Complex Systems **2** (1988) 321–355

Multivariable Functional Interpolation and Adaptive Networks

D. S. Broomhead*
Royal Signals and Radar Establishment,
St. Andrews Road, Great Malvern, Worcs. WR14 3PS, United Kingdom

David Lowe†
Speech Research Unit, Royal Signals and Radar Establishment,
St. Andrews Road, Great Malvern, Worcs. WR14 3PS, United Kingdom

Abstract. The relationship between "learning" in adaptive layered networks and the fitting of data with high dimensional surfaces is discussed. This leads naturally to a picture of "generalization" in terms of interpolation between known data points and suggests a rational approach to the theory of such networks. A class of adaptive networks is identified which makes the interpolation scheme explicit. This class has the property that learning is equivalent to the solution of a set of *linear* equations. These networks thus represent nonlinear relationships while having a guaranteed learning rule.

1. Introduction

The strong resurgence of interest in "self-learning machines", whose ancestors include the perceptrons of the 1960's, is at least partly driven by the expectation that they will provide a new source of algorithms/architectures for the processing of complex data. This expectation is based on an analogy with connectionist models of information processing in animal brains. The brain is able to cope with sophisticated recognition and inductive tasks far beyond the capabilities of systems based on present computing logic and architectures.

An example of an everyday task for the human brain which illustrates this point, is the recognition of human speech. For automatic speech recognition one wishes to deduce properties (the implied message) from the statistics of a finite input data set even though the speech will be subject to nonlinear

*Electronic mail address from USA: `dsb%rsre.mod.uk@relay.mod.uk`; electronic mail address from the UK: `dsb%rsre.mod.uk@uk.ac.ucl.cs.nss`.

†Electronic mail address from USA: `dl%rsre.mod.uk@relay.mod.uk`; electronic mail address from the UK: `dl%rsre.mod.uk@uk.ac.ucl.cs.nss`.

distortions due to noise, sex, age, health, and accent of the speaker. This has proved to be notoriously difficult. Over the past decade, one of the most successful techniques developed for speech recognition has been based around Hidden Markov Models (see for instance [1,2,3]). In this scheme, speech is modelled as a sequence of causal stationary stochastic processes determined by a finite number of allowable states, state transitions, a matrix of stationary transition probabilities and an initial state distribution. There is, in addition, a learning algorithm (Baum–Welch iteration) to fine tune the parameters of the model which increases the likelihood that each model produces its associated data. However, one of the problems with this approach involves the *a priori* assumptions made regarding the topology of the Hidden Markov model (e.g., number of states, allowable transitions). As long as the total possible input data is consistent with the assumptions made in the original model, then one can expect a faithful representation of the data. This unfortunately presupposes that we already know how to model speech adequately.

In an attempt to circumvent this problem, self-learning machines have been employed. The virtue of these is that no *explicit* model of speech is required. For instance, the multilayer perceptron (which is a layered nonlinear network) embodies a set of nonlinear, "hidden" units whose task is to encode the higher-order constraints of the input data [4]. This is achieved by varying weights governing the strengths of connections between the units in order to minimize the error in relating known input-output pairs (the "training" set). This process has become known as "learning." The ability of the network to give subsequently reasonable (in some sense) outputs for inputs not contained in the training set is termed "generalization." In effect, a multilayer perceptron of given geometry with given nonlinear responses of the units constitutes an M-parameter family of models (where M is the total number of weights which are varied). It is currently an act of faith, based on encouraging practical results, that such families are broad enough to include models of speech which are adequate for the purposes of classification. This means, however, that the design of a multilayer perceptron for a specific task remains an empirical art. In particular, how many hidden units should be employed and in what configuration, how much training data is needed, and what initial interconnection strengths have to be assigned? Some experimental work has been performed which addresses these problems (for instance, [5,6]) although it is fair to comment that an understanding of these issues is still lacking. The source of the difficulty is the implicit relationship between these externally controllable factors, and the model ultimately represented by the network.

The present paper investigates the implicit assumptions made when employing a feed-forward layered network model to analyze complex data. The approach will be to view such a network as representing a map from an n-dimensional input space to an n'-dimensional output space, say $s : \mathbb{R}^n \to \mathbb{R}^{n'}$. This map will be thought of as a graph $\Gamma \subset \mathbb{R}^n \times \mathbb{R}^{n'}$ (in the same way that $s : \mathbb{R} \to \mathbb{R}$, where $s(x) = x^2$ may be thought of as a parabola drawn

in $\mathbb{R}^2$). From this point of view, "error free" training data presented to the network in the form of input-output pairs represent points on the graph, Γ and the learning phase of an adaptive network constitutes the optimization for a fitting procedure for Γ based on the known data points. Generalization is therefore synonymous with *interpolation* between the data points with the interpolation being along the constrained surface generated by the fitting procedure as the optimum approximation to Γ. In this picture the implicit assumptions made when using a multilayer perceptron concern the nature of the fitting procedure, and clearly relate directly to the way in which the network generalizes. Thus, we are led to the theory of multivariable interpolation in high dimensional spaces. In subsequent sections, we shall exploit some of the mathematics of this expanding field of research to develop a new type of layered network model in which the nature of the fitting procedure is explicit. This class of layered network model will be shown to be of considerable interest in itself. In addition, however, it is hoped that the explicit nature of the fitting procedure will allow us to develop a better understanding of the general properties of layered nonlinear networks which perform an equivalent function.

2. Multivariable functional interpolation using radial basis functions

This section introduces briefly the method of *Radial Basis Functions*, a technique for interpolating in a high dimensional space which has recently seen important developments. Further details may be obtained from the review article of Powell [7] and the important contribution of Micchelli [8].

In the cited references the radial basis function approach is applied to the strict interpolation problem which may be summarized as follows:

Problem. *Given a set of m distinct vectors (data points), $\{\underline{x}_i; i = 1, 2, \ldots, m\}$ in $\mathbb{R}^n$ and m real numbers $\{f_i; i = 1, 2, \ldots, m\}$, choose a function $s : \mathbb{R}^n \to \mathbb{R}$ which satisfies the interpolation conditions*

$$s(\underline{x}_i) = f_i \quad i = 1, 2, \ldots, m \tag{2.1}$$

Note that the function, s, is constrained to go through the known data points.

There are clearly many criteria one could impose which would restrict the possible functional form of $s(\underline{x})$ (see for instance [9]). The Radial Basis Function approach constructs a linear function space which depends on the positions of the known data points according to an arbitrary distance measure. Thus, a set of m arbitrary (generally nonlinear) "basis" functions $\phi(||\underline{x} - \underline{y}_i||)$ is introduced, where $\underline{x} \in \mathbb{R}^n$ and $||\ldots||$ denotes a norm imposed on $\mathbb{R}^n$ which is usually taken to be Euclidean. The vectors $\underline{y}_i \in \mathbb{R}^n$,$i = 1, 2, \ldots, m$ are the centers of the basis functions and taken to be sample data points. In terms of these basis functions, we consider interpolating functions of the form:

$$s(\underline{x}) = \sum_{j=1}^{m} \lambda_j \phi(||\underline{x} - \underline{y}_j||) \quad \underline{x} \in \mathbb{R}^n \tag{2.2}$$

Inserting the interpolation conditions, equation (2.1) into equation (2.2), gives the following set of linear equations for the coefficients $\{\lambda_j\}$,

$$\begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix} = \begin{pmatrix} A_{11} & \dots & A_{1m} \\ \vdots & \ddots & \vdots \\ A_{m1} & \dots & A_{mm} \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_m \end{pmatrix} \tag{2.3}$$

where

$$A_{ij} \triangleq \phi(||\underline{x}_i - \underline{y}_j||) \quad i,j = 1,2,\dots,m \tag{2.4}$$

Given the existence of the inverse to matrix $\mathbf{A}$ with elements A_{ij}, equation (2.3) uniquely defines the coefficients λ_j through $\underline{\lambda} \equiv \mathbf{A}^{-1}\underline{f}$.

In general, one might expect that for an arbitrary choice of ϕ the matrix $\mathbf{A}$ could be singular. However, the results of Micchelli prove that for all positive integers m, n and for a large class of functions ϕ, the matrix $\mathbf{A}$ is non-singular if the data points are all distinct .

This discussion readily generalizes to maps $s : \mathbb{R}^n \to \mathbb{R}^{n'}$. In this case the m distinct data points in $\mathbb{R}^n$ are associated with m vectors $\underline{f}_i \in \mathbb{R}^{n'}$. The interpolation condition of equation (2.1) thus generalizes to

$$s_k(\underline{x}_i) = f_{ik} \quad i = 1,2,\dots,m \;\; k = 1,2,\dots,n' \tag{2.5}$$

which leads to interpolating functions of the form

$$s_k(\underline{x}) = \sum_{j=1}^{m} \lambda_{jk} \phi(||\underline{x} - \underline{y}_j||) \quad \underline{x} \in \mathbb{R}^n \quad \mathrm{k} = 1,2,\dots,\mathrm{n}' \tag{2.6}$$

The expansion coefficients λ_{jk} are obtained using the inverse of the same matrix $\mathbf{A}$ defined in equation (2.4).

Once a suitable choice of the function ϕ is made, and a convenient distance measure imposed, the above relations exactly specify the interpolation problem which has a guaranteed solution.

However, for certain classes of problem, the above analysis may not be a good strategy for the following reason. A basic consideration when fitting data is the number of degrees of freedom required. That is, the minimum number of basis functions needed to generate an acceptable fit to the data. In the situation where the number of data points far exceeds the number of degrees of freedom there will be redundancy since we are constrained to use as many radial basis functions as data points. In this case the strict interpolation conditions generally result in this redundancy being used to fit misleading variations due to imprecise, or noisy data.

It is possible to avoid this difficulty by weakening the interpolation conditions. We suggest the following generalizations to the conventional radial basis function approach. First, it may be necessary to distinguish between

the data points, $(\underline{x}_i,\ i = 1, 2, \ldots, m)$ and the radial basis function centers, $(\underline{y}_j,\ j = 1, 2, \ldots, n_0 \quad n_0 < m)$ [1]. The problem thus becomes overspecified, the matrix $\mathbf{A}$ is not square, a unique inverse no longer exists, and the previous exact problem becomes one of linear optimization. In the following, we shall adopt a minimum norm least squares method by introducing the Moore-Penrose pseudo-inverse, $\mathbf{A}^+$ of matrix $\mathbf{A}$. For the case where $rank\mathbf{A} = n_0$, this has the property that $\mathbf{A}^+\mathbf{A} = \mathbf{1}$ where $\mathbf{1}$ denotes the $n_0 \times n_0$ identity matrix. More generally, the pseudo-inverse provides a unique solution to the linear least squares problem [10] in the following sense,

Of all the vectors $\underline{\lambda}$ which minimize the sum of squares $||\mathbf{A}\underline{\lambda} - \underline{f}||^2$, the one which has the smallest norm (and hence minimizes $||\underline{\lambda}||^2$) is given by $\underline{\lambda} = \mathbf{A}^+\underline{f}$.

