine expansion of the 2 inputs, comprising $x_k \cos(x_i)\sin(x_j)$, $x_k \cos(x_i)\sin(2x_j)$, $x_k \cos(2x_i)\sin(x_j)$ terms for $i \neq j$, $i, j, k = 1, 2$ (12 terms).
(11) $\cos(x_2)x_1x_2$, $\cos(2x_2)x_1x_2$, $\cos(3x_2)x_1x_2$, $\sin(x_1)x_1x_2$, and $\sin(3x_1)x_1x_2$ (5 terms).
(12) x_1x_2, $(x_1)^2(x_2)^2$, $(x_1)(x_2)^2$, $(x_1)^2(x_2)$, $(x_1)^3$, and $(x_2)^3$ (5 terms).
(13) $(1 - x_1)\cos(x_2)$ and $(1 - x_2)\cos(x_1)$ (2 terms).

The set of equations described above comprise the full cross-terms functional expansion. However, a number of these can be discarded due to the symmetric properties of the functions. The following functions may be excluded after visual inspection of the surface characteristics of all 80 function terms: $\sin(x_2)\sin(2x_1)$, $\cos(x_2)\cos(2x_1)$, $\cos(x_1)\sin(x_2)$, $x_2(x_1)^2$, $(x_1)^3$, $(x_2)^3$, $x_2\sin(x_2)\cos(x_1)$, $x_2\cos(x_2)\sin(x_1)$, $x_1\cos(x_2)\sin(x_1)$, $x_1\sin(x_2)\cos(x_1)$, $x_2\cos(2x_1)$, $x_2\sin(3x_1)$, $(1 - x_2)\cos(x_1)$, $\cos(2x_2)\sin(2x_1)$, $\sin(2x_2)\cos(3x_1)$, $\sin(x_2)\cos(3x_1)$, $\cos(2x_1)\sin(2x_2)$, $x_1\sin(x_2)\cos(x_1)$, $\sin(x_2)\cos(2x_1)$, $x_2\sin(2x_1)$, $x_2\cos(3x_1)$.

The final functional expansion has been reduced to 59 functions in total.

2.2. Multilevel 2D FFENN design

The multilevel 2D FFENN design is an extended version of the generic 2D FFENN model. It incorporates extra function groups at different scales in order to enhance its performance and to cope with "spikier" data. The topology of the multilevel 2D FFENN design is depicted in Figure 2. The 2D FFENN scale-set-1 unit (low-scales functional expansion) shown in Figure 2 refers to exactly the same design described so far by the generic 2D FFENN using 59 functions. Scale-set-2 (medium-scales functional expansion) and scale-set-3 (high-scales functional expansion) are two additional function groups that have been included in order to be introduced to the model functions at different scales. The network can produce results by either using scale-set-1 only, by using scale-sets-1 and -2, or by using all three sets together. Scale-set-2 achieves a further functional expansion of 33 functions and scale-set-3 adds an extra 42 functions at a higher scale.

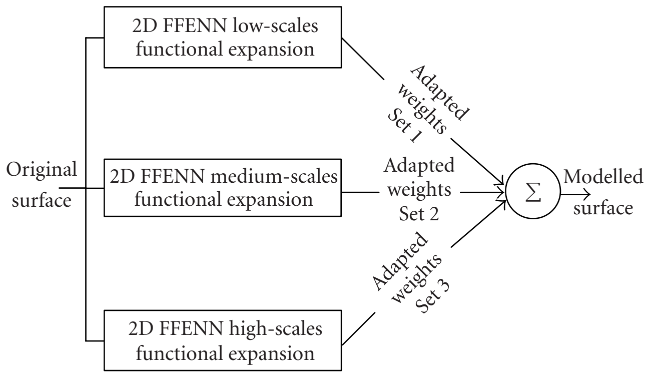
Original surface
2D FFENN low-scales functional expansion
2D FFENN medium-scales functional expansion
2D FFENN high-scales functional expansion
Adapted weights Set 1
Adapted weights Set 2
Adapted weights Set 3
Σ
Modelled surface

Figure 2: The multilevel 2D FFENN design.

Extended 2D FFENN functional expansion

The extended functional expansion of the multilevel 2D FFENN is constituted by the following terms.

