MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY
and
CENTER FOR BIOLOGICAL INFORMATION PROCESSING
WHITAKER COLLEGE

A.I. Memo No.1140 July 1989
C.B.I.P. Paper No. 31

A Theory of Networks for Approximation and Learning

Tomaso Poggio and Federico Girosi

Abstract

Learning an input-output mapping from a set of examples, of the type that many neural networks have been constructed to perform, can be regarded as synthesizing an approximation of a multi-dimensional function, that is solving the problem of hypersurface reconstruction. From this point of view, this form of learning is closely related to classical approximation techniques, such as generalized splines and regularization theory. This paper considers the problems of an exact representation and, in more detail, of the approximation of linear and nonlinear mappings in terms of simpler functions of fewer variables. Kolmogorov's theorem concerning the representation of functions of several variables in terms of functions of one variable turns out to be almost irrelevant in the context of networks for learning. We develop a theoretical framework for approximation based on regularization techniques that leads to a class of three-layer networks that we call Generalized Radial Basis Functions (GRBF), since they are mathematically related to the well-known Radial Basis Functions, mainly used for strict interpolation tasks. GRBF networks are not only equivalent to generalized splines, but are also closely related to pattern recognition methods such as Parzen windows and potential functions and to several neural network algorithms, such as Kanerva's associative memory, backpropagation and Kohonen's topology preserving map. They also have an interesting interpretation in terms of prototypes that are synthesized and optimally combined during the learning stage. The paper introduces several extensions and applications of the technique and discusses intriguing analogies with neurobiological data.

This paper describes research done within the Center for Biological Information Processing, in the Department of Brain and Cognitive Sciences, and at the Artificial Intelligence Laboratory. This research is sponsored by a grant from the Office of Naval Research (ONR), Cognitive and Neural Sciences Division; by the Artificial Intelligence Center of Hughes Aircraft Corporation; by the Alfred P. Sloan Foundation; by the National Science Foundation. Support for the A. I. Laboratory's artificial intelligence research is provided by the Advanced Research Projects Agency of the Department of Defense under Army contract DACA76–85–C–0010, and in part by ONR contract N00014–85–K–0124.

Contents

1 Learning as Approximation

The problem of learning a mapping between an input and an output space is essentially equivalent to the problem of synthesizing an associative memory that retrieves the appropriate output when presented with the input and *generalizes* when presented with new inputs. It is also equivalent to the problem of estimating the system that transforms inputs into outputs given a set of examples of input-output pairs. A classical framework for this problem is *approximation theory*. Related fields are *system identification techniques* when it is possible to choose the input set and *system estimation techniques* when the input-output pairs are given. A suggestive point of view on networks and classical representation and approximation methods is provided by Omohundro (1987) and an interesting review of networks, statistical inference, and estimation techniques can be found in Barron and Barron, 1988. Learning from the point of view of approximation has been also considered among others by J. Schwartz (1988), Poggio et al. (1988, 1989), Aloimonos (1989), Hurlbert and Poggio (1988) and Poggio (1975).

Approximation theory deals with the problem of *approximating* or *interpolating* a continuous, multivariate function $f(X)$ by an approximating function $F(W, X)$ having a fixed number of parameters W (X and W are real vectors $X = x_1, x_2, ..., x_n$ and $W = w_1, w_2, ..., w_m$). For a choice of a specific F, the problem is then to find the set of parameters W that provides the best possible approximation of f on the set of "examples". This is the *learning* step. Needless to say, it is very important to choose an approximating function F that can represent f as well as possible. There would be little point in trying to learn, if the chosen approximation function $F(W, X)$ could only give a very poor representation of $f(X)$, even with optimal parameter values. Therefore, it is useful to separate three main problems:

1) the problem of which approximation to use, i.e. which classes of functions $f(X)$ can be effectively approximated by which approximating functions $F(W, X)$. This is a *representation* problem. Our motivation is similar to Minsky's and Papert's in studying the properties and limitations of Perceptrons to deal with specific classes of problems. The issue of *complexity* of the approximation arises naturally at this point. The complexity of the approximation – measured, for instance, by number of terms, is directly related to the *scaling* problem of the neural network literature (Rumelhart et al., 1986), to the concept of *order*, a central point in **Perceptrons** and to the *curse of dimensionality*, well-known in statistical estimation.

2) the problem of *which* algorithm to use for finding the optimal values of the parameters W for a given choice of F.

3) the problem of an efficient implementation of the algorithm in parallel, possibly analog hardware.

This paper deals with the first two of these problems and is especially focused on the first.

1.1 Networks and Approximation Schemes

Almost all approximation schemes can be mapped into some kind of network that can be dubbed as a "neural network" [1]. Networks, after all, can be regarded as a graphic notation for a large class of algorithms. In the context of our discussion, a network is a function

[1] Many instances of Neural Networks should be called Non-Neural-Networks, since their relation to biological neurons is weak at best

represented by the composition of many basic functions. To see how the approximation problem maps into a network formulation, let us introduce some definitions.

To measure the quality of the approximation, one introduces a *distance function* ρ to determine the distance $\rho[f(X), F(W, X)]$ of an approximation $F(W, X)$ from $f(X)$. The distance is usually induced by a norm, for instance the standard L_2 norm. The approximation problem can be then stated formally as:

Approximation problem *If $f(X)$ is a continuous function defined on set* $\mathbf{X}$, *and $F(W, X)$ is an approximating function that depends continuously on $W \in P$ and X, the approximation problem is to determine the parameters W^* such that*

$$\rho[F(W^*, X), f(X)] < \rho[F(W, X), f(X)]$$

for all W in the set P.

With these definitions we can consider a few examples of $F(W, X)$, shown in the figure 1:

- the classical linear case is

  $$F(W, X) = W \cdot X$$

  where W is an $m \times n$ matrix and X is an n-dimensional vector. It corresponds to a network without hidden units;

- the classical approximation scheme is linear in a suitable basis of functions $\Phi_i(X)$ of the original inputs X, that is

  $$F(W, X) = W \cdot \Phi_i(X)$$

  and corresponds to a network with one layer of hidden units. Spline interpolation and many approximation schemes, such as expansions in series of orthogonal polynomials, are included in this representation. When the Φ_i are products and powers of the input components X_i, F is a polynomial.

- the nested sigmoids scheme (usually called backpropagation, BP in short) can be written as

  $$F(W, X) = \sigma(\sum_n w_n \sigma(\sum_i v_i \sigma(...\sigma(\sum_j u_j X_j)...)))$$

  and corresponds to a multilayer network of units that sum their inputs with "weights" $w, v, u, \ldots$ and then perform a sigmoidal transformation of this sum. This scheme (of nested nonlinear functions) is unusual in the classical theory of the approximation of continuous functions. Its motivation is that

$$F(W, X) = \sigma(\sum_n w_n \sigma(\sum_j u_j X_j))$$

with σ being a linear threshold function, can represent all Boolean functions (any mapping S from $I = \{0, 1\}^N$ into $\{0, 1\}$ can be written as a disjunction of conjunctions, which in terms of threshold elements becomes the above expression, where biases or dummy inputs are allowed). Networks of this type, with one layer of hidden units, can approximate uniformly any continuous d-variate functions (see, for instance, Cybenko, 1989 or Funahashi, 1989; Cybenko, 1988 and Moore and Poggio, 1988, among others, proved the same result for the case of two layers of hidden units).