For this particular solution set, an expression may be derived for the normalized error $\mathcal{E}$, specifically,

$$\begin{aligned} \mathcal{E} &= \sqrt{\frac{\sum_{i=1}^{m} ||\underline{s}(\underline{x}_i) - \underline{f}_i||^2}{\sum_{i=1}^{m} ||\underline{f}_i - \langle \underline{f} \rangle||^2}} \\ &\equiv \sqrt{\frac{\sum_{i=1}^{m} ||\sum_{i'=1}^{m} (\mathbf{A}\mathbf{A}^+)_{ii'} \underline{f}_{i'} - \underline{f}_i||^2}{\sum_{i=1}^{m} ||\underline{f}_i - \langle \underline{f} \rangle||^2}} \end{aligned} \tag{2.7}$$

where $\langle \underline{f} \rangle$ is the mean value of the response vector over all the *training* data points. Note that if the pseudo-inverse equals the exact inverse, then the left and right inverses are the same and hence the matrix product $\mathbf{A}\mathbf{A}^+$ is the m-dimensional identity matrix, and this error is zero.

An additional modification, which is useful particularly when $\underline{f}(\underline{x})$ has a large $\underline{x}$-independent component, is to incorporate constant offsets $\{\lambda_{0k}\}$ into the form of the interpolating functions

$$s_k(\underline{x}) = \lambda_{0k} + \sum_{j=1}^{m} \lambda_{jk} \phi(||\underline{x} - \underline{y}_j||) \quad \underline{x} \in \mathbb{R}^n \quad k = 1, 2, \ldots, n' \tag{2.8}$$

These coefficients enter the least squares formalism through an additional column in $\mathbf{A}$

$$A_{i0} = 1 \quad i = 1, 2, \ldots, m \tag{2.9}$$

In this form, the radial basis function approach to multivariable functional interpolation has a close resemblance to adaptive network theory. This will be discussed in the following sections. An important consequence of this approach which should be emphasized is that the determination of the nonlinear map $\underline{s}(\underline{x})$ has been reduced to a problem in linear algebra. The coefficients λ_{jk} appear *linearly* in the functional form of the mapping, therefore the problem of determining the precise values, even in an overdetermined situation,

[1] In particular, we do not necessarily require that the radial basis function centers correspond to any of the data points.

has been reduced to one of a *linear* least squares optimization which has a "guaranteed learning algorithm" through the pseudo-inverse technique [2]. Of course, this reduction has assumed suitable choices for the centers, $\{\underline{y}_j\}$ and the function ϕ. It may be argued, therefore, that the restriction of the optimization by fixing these quantities is excessive and must limit the range of applicability of the approach. In the case of the strict interpolation this does not seem to be the case, at least as far as the choice of ϕ is concerned [11]. There is evidence to show [12], again for strict interpolation, that the effect of a suboptimal choice of the $\{\underline{y}_j\}$ is to reduce the rate of convergence of the expansion given in equation (2.6). For the least squares extensions described here, much less is known.

3. The radial basis function method viewed as a layered network

Much of the power and attraction of current adaptive network models is contained in the nonlinear aspects of the hidden units so that nonlinear input-output relationships may be modelled. Unfortunately, since the error criterion, or cost function, depends on the response of the nonlinear elements, the problem of finding a globally optimum solution becomes one of unconstrained nonlinear least squares minimization. Such problems can be solved usually only by iterative techniques. For instance, "error back-propagation" is a first-order (only depends on the gradient of the function of interest) steepest descent technique which suffers from slow convergence properties due to its tendency to zig-zag about the true direction to a minimum. More sophisticated iterative schemes derived from second-order approximations have been proposed which improve convergence properties (the *quasi-Newton*, or *variable metric* algorithms–see for instance [13], Chapter 15 and [14]). Recent works of note which have considered more general iterative strategies and found them to be orders of magnitude superior to back-propagation when applied to specific layered network problems are those of Lapedes and Farber [15] (*conjugate gradients*) and Watrous [16] (*Davidon-Fletcher-Powell* and *Broyden-Fletcher-Goldfarb-Shanno*). In spite of this effort, the difficulty remains that the solution obtained by such methods is not guaranteed to be the global optimum since local minima may be found. There is no reason in the current iterative schemes why a least squares minimization solution obtained, will necessarily have the required form. Even choosing a good initial starting point for the iteration schemes will not necessarily imply that a good approximation to the global minimum will be obtained. It is important to know how "good" a solution is obtained by settling for a local minimum, and under what conditions the solution at such a minimum has to be deemed unsatisfactory.

In the previous section it was shown that because of the linear dependence on the weights in the radial basis function expansion, a globally optimum

[2] As a technical numerical point, the solution will not generally be obtained from the normal equations (which may be ill-conditioned), but would be obtained via the efficient procedure of singular valued decomposition.

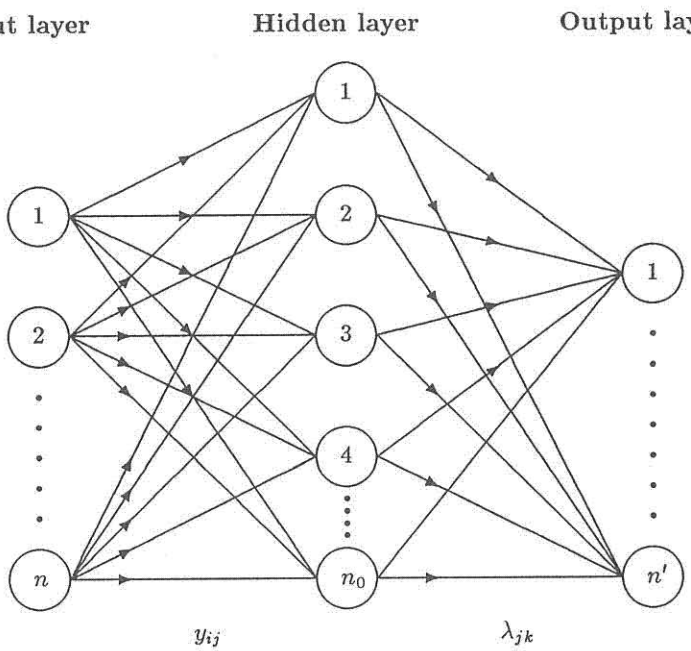

Figure 1: A schematic diagram of the feed-forward layered network model represented by the radial basis function expansion.

least squares interpolation of nonlinear maps can be achieved. The relevance of this to layered network models is that the mapping produced by the radial basis function expression equation (2.6), has the form of a weighted sum over nonlinear functions. There is thus a natural correspondence with the following general three-layer network system, in which the layers are fully interconnected with adjacent layers, but there are no interconnections within the layers (see figure 1).

The input layer of this network model is a set of n-nodes waiting to accept the components of the n-dimensional vector $\underline{x}$. These input nodes are directly connected to all of the hidden nodes. Associated with each connection is a scalar (y_{ij} for the link between the i^{th} input node and the j^{th} hidden node) such that the fan-in to a given node has the form of a hypersphere, i.e. in the case of a Euclidean norm, the input to the node is $\theta_j = \sum_{i=1}^{n}(x_i - y_{ij})^2$ where the x_i are components of $\underline{x}$. The "hidden layer" consists of a set of n_0 nodes, one for each radial basis function center. The output of each of these is a scalar, generally nonlinear function of θ_j. The hidden layer is fully connected to an output layer corresponding to the n'-components of the n'-dimensional response vector $\underline{s}(\underline{x})$ of the network. The input value received by each output unit is a weighted sum of all the outputs of the hidden units, where the strengths of connections from the j^{th} hidden unit to the k^{th} output unit are denoted by λ_{jk}. The response of each output unit is

a linear function of its net input which may include the bias λ_{0k}. A natural extension is to allow for nonlinearity in the response of the output units. Clearly, if the transfer function of these units is invertible, then it can be accounted for by a suitable modification of the interpolation conditions used to derive the weights $\{\lambda_{jk}\}$. This specifies, as a layered network model, the radial basis function expansion, equation (2.8), which produces a mapping from an n-dimensional input space to an n'-dimensional target space.

This type of network falls within the general class of nonlinear layered feedforward networks. In particular we have specialized the geometry to a single hidden layer and fixed the fan-in and fan-out to the hidden units as indicated. The choice of hyperspherical fan-in to the hidden units has the effect of sectioning the decision space into hyperspherical regions rather than hyperplanes which result from the more usual choice of a scalar product type of fan-in. This has the advantage of allowing disjoint regions in the decision space, but which pertain to the same classification, to be satisfactorily resolved by the single "hidden" adaptive layer. Problems without simple connectivity would traditionally require *two* hidden adaptive layers, as discussed in [17], whereas the approach described in this paper can deal with such problems by employing a single hidden layer.

It is interesting to compare the radial basis function network with the traditional multilayer perceptron and the linear perceptron as discussed by Minsky and Papert [18]. The "learning scheme" of conventional multilayer perceptrons consists of an unconstrained nonlinear least squares optimization process and as such there are no global theorems regarding convergence to the correct minimum error solution. In contrast, the radial basis function network has a guaranteed learning procedure since there is only a single layer of adjustable weights which may be evaluated according to linear optimization techniques for which global existence theorems are known. In this sense, the radial basis function networks are more closely related to the early linear perceptrons. However, in contrast to these early networks, the radial basis function network is capable of representing arbitrary nonlinear transformations as determined by a finite training data set of input-output patterns, since the results of Micchelli show that the radial basis function approach can produce an interpolating surface which exactly passes through all the pairs of the training set. Of course, in application the exact fit is neither useful nor desirable since it may produce anomalous interpolation properties. In practical situations a regularization, equivalent to assuming the existence of a smooth interpolation between known data pairs, is imposed. One such regularization is the least squares construction introduced in section 2. The ability of radial basis function networks to explicitly model nonlinear relationships will be demonstrated in subsequent sections by application to specific problems which demand the construction of a nonlinear mapping in order to produce the correct solution.