Medium-scales functional expansion

(1) Sine expansion of the 2 inputs, comprising $\sin(10x_i)$ and $\sin(20x_i)$ terms for $i = 1, 2$ (4 terms).
(2) Cosine expansion of the 2 inputs, comprising $\cos(10x_i)$ and $\cos(20x_i)$ terms for $i = 1, 2$ (4 terms).
(3) Cross sine expansion of the 2 inputs, comprising $\sin(x_i)\sin(10x_j)$, $\sin(x_i)\sin(20x_j)$, $\sin(10x_i)\sin(10x_j)$, $\sin(20x_i)\sin(20x_j)$, and $\sin(10x_i)\sin(20x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (8 terms).
(4) Cross cosine expansion of the 2 inputs, comprising the terms $\cos(x_i)\cos(10x_j)$, $\cos(x_i)\cos(20x_j)$, $\cos(10x_i)\cos(10x_j)$, $\cos(20x_i)\cos(20x_j)$ and $\cos(10x_i)\cos(20x_j)$ for $i \neq j$, $i, j = 1, 2$ (8 terms).
(5) Cross cosine and sine expansion of the 2 inputs, comprising $\cos(10x_i)\sin(10x_j)$, $\cos(10x_i)\sin(20x_j)$, $\cos(20x_i)\sin(10x_j)$, and $\cos(20x_i)\sin(20x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (8 terms).
(6) Exponential expansion of the 2 inputs, comprising the following term: $e^{\cos(2\cdot x_i)+\cos(2\cdot x_j)}$, for $i \neq j$, $i, j = 1, 2$ (1 term).

High-scales functional expansion

(1) Sine expansion of the 2 inputs, comprising $\sin(30x_i)$ and $\sin(40x_i)$ terms for $i = 1, 2$ (4 terms).
(2) Cosine expansion of the 2 inputs, comprising $\cos(30x_i)$ and $\cos(40x_i)$ terms for $i = 1, 2$ (4 terms).
(3) Cross sine expansion of the 2 inputs, comprising $\sin(x_i)\sin(30x_j)$, $\sin(x_i)\sin(40x_j)$, $\sin(30x_i)\sin(30x_j)$, $\sin(40x_i)\sin(40x_j)$, and $\sin(30x_i)\sin(40x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (8 terms).
(4) Cross cosine expansion of the 2 inputs, comprising the terms $\cos(x_i)\cos(30x_j)$, $\cos(x_i)\cos(40x_j)$, $\cos(30x_i)\cos(30x_j)$, $\cos(40x_i)\cos(40x_j)$, and $\cos(30x_i)\cos(40x_j)$ for $i \neq j$, $i, j = 1, 2$ (8 terms).
(5) Cross cosine and sine expansion of the 2 inputs, comprising $\cos(30x_i)\sin(30x_j)$, $\cos(30x_i)\sin(40x_j)$, $\cos(40x_i)\sin(30x_j)$, and $\cos(40x_i)\sin(40x_j)$ terms for $i \neq j$, $i, j = 1, 2$ (8 terms).

(6) Sigmoid expansion of the 2 inputs, comprising $\tanh(x_i)$, $\tanh(10x_i)$, $\tanh(30x_i)$, $\tanh(50x_i)$, $\tanh(60x_i)$, and $\tanh(100x_i)$ terms for $i = 1, 2$ (12 terms).

3. NETWORK WEIGHT ADAPTATION

FFENN network weight adaptation is achieved recursively by the exponentially weighted RLS algorithm [13, Chapter13, pages 562–587].

The exponentially weighted RLS estimator can be derived by minimizing the following cost function with respect to the weight coefficient vector $W(k)$:

$$E(k) = \sum_{t=1}^{k} \lambda^{k-t} e(t)^2, \tag{6}$$

where the ensemble averages have now been replaced by time averages and λ represents a weighting or forgetting factor belonging to $(0, 1)$.

The computation is launched with known initial conditions and then progresses recursively using information contained in new data samples that used to update the old estimates. The use of the exponential weighting factor or forgetting factor λ, in general, is intended to ensure that the data in the distant past are "forgotten," in order to allow modeling of time-variant systems.

The optimum weight vector $W(k)$, for which the cost function $E(k)$ attains its minimum value, is defined by the set of "normal" equations

$$W_{\text{opt}} = R^{-1}(k) \cdot \Phi(k), \tag{7}$$

where the autocorrelation matrix $R(k)$ is defined as

$$R(k) = \sum_{t=1}^{k} \lambda^{k-t} \cdot F(t) \cdot F^T(t) \tag{8}$$

and the cross-correlation vector $\Phi(k)$ as

$$\Phi(k) = \sum_{t=1}^{k} \lambda^{k-t} \cdot d(t) \cdot F(t). \tag{9}$$

Notice that both $R(k)$ and $\Phi(k)$ can also be recursively expressed as

$$R(k) = \lambda \cdot R(k-1) + F(k) \cdot F^T(k), \tag{10}$$

$$\Phi(k) = \lambda \cdot \Phi(k-1) + d(k) \cdot F(k). \tag{11}$$

The inverse of $R(k)$ can then be estimated recursively using the matrix inversion lemma [13, Chapter13, pages 562–587] as follows. Let $P(k) = R^{-1}(k)$ and assume $\Phi(k)$ is positive definite and therefore nonsingular:

$$P(k) = \frac{1}{\lambda}\left[P(k-1) - \frac{P(k-1) \cdot F(k) \cdot F^T(k) \cdot P(k-1)}{\lambda + F^T(k) \cdot P(k-1) \cdot F(k)}\right]. \tag{12}$$