In general, each approximation scheme has some specific algorithm for finding the optimal set of parameters W. An approach that works in general, though it may not be the most efficient in any specific case, is some relaxation method, such as gradient descent or conjugate gradient or simulated annealing, in parameter space, attempting to minimize the error ρ over the set of examples. In any case, our discussion suggests that networks of the type used recently for simple learning tasks can be considered as specific methods of function approximation. This observation suggests that we approach the problem of learning from the point of view of classical approximation theory.

In this paper, we will be mainly concerned with the first of the problems listed earlier, that is the problem of developing a well-founded and sufficiently general approximation scheme, which maps into multilayer networks. We will only touch upon the second problem of characterizing efficient "learning" algorithms for estimating parameters from the data.

The plan of the paper is as follows. We first consider the question of whether *exact*, instead of approximated, representations are possible for a large class of functions, since a theorem of Kolmogorov has been sometime interpreted as supporting this claim. We conclude that exact representations with the required properties do not exist. Good and general approximating representations, however, may exist. Thus section 3 discusses the formulation of the problem of learning from examples as the problem of approximation of mappings and, in particular, as hypersurface reconstruction. From this point of view, regularization techniques used for surface reconstruction can be applied also to the problem of learning. The problem of the connection between regularization techniques and feedforward, multilayer networks is left open. Sections 4 and 5 provide an answer to this question by showing that regularization leads to an approximation scheme, called Generalized Radial Basis Functions (GRBFs), which is general, powerful and maps into a class of networks with one layer of hidden units. We show that GRBFs are mathematically strictly related to the well-known interpolation method of Radial Basis Functions. Section 4 reviews some of the existing results about RBF, while Section 5 derives the main result of this paper, that is the derivation of GRBFs from regularization. In section 6 we discuss how the framework of GRBFs encompasses several existing "neural network" schemes. The possible relevance of the work to neurophysiology is then briefly outlined, together with a number of interesting properties of gaussian radial basis functions. Section 8 discusses several extensions and applications of the method. We conclude with some comments on the crucial problem of dimensionality faced by this and any other learning or approximation technique[2].

[2]A preliminary version of the ideas developed here have appeared in (Poggio and the staff, 1989).

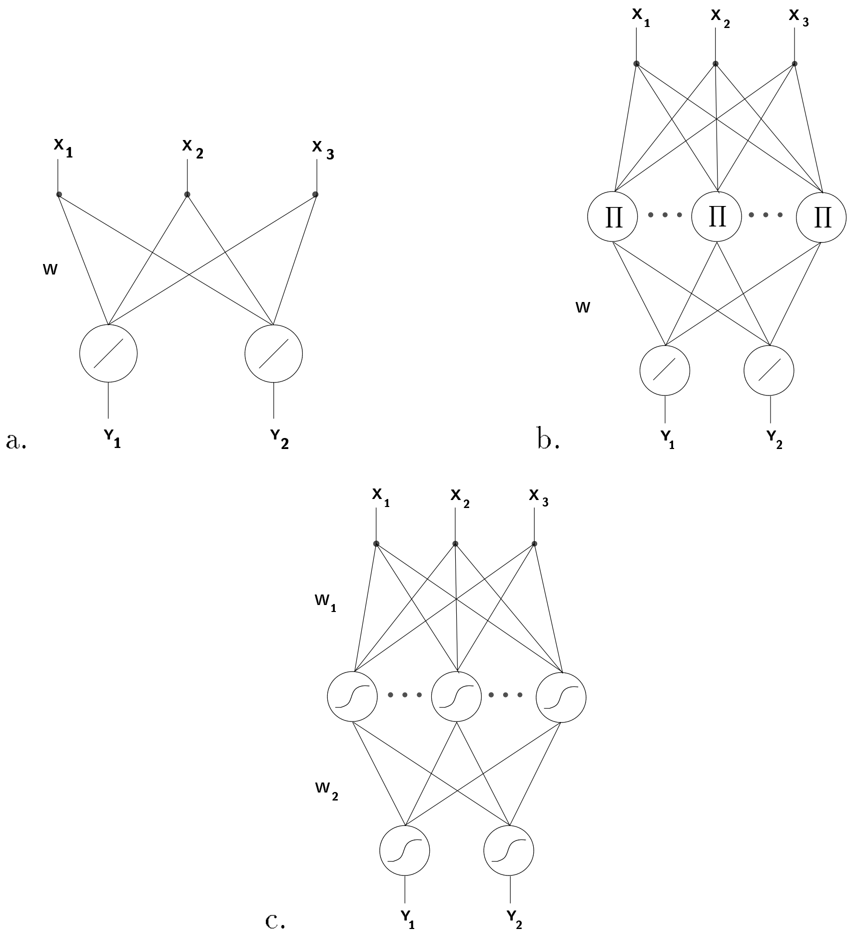

Figure 1: (a) A linear approximating function maps into a network without a hidden layer with linear units. The "weights" of the connections correspond to the matrix W – the linear estimator. (b) Polynomial estimators and other linear combinations of nonlinear "features" of the input correspond to networks with one hidden layer. In the case of polynomials the hidden units correspond to products (Π) and powers of the components of the input vector. The first layer of connections is fixed; the second is modified during "training". (c) A back-propagation network with one layer of hidden sigmoid units. Both sets of connections – from the input layer to the hidden units (w_1) and from there to the output layer (w_2) – are modified during training.

2 Kolmogorov's Theorem: An Exact Representation is Hopeless

Before discussing more extensively the approximation problem, it is obviously important to answer the question of whether an *exact* representation exists for continuous functions in terms of simpler functions. For instance, if all multivariate functions could be represented exactly and *nicely* as the sum or product of univariate ones, we could use networks consisting of units with just one input and one output. Recently, it has been claimed that a theorem of this type, due to Kolmogorov (1957), could be used to justify the use of multilayer networks (Hecht-Nielsen, 1987; Barron and Barron, 1988; see also Poggio, 1982). The original statement (Lorentz, 1976) is the following:

Theorem 2.1 (Kolmogorov, 1957) *There exist fixed increasing continuous functions $h_{pq}(x)$, on $I = [0,1]$ so that each continuous function f on I^n can be written in the form*

$$f(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} h_{pq}(x_p)),$$

where g_q are properly chosen continuous functions of one variable.

This result asserts that every multivariate continuous function can be represented by the superposition of a small number of univariate continuous functions. In terms of networks this means that every continuous function of many variables can be computed by a network with two hidden layers whose hidden units compute continuous functions (the functions g_q and h_{pq}). Can this be considered as a proof that a network with two hidden layers is a good and powerful representation? The answer is no. There are at least two reasons for this.