Our radial basis function strategy may be applied to the general multilayer perceptron for which the output units have an invertible nonlinearity. In particular, when extended to allow for variation in the input-hidden

Input Pattern		Number of "ON" bits in input		Output Pattern
00	→	0	→	0
01	→	1	→	1
10	→	1	→	1
11	→	2	→	0

Table 1: Symbolic mapping of the exclusive-OR problem.

weights, this method provides an interesting picture of learning. The hidden-output weights may be visualized as evolving on a different "time scale" to the input-hidden weights. Thus, as the input-hidden weights evolve slowly by some nonlinear optimization strategy, the hidden-output weights adjust themselves rapidly through linear optimization so as to always remain in the global minimum of an evolving error surface over the hidden-output weights which is parametrically controlled by the input-hidden weights. The hidden-output weights are "slaved" to the behavior of the input-hidden weights.

The rest of this paper is concerned with various simple applications of the radial basis function network. The total set of adjustable parameters include the set of n_0-radial basis function centers, $\underline{y}_j$, as well as the set of $(n_0 + 1) \times n'$ weights, λ_{jk}. However, only the latter are included in the least squares analysis in this paper in order to preserve the linearity of the learning procedure. In the absence of any *a priori* knowledge, the centers, $\{\underline{y}_j\}$ are either distributed uniformly within the region of $\mathbb{R}^n$ for which there is data, or they are chosen to be a subset of the training points by analogy with strict interpolation. We expect that with additional knowledge of the surface to be fitted, the freedom to position the centers may be used to advantage to improve the convergence of the expansion (although not necessarily to improve the "fit" to the unseen data). Evidence for this as well as insight into the significance of the centers follows from the work of Powell [11] who showed that for strict interpolation when $n = n' = 1$ and when $\phi(r) = r^{2k+1}$, $(k = 0, 1, \ldots)$, the radial basis function method is equivalent to interpolation with natural splines. In this case the $\{\underline{y}_j\}$ are the knots of the spline fit. Naturally, when the strict interpolation is weakened to give a least squares interpolation, the significance of the "knots" in constraining the surface is also weakened. In what follows, we shall attempt to be as general as possible in the analytic work by assuming an arbitrary form of ϕ. Where numerical work necessitates a specific choice, we have chosen to employ either a Gaussian form ($\phi(r) \approx exp[-r^2]$) or a multiquadric ($\phi(r) \approx \sqrt{c^2 + r^2}$).

4. Specific example (i): the exclusive-OR problem and an exact solution

The exclusive-OR problem defined by the symbolic mapping in table 1 has been considered interesting because points which are closest (in terms of the Hamming distance) in the input space, map to regions which are maximally

apart in the output space. This is a classic example of a logic function that cannot be represented by a linear perceptron. Clearly, a function which interpolates between the points given in Table 1 must oscillate, and may be hard to represent using a subset of data points unless there is some built in symmetry in the radial basis functions.

In what follows, we initially take one radial basis function center determined by each piece of input data, so that both $\underline{y}_j, \underline{x}_i$ are selections of the four ordered input patterns of table 1. We choose to number the four possible input patterns as $(0,0) \rightarrow 1$, $(0,1) \rightarrow 2$, $(1,1) \rightarrow 3$, $(1,0) \rightarrow 4$ which we visualize to be the cyclically ordered corners of a square. The action of the network may be represented by

$$s(\underline{x}) = \sum_{j=1}^{4} \lambda_j \phi(||\underline{x} - \underline{y}_j||)$$

so that the set $\{\lambda_j\}$ may be found from

$$f_i = \sum_{j=1}^{4} \lambda_j \phi(||\underline{x}_i - \underline{y}_j||)$$

i.e.

$$\underline{\lambda} = \mathbf{A}^{-1} \underline{f}$$

For the ordering we have chosen, the vector $\underline{f}$ and the matrix $\mathbf{A}$ take the specific forms

$$\underline{f} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \tag{4.1}$$

and

$$\mathbf{A} = \begin{pmatrix} \phi_0 & \phi_1 & \phi_{\sqrt{2}} & \phi_1 \\ \phi_1 & \phi_0 & \phi_1 & \phi_{\sqrt{2}} \\ \phi_{\sqrt{2}} & \phi_1 & \phi_0 & \phi_1 \\ \phi_1 & \phi_{\sqrt{2}} & \phi_1 & \phi_0 \end{pmatrix} \tag{4.2}$$

where a labeling notation has been introduced so that ϕ_0 denotes the value of $\phi(||\underline{x}_i - \underline{x}_i||)$, ϕ_1 denotes the value $\phi(||\underline{x}_i - \underline{x}_{i\pm1}||)$ and $\phi_{\sqrt{2}}$ represents $\phi(||\underline{x}_i - \underline{x}_{i\pm2}||)$, all counting being performed cyclically around the square. Note that this is a labeling device and we will not exploit the properties of any particular distance function at this stage (although the notation $\phi_{\sqrt{2}}$ indicates that we have in mind a Euclidean metric as an example).

It will be convenient to construct $\mathbf{A}^{-1}$ from the eigenvalue decomposition (see Appendix A for further details)

$$\mathbf{A} \equiv \mathbf{V} \underline{\underline{\mu}} \mathbf{V}^T$$

Since **A** is real symmetric, **V** is an orthogonal matrix with columns composed out of the orthogonal eigenvectors; in this case,

$$\mathbf{V} = \frac{1}{2}\begin{pmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & -1 & 0 & -\sqrt{2} \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & -\sqrt{2} \end{pmatrix} \tag{4.3}$$

and $\underline{\underline{\mu}}$ is the real, diagonal matrix

$$\underline{\underline{\mu}} = \begin{pmatrix} \mu_A & 0 & 0 & 0 \\ 0 & \mu_B & 0 & 0 \\ 0 & 0 & \mu_E & 0 \\ 0 & 0 & 0 & \mu_E \end{pmatrix}$$

where

$$\begin{aligned} \mu_A &= (\phi_0 + 2\phi_1 + \phi_{\sqrt{2}}) \\ \mu_B &= (\phi_0 - 2\phi_1 + \phi_{\sqrt{2}}) \\ \mu_E &= (\phi_0 - \phi_{\sqrt{2}}) \end{aligned} \tag{4.4}$$

Note that at this stage it is possible to decide how well posed the original problem was, by seeing under what conditions an inverse matrix $\mathbf{A}^{-1}$ exists. It is clear from the form of the derived eigenvalues of the problem, that an inverse exists as long as

$$\phi_0 \neq \phi_{\sqrt{2}} \tag{4.5}$$

or

$$\phi_1 \neq \pm\left(\frac{\phi_0 + \phi_{\sqrt{2}}}{2}\right) \tag{4.6}$$

are satisfied. Thus far, the analysis has been carried out for an arbitrary choice of nonlinear radial basis function and metric, therefore the above conditions can be taken to be restrictions on the various combinations of radial basis function and metric that may be reasonably employed for this problem. It is interesting to point out two situations where an inverse does *not* exist:

$\phi(x) = mx + c$ combined with a city-block metric

and $\phi(x) = \alpha x^2 + c$ combined with a Euclidean distance function, $(||\underline{x}|| \equiv \sqrt{\sum_i x_i^2})$

as may be easily verified by direct substitution into equation (4.6). These instances correspond to the network structure of a *linear* perceptron and are thus unable to represent the exclusive-OR function.

Given the inverse of $\mathbf{A}$, we may proceed to obtain the "weights" of the network as

$$\begin{aligned} \underline{\lambda} &= \mathbf{A}^{-1}\underline{f} \\ &\equiv \mathbf{V}\underline{\underline{\mu}}^{-1}\mathbf{V}^T\underline{f} \\ &\equiv \mathbf{V}\underline{\underline{\mu}}^{-1}\mathbf{V}^T(\underline{v}_A - \underline{v}_B) \end{aligned} \tag{4.7}$$

where we have exploited the fact that the response vector $\underline{f}$, equation (4.1), may be decomposed into the difference of the basis vectors $\underline{v}_A$ and $\underline{v}_B$ derived in the appendix (Appendix A). Performing the above operations (which is simplified since vectors orthogonal to $\underline{v}_A, \underline{v}_B$ do not contribute) gives the final result that

$$\underline{\lambda} = \frac{1}{2}\begin{pmatrix} \mu_A^{-1} - \mu_B^{-1} \\ \mu_A^{-1} + \mu_B^{-1} \\ \mu_A^{-1} - \mu_B^{-1} \\ \mu_A^{-1} + \mu_B^{-1} \end{pmatrix} = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_1 \\ \lambda_2 \end{pmatrix} \tag{4.8}$$

where, explicitly

$$\begin{aligned} \lambda_1 &= \frac{-2\phi_1}{\left[\phi_0+\phi_{\sqrt{2}}\right]^2 - 4\phi_1^2} \\ \lambda_2 &= \frac{\left(\phi_0+\phi_{\sqrt{2}}\right)}{\left[\phi_0+\phi_{\sqrt{2}}\right]^2 - 4\phi_1^2} \end{aligned} \tag{4.9}$$

Equation (4.8) specifies the choice of network weights which exactly solves the exclusive-OR problem. The weights are still dependent on the precise choice of radial basis function and distance measure. Clearly we are free to choose these subject to condition (4.6) without affecting the solution of exclusive-OR. This choice does however influence the output of the network for *arbitrary*, real-valued inputs.

Figure 2 illustrates the solution for the specific choice of a Euclidean distance function and Gaussian radial basis functions ($\phi(x) = \exp[-x^2/\sigma]$). Similarly, figure 3 shows the output using the same distance function, but employing multiquadric radial basis functions ($\phi(x) = \sqrt{1 + x^2/\sigma}$). In both instances, the mapping surfaces have two planes of reflection symmetry through the diagonals of the unit square. The difference is that the Gaussian choice produces two maxima near to the odd parity inputs and two shallow minima close to the even parity inputs. The multiquadric does not have these maxima and moreover diverges rapidly outside the unit square. This distinction is of no relevance to the exclusive-OR function itself. It would however, be significant were it attempted to give meaning to input and output values other than those represented by zero and unity. Clearly, the details of the generalization would then be dependent on the specific interpolation scheme represented by the network.

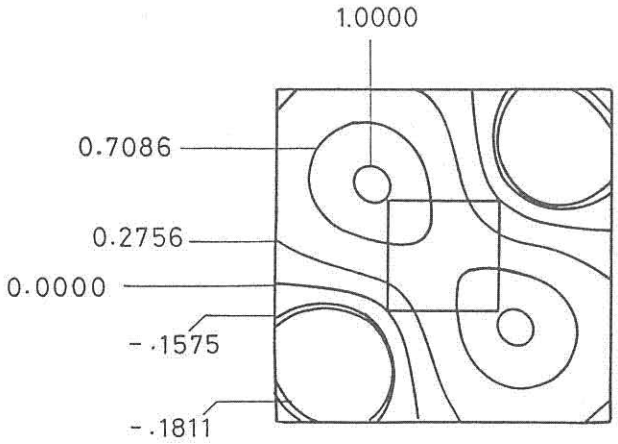

Figure 2: The exclusive-OR solution (i). Obtained using a Euclidean distance function and *Gaussian* radial basis functions centered at the corners of the unit square.