Thus, substituting for $\Phi(k)$ from (11) into (7) yields

$$W_{\text{opt}}(k) = \lambda \cdot P(k) \cdot \Phi(k-1) + d(k) \cdot P(k) \cdot F(k), \tag{13}$$

where $P(k)$ is given from (12). Also, $P(k-1) \cdot \Phi(k-1) = W(k-1)$.

Hence, the final recursive update for the FFENN weight vector $W(k)$, for a single output, can be expressed as

$$W(k) = W(k-1) + P(k) \cdot F(k) \cdot e(k). \tag{14}$$

The RLS algorithm has been preferred due to its faster convergence rates, even though it is computationally more complex. A simpler algorithm such as the least mean squares (LMS) or the normalized least mean squares (NLMS) can equally be used.

The LMS algorithm uses a simple equation to update the weight vector $W(k)$ as follows:

$$W(k+1) = W(k) + 2 \cdot \mu \cdot e(k) \cdot F(k), \tag{15}$$

where μ is the step-size parameter. Notice that the larger the value of μ, the faster the convergence of the weight vector, but the more susceptible to noise.

For the NLMS algorithm, we need also to introduce the normalized step-size parameter α. Thus, the step-size parameter μ of (15) is now defined as

$$\mu = \frac{\alpha}{F(k) \cdot F(k)^T}, \quad 0 < \alpha \leq 1. \tag{16}$$

4. PRUNING OF THE FULLY EXPANDED 2D FFENN

A large functional expansion can achieve better prediction results. Nevertheless, depending on the type and level of complexity of the surface to be modeled, the network may assume too many free parameters. This is because a much smaller number of functions are probably needed to characterize the specific function or surface. For this reason, a pruning scheme is utilized. Its task is to select only those functions which have a significant contribution to the output of the network. In other words, we want to choose only the dominant weights of the functional expansion.

Pruning is performed by an iterative pruning-retraining approach. Initially, the fully expanded network structure is trained on the training data set, and the maximum surface level error (MSLE) value on the training set is computed. For other methods, compute the mean square error (MSE) instead, which is defined as follows:

$$\text{MSE} = \frac{1}{k} \sum_{k} [e(k)^2]. \tag{17}$$

Based on our experiments, we suggest that the MSLE takes under consideration the worst possible error and not an average measurement. In the sea environment for example, target signal returns are mixed with the "spiky" signal characteristics of the sea clutter; therefore, averaging cannot assure optimal detection results because the MSE occasionally fails to

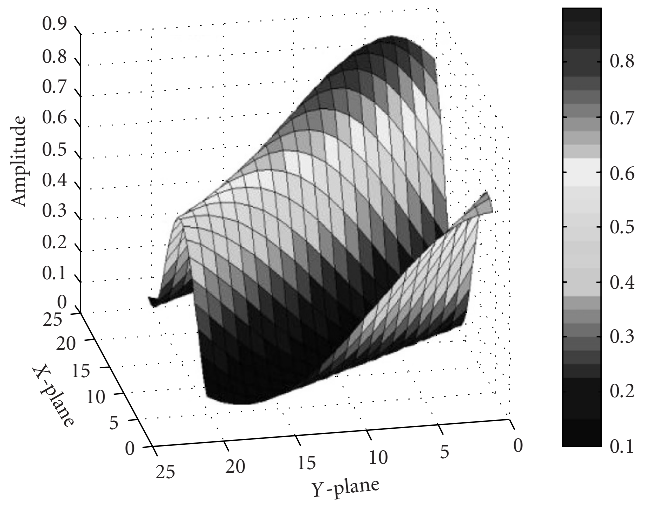

Figure 3: Original test surface $T(\bar{t}_1, \bar{t}_2)$.