First, in a network implementation that has to be used for learning and generalization, some degree of smoothness is required for the functions corresponding to the units in the network. Smoothness of the h_q and of g is important because the representation must be smooth in order to generalize and be stable against noise. A number of results of Vituskin (1954, 1977) and Henkin (1967) show, however, that the inner functions h_{pq} of Kolmogorov's theorem are highly nonsmooth (they can be regarded as "hashing" functions, see Appendix A and Abelson, 1978). In fact Vitushkin (1954) had proved that there are functions of more than one variable which are not representable as the superposition of differentiable functions of one variable. Little seems to be known about the smoothness of g. Kahane (1975) shows that g can be represented as an absolutely convergent Fourier series. It seems that g could be either smooth or non-smooth, even for differentiable functions f.

The second reason is that useful representations for approximation and learning are *parametrized*: they correspond to networks with fixed units and modifiable parameters. Kolmogorov's network is not of this type: the form of g depends on the specific function f to be represented (the h_q are independent). g is at least as complex, in terms of bits needed to represent it, as f.

A stable and usable *exact* representation of a function in terms of networks with two or more layers seems hopeless. The result obtained by Kolmogorov can then be considered as a "pathology" of the continuous functions. Vitushkin's results however, leave completely open the possibility that multilayer *approximations* exist. In fact it has been recently proved (see, for instance, Funahashi, 1989; Moore and Poggio, 1988) that a network with two layers of hidden sigmoidal units can approximate arbitrarily well any continuous function. It is interesting to notice that this statement still holds true if there is just one hidden layer (Carrol and Dickinson, 1989; Cybenko, 1989; Funahashi, 1989). The expansion in terms of sigmoid functions can then be regarded as one of the possible choices for the representation of a function, although little is known about its properties. The problem of finding good and well founded approximate representations will be considered in the rest of the paper.

3 Learning as Hypersurface Reconstruction

If we consider learning from the perspective of approximation, we can draw an equivalence between learning smooth mappings and a standard approximation problem, surface reconstruction from sparse data points. In this analogy, learning simply means collecting the *examples*, i.e., the input coordinates x_i, y_i and the corresponding output values at those locations, the height of the surface d_i. This builds a look-up table. *Generalization* means estimating d in locations x, y where there are no examples, i.e. no data. This requires interpolating or, more generally, approximating the surface between the data points. Interpolation is the limit of approximation when there is no noise in the data. This example, given for a surface, i.e., the graph in $R^2 \times R$, corresponding to the mapping from R^2 to R, can be immediately extended to mappings from R^n to R^m (and graphs in $R^n \times R^m$). In this sense learning is a problem of *hypersurface reconstruction*. Notice that the other tasks of classification and of learning boolean functions may be regarded in a similar way. They correspond to the problems of approximating a mapping $R^n \rightarrow \{0, 1\}$ and a mapping $\{0, 1\}^n \rightarrow \{0, 1\}$, respectively.

3.1 Approximation, Regularization, and Generalized Splines

From the point of view of learning as approximation, the problem of learning a smooth mapping from examples is ill-posed (Courant and Hilbert, 1962; Hadamard, 1964; Tikhonov and Arsenin, 1977) in the sense that the information in the data is not sufficient to reconstruct uniquely the mapping in regions where data are not available. In addition, the data are usually noisy. *A priori* assumptions about the mapping are needed to make the problem well-posed. Generalization is not possible if the mapping is completely *random*. For instance, examples of the mapping represented by a telephone directory (people's names into telephone numbers) do not help in estimating the telephone number corresponding to a new name. Generalization is based on the fact that the world in which we live is usually – at the appropriate level of description – redundant. In particular, it may be *smooth*: small changes in some input parameters determine a correspondingly small change in the output (it may be necessary in some cases to accept *piecewise smoothness*). This is one of the most general and weakest constraints that makes approximation possible. Other, stronger *a priori* constraints may be known before approximating a mapping, for instance that the mapping is linear, or has a positive range, or a limited domain or is invariant to some group of transformations. Smoothness of a function corresponds to the function being not fully local: the value at one point depends on other values nearby. The results of Stone (1982, see section 9.2) suggest that if nothing else is known about the function to be approximated with dimensions, say, higher than about 10, the only option is to assume a high degree of smoothness. If the function to be approximated is not sufficiently smooth, the number of examples required would be totally unpractical.

3.2 Regularization Techniques for Learning

Techniques that exploit smoothness constraints in approximation problems are well known under the term of standard regularization. Consider the inverse problem of finding the hypersurface values z, given sparse data d. Standard regularization replaces the problem with the variational problem of finding the surface that minimizes a cost functional consisting of two terms (Tikhonov, 1963; Tikhonov and Arsenin, 1977; Morozov, 1984; Bertero, 1986; the first to introduce this technique in computer vision was Eric Grimson, 1981). The first term measures the distance between the data and the desired solution z; the second term measures the cost associated with a functional of the solution $\|Pz\|^2$ that embeds the *a priori* information on z. P is usually a differential operator. In detail, the problem is to find the hypersurface z that minimizes

$$\sum_i (z_i - d_i)^2 + \lambda \|Pz\|^2 \tag{1}$$

where i is a collective index representing the points in feature space where data are available and λ, the regularization parameter, controls the compromise between the degree of smoothness of the solution and its closeness to the data. Therefore λ is directly related to the degree of generalization that is enforced. It is well known that standard regularization provides solutions that are equivalent to generalized splines (Bertero et al., 1988). A large body of results in fitting and approximating with splines may be exploited.

3.3 Learning, Bayes Theorem and Minimum Length Principle

The formulation of the learning problem in terms of regularization is satisfying from a theoretical point of view. A variational principle such as equation (1) can be solidly grounded on Bayesian estimation (see Appendix D). Using Bayes theorem one expresses the conditional probability distribution $P_{z/d}(z;d)$ of the hypersurface z given the examples d in terms of a prior probability $P_z(z)$ that embeds the constraint of smoothness and the conditional probability $P_{d/z}(d;z)$ of d given z, equivalent to a model of the noise:

$$P_{z/d}(z;d) \propto P_z(z)\ P_{d/z}(d;z) \quad .$$

This can be rewritten in terms of *complexities* of hypothesis, defined as $C(\cdot) = -\log P(\cdot)$

$$C(z|d) = C(z) + C(d|z) + c \tag{2}$$

where c, which is related to $P_d(d)$, depends only on d. The MAP estimate corresponds to considering the z with minimum complexity $C(z|d)$. Maximum likelihood is the special case of MAP for uniform $C(z)$ (perfect *a priori* ignorance).

The maximum of this posterior probability (the MAP estimate) coincides with standard regularization, that is equation (1), provided that the noise is additive and gaussian and the prior is a gaussian distribution of a linear functional of z (see Appendix D). Under these conditions, the first term – $\sum_i \|z_i - d_i\|^2$ – in the regularization principle equation 1 corresponds to $C(d|z)$, whereas the second term – $\|Pz\|^2$ – corresponds to the prior $C(z)$.

Outside the domain of standard regularization, the prior probability distribution may represent other *a priori* knowledge than just smoothness. Piecewise constancy, for instance, could be used for classification tasks. Positivity, convexity, local behaviors of various types may be captured by an appropriate prior. Markov Random Field models, which can be considered as an extension of regularization, allow more flexibility in the underlying *generalization conditions*, for instance in terms of *piecewise smoothness*, by using *line processes* (see Geman and Geman, 1985 and Marroquin et al., 1987).