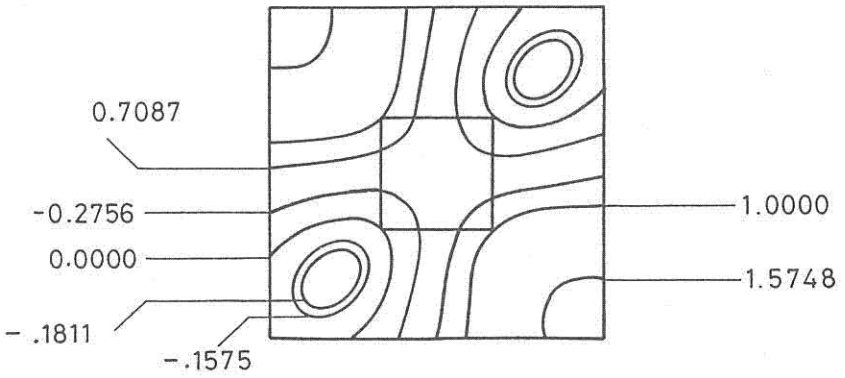

Figure 3: The exclusive-OR solution (ii). Obtained using a Euclidean distance function and *multiquadric* radial basis functions centered at the corners of the unit square.

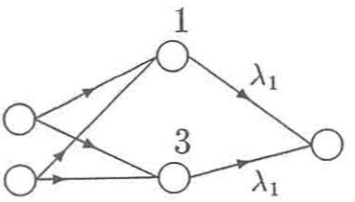

Figure 4: Equivalent network for exclusive-OR with λ_2 set to zero.

We conclude this section with a discussion on the number of "hidden units" employed to solve the problem. Note that the problem has been solved *exactly*; given the weights as determined by equation (9) and a specific choice of a radial basis function, applying any of the input pattern pairs will guarantee to get the correct output answer. On preliminary inspection this may not seem so surprising since each possible input data point was used as a center for a radial basis function and so a "dictionary" of possibilities could be encoded.[3]

One can exploit the symmetry of the solution however, to show how it is still possible to solve the exclusive-OR problem *exactly* without explicitly specifying the response of the whole set of input states. Specifically, from equation (9) and by a judicious or "fortuitous" choice of nonlinear function ϕ (for instance if $\phi_1 = 0$ or $\phi_0 = -\phi_{\sqrt{2}}$) then two of the four possible weights would be zero. This uncouples the corresponding pair of hidden units from the system, with the result that the remaining network satisfies the exclusive-OR function without being explicitly "trained" on the entire possible set of input/output pairs.

For the case that $\phi_1 = 0$ (figure 5) the two identical weights connecting the two hidden units to the output unit have a value of $1/[\phi_0 + \phi_{\sqrt{2}}]$. In this case, the hidden units centered on the patterns $(0,0),(1,1)$ have no connections to the output and hence cannot contribute. Thus, when these patterns are presented to the network, the two units which would react most strongly to their presence have been disconnected from the output unit while the remaining two respond with $\phi_1 = 0$ as expected. Alternatively, if the patterns $(0,1),(1,0)$ are presented, one hidden unit contributes a value of ϕ_0 and the other a value of $\phi_{\sqrt{2}}$. Since their sum is just $1/\lambda_2$ the result of the network is to give the answer 1 as it should. A similar argument may be presented for the case when $\phi_0 = -\phi_{\sqrt{2}}$.

In either case, the surfaces shown in figure 2 and figure 3 are constrained

[3]However, note that this scheme has achieved a fit with four adjustable parameters, the weights λ_j, whereas the standard 2-2-1 multilayer perceptron would employ nine adjustable parameters.

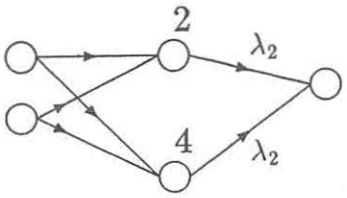

Figure 5: Equivalent network for exclusive-OR with λ_1 set to zero.

sufficiently for the specification of just two points to fix the remaining pair of output values. Here we have, by a suitable choice of ϕ, adjusted the form of the fitting algorithm to admit a network which "generalizes" the incomplete training data to give the entire exclusive-OR function. This sort of procedure can of course be employed by a multilayer perceptron . However, it should be clear that for a strongly folded surface such as represents the exclusive-OR function (or the more general n-bit parity function shown in Appendix B) the presence or absence of redundant points which may be omitted from the training set must depend sensitively on the implicit fitting procedure employed by the network. Moreover, the question of *which* points are redundant must also require detailed specific knowledge of the network and the relationship it is being used to represent. As a rule, one can expect a network to be capable of "correctly" generalizing, only when there is sufficient training data *appropriately distributed* to enable an adequate fit to significant turning points of the underlying graph. Clearly, the more folded this graph is, the more demanding will be the requirement on the data.

5. An analytic solution to a non-exact problem: the exclusive-OR problem with two centers

The previous section, with its related appendices, dealt with exact solutions to *strict* interpolation problems. For strict interpolation the interpolating surface is constrained to pass through *all* the training data points. This is achieved by using the formalism described in the first part of section 2 which requires the use of a radial basis function (or hidden unit) for each distinguishable data pair. It was noted that this rule may be relaxed in special circumstances where the symmetry and other details of the problem may be employed.

In this section, we shall consider a specific example of the more general approach discussed at the end of section 2 which relaxes the strict interpolation of the data. Recall that sufficient scope was allowed in the variant of radial basis function techniques to accommodate an *approximate*interpolating surface whereby this surface is not directly constrained to go through all (or

any) of the training set. This is clearly an advantageous strategy when the input data is corrupted in some fashion by external sources and it would not be desirable to try and fit the noise added onto the (presumably) structured data. In addition, where the true data actually represents a smooth map, it allows the use of far fewer hidden units than data points.

We repeat the analysis of the exclusive-OR problem considered in the previous section, but using just two radial basis function centers. It is clear that there are two distinct choices how the two centers may be positioned: either on opposing vertices, or adjacent vertices on the ordered corners of the unit square. We choose the locations of the centers to be on opposing vertices at $(0,0)$ and $(1,1)$. This choice allows us to exploit the symmetry of the exclusive-OR function to allow its solution with fewer "training" points than data points.

The total training data is the mapping depicted in Table 1. The calculations are performed using the pseudo-inverse technique with, and without, the use of an adjustable bias on the output unit.

5.1 The approximate exclusive-OR without an output bias

Following section 2, we use the following form of interpolating functions:

$$s(\underline{x}) = \sum_{j=1,3} \lambda_j \phi(||\underline{x} - \underline{y}_j||) \tag{5.1}$$

where $\underline{y}_1 = [0,0]^T$, $\underline{y}_3 = [1,1]^T$. The set $\{\lambda_j\}$ is given by

$$f_i = \sum_{j=1,3} \lambda_j \phi(||\underline{x}_i - \underline{y}_j||) \quad i = 1,2,3,4 \tag{5.2}$$

so that

$$\underline{\lambda} = \mathbf{A}^+ \underline{f} \tag{5.3}$$

where $\underline{f}$ is the same response vector as in the exact case, equation (4.1), and $\mathbf{A}^+$ is the pseudo-inverse of the (non-square) transformation matrix

$$\mathbf{A} = \begin{pmatrix} \phi_0 & \phi_2 \\ \phi_1 & \phi_1 \\ \phi_2 & \phi_0 \\ \phi_1 & \phi_1 \end{pmatrix} \tag{5.4}$$

From Appendix A, given the singular value decomposition of $\mathbf{A}$

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T$$

the pseudo-inverse is obtained as

$$\begin{aligned} \mathbf{A}^+ &= \mathbf{V}(\mathbf{S}^{-1})\mathbf{U}^T \\ &= \mathbf{V}(\mathbf{S}^{-1})^2\mathbf{V}^T\mathbf{A}^T \end{aligned} \tag{5.5}$$

The matrix $\mathbf{V}$ is composed of the normalized eigenvectors of the matrix

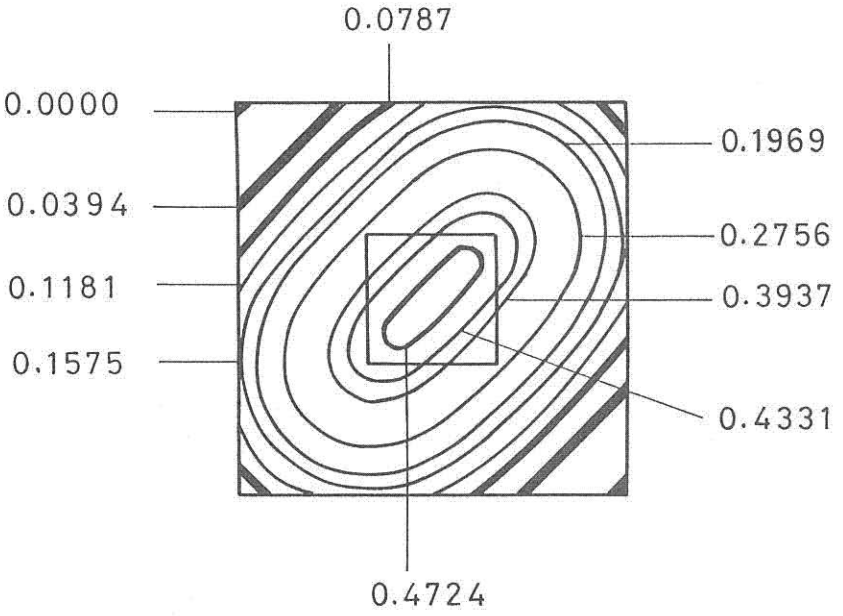

Figure 6: Contours of the approximate exclusive-OR solution, without an output bias.

$$\mathbf{A}^T\mathbf{A} = \begin{pmatrix} a & b \\ b & a \end{pmatrix} \tag{5.6}$$

where

$$\begin{aligned} a &= \phi_0^2 + 2\phi_1^2 + \phi_2^2 \\ b &= 2\phi_1^2 + 2\phi_0\phi_2 \end{aligned} \tag{5.7}$$

and the diagonal matrix $(\mathbf{S}^{-1})^2$ is made up of the reciprocal of the eigenvalues of the corresponding eigenvectors.