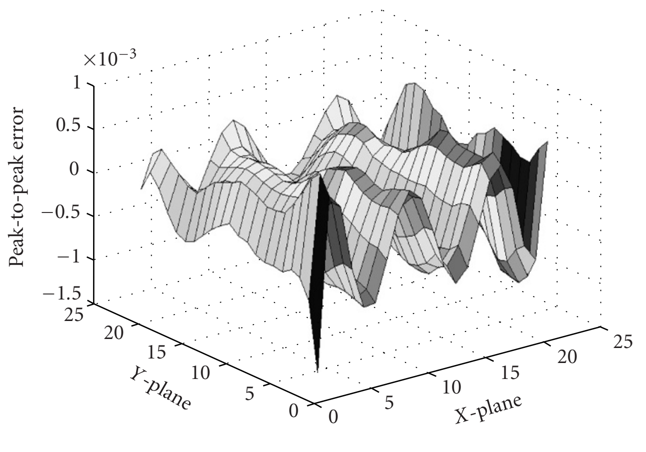

Figure 4: Error surface of 2D FFENN (no pruning, 59 functions, 5 epochs).

minimize localized error responses. In mathematical terms, the MSLE is defined as follows:

$$\mathrm{MSLE} = \left\| \max\left[e(k)\right] - \min\left[e(k)\right]\right\|. \qquad (18)$$

The insignificant functions in the expansion model are associated with the smallest weights; these are successively pruned one by one starting with the least significant one. After each insignificant function is being pruned, the output of the network is computed. Moreover, the resulting MSLE is also computed at each pruning stage. The pruning process is stopped at the stage when a pruned network structure is found to be incapable of reducing the output MSLE or MSE on the training set to the desired level. The network structure can be retrained after each time pruning is applied, with the same train set, in order to determine the optimal weights for the remaining unpruned functions.

Furthermore, it is important to note that pruning is only an optimization strategy and can be omitted when there is no advantage to be gained. It effectively achieves to reduce the size of the functional expansion, but in the downside, requires offline training (supervised learning) and much longer computation time.

5. SIMULATION RESULTS

5.1. *Application to computer generated 2D data*

5.1.1. *Function approximation by the 2D FFENN and multilevel FFENN*

In this section, we present simulation results for both the 2D FFENN and multilevel 2D FFENN structures. In order to illustrate the modeling capability of the 2D FFENN structure and the effectiveness of the pruning strategy, we produce a model for the smooth continuous surface described by (19). The choice of this function is based on its surface characteristics that resemble the nature of a swell wave. It has also been used in [14, Chapter 5, page 75] for testing the performance of feed-forward neural networks:

$$T(\bar{t}_1, \bar{t}_2) = 0.1 + \frac{1 + \sin\left(2 \cdot \bar{t}_1 + 3 \cdot \bar{t}_2\right)}{3.5 + \sin\left(\bar{t}_1 - \bar{t}_2\right)}. \qquad (19)$$

Figure 3 displays the surface characteristics of this 2D function. The training set is constructed from the function $T(\bar{t}_1, \bar{t}_2)$ by sampling the domain from -1.0 to $+1.0$ in two dimensions at equally spaced grid points, at an interval of 0.1 for both $\bar{t}_1$ and $\bar{t}_2$. The 21 spacing indices generated for each one of the dimensions correspond to the network inputs.

In Figure 4, the 2D FFENN modeling error is presented. Modeling was achieved by a functional expansion of 59 functions and batch training was performed for 5 times. Pruning has not been considered, so at this stage, all 59 functions contribute to the model. Based on the results, we can conclude that the 2D FFENN is able to produce very good surface mappings even after being trained for a small number of epochs.

In order to test the efficiency of the pruning strategy employed by the 2D FFENN, we simulate for the same surface, but this time using the MSE and MSLE to choose the most appropriate functions for modeling. The results are shown in Figures 5 and 6, respectively. From the results, it is evident that pruning can effectively reduce the dimensionality of the functional expansion. MSLE measure considers the worst possible error (peak-to-peak error) achieving a fairly flat error response, while the MSE measures the average error and consequently fails to minimize localized error responses. In this example, 29 functions have been discarded. The size of pruning achieved mainly depends on the specific characteristics of the surface to be predicted and the number of training epochs. The algorithm allows the user to retrain also after each time pruning is performed. This leads to a better weight adaptation achieving a smaller mapping error. However, the downside is that the number of candidate-pruned functions is reduced. In general, the minimum the network training performed, the maximum the pruning that can be achieved. Obviously, this tradeoff is directly reflected to the modeling accuracy requirement. Finally, Figure 7 shows the modeling

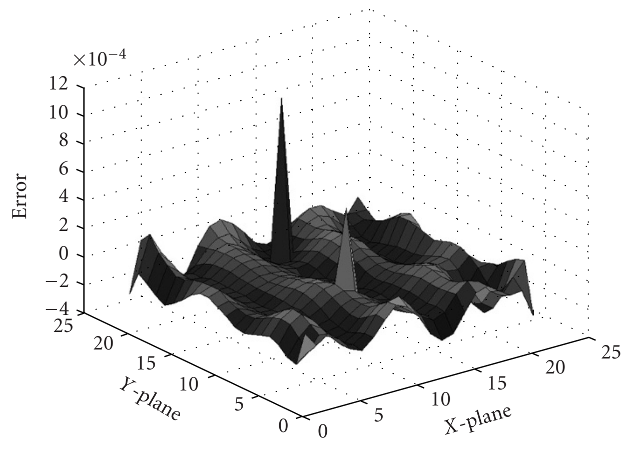

FIGURE 5: Error surface of 2D FFENN (MSE pruning, 30 functions, 5 epochs).