Notice that in practice additional a priori information must be supplied in order to make the learning problem manageable. *Space invariance* or other invariances to appropriate groups of transformations can play a very important role in effectively countering the dimensionality problem (see Poggio, 1982).

As pointed out by Rivest (in preparation), one can reverse the relationship between prior probabilities and complexity (see equation (2)). Instead of determining the complexity $C(z)$ in equation 2 from the prior, one may measure the *complexity* of the *a priori* hypotheses to determine the prior probabilities. Rissanen (1978), for instance, proposes to measure the complexity of a hypothesis in terms of the bit length needed to encode it. In this sense, the MAP estimate is equivalent to the Minimum Description Length Principle: the hypothesis z which for given d can be described in the most compact way is chosen as the "best" hypothesis. Similar ideas have been explored by others (for instance Solomonoff, 1978). They connect data compression and coding with Bayesian inference, regularization, hypersurface reconstruction and learning.

3.4 From Hypersurface Reconstruction to Networks

In the section above we have sketched the strict relations between learning, Bayes estimation, MRFs, regularization and splines; splines are equivalent to standard regularization, itself a special case of MRF models, which are a subset of Bayesian estimators. All these methods can be implemented in terms of parallel networks: in particular, we have argued that MRFs can be implemented in terms of hybrid networks of coupled analog and digital elements (Marroquin et al., 1987). Standard regularization can be implemented by resistive grids, and has been implemented on an analog VLSI chip (Harris, 1989). It is then natural to ask if splines, and more generally standard regularization, can be implemented by feedforward multilayer networks. The answer is positive, and will be given in the next few sections in terms of what we call Generalized Radial Basis Functions (GRBF). GRBFs are closely related to an interpolation technique called Radial Basis Functions (RBF), which has recent theoretical foundations (see Powell, 1987 for a review) and has been used with very promising results (Hardy, 1971; Franke, 1982; Rippa, 1984; Broomhead and Lowe, 1988; Renals and Rohwer, 1989; Casdagli, 1989) .

4 Radial Basis Functions: A Review

In the following sections we will describe the RBF technique, its feedforward network implementation and a straightforward extension that makes it usable for approximation rather than for interpolation.

4.1 The interpolation problem and RBF

The Radial Basis Function (RBF) method is one of the possible solutions to the real multivariate *interpolation problem*, that can be stated as follows:

Interpolation problem *given N different points $\{\mathbf{x}_i \in R^n | i = 1, \ldots N\}$ and N real numbers $\{y_i \in R | i = 1, \ldots N\}$ find a function F from R^n to R satisfying the interpolation conditions:*

$$F(\mathbf{x}_i) = y_i \quad i = 1, \ldots, N.$$

The RBF approach consists in choosing F from a linear space of dimension N, depending on the data points $\{\mathbf{x}_i\}$. The basis of this space is chosen to be the set of functions

$$\{h(\|\mathbf{x} - \mathbf{x}_i\|) | i = 1, \ldots N\}$$

where h is a continuous function from R^+ to R, usually called the *radial basis function*, and $\|\cdot\|$ is the Euclidean norm on R^n. Usually a polynomial is added to this basis, so that the solution to the interpolation problem has the following form:

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i h(\|\mathbf{x} - \mathbf{x}_i\|) + \sum_{i=1}^{m} d_i p_i(\mathbf{x}) \quad m \leq n \tag{3}$$

where $\{p_i | i = 1, \ldots, m\}$ is a basis of the linear space $\pi_{k-1}(R^n)$ of algebraic polynomials of degree at most $k-1$ from R^n to R, and k is given.

The interpolation conditions give N linear equations for the $(N+m)$ coefficients c_i and d_i in equation (3), so that the remaining degrees of freedom are fixed by imposing the following constraints:

$$\sum_{i=1}^{N} c_i p_j(\mathbf{x}_i) = 0, \quad j = 1, \ldots, m.$$

In order to discuss the solubility of the interpolation problem by means of this representation we need the following definition (Gelfand and Vilenkin, 1964; Micchelli, 1986):

Definition 4.1 *A continuous function $f(t)$, defined on $[0, \infty)$, is said to be conditionally (strictly) positive definite of order k on R^n if for any distinct points $\mathbf{x}_1, \ldots, \mathbf{x}_N \in R^n$ and scalars $c_1, \ldots, c_N$ such that $\sum_{i=1}^N c_i p(\mathbf{x}_i) = 0$ for all $p \in \pi_{k-1}(R^n)$, the quadratic form $\sum_{i=1}^N \sum_{j=1}^N c_i c_j f(\|\mathbf{x}_i - \mathbf{x}_j\|)$ is (positive) nonnegative.*

Notice that for $k = 0$ this class of functions, that we denote by $\mathcal{P}_k(R^n)$, reduces to the class of the (strictly) positive definite functions, that is the class of functions such that the quadratic form $\sum_{i=1}^N \sum_{j=1}^N c_i c_j f(\|\mathbf{x}_i - \mathbf{x}_j\|)$ is (positive) nonnegative (Schoenberg, 1938).

Well known results of approximation theory assert that a sufficient condition for the existence of a solution of the form (3) to the interpolation problem is that $h \in \mathcal{P}_k(R^n)$. It is then an important problem to give a full characterization of this class. In particular we are interested in characterizing the set of functions that are conditionally positive definite of order k over any R^n, that we define as simply $\mathcal{P}_k$.

4.1.1 RBF and Positive Definite Functions

The class $\mathcal{P}_0$ has been extensively studied (see Stewart, 1976, for a review), and we mention here one relevant result obtained by Schoenberg in 1938. Before stating his result we first give the following

Definition 4.2 *A function f is said to be completely monotonic on $(0, \infty)$ provided that it is $C^\infty(0, \infty)$ and $(-1)^l \frac{\partial^l f}{\partial x^l}(x) \geq 0$, $\forall x \in (0, \infty)$, $\forall l \in \mathcal{N}$, where $\mathcal{N}$ is the set of natural numbers.*

We define $\mathcal{M}_0$ as the set of all the functions that are completely monotonic on $(0, \infty)$. In 1938 Schoenberg was able to show the deep connection between $\mathcal{M}_0$ and $\mathcal{P}_0$. In fact he proved the following theorem:

Theorem 4.1 (Schoenberg, 1938) *A function $f(r)$ is completely monotonic on $(0, \infty)$ if and only if $f(r^2)$ is positive definite.*

This theorem asserts that the classes $\mathcal{M}_0$ and $\mathcal{P}_0$ are the same class, but Schoenberg went further, proving that in his theorem positive definitess can be replaced by *strictly* positive definitess, except for trivial cases. We can then conclude that it is possible to solve the interpolation problem with an expansion of the type

$$F(\mathbf{x}) = \sum_{i=1}^N c_i h(\|\mathbf{x} - \mathbf{x}_i\|) \tag{4}$$

if the function $h(\sqrt{t})$ is completely monotonic. The unknown coefficients c_i. can be recovered imposing the interpolation conditions $F(\mathbf{x}_j) = y_j$ $(j = 1, ...N)$, that substituted in equation (4) yields the linear system

$$y_j = \sum_{i=1}^N c_i h(\|\mathbf{x}_j - \mathbf{x}_i\|) \quad j = 1, ..., N.$$