It is straightforward to verify that

$$V = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{5.8}$$

and

$$(\mathbf{S}^{-1})^2 = \begin{pmatrix} 1/[a+b] & 0 \\ 0 & 1/[a-b] \end{pmatrix} \tag{5.9}$$

Substituting these matrices into the pseudo-inverse expression, equation (5) and then into equation (5.3) gives,

$$\underline{\lambda} = \begin{pmatrix} \lambda_1 \\ \lambda_1 \end{pmatrix} \tag{5.10}$$

where

$$\begin{aligned} \lambda_1 &= 2\phi_1/[a+b] \\ &= 2\phi_1/(4\phi_1^2 + [\phi_0 + \phi_2]^2) \end{aligned} \tag{5.11}$$

This is the set of weights which minimizes the least mean squared error to all the training data. In fact the error, $\mathcal{E}$, may be analytically evaluated to give

$$\mathcal{E} = \frac{(\phi_0 + \phi_2)}{\sqrt{2\phi_1^2 + [\phi_0 + \phi_2]^2/2}} \tag{5.12}$$

An interesting point about this error, is that it will be zero if the radial basis functions are chosen to be such that $\phi_0 = -\phi_2$. This is precisely the condition mentioned at the end of the previous section in the discussion of how the exact exclusive-OR problem could be solved with only two radial basis function centers. In both instances the error is zero and the interpolating map manages to perfectly "learn" the training data. However, for general choices of radial basis function the solution as derived, does not reproduce the desired output of exclusive-OR satisfactorily. For instance, figure 6 depicts this solution using Gaussian radial basis functions and a Euclidean norm. The figure plots contours of equal "height" produced at the output of the radial basis function mapping. The vertices of the unit square in the figure represent the logical values 00, 01, 11, 10. As seen, although the output discriminates between even and odd parity inputs, the relative magnitudes have not been preserved (the output of 00 is greater than the output of 01 for instance).

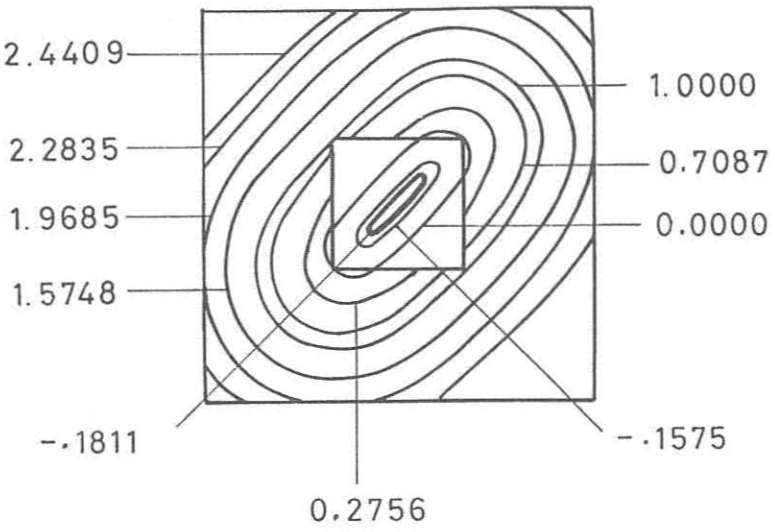

Figure 7: Contours of the approximate exclusive-OR solution, including an output bias.

5.2 The approximate exclusive-OR including an output bias

Consider the same situation as in the previous subsection, but where now a data independent variable is allowed, effectively a weighted bias at the output node through a connection to a node which gives a constant unit output. The significance of the output bias is that the desired output states of the exclusive-OR function have non-zero mean. The analysis without the inclusion of bias achieves a minimum error solution which matches the output *in the mean*. However, since positive weights are needed to achieve this, the resulting $s(\underline{x})$ naturally has maxima near to the centers $(0,0)$, $(1,1)$. Therefore, $s(\underline{x})$ does not reproduce the required qualitative details of the exclusive-OR function. In contrast, one might expect the inclusion of a bias to allow the whole surface $s(\underline{x})$ to be "floated" to the correct mean level while the remaining parameters adjust its qualitative form. Following this, the interpolating function is taken to have the form

$$s(\underline{x}) = \lambda_0 + \sum_{j=1,3} \lambda_j \phi(||\underline{x} - \underline{y}_j||) \tag{5.13}$$

where $\{\lambda_{1,3}\}$ are as previously assumed.

The matrix of distances is now

$$\mathbf{A} = \begin{pmatrix} 1 & \phi_0 & \phi_2 \\ 1 & \phi_1 & \phi_1 \\ 1 & \phi_2 & \phi_0 \\ 1 & \phi_1 & \phi_1 \end{pmatrix} \tag{5.14}$$

Consequently, the singular value decomposition is determined by the eigenvectors and eigenvalues of the matrix

$$\mathbf{A}^T\mathbf{A} = \begin{pmatrix} 4 & c & c \\ c & a & b \\ c & b & a \end{pmatrix} \tag{5.15}$$

where

$$\begin{aligned} a &= \phi_0^2 + 2\phi_1^2 + \phi_2^2 \\ b &= 2\phi_1^2 + 2\phi_1\phi_2 \\ c &= \phi_0 + 2\phi_1 + \phi_2 \end{aligned} \tag{5.16}$$

Note that c here has the interpretation of being proportional to the *mean* value of radial basis functions evaluated at any training point.

Consider the eigenvalue equation

$$\mathbf{A}^T\mathbf{A}\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix} = \mu \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{pmatrix}$$

From the characteristic equation one finds that the eigenvalue problem factorizes, giving

$$\begin{aligned} \mu_0 &= a - b \\ \mu_\pm &= \frac{(a+b+4) \pm \sqrt{(a+b-4)^2 + 8c^2}}{2} \end{aligned} \tag{5.17}$$

The normalized eigenvector corresponding to $\mu = a - b$ is then

$$a_0 = \frac{1}{\sqrt{2}}\begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix} \tag{5.18}$$

For the case $\mu = \mu_\pm$, we have $\alpha_2 = \alpha_3$, since the resulting eigenvectors are orthogonal to a_0. Setting $\alpha_2 = 1$ without loss of generality implies that the (unnormalized) component α_1 is

$$\alpha_1^\pm = \frac{2c}{\mu_\pm - 4}$$

Thus the resulting orthonormal matrices $\mathbf{V}$ and $(\mathbf{S}^{-1})^2$ take the form

$$\mathbf{V} = \begin{pmatrix} 0 & \alpha_1^+/\Delta_+ & \alpha_1^-/\Delta_- \\ 1/\sqrt{2} & 1/\Delta_+ & 1/\Delta_- \\ -1/\sqrt{2} & 1/\Delta_+ & 1/\Delta_- \end{pmatrix} \tag{5.19}$$

$$(\mathbf{S}^{-1})^2 = \begin{pmatrix} \mu_0 & 0 & 0 \\ 0 & \mu_+ & 0 \\ 0 & 0 & \mu_- \end{pmatrix} \tag{5.20}$$

where the normalization factors $\Delta_\pm$ are given explicitly by

$$(\Delta_\pm)^2 = 2 + (\alpha_1^\pm)^2 \tag{5.21}$$

Using these matrices to construct the pseudo-inverse of $\mathbf{A}$ results finally in the set of weights,

$$\underline{\lambda} = \begin{pmatrix} \lambda_0 \\ \lambda_1 \\ \lambda_1 \end{pmatrix} \tag{5.22}$$

where

$$\begin{aligned} \lambda_0 &= \tfrac{2\alpha_1^+}{\mu_+\Delta_+^2}(\alpha_1^+ + 2\phi_1) + \tfrac{2\alpha_1^-}{\mu_-\Delta_-^2}(\alpha_1^- + 2\phi_1) \\ \lambda_1 &= \tfrac{2}{\mu_+\Delta_+^2}(\alpha_1^+ + 2\phi_1) + \tfrac{2}{\mu_-\Delta_-^2}(\alpha_1^- + 2\phi_1) \end{aligned} \tag{5.23}$$

This result is shown in figure 7 and may be compared directly with that of the previous subsection shown in figure 6. Note that the network now successfully discriminates states of opposite parity and moreover returns precisely the correct magnitude for each corner of the unit square. However, the symmetry of placing the centers on the diagonal of the unit square, means that the solution obtained in this case is *exact*. There are only three independent equations we need to solve, and three adjustable parameters at our disposal. In this sense the XOR problem is rather too simple.

It is interesting to repeat the calculation of section 5.1 using a response vector $\tilde{f} = (-\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, \frac{1}{2})^T$. This allows one to study an equivalent problem with the mean artificially removed without the introduction of an additional parameter. This produces, again, two equal weight values (compare with (11))

$$\lambda_1 = \frac{\phi_1 - [\phi_0 + \phi_2]/2}{8\phi_1^2 + 2[\phi_0 + \phi_2]^2}$$

The resultant fit clearly does not reproduce the training values as well as the three parameter model given above. It does however have the correct qualitative form.

In general, a bias parameter may be expected to provide a compensation for a global shift which is hard to achieve through weighting the individual basis functions. The consequences of this observation may also be noted in conventional multilayer perceptron studies where the performance of the network is enhanced if the input data is previously scaled to have zero mean.

6. Numerical examples: the prediction of chaotic time series

Lapedes and Farber [15] have recently used multilayer perceptrons for the prediction of time series data generated by nonlinear dynamical systems. Extensive comparisons with other prediction techniques showed that multilayer perceptrons were more accurate than the classical (linear) methods and were comparable with the locally linear approach of Farmer and Sidorowich [19].

In this section, we shall, following Lapedes and Farber, use nonlinear prediction as a non-trivial example of the application of adaptive networks. We note that in this application our approach is very close to that of Casdagli [12] who has applied radial basis functions to the construction of nonlinear maps from time series data. Unlike Casdagli who used strict interpolation, we shall employ the least squares generalization given in section 2.

Specifically, consider T_d, an ordered sequence of iterates of the doubling map:

$$x_{n+1} = 2x_n \ \text{(modulo 1)} \tag{6.1}$$

and T_q, a sequence of iterates of the quadratic map

$$x_{n+1} = 4x_n(1 - x_n) \tag{6.2}$$

These maps are known to be chaotic on the interval $[0, 1]$: in both cases the iterates of generic initial conditions are distributed according to continuous invariant measures. For T_d the autocorrelation $\langle x_0 x_n \rangle$ decays as 2^{-n} while for T_q, $\langle x_0 x_n \rangle \approx \delta_{0,n}$ where $\delta_{i,j}$ is the Kroenecker delta. Therefore, given only the data T_q, second-order statistics would convey the impression that the sequence is random broadband noise (see Appendix C for further details). Naïvely (and in fact, erroneously) one might expect from this that the prediction of T_q is harder than the prediction of T_d.