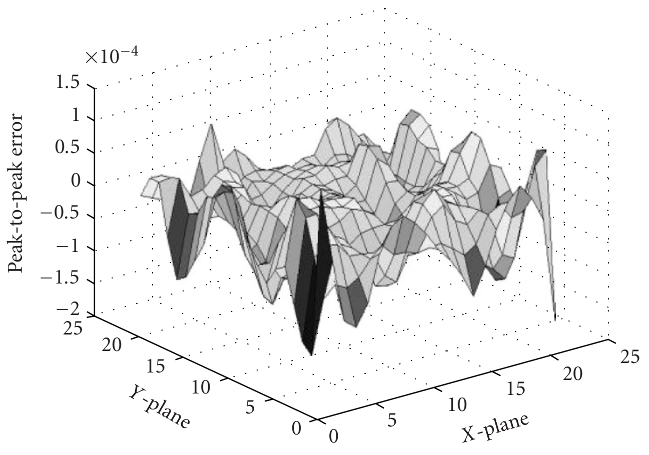

FIGURE 7: Error surface of multilevel 2D FFENN (MSLE pruning, 60 functions, 5 epochs).

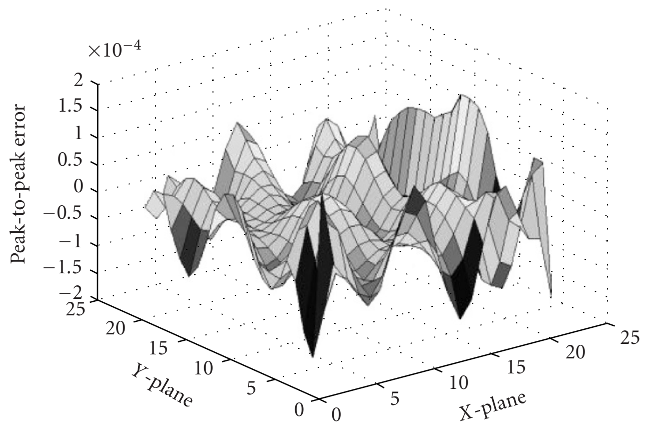

FIGURE 6: Error surface of 2D FFENN (MSLE pruning, 30 functions, 5 epochs).

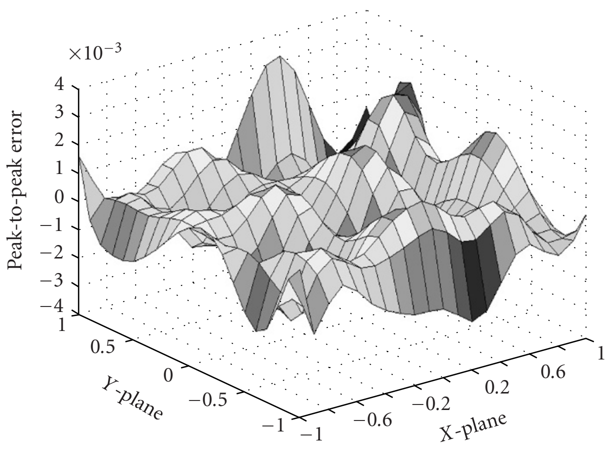

FIGURE 8: MSLE error surface of MLP (500 epochs, 15 hidden neurons).

results produced by the multilevel 2D FFENN design. The network is initially trained for a functional expansion of 134 functions and after 5 epochs of training, pruning manages to reduce the functional expansion down to 60 significant functions.

5.1.2. *Comparative study*

In this section, we compare the results presented in Section 5.1.1 with the performance of the MLP and RBF neural networks [7]. In order to perform a quantitative relative comparison, we try to achieve the best surface approximation that both the RBF and MLP can produce under a realistic network design of a strength similar to that of the 2D FFENN.

In Figure 8, the MLP network training error is given. From the results, it appears to be inferior to the 2D FFENN. It requires 500 epochs under supervised batch training and approximately a network expansion of 15 hidden neurons (i.e., 46 weights in total) to produce good but not better results than the 2D FFENN.

On the other hand, the RBF network can level the performance of the 2D FFENN at the expense of a large hidden neurons expansion. The associated training error for an RBF network of 42 hidden neurons (i.e., 127 weights in total) is shown in Figure 9.

5.1.3. *System validation*

In order to validate the performance of the 2D FFENN, multilevel 2D FFENN, MLP, and RBF networks, a test data set was constructed using grid spacing of 0.03195, a value chosen to avoid replication of training set points, forcing the network to interpolate. MSLE pruning was performed as it has proven to be a more reliable measurement in comparison to the MSE. All results are shown in Table 1.