Defining the vectors $\mathbf{y}$, $\mathbf{c}$ and the symmetric matrix H as follows

$$(\mathbf{y})_j = y_j, \qquad (\mathbf{c})_i = c_i, \qquad (H)_{ij} = h(\|\mathbf{x}_j - \mathbf{x}_i\|)$$

the coefficients of the expansion (4) are given by

$$\mathbf{c} = H^{-1}\mathbf{y}. \tag{5}$$

The theorem of Schoenberg ensures that the solution of system (5) always exists, since the matrix H can be inverted, being strictly positive definite. As an example of application of the theorem of Schoenberg we mention the functions e^{-r} and $(1+r)^{-\alpha}$ with $\alpha > 0$: since they are evidently completely monotonic the functions e^{-r^2} (Gaussians) and $(1+r^2)^{-\alpha}$ are strictly positive definite, and can be used as radial basis functions to interpolate any set of n-dimensional data points.

From equation (5) it turns out that a necessary and sufficient condition to solve the interpolation problem is the invertibility of the matrix H. Schoenberg's theorem, however, gives only a sufficient condition, so that many other functions could be used as radial basis functions without being strictly positive definite. Other sufficient conditions have been recently given by Micchelli, that in 1986 proved the following theorem:

Theorem 4.2 (Micchelli, 1986) *Let h be a continuous function on $[0, \infty)$ and positive on $(0, \infty)$. Suppose its first derivative is completely monotonic but not constant on $(0, \infty)$. Then for any distinct vectors $\mathbf{x}_1, ..., \mathbf{x}_N \in R^n$*

$$(-1)^{N-1} det \quad h(\|\mathbf{x}_i - \mathbf{x}_j\|^2) > 0 \ .$$

The essence of this theorem is that if the first derivative of a function is completely monotonic this function can be used as radial basis function, since the matrix H associated to it can be inverted. A new class of functions is then allowed to be used as radial basis functions. For instance the function $(c^2 + r)^\alpha$, with $0 < \alpha < 1$ and c possibly zero, is not completely monotonic, but satisfies the conditions of theorem (4.2), so that the choice $(c^2 + r^2)^\alpha$ is possible for the function h in (4).

A list of functions that can be used in practice for data interpolation is given below, and their use is justified by the results of Schoenberg or Micchelli:

$$h(r) = e^{-(\frac{r}{c})^2} \qquad (gaussian)$$

$$h(r) = \frac{1}{(c^2 + r^2)^\alpha} \qquad \alpha > 0$$

$$h(r) = (c^2 + r^2)^\beta \qquad 0 < \beta < 1$$

$$h(r) = \sqrt{r^2 + c^2} \qquad (multiquadric)$$

$$h(r) = r \qquad (linear)$$

Notice that the linear case corresponds, in one dimension, to piecewise linear interpolation, that is the simplest case of spline interpolation (further and stronger connections to spline interpolation will be discussed later). Notice that even the case $\beta = \frac{1}{2}$ has been explicitly mentioned, since it corresponds to interpolation by means of "multiquadric" surfaces. Multiquadrics have been introduced by Hardy (1971) and extensively used in surface interpolation with very good results (Franke, 1982; Rippa, 1984). Some of the functions listed above have been used in practice. Gaussian RBF have been used by Agterberg (1974) and Schagen (1979), and an approximating RBF expansion has been studied by Klopfenstein and Sverdlove (1983). The latter considered the case of equally spaced data and succeeded in giving some error estimates. RBF have even been used by Broomhead and Lowe (1988) and

Casdagli (1989) to predict the behavior of dynamical systems and by Renals and Rohwer (1989) for phoneme classification. A crude form of RBF performed better on the Nettalk task (Sejnowski and Rosenfeld, 1987) than Nettalk itself (Wolpert, 1988).

Almost all of these functions share the unpleasant property of depending on a parameter, that will generally depend on the distribution of the data points. However it has been noticed (Franke, 1982) that the results obtained with Hardy's multiquadrics (in 2 dimensions) seem not to depend strongly on this parameter, and that the surfaces obtained are usually very smooth. It is interesting to notice that, in spite of the excellent results, no theoretical basis existed for Hardy's multiquadrics before Micchelli's theorem. On the contrary, in the case of several functions, including the gaussian, a mathematical justification can be given in the context of regularization theory, as we shall see in section (6).

4.1.2 RBF and Conditionally Positive Definite Functions

The theorem of Schoenberg on the equivalence of $\mathcal{M}_0$ and $\mathcal{P}_0$ has been recently extended, to obtain an interesting characterization of $\mathcal{P}_k$. In fact in 1986 Micchelli proved:

Theorem 4.3 (Micchelli, 1986) *$h(r^2) \in \mathcal{P}_k$ whenever $h(r)$ is continuous on $[0, \infty)$ and $(-1)^k \frac{\partial^k h(r)}{\partial r^k}$ is completely monotonic on $(0, \infty)$*

To our extents the practical implication of this theorem is the following: if the function $h(r^2)$ is not positive definite we do not know if it can be used as radial basis function, since the matrix H could be singular, but if the k-th derivative of $h(r)$ is completely monotonic a polynomial of degree at most $k-1$ can be added to the expansion (4), (see equation (3)), so that it can be used to solve the interpolation problem. Notice that, according to a remark of Micchelli (1986), the converse of theorem (4.3) holds true: denoting by $\mathcal{M}_k$ the functions whose k-th derivative belongs to $\mathcal{M}$, $f(r^2) \in \mathcal{P}_k$ if and only if $f(r) \in \mathcal{M}_k$, so that $\mathcal{P}_k \equiv \mathcal{M}_k$.

It has been noticed (Micchelli, 1986; Powell, 1988) that this theorem embeds the results obtained by Duchon (1976, 1977) and Meinguet (1979, 1979a) in their variational approach to splines. For instance the functions $h(r) = r^{\frac{3}{2}}$ and $g(r) = \frac{1}{2} r \log \sqrt{r}$ are not completely monotonic, but this property holds for their second derivatives (that is they belongs to $\mathcal{M}_2$). By theorem 4.3 the functions $h(r^2) = r^3$ and $g(r^2) = r^2 \log r$ ("thin plate splines") belong to $\mathcal{P}_2$): It is then possible to interpolate any set of data points using $h(r^2)$ and $g(r^2)$ as radial basis functions and a linear term (polynomial of degree at most 2 -1) added to the expansion (4). This corresponds exactly to the expressions derived by Duchon and Meinguet, but without some of their limitations (see Example 2 in section 5.1.2). Since this method has been shown to embody natural spline interpolation in one dimension (Powell, 1988), can then be considered as an extension of natural splines to multivariable interpolation.

4.2 RBF and Approximation Theory

It is natural to ask if the expansion (3) is a good approximation method. The interpolation property of the RBF expansion, ensured by Micchelli's theorems, is neither sufficient nor necessary to guarantee good results. In particular, a very important question that is always addressed in approximation theory is whether the solution can be prevented from badly oscillating between the interpolation points when they become dense. This does not happen when spline functions are used, due to the smoothness constraint, but it could happen when other radial basis functions are chosen.