A radial basis function network for predicting one time step into the future was constructed as follows. The basis function centers $\{y_j\}$ were chosen to be uniformly spaced on $(0, 1)$; the number of centers was an adjustable parameter. A set of input values $\{x_i \in [0, 1] | i = 1, 2, \ldots, 250\}$ was used to calculate the matrix $\mathbf{A}$ using equation (2.4). The singular value decomposition of $\mathbf{A}$, calculated numerically by a Golub-Reinsch algorithm, was used to form the pseudo inverse $\mathbf{A}^+$ using equation (A.2). This was then applied to the vector of outputs:

$$\underline{f} = (f(x_1), \ldots, f(x_i), \ldots, f(x_{250}))^T$$

(where $f(x)$ is the map given by either equation (6.1), or equation (6.2)) to obtain the weights $\{\lambda_j\}$ according to $\underline{\lambda} = \mathbf{A}^+ \underline{f}$. The accuracy of this mapping were then analyzed for an extra 250 different "test" points.

Figures 8 and 9, which show the output of the networks as a function of inputs, illustrate the relationship with curve fitting for these simple one-dimensional problems. It is clear that the basis of the difference between predicting T_d and predicting T_q is that the doubling map is discontinuous and

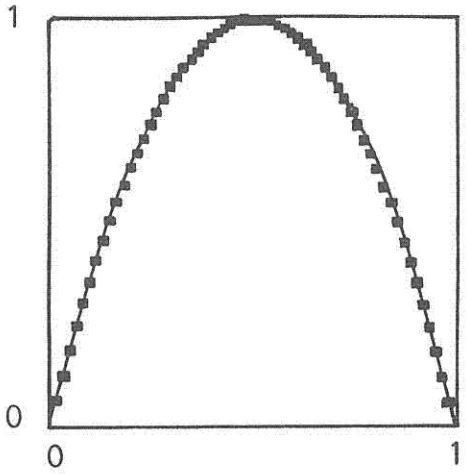

Figure 8: The Quadratic Map: A figure showing the actual (solid line), and predicted (filled squares) outputs of the network over the interval $[0, 1]$ for one iterate.

therefore hard to fit. Multilayer perceptrons also have difficulty with trying to find an appropriate set of weight values which allows a good fit to T_d (in fact the overwhelming tendency is for the multilayer perceptron to get stuck in an unsuitable local minimum, M.D. Bedworth, private communication).

For prediction further into the future, the situation is further exacerbated and rapidly becomes hard even in the case of T_q. The problem is now one of fitting a graph of the n^{th} order iterate of equation (6.1) or (6.2). In either case the graph has 2^{n-1} oscillations of unit amplitude. In terms of the radial basis function network, this would require at least 2^n hidden units with centers appropriately positioned. An alternative to this strategy is to iterate the one-step network. This however, is inaccurate since errors in chaotic systems grow exponentially because of the local instability of the evolution.

The accuracy of the network can be quantified by the following index, $\mathcal{I}$:

$$\mathcal{I} = \sqrt{\frac{\langle [x_{predicted}(t) - x_{expected}(t)]^2 \rangle}{\langle (x - \langle x \rangle)^2 \rangle}} \tag{6.3}$$

This quantity is a measure of how well the network generalizes beyond the training data. The error expression given in section 2 has the same form, but since it is based on the training data, shows how well the network reproduces the training data. It is of interest to compare the two since the difference quantifies the degradation of the predictive power of the network when it is required to generalize. The graphs shown in figures 10 and 11 summarize both kinds of error analysis for networks trained on T_d, and T_q.

The most obvious difference between these figures is the scale. It is clear that prediction of T_q, whichever error criterion is used, is much easier than

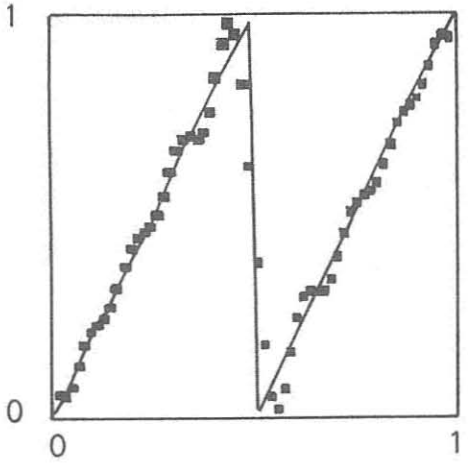

Figure 9: The Doubling Map: A figure showing the actual (solid line), and predicted (filled squares) outputs of the network over the interval $[0, 1]$ for one time step into the future for the doubling map.

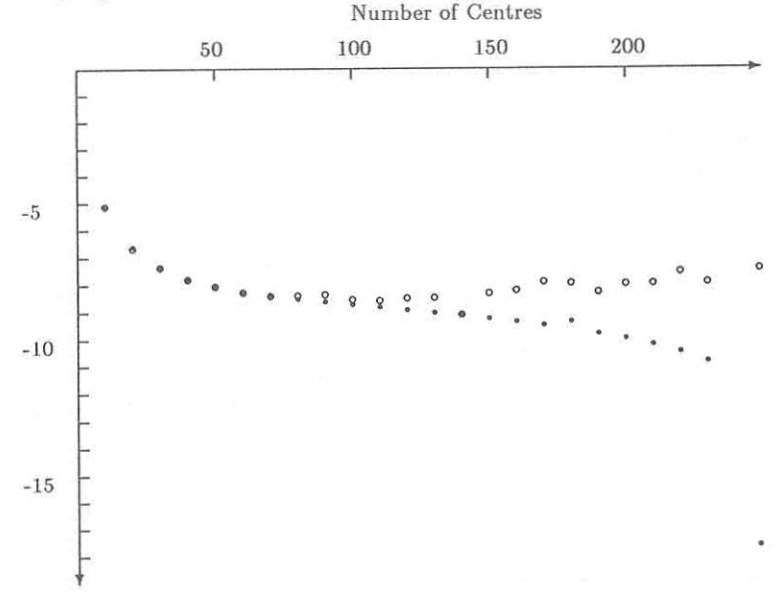

Figure 10: The Quadratic Map: The log normalized error showing the training (solid circles) and recognition data (open circles) as a function of the number of radial basis function centers. Euclidean norm and a Gaussian radial basis function ($\phi = \exp[-z^2 n_0^2/16]$) were used.

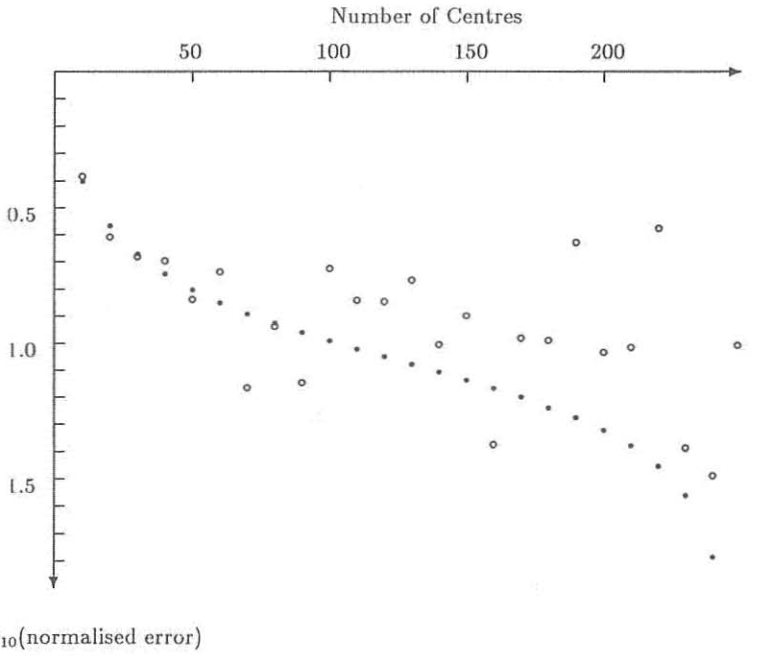

Figure 11: The Doubling Map: The log normalized error showing the training (solid circles) and recognition data (open circles) as a function of the number of radial basis function centers. Euclidean norm and a Gaussian radial basis function ($\phi = \exp[-z^2 n_0^2/16]$) were used.

the prediction of T_d by several orders of magnitude. Beyond this, we see in both cases that the training error of section 2 has the same basic dependence on the number of hidden units; that is, a fast improvement as n_0 increases to about 30 followed by a plateau region where the relative improvement is small. As n_0 approaches the number of data points used in the training (250 in this example), the training error again drops rapidly as the problem approaches the strict interpolation limit. This drop is not, however, mirrored by a drop in the recognition error. Although initially, the recognition error follows the training error very closely, a saturation plateau is reached and approximately maintained irrespective of how many hidden units are employed. This can be understood since the capability of the model to generalize, is connected with the underlying "smoothness" of the true map and the level of "smoothness" built into the model through the choice of metric and radial basis function (and indeed the assumption that an arbitrary function may be approximately represented by the radial basis function expansion). Therefore one can surmise that in most instances, there will be a limiting accuracy to which it is possible to model unseen data generated by a mapping. This is not true for the training points themselves, since it is possible by strict interpolation to produce a mapping surface which exactly passes through all the points. However, all that this accomplishes is a fit to the noise on the training points which may oscillate wildly between the constraining "knots." It was for this very reason that we introduced the least squares solution of the radial basis function construction in section 2.

7. Conclusion

The object of this paper has been to introduce a simple view of network models as devices for the interpolation of data in multidimensional spaces. The purpose of this is to allow the application of a large body of intuition and knowledge from the theory of fitting and interpolation to the understanding of the properties of nonlinear networks. Thus we associate the concept of generalization in networks with the simple idea of interpolation (extrapolation) between known data points. The details of the generalization are then dependent upon the implicit interpolation scheme employed by a given network. Generalization is hard where the relationship has strong oscillations or discontinuities. This suggests that, particularly in the case of abstract problems for which the topology of the input and output spaces may not be clear *a priori*, it may be advantageous to attempt to code the data so as to produce a relationship which is as smooth as possible. Further we expect the training data, where possible, would best be distributed to give information about all the turning points of the graph and need not be tightly clustered where, for example, the relationship is smooth or monotone.

Motivated by this philosophy, we introduce a network model based on the radial basis function approach to curve fitting. This model has two main advantages. First, it is firmly attached to a well established technique for fitting, but, since it is contained within a general class of nonlinear networks, it

may be used as a source of "existence proofs" for such networks. For instance, we know that networks of this form can be used to model relationships which lie in the function space spanned by the chosen set of radial basis functions. The characterization of this space and quantification of such things as the rate of convergence of radial basis function expansions is currently receiving much attention and is seen to be of direct relevance to the theory of networks.