5.2. *Application to real sea surface*

5.2.1. *Sea surface approximation*

Detection of small targets in an oceanic environment using a radar system is a difficult task [15, 16, 17, 18, 19, 20], primarily due to the phenomenon of sea clutter, which limits

Table 1: Network training and validation errors.

Network	Pruning	Total no. of weights	Training MSLE	Validation MSLE
2D FFENN	Not applied	59	0.00259	0.00221
2D FFENN	MSLE	30	0.00033	0.00030
Multilevel 2D FFENN	MSLE	60	0.00029	0.00021
MLP	Not applied	46	0.00785	0.00722
RBF	Not applied	127	0.00104	0.00093

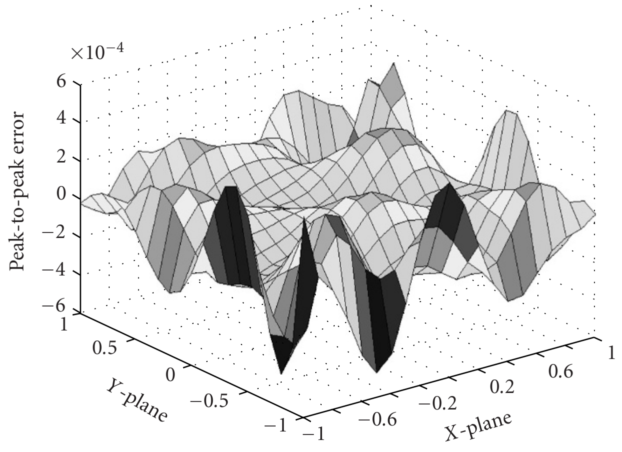

Figure 9: MSLE error surface of RBF (42 hidden neurons).

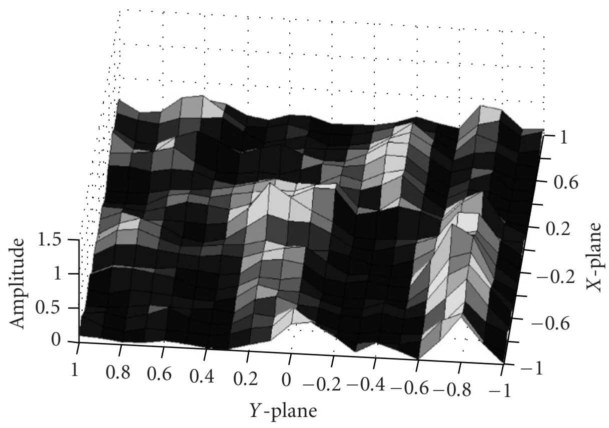

Figure 10: Original sea surface.

the performance of such systems. Sea clutter refers to the backscattered energy returns from a radar illuminated sea surface. A conventional radar system directs a beam of microwave pulses illuminating a patch of the sea surface. The backscattered energy is collected and based on its strength and round trip delay targets can be detected and located. Signal returns from the actual sea surface are widely known as sea clutter, whilst signal returns from an object are known as targets. Therefore, for successful target detection, sea clutter suppression must be achieved. Because sea clutter refers to radar signal returns from the sea surface itself, the problem casts explicitly to sea surface modeling.

In the previous section, we have demonstrated the modeling capability of the 2D FFENN design over two of the most well-known neural network architectures described in the literature. In this section, we investigate the performance of the proposed 2D FFENN design to a real sea surface patch. The actual sea surface texture segment is shown in Figure 10. The data has been taken from a much larger radar data set, provided by QinetiQ, Malvern, with characteristics shown in Table 2. Results have been produced for both the method of surface approximation by the 2D FFENN and MLP. The corresponding modeled surfaces are shown in Figures 11 and 12, respectively. The 2D FFENN network is initially trained for a functional expansion of 134 functions and after 10 epochs of training, pruning achieves reduction of the functional expansion down to 97 significant functions (i.e., 97 weights in total). The MLP is trained for 500 epochs with a hidden layer expansion of 32 neurons (i.e., 97 weights in total).

Table 2: Staring radar data (QinetiQ, Malvern, UK).

Range bin size (m)	0.3
Sea state	2–3
Wind speed (Knots)	10
Wind direction	NE
Polarisation	HH
PRF (Hz)	1000
Sampling period (s)	0.001
Radar mode	Staring
Radar height (m)	50

From the surface approximation results, it is evident that both models are capable of estimating the main surface characteristics of the sea surface segment. Both models experience some difficulty in estimating the sharp peaks of the actual sea surface, however, the main peaks are clearly identified. Modeling with the RBF network was not feasible, as the network was not able to approximate the original surface leading to a very large neuron expansion approaching the length of the original data.