Several results have been obtained about this question in recent years. Perhaps the most important of them has been obtained by Jackson (1988). He addressed a more fundamental and general question: given a multivariate function $f(\mathbf{x})$ and a set of N data points $\{\mathbf{x}_i | i = 1, ..., N\}$, do there exist a sequence of functions $F_N(\mathbf{x})$, with

$$F_N(\mathbf{x}) = \sum_{i=1}^{N} c_i^N h(\|\mathbf{x} - \mathbf{x}_i\|) + \sum_{j=1}^{k} \alpha_j^N p_j(\mathbf{x})$$

and some bounded open domain on which

$$|F_N(\mathbf{x}) - f(\mathbf{x})| \rightarrow 0 \quad as \quad N \rightarrow \infty \ ?$$

Jackson gave sufficient conditions on h for this result to hold. In particular he considered the function $h(r) = r$ and showed that these conditions are satisfied in R^{2n+1} but not in R^{2n}. These results are encouraging and make this approach a highly promising way of dealing with irregular sets of data in multi-dimensional spaces.

Finally we mention the problem of noisy data. Since in this case a strict interpolation is meaningless, the RBF method must be modified. The problem consists in solving the linear system $H\mathbf{c} = \mathbf{y}$ in a way that is robust against noise. A very simple solution to this problem has been provided in the context of regularization theory (Tikhonov and Arsenin, 1977). It consists in replacing the matrix H by $H + \lambda I$, where I is the identity matrix, and λ is a "small" parameter, whose magnitude is proportional to the amount of noise in the data points. The coefficients of the RBF expansion are then given by

$$\mathbf{c} = (H + \lambda I)^{-1}\mathbf{y}. \tag{6}$$

This equation gives an *approximating* RBF expansion, and the original interpolating expansion is recovered by letting λ go to zero. Since data are usually noisy, from now on we will refer to the RBF expansion as the one computed by means of equation (6).

4.3 Network Implementation

A remarkable property of this technique is that it can be implemented by a simple network with just one layer of hidden units, as shown in figure 2 (see Broomhead and Lowe, 1988). For simplicity we restrict ourselves to expansions of the type (4), disregarding the polynomial term, the results being valid even in the more general case in which it is included.

The first layer of the network consists of "input" units whose number is equivalent to the number of independent variables of the problem. The second layer is composed by nonlinear "hidden" units fully connected to the first layer. There is one unit for each data point $\mathbf{x}_i \equiv (x_i, y_i, z_i, \cdots)$ parametrized by its "center", which has the coordinates $(x_i, y_i, z_i, \cdots)$ of the data point itself. The output layer, fully connected to the hidden layer, consists of one (or more) linear unit(s), whose "weights" are the unknown coefficients of the RBF expansion. These output units may also represent a fixed, nonlinear, invertible function, as already observed by Broomhead and Lowe (1988) and discussed in a later section. This is

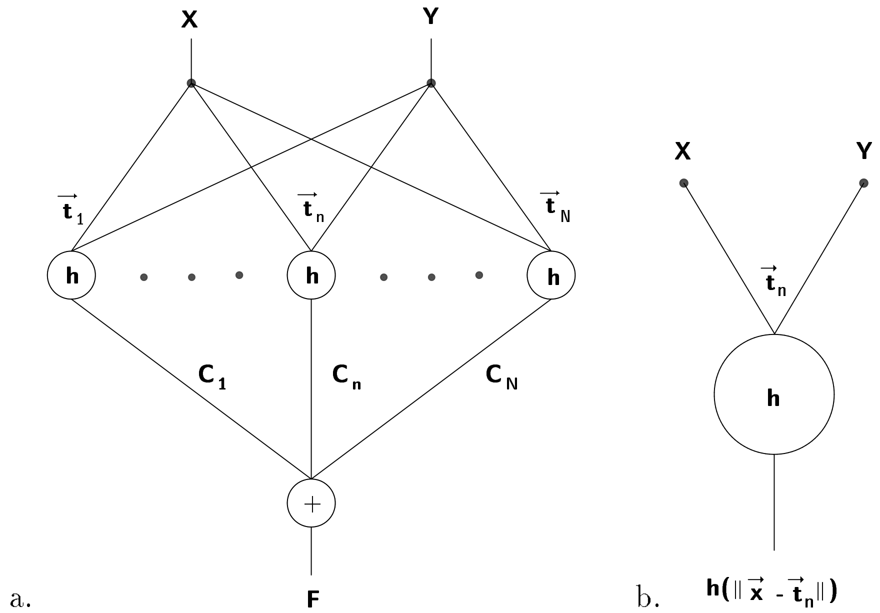

Figure 2: *a) The Radial Basis Function network for the interpolation of a bivariate function F. The radial hidden units h evaluate the functions $h(\|\mathbf{x}-\mathbf{t}_n\|)$. A fixed invertible nonlinear function may be present after the final summation. b) A radial hidden unit h. The input is given by $\mathbf{x} \equiv (x, y)$ and its parameters are the coordinates of the n-th "center" $\mathbf{t}_n$. The output is the value of the radial basis function h, centered on $\mathbf{t}_n$, at point $\mathbf{x}$. The centers $\mathbf{t}_n$ coincide in this RBF case with the data points $\mathbf{x}_n$*

.

useful for instance in the case of classification tasks. All the learning algorithms discussed later extend trivially to this case.

Notice that the architecture of the network is completely determined by the learning problem, and that, unlike most of the current "neural" networks, there are no unknown weights connecting the input and the hidden layer. Since spline interpolation can be implemented by such a network, and splines are known to have a large power of approximation, this shows that a high degree of approximation can be obtained by just one hidden layer network.

Of course this method has its drawbacks. The main one is that the method is global, that is each data point contributes to the value of the interpolating function at every other point. The computation of the coefficients of the RBF expansion can become then a very time consuming operation: its complexity grows polynomially with N, (roughly as N^3) since an $N \times N$ matrix has to be inverted. In a typical application to surface reconstruction from sparse stereo data the number of data points can easily be more than 3000: to invert a sparse 3000×3000 matrix not only is a formidable task, but may not be meaningful, since we know that the probability of ill-conditioning is higher for larger and larger matrices (it grows like N^3 for a $N \times N$ uniformly distributed random matrix) (Demmel, 1987). Another problem with the Radial Basis Function method, and with interpolating methods in general, is that data are usually noisy and therefore not suitable for interpolation; an approximation of the data would be preferable. In Section 5 a solution to these problems will be given in the context of regularization theory. The next section presents a method that has been proposed by Broomhead and Lowe (1988) to reduce computational complexity and gives as result an *approximation* instead of an interpolation. A new result is given, supporting its validity.

4.4 Approximated Radial Basis Function

In this section we show how the Radial Basis Functions can also be used for approximation rather than for interpolation. In the RBF approach the basis on which the interpolating function is expanded is given by a set of radial functions h translated and centered on the data points. The interpolating function is then a point in a multidimensional space, whose dimension is equal to the number of data points, which could be very large. As usual when dealing with spaces of a such high dimensionality we could ask if all the dimensions are really significant.