The second advantage of this network is in practical application. The basis of its simplicity is that it combines a linear dependence on the variable weights with an ability to model explicitly nonlinear relationships such as for example, the exclusive-OR function. Thus, in the least squares context, training the network is equivalent to solving a linear matrix equation. If we specialize to a minimum norm solution, the solution is unique and in this sense the network may be said to have a guaranteed learning algorithm.

This general approach, whereby optimization is carried out on the subset of the weights for which the problem is linear, may be taken with other network models. It would be interesting to study how much this restricts their generality. Work along these lines is currently in progress. In the present case, on the other hand, the inclusion of the basis function centers into the optimization calculation may be carried out using a nonlinear optimization technique in conjunction with the linear analysis described above. By analogy with spline fitting of curves, this may produce some advantage, perhaps in the form of needing fewer hidden units, but it is questionable whether this would compensate for the added complexity of performing the nonlinear optimization. We have not approached here the general question of what form of ϕ is best, or where and how many centers should be used in the expansion. Work is currently in progress to assess the sensitivity of convergence of these factors and the use of the error function given in equation (7) as a cost function for nonlinear optimization using the basis function centers.

Acknowledgements

We would like to thank Professor M.J.D. Powell at the Department of Applied Mathematics and Theoretical Physics at Cambridge University for providing the initial stimulus for this work. Also, we appreciate the help given to us by Mark Bedworth, John Bridle and Andrew Webb of the Speech Research Unit at RSRE for sharing their insights into the workings of layered network models in general, and the difficulties associated with the practical application of the logistic, semi-linear multilayer perceptron in particular.

Appendix A. Solving linear inverse problems

The appendix looks at linear inverse problems as they arise in the radial basis function approach to nonlinear networks.

In applying the radial basis function method, we need to invert an $m \times n$ $(m \geq n)$ matrix with elements $A_{ij} = \phi(||\underline{x}_i - \underline{y}_j||)$. Since A may be rank

$\mathbf{C}_4$	E	C_4	C_2	C_4^3
A	1	1	1	1
B	1	-1	1	-1
E	$\begin{Bmatrix}1\\1\end{Bmatrix}$	$\begin{matrix}i\\-i\end{matrix}$	$\begin{matrix}-1\\-1\end{matrix}$	$\begin{Bmatrix}-i\\i\end{Bmatrix}$

Table 2: The character table for $\mathbf{C}_4$

deficient, it is necessary to account for the possibility of ill-conditioning. Consider the singular value decomposition of $\mathbf{A}$

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T \tag{A.1}$$

where $\mathbf{V}$ is an $n \times n$ orthogonal matrix, $\mathbf{S}$ is an $n \times n$ diagonal matrix of singular values and $\mathbf{U}$ is an $m \times n$ matrix with orthonormal columns. The pseudo inverse of $\mathbf{A}$, $\mathbf{A}^+$, may be constructed as

$$\mathbf{A}^+ = \mathbf{V}\mathbf{S}^+\mathbf{U}^T \tag{A.2}$$

where $\mathbf{S}^+$ is obtained from $\mathbf{S}$ by reciprocating the non-zero diagonal elements. Clearly if $rank\mathbf{A} = m$ then $\mathbf{A}^+\mathbf{A} = \mathbf{1}$ where $\mathbf{1}$ is the $m \times m$ unit matrix. On the other hand, $\mathbf{A}\mathbf{A}^+ = \mathbf{U}\mathbf{U}^T$ a superposition of projections onto a subspace of $\mathbb{R}^n$ spanned by the columns of $\mathbf{U}$. If $rank\mathbf{A} < m$ then $\mathbf{A}^+\mathbf{A}$ and $\mathbf{A}\mathbf{A}^+$ give projections onto subspaces of $\mathbb{R}^m$ and $\mathbb{R}^n$ respectively.

In the case of the exact exclusive-OR function the question of ill-conditioning does not arise. In this case it is convenient to calculate the inverse of $\mathbf{A}$ through its eigenvalue decomposition since the symmetry of the problem may be exploited to obtain the solution. Here

$$\mathbf{A} = \mathbf{V}\underline{\underline{\mu}}\mathbf{V}^T \tag{A.3}$$

where, since in this case $\mathbf{A}$ is a real symmetric matrix, the matrix of eigenvectors $\mathbf{V}$ is orthogonal. It follows that

$$\mathbf{A}^{-1} = \mathbf{V}\underline{\underline{\mu}}^{-1}\mathbf{V}^T \tag{A.4}$$

assuming that $\mathbf{A}$ is full rank. The rest of the appendix deals with the calculation of the eigenvectors and eigenvalues of $\mathbf{A}$ using the symmetry of the exclusive-OR function.

Our choice of ordering of the input points in section 4 is somewhat arbitrary. It should be clear that we can perform a sequence of rotations on the original orderings while retaining the same matrix $\mathbf{A}$. In other words, $\mathbf{A}$ is invariant to a certain class of transformations, in particular, it is invariant to operations in the group $\mathbf{C}_4 = \{E, C_4, C_2, C_4^3\}$, where E is the identity transformation, C_4 denotes rotations by $\pi/2$, C_2 by π and C_4^3 rotations by $3\pi/2$. The character table for the group $\mathbf{C}_4$ is shown in table 2 (for an introduction to the theory and application of groups see [20]).

The character table may be exploited to solve the eigenvalue problem for $\mathbf{A}$ by obtaining a symmetry adapted basis. We can see this as follows. The representation of the group operations using the standard basis,

$$\underline{e}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \underline{e}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad \underline{e}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \quad \underline{e}_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

is not irreducible since its dimension is four and the maximum dimension of an irreducible representation of $\mathbf{C}_4$ is two. In fact this representation of the group has the form $\Gamma = A + B + E$ and so the basis has irreducible components A, B, E. From the character table , one can ascertain that the appropriate symmetry adapted basis is just

$$\underline{v}_A = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \underline{v}_B = \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix} \quad \underline{v}_E = \begin{pmatrix} 1 \\ i \\ -1 \\ -i \end{pmatrix} \quad \underline{v}_E^* = \begin{pmatrix} 1 \\ -i \\ -1 \\ i \end{pmatrix}$$

or, by normalizing and replacing the degenerate vectors $\underline{v}_E$ and $\underline{v}_E^*$ by simple linear combinations, we arrive at a symmetry adapted set of basis vectors,

$$\underline{v}_A = \frac{1}{2}\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \quad \underline{v}_B = \frac{1}{2}\begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix} \quad \underline{v}_E^1 = \frac{1}{2}\begin{pmatrix} \sqrt{2} \\ 0 \\ -\sqrt{2} \\ 0 \end{pmatrix}$$

$$\underline{v}_E^2 = \frac{1}{2}\begin{pmatrix} 0 \\ \sqrt{2} \\ 0 \\ -\sqrt{2} \end{pmatrix} \tag{A.5}$$

It is clear that these basis vectors are orthogonal and they are eigenvectors of the matrix $\mathbf{A}$ since, explicitly,

$$\begin{array}{lcll} \mathbf{A}\underline{v}_A & = & (\phi_0 + 2\phi_1 + \phi_{\sqrt{2}})\underline{v}_A & \Rightarrow \quad \mu_A = (\phi_0 + 2\phi_1 + \phi_{\sqrt{2}}) \\ \mathbf{A}\underline{v}_B & = & (\phi_0 - 2\phi_1 + \phi_{\sqrt{2}})\underline{v}_B & \Rightarrow \quad \mu_B = (\phi_0 - 2\phi_1 + \phi_{\sqrt{2}}) \\ \mathbf{A}\underline{v}_E^1 & = & (\phi_0 - \phi_{\sqrt{2}})\underline{v}_E^1 & \Rightarrow \quad \mu_E = (\phi_0 - \phi_{\sqrt{2}}) \end{array} \tag{A.6}$$

These basis vectors and eigenvalues are employed in section 4 to obtain a set of weights analytically, which exactly solves the exclusive-OR function.

Appendix B. An analytic solution of the exact n-bit parity problem

The n-bit parity problem is a generalization of the exclusive-OR problem discussed in section 4. It may be defined by the mapping depicted in table 3, that is, the output is unity if the total number of input bits having value one is odd, otherwise the output is zero. Thus changing one bit in the input pattern produces a maximal change in the output.

With the benefit of insight developed from the exclusive-OR problem, this section obtains an exact representation of the n-bit parity problem as a

Input Pattern		Number of "ON" bits in input		Output Pattern
000	→	0	→	0
001	→	1	→	1
010	→	1	→	1
100	→	1	→	1
011	→	2	→	0
101	→	2	→	0
110	→	2	→	0
111	→	3	→	1

Table 3: Symbolic mapping of the n-bit parity problem for the case $n = 3$.

network based on radial basis functions. The network will have the general form of n-inputs for an n-bit word, and 2^n hidden units all connected to one output. The centers of the 2^n hidden units correspond to the possible input patterns. Thus, an exact solution may be obtained once the 2^n-dimensional vector $\underline{\lambda}$ of weights has been determined. All possible input states may be put in a $1:1$ correspondence with the vertices of a unit n-dimensional hypercube. This is conveniently achieved by aligning the hypercube with the Cartesian basis so that one vertex resides at the origin. The Cartesian co-ordinates of each vertex then directly maps to a unique binary sequence, for instance $(0,0,0) \rightarrow 000$, $(0,1,0) \rightarrow 010$, $(1,1,0) \rightarrow 110$, etc. The vertices may be ordered by treating the set of sequences as a cyclic Grey code of the first 2^n integers. Thus all nearest neighbors in this scheme correspond to points of opposite parity and the use of the cyclic Grey code ensures that entries across the rows of $\mathbf{A}$ represent points of alternating parity.

The rows of $\mathbf{A}$ are permutations of each other because of the symmetry of the hypercube. It follows that there is a totally symmetric eigenvector, $v_+ = 2^{-n/2}[1,1,\ldots]^T$ for which the corresponding eigenvalue is the sum of the row elements of $\mathbf{A}$

$$\mu_+ = \sum_{j=0}^{n} p_j^n \phi_j \tag{B.1}$$

where p_j^n is the number of j^{th} nearest neighbors to an arbitrary vertex.

A second eigenvector may be found using the division of the vertices into two groups, differentiated by their parity. The form of this eigenvector follows from the use of the cyclic Grey code in ordering the vertices: $v_- = 2^{-n/2}[1,-1,1,\ldots]^T$. This antisymmetric form distinguishes sites of opposite parity and thus has a corresponding eigenvalue

$$\mu_- = \sum_{j\ even} p_j^n \phi_j - \sum_{j\ odd} p_j^n \phi_j \tag{B.2}$$

since there are $\sum_{j\ even} p_j^n$ sites of the same parity, and $\sum_{j\ odd} p_j^n$ sites with opposite parity.