6. CONCLUSION

A 2D functionally expanded neural network was presented in this paper. The network's backbone architecture was described and computer simulations for both computer

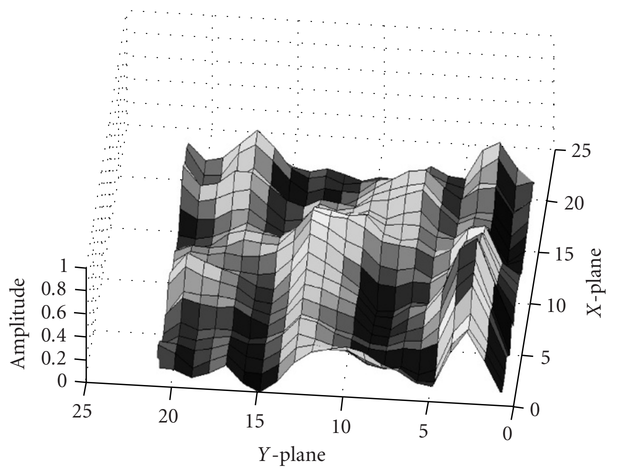

Figure 11: Error surface of multilevel 2D FFENN (MSLE pruning, 97 functions, 10 epochs).

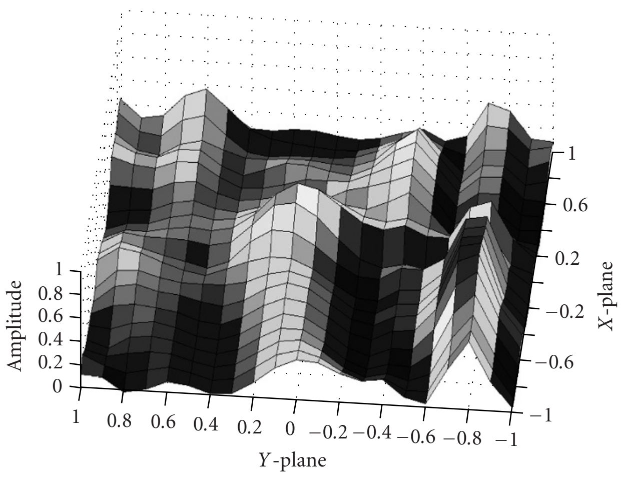

Figure 12: Error surface of MLP (500 epochs, 32 hidden neurons).

generated data and real sea clutter data were presented. A multiscaled functionally expanded structure of 2D FFENN design was also presented, designed to enhance its nonlinear modeling ability for surfaces where discontinuities and spikiness are present. An efficient function pruning strategy was also devised. The results obtained by the proposed system demonstrate the effectiveness of such a network structure to produce surface mappings under short training times. Comparative simulation results were also produced for MLP and RBF networks, validating the surface modeling potential of the 2D FFENN network architecture.

FFENN's flexible functional expansion and short training time makes it a very attractive design for many 2D signal processing applications. We are currently extending the proposed network design to a wavelet-based functionally expanded structure. Our future development plans include first the extension of the static 2D FFENN architecture to a recurrent 2D FFENN design, which will be able to produce dynamic models and second the extension of the 2D structure to a three-dimensional volume modeler, 3D FFENN, which lies on straightforward modifications of the existing system.

ACKNOWLEDGMENTS

The authors are very grateful to QinetiQ, Malvern, in particular Professor I. Proudler and Dr. S. Lycett, for their valuable suggestions and for providing the real sea clutter data used in Section 5.2. This research was sponsored by QinetiQ under the United Kingdom Ministry of Defence Corporate Research Programme CISP.

REFERENCES

[1] D. R. Hush and B. G. Horne, "Progress in supervised neural networks: What's new since lippmann," *IEEE Signal Processing Magazine*, vol. 10, no. 1, pp. 8–39, 1993.

[2] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," *Neural Computation*, vol. 3, no. 2, pp. 246–257, 1991.

[3] D. Lowe and A. R. Webb, "Time series prediction by adaptive networks: a dynamical systems perspective," *IEE Proceedings Part F: Radar and Signal Processing*, vol. 138, no. 1, pp. 17–24, 1991.

[4] S. Haykin and J. Principe, "Making sense of a complex world [chaotic events modeling]," *IEEE Signal Processing Magazine*, vol. 15, no. 3, pp. 66–81, 1998.

[5] S. Haykin, S. Puthusserypady, and P. Yee, "Dynamic reconstruction of sea clutter using regularized RBF networks," in *Proc. IEEE 32nd Asilomar Conference on Signals, Systems & Computers*, vol. 1, pp. 19–23, Pacific Grove, Calif, USA, November 1998.

[6] S. Haykin and H. Leung, "Chaotic model of sea clutter using a neural network," in *Advanced Algorithms and Architectures for Signal Processing IV*, vol. 1152 of *Proceedings of SPIE*, pp. 18–21, San Diego, Calif, USA, 1989.

[7] S. Haykin, *Neural Networks—A Comprehensive Foundation*, pp. 208, 293 & 762, Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 1999.

[8] A. Hussain, J. J. Soraghan, and T. S. Durrani, "A new hybrid neural network structure for non-linear time series modeling," *Journal of Computational Intelligence in Finance*, vol. 5, no. 1, pp. 16–26, 1997, Special issue on hybrid neural networks for financial time series forecasting.

[9] A. Hussain, J. J. Soraghan, T. S. Durrani, and D. R. Campbell, "A new neural network structure for modeling non-linear dynamical systems," in *Proc. IEEE International Workshop on Image and Signal Processing, Advances in Computational Intelligence*, B. G. Mertzios and P. Liatsis, Eds., pp. 119–122, Elsevier Science Publishers, Manchester, UK, November 1996.

[10] A. Hussain, J. J. Soraghan, and T. S. Durrani, "A new adaptive functional-link neural-network-based DFE for overcoming co-channel interference," *IEEE Trans. Communications*, vol. 45, no. 11, pp. 1358–1362, 1997.

[11] A. Hussain and J. J. Soraghan, "FENN methodology: Improved neural network," UK Patent no. 9624298.7, November 1996.

[12] A. Hussain, *Novel artificial neural-network architectures and algorithms for non-linear dynamical system modelling and digital communications applications*, Ph.D. thesis, Strathclyde University, Glasgow, UK, September 1996.

[13] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, Upper Saddle River, NJ, USA, 3rd edition, 1996.

[14] T. Masters, *Practical Neural Network Recipes in C++*, Academic Press, San Diego, Calif, USA, 1993.
[15] K. D. Ward, "Compound representation of high resolution sea clutter," *Electronics Letters*, vol. 17, no. 16, pp. 561–563, 1981.
[16] H. Leung and S. Haykin, "Is there a radar clutter attractor?" *Applied Physics Letters*, vol. 56, no. 6, pp. 593–595, 1990.
[17] J. L. Noga, *Bayesian state-space modelling of spatio-temporal non-Gaussian radar returns*, Ph.D. thesis, Cambridge University, Cambridge, UK, March 1999.
[18] M. Davies, "Looking for nonlinearities in sea clutter," in *IEE/EUREL Workshop on Radar and Sonar Signal Processing*, Peebles, Scotland, July 1998.
[19] S. Haykin and S. Puthusserypady, *Chaotic Dynamics of Sea Clutter*, John Wiley & Sons, New York, NY, USA, 1999.
[20] M. Cowper and B. Mulgrew, "Nonlinear processing of high resolution radar sea clutter," in *Proc. IEEE International Joint Conference on Neural Networks (IJCNN '99)*, vol. 4, pp. 2633–2638, Washington, DC, USA, July 1999.

S. Panagopoulos was born in Aegina, Greece, in 1977. He received the B.Eng. (honors) degree in electronic and electrical engineering from Strathclyde University, Glasgow, UK, in 2000. He is currently working toward the Ph.D. degree in the Institute for Communications and Signal Processing, University of Strathclyde, Glasgow, UK. His current research interests include nonlinear time series modeling, wavelets, neural network design, chaotic analysis, radar sea clutter suppression, and target detection.

J. J. Soraghan holds the Texas Instruments Chair in signal processing in the Institution of Communications and Signal Processing, Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, Scotland. He received his B.Eng. degree with first-class honours in 1978 and his M.Eng.Sc. degree in 1982 both from University College Dublin, Ireland. He received his Ph.D. degree in electronic engineering in 1989 from the University of Southampton, UK. From 1979 to 1980, he worked with Westinghouse Electric Corporation, USA. He joined the Department of Electronic and Electrical Engineering in 1986 as a Lecturer in the Signal Processing Division, became a Senior Lecturer in 1999, a Reader in 2001, and a Professor in 2003. From 1989 to 1991, he was a Manager of the Scottish Transputer Centre and from 1991 to 1995, he was a Manager of the DTI Centre for Parallel Signal Processing. Since 1996, he has been a Manager of the Texas Instruments' DSP Elite Centre in the University. His main research interests include advanced linear and nonlinear signal, image, and video processing algorithms, and wavelets and fuzzy systems with applications to telecommunications, biomedical, and remote sensing. He has supervised 17 Ph.D. students to graduation, holds 3 patents, and has published over 170 technical papers. He is a Member of the Institute of Electrical Engineers (IEE) and a Senior Member of the Institute of Electrical and Electronic Engineers (IEEE).