This suggests that the RBF expansion could be approximated by an expansion in a basis with a smaller number of dimensions (Broomhead and Lowe, 1988). This can be accomplished by an expansion of the following form:

$$F(\mathbf{x}) = \sum_{\alpha=1}^{K} c_\alpha h(\|\mathbf{x} - \mathbf{t}_\alpha\|) \tag{7}$$

where the $\mathbf{t}_\alpha$ are K points, that we call "centers" or "knots", whose coordinates have to be chosen and $K < N$. It is clear from equation (7) that the interpolation conditions can no longer be satisfied. Imposing $F(\mathbf{x}_i) = y_i$ leads to the following linear system:

$$y_j = \sum_{\alpha=1}^{K} c_\alpha h(\|\mathbf{x}_j - \mathbf{t}_\alpha\|) \quad j = 1, ..., N.$$

This system is overconstrained, being composed of N equations for K unknowns, and the problem must be then regularized. A least-squares approach can be adopted (see also Broomhead and Lowe, 1988) and the optimal solution can be written as

$$\mathbf{c} = H^+ \mathbf{y} \tag{8}$$

where $(H)_{i\alpha} = h(\|\mathbf{x}_i - \mathbf{t}_\alpha\|)$ and H^+ is the Moore-Penrose pseudoinverse of H (Penrose, 1955; Ben-Israel and Greville, 1974). The matrix H is rectangular ($N \times K$) and its pseudoinverse can be computed as

$$H^+ = (H^T H)^{-1} H^T$$

provided $(H^T H)^{-1}$ exists. The matrix $H^T H$ is square and its dimensionality is K, so that it can be inverted in time proportional to K^3. A rough estimate suggests that this technique could speed the computations by a factor $(\frac{N}{K})^3$. Of course, other methods for computing the pseudoinverse exist, including recursive ones (see Albert, 1972).

As in the previous case this formulation makes sense if the matrix $H^T H$ is nonsingular. Micchelli's theorem is still relevant to this problem, since we prove the following corollary:

Corollary 4.4.1 *Let G be a function satisfying the conditions of Micchelli's theorem and $\mathbf{x}_1, ..., \mathbf{x}_N$ an N-tuple of vectors in R^n. If H is the $(N - s) \times N$ matrix H obtained from the matrix $G_{i,j} = G(\|\mathbf{x}_i - \mathbf{x}_j\|)$ deleting s arbitrary rows, then the $(N - s) \times (N - s)$ matrix $H^T H$ is not singular.*

To prove this corollary it is sufficient to notice that, since Micchelli's theorem holds, the rank of G_{ij} is N. A theorem of linear algebra states that deleting s rows from a matrix of rank N yields a matrix of rank $N - s$. Remembering that $rank(AA^T) = rank(A)$ for every rectangular matrix A we have that $rank(HH^T) = rank(H) = N - s$; then HH^T is not singular.

This result asserts that if the set of knots is chosen to be a subset of the set of data, and if the conditions of Micchelli's theorem are satisfied, the pseudoinverse can be computed as $H^+ = (H^T H)^{-1} H^T$. Other choices for the set of knots seem possible in practice: for example Broomhead and Lowe (1988) use uniformly spaced knots to predict successfully chaotic time series, in conjunction with a gaussian Radial Basis Function.

We remark that while this technique is useful for dealing with large sets of data, it has been proposed as a way of dealing with noise. In the next section, we will show, however, that if the only problem is noise, an approximating technique of radial basis functions with as many centers as data points can be derived in a rigorous way with the aid of regularization theory. An extension of the RBF method will be presented in the general framework of regularization theory.

5 Regularization Approach and Generalized Radial Basis Functions

In this section we derive from regularization theory an alternative approximation method based on a basis of radial functions. We apply regularization theory to the approximation/interpolation problem and we show that for a large class of stabilizers the regularized solution is an expansion of the radial basis function type. The approach leads to a representation, that we call Generalized Radial Basis Functions (GRBFs), which is very similar to RBFs while overcoming the problem of the computational complexity of RBFs for large data sets. The GRBF technique can then be considered the point of contact between multilayer feedforward networks and the mathematical apparatus of standard regularization theory.

5.1 Regularization Theory

Let $S = \{(\mathbf{x}_i, y_i) \in R^n \times R | i = 1, ...N\}$ be a set of data that we want to approximate by means of a function f. The regularization approach (Tikhonov, 1963; Tikhonov and Arsenin, 1977; Morozov, 1984; Bertero, 1986) consists in looking for the function f that minimizes the functional

$$H[f] = \sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))^2 + \lambda \|Pf\|^2$$

where P is a constraint operator (usually a differential operator), $\|\cdot\|$ is a norm on the function space to whom f belongs (usually the L^2 norm) and λ is a positive real number, the so called *regularization parameter*. The structure of the operator P embodies the a priori knowledge about the solution, and therefore depends on the nature of the particular problem that has to be solved. Minimization of the functional H leads to the associated Euler-Lagrange equations. For a functional $H[f]$ that can be written as

$$H[f] = \int_{R^n} dxdy...\mathcal{L}(f, f_x, f_y, ..., f_{xx}, f_{xy}, f_{yy}, ...f_{yy...y})$$

the Euler-Lagrange equations are the following (Courant and Hilbert, 1962)

$$\mathcal{L}_f - \frac{\partial}{\partial x}\mathcal{L}_{f_x} - \frac{\partial}{\partial y}\mathcal{L}_{f_y} + \frac{\partial^2}{\partial x^2}\mathcal{L}_{f_{xx}} + \frac{\partial^2}{\partial y^2}\mathcal{L}_{f_{yy}}$$
$$+\frac{\partial^2}{\partial x \partial y}\mathcal{L}_{f_{xy}} + \cdots + (-1)^n \frac{\partial^n}{\partial y^n}\mathcal{L}_{f_{yy\ldots y}} + \cdots = 0 \ .$$

The functional H of the regularization approach is such that they can always be written as

$$\hat{P}P\ f(\mathbf{x}) = \frac{1}{\lambda}\sum_{i=1}^{N}(y_i - f(\mathbf{x}))\delta(\mathbf{x} - \mathbf{x}_i) \tag{9}$$

where $\hat{P}$ is the adjoint of the differential operator P and the right side comes from the functional derivative with respect to f of the data term of H.

Equation (9) is a partial differential equation, and it is well known that its solution can be written as the integral trasformation of its right side with a kernel given by the Green's function of the differential operator $\hat{P}P$, that is the function G satisfying the following distributional differential equation:

$$\hat{P}P\ G(\mathbf{x};\xi) = \delta(\mathbf{x}-\xi)\ .$$

Because of the delta functions appearing in equation (9) the integral transformation becomes a discrete sum and f can then be written as

$$f(\mathbf{x}) = \frac{1}{\lambda}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i))G(\mathbf{x};\mathbf{x}_i)\ . \tag{10}$$

It is important to notice that a polynomial term should in general be added to the right-hand side of equation (10), depending on the specific stabilizer. Equation (10) says that the solution of the regularization problem lives in an N-dimensional subspace of the space of smooth functions. A basis for this subspace is given by the N functions $G(\mathbf{x};\mathbf{x}_i)$, that is by the Green's function G "centered" on the data points $\mathbf{x}_i$.

A set of equations for the unknown coefficients $c_i = \frac{y_i - f(\mathbf{x}_i)}{\lambda}$ is easily obtained by evaluating equation (10) at the N data points $\mathbf{x}_i$. A straightforward calculation yields the following linear system:

$$(G+\lambda I)\mathbf{c} = \mathbf{y} \tag{11}$$

where I is the identity matrix, and we have defined

$$(\mathbf{y})_i = y_i\ , \qquad (\mathbf{c})_i = c_i\ , \qquad (G)_{ij} = G(\mathbf{x}_i;\mathbf{x}_j)\ .$$

We then conclude that the solution to the regularization problem is given by the following formula

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i G(\mathbf{x};\mathbf{x}_i) \tag{12}$$

where the coefficients satisfy the linear system (11) and a polynomial term is in general added.

Notice that since $\|Pf\|^2$ is quadratic the corresponding operator in equation (9) is self-adjoint and can be written as $\hat{P}P$. Then the Green's function is symmetric: $G(\mathbf{x};\xi) = G(\xi;\mathbf{x})$. If $\|Pf\|^2$ is translationally invariant G will depend on the difference of its arguments ($G = G(\mathbf{x}-\xi)$) and if $\|Pf\|^2$ is rotationally and translationally invariant G will be a radial function: $G = G(\|\mathbf{x}-\xi\|)$.

Let us now compare equations (12) and (11) with equations (4) and (6). It is clear that *if the stabilizers P is rotationally invariant then the regularized solution is given by an expansion in radial functions*. The requirement of rotational and translational invariance on $\|Pf\|^2$ is very common in practical applications. Clearly regularization with a non-radial stabilizer P justifies the use of appropriate non-radial basis functions, retaining all the approximation properties associated with the Tikhonov technique. Examples of P and corresponding G will be given in section 5.1.2.

5.1.1 Micchelli's condition and regularization define almost identical classes of radial basis functions

At this point it is natural to ask about the relation between the class of functions defined by stabilizers P of the Tikhonov type in regularization theory (the class $\mathcal{T}$) and the class $\mathcal{P}_k$ (for all k) of conditionally positive definite functions of order k (which is identical with $\mathcal{M}_k$). The two classes have to be very closely related, since they originate from the same problem – optimal approximation. The regularization approach gives – if the stabilizer is radially symmetric – radial functions G that satisfy

$$\hat{P}P\ G(\|\mathbf{x}-\xi\|) = \delta(\mathbf{x}-\xi)\ . \tag{13}$$

Notice that if P contains a term proportional to the function itself, then $\hat{P}P$ contains a constant term; by taking the Fourier transform of equation (13) and applying Bochner's theorem (1932, 1959) on the representation of positive definite functions (Stewart, 1976), it turns out that G is positive definite, that is $G \in \mathcal{P}_0$. (See section 5.1.2 for details). In general equation (13) implies (Gelfand and Vilenkin, 1964) that G is conditionally positive definite (of an order determined by P). This discussion suggests that $\mathcal{P}_k \equiv \mathcal{M}_k \supset \mathcal{T}$ (for all k): in fact a function G may satisfy the conditions of Micchelli's theorem 4.3 ($G \in \mathcal{M}_k$), and therefore be conditionally positive definite of order k ($G \in \mathcal{P}_k$), without satisfying equation (13) for any operator P ($G \notin \mathcal{T}$). We conjecture that these functions G, $G \in \mathcal{P}_k$, $G \notin \mathcal{T}$ are interpolating functions but not good approximating functions, since they do not come from a regularization approach. We have little reasons for this conjecture, apart from Jackson's result (1988): in an even number of dimensions the function $h(r) = r$ does not satisfy his (sufficient!) conditions for good approximation and is not the Green's function of any known Tikhonov stabilizer, though it can be used as radial basis function, according Micchelli's theorem (4.2), since $h(\sqrt{r}) \in \mathcal{M}_1$. Notice that in R^{2n+1}, $h(r) = r$ satisfies Jackson's conditions, is the Green's function of the thin-plate stabilizer and satisfies Micchelli's conditions. Hardy's multiquadrics $H(r) = (c^2 + r^2)^{\frac{1}{2}}$ are a possible counterexample to our conjecture, since they are conditionally positive definite of order one ($H \in \mathcal{P}_1$), numerical work suggests that they have good approximation properties and we have been so far unable to obtain an expansion in terms of multiquadric from any regularization principle. We are happy to leave the answer to these questions to real mathematicians.

5.1.2 Examples

Example 1

We now consider a wide class of stabilizers and show that they lead to a solution of the regularization problem that has the form of a radial basis function expansion of the type of equation (4), with a positive definite basis function and without the polynomial term.

Let us consider the class of constraint operators defined by

$$\|P_1 f\|^2 = \int_{R^n} d\mathbf{x} \sum_{m=0}^{\infty} a_m (P^m f(\mathbf{x}))^2 \tag{14}$$

where $P^{2m} = \nabla^{2m}$, $P^{2m+1} = \nabla\nabla^{2m}$, ∇^2 is the Laplacian operator and the coefficients a_m are real positive numbers. The stabilizer is then translationally invariant and the Green's function satisfies the distributional differential equation:

$$\sum_{m=0}^{\infty}(-1)^m a_m \nabla^{2m} G(\mathbf{x}-\xi) = \delta(\mathbf{x}-\xi) \quad . \tag{15}$$

By Fourier transforming both sides of equation (15) we obtain:

$$\sum_{m=0}^{\infty} a_m(\omega \cdot \omega)^m \; G(\omega) = 1 \tag{16}$$

and by Fourier anti-transforming $G(\omega)$ we have for the Green's function $G(\mathbf{x})$:

$$G(\mathbf{x}) = \int_{R^n} d\omega \frac{e^{i\omega\cdot\mathbf{x}}}{\sum_{m=0}^{\infty} a_m(\omega\cdot\omega)^m} = \int_{R^n} d\omega e^{i\omega\cdot\mathbf{x}} dV(\omega) \tag{17}$$

where $V(\omega)$ is a bounded non-decreasing function if $a_0 \neq 0$. Now we can apply Bochner's theorem (1932), which states that a function is positive definite if and only if it can be written in the form (17), to conclude that $G(\mathbf{x})$ is positive definite. Notice that the condition $a_0 \neq 0$ is crucial in this particular derivation, and, as it has been pointed out by Yuille and Grzywacz (1988), *it is a necessary and sufficient condition for the Green's function to fall asymptotically to zero.* Let us now see some examples.

One example is provided by the following choice of the coefficients:

$$a_0 = 1, \quad a_1 = 1, \quad a_n = 0 \;\; \forall n \geq 2 \; .$$

In this case the Green's function (here in one dimension) becomes the Fourier transform of $\frac{1}{1+\omega^2}$, and then

$$G(x) \propto e^{|x|}.$$

Clearly this function is not very smooth, reflecting the fact that the stabilizer consists of derivatives