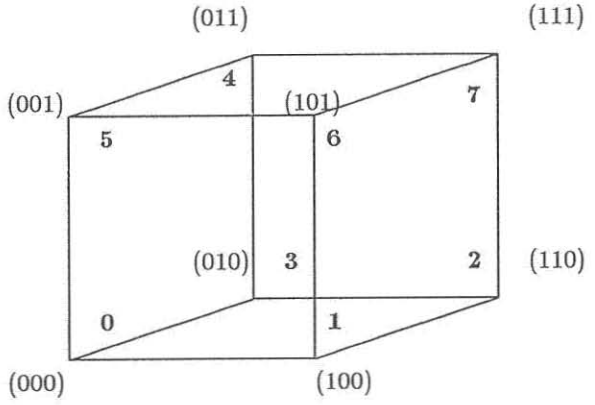

Figure 12: The hypercube of the three-bit parity problem.

The coefficients p_j^n may be obtained by comparing two arbitrary n-bit sequences which differ in j-locations. The number of ways of permuting j-locations within n-bits is just $p_j^n = \binom{n}{j}$. Note that $\sum_0^n p_j^n = 2^n$, the total number of vertices of the hypercube.

The response vector, $\underline{s} = [0, 1, 0, 1, 0, \ldots]$ may be decomposed as the difference of the symmetric and antisymmetric eigenvectors. Thus, v_+, v_- are the only eigenvectors which are relevant in evaluating the weights. Consequently, as in the exclusive-OR example, the vector of weights of the n-bit parity problem may be evaluated as

$$\underline{\lambda} = \frac{1}{2} \begin{pmatrix} \mu_+^{-1} - \mu_-^{-1} \\ \mu_+^{-1} + \mu_-^{-1} \\ \mu_+^{-1} - \mu_-^{-1} \\ \mu_+^{-1} + \mu_-^{-1} \\ \vdots \end{pmatrix} = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_1 \\ \lambda_2 \\ \vdots \end{pmatrix} \tag{B.3}$$

where,

$$\lambda_1 = \frac{-\sum_{j\ odd}\binom{n}{j}\phi_j}{\left[\sum_{j\ even}\binom{n}{j}\phi_j\right]^2 - \left[\sum_{j\ odd}\binom{n}{j}\phi_j\right]^2} \tag{B.4}$$

$$\lambda_2 = \frac{\sum_{j\ even}\binom{n}{j}\phi_j}{\left[\sum_{j\ even}\binom{n}{j}\phi_j\right]^2 - \left[\sum_{j\ odd}\binom{n}{j}\phi_j\right]^2}$$

Equation (B.4) accomplishes what the section set out to obtain: an *exact* solution to the n-bit parity problem in the sense that there is a guaranteed set of values for the weights of the matrix which ensures that the result of the network is to reproduce, at least, the values exhibited in table 3. It should be noted that although this task has been achieved with a network involving 2^n hidden units, it is still possible to solve the problem exactly with fewer than this number of hidden units. For instance, and by analogy with the exclusive-OR problem, if the radial basis function and metric were chosen in such a way that either of

$$\sum_{j\ even}\binom{n}{j}\phi_j = 0 \tag{B.5}$$

$$\sum_{j\ odd}\binom{n}{j}\phi_j = 0$$

are satisfied, then an exact solution may be obtained with only 2^{n-1} hidden units.

Appendix C. The quadratic and doubling maps

This appendix lists a few details relevant to the chaotic maps discussed in section 6 for time series prediction. The quadratic map, as discussed, determines the signal at time $t+1$ in terms of the signal at time t by the explicit iteration scheme

$$x_{t+1} = 4x_t(1 - x_t) \tag{C.1}$$

By a simple linear transformation, $x \rightarrow y$, this map is transformed into the form

$$y_{n+1} = 2y_n^2 - 1 \tag{C.2}$$

Since these two mappings are related by a linear transformation ($x = 0.5[1 - y]$), the behavior of equation (C.2) determines the behavior of the quadratic map. Equation (C.2) is interesting because it illustrates that the mapping is explicitly given in terms of Chebyshev polynomials [21], specifically

$$y_{n+1} = T_2(y_n) \tag{C.3}$$

From this relationship and exploiting the property of Chebyshev polynomials that

$$2T_m T_n = T_{n+m} + T_{n-m} \tag{C.4}$$

or, specifically

$$2T_n^2 = T_{2n} + T_0 \tag{C.5}$$

one finds that the n^{th} iterate, y_n, is connected with the starting value y_0 through the 2^n Chebyshev polynomial,

$$y_n = T_{2^n}(y_0) \tag{C.6}$$

This just makes explicit the fact that the value of the map at any time in the future is uniquely determined by the starting value. However, the map is known to be ergodic, and thus time averages are equivalent to phase averages. To obtain the phase average, one needs the fact that the invariant measure of the map, equation (C.2), is

$$m(y) = \frac{1}{\pi\sqrt{1-y^2}} \tag{C.7}$$

Therefore, the correlation function $\langle y_k y_{k+j}\rangle$ may be determined by the expectation value

$$\langle y_k y_{k+j}\rangle = \int_{-1}^{1} \frac{T_{2^k}(y_0)T_{2^{k+j}}(y_0)}{\pi\sqrt{1-y_0^2}}\, dy_0 \tag{C.8}$$

However, this is precisely the orthogonality relationship between Chebyshev polynomials of the first kind, and hence the integral yields a Kroeneker delta,

$$\langle y_k y_{k+j}\rangle = \frac{1}{2}\delta_{j,0} \tag{C.9}$$

Consequently, as far as second-order statistics are concerned, the time series generated by the quadratic map totally loses its memory between each time step, and hence would appear to be an infinite bandwidth, noise signal (this is of course not the case when higher-order correlations are taken into account).

In the case of the doubling map, the value of the time series at time $t+1$ is determined by the time series at t via

$$x_{t+1} = 2x_t \mod 1 \tag{C.10}$$

The correlation function may be expressed as

$$\begin{aligned}\langle x_0 x_t\rangle &= \int_0^1 x_0[2^t x_0]\, dx \\ &= \sum_{j=0}^{2^t-1} \int_{j/2t}^{(j+1)/2t} x[2^t x]\, dx\end{aligned} \tag{C.11}$$

where $[x]$ denotes the fractional part of x (note that the invariant measure is uniform in this case and the map is known to be chaotic so that time averages and ensemble averages are equivalent). By a change of variables, $y = 2^t x - j$ the above integral is readily performed to give:

$$\begin{aligned} \langle x_0 x_t \rangle &= 2^{-2t} \sum_{j=0}^{2^t-1} \left\{ \frac{1}{3} + \frac{j}{2} \right\} \\ &= \frac{1}{4}\left(1 + \frac{2^{-t}}{3}\right) \end{aligned} \tag{C.12}$$

Thus, in contrast to the quadratic map, the correlation function for the doubling map decays exponentially in time.

References

[1] F. Jelinek, R. L. Mercer and L. R. Bahl, "Continuous Speech Recognition: Statistical Methods" in *"Handbook of Statistics"*, **2**, edited by P. R. Krishnaiah and L. N. Kanal, (North-Holland Publishing Company, 1982) 549–573.

[2] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, "An Introduction to the Application of the Theory of Probabilistic Functions of a Markov process to Automatic Speech Recognition", *The Bell System Technical Journal*, **62(4)** (1983) 1036–1073.

[3] L. R. Rabiner, J. G. Wilson, and B. H. Juang, "A model-based connected-digit recognition system using either hidden Markov models or templates", *Computer Speech & Language*, **1(2)** (1986) 167–197.

[4] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation", ICS Report 8506 (Institute for Cognitive Science, University of California, San Diego, 1985).

[5] Mark D. Bedworth and John S. Bridle, "Experiments with the Back-Propagation Algorithm: A Systematic Look at a Small Problem", RSRE Memorandum 4049 (1987) available from R.S.R.E., St. Andrews Rd., Great Malvern, Worcs. WR14 3PS, England.

[6] D. C. Plaut, S. J. Nolan, and G. E. Hinton, "Experiments on Learning by Back Propagation", CMU-CS-86-126 (Carnegie-Mellon University, 1986).

[7] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review", *IMA conference on "Algorithms for the Approximation of Functions and Data", RMCS Shrivenham* (1985).

[8] Charles A. Micchelli, "Interpolation of Scattered Data: Distance Matrices and Conditionally Positive Definite Functions", *Constructive Approximation*, **2** (1986) 11–22.

[9] R. Franke, "Scattered Data Interpolation: Tests of some methods", *Mathematics and Computing*, **38** (1982) 181–200.

[10] G. Golub and W. Kahan, "Calculating the Singular Values and Pseudo-Inverse of a Matrix", *Journal SIAM Numerical Analysis, Series B*, **2(2)** (1965) 205–224.

[11] M. J. D. Powell, "Radial basis function approximations to polynomials", DAMPT preprint 1987 NA/6 (presented at the 1987 Dundee biennial Numerical Analysis Conference).

[12] Martin Casdagli, "Nonlinear prediction of chaotic time series", submitted to *Physica D* 1988.

[13] J. C. Nash, "Compact Numerical Methods for Computers: linear algebra and function minimisation", (Adam-Hilger Ltd, Bristol, 1980).

[14] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, "Numerical Recipes: The Art of Scientific Computing", (Cambridge University Press, 1986).

[15] Alan Lapedes and Robert Farber, "Nonlinear Signal Processing using Neural Networks: Prediction and System Modelling", (submitted to Proceedings of IEEE, July 1987).

[16] Raymond L. Watrous, "Learning Algorithms for Connectionist Networks:Applied Gradient Methods of nonlinear Optimisation" (1987).

[17] Ian D. Longstaff and John F. Cross, "A pattern recognition approach to understanding the multilayer perceptron", *Pattern Recognition Letters*, **5** (1987) 315–319

[18] Marvin L. Minsky and Seymour A. Papert, *Perceptrons: An Introduction to Computational Geometry* (MIT Press, expanded edition, 1988).

[19] J. Doyne Farmer and John J. Sidorowich, "Predicting Chaotic Time Series", *Physical Review Letters*, **59(8)** (1987) 845–848.

[20] L. D. Landau and E. M. Lifshitz, "Quantum Mechanics: Non-Relativistic Theory", (Pergammon Press, 2nd edition 1965, Chapter XII).

[21] Milton Abramowitz and Irene A. Stegun, "Handbook of Mathematical Functions", (Dover Publications, Ninth Edition, 1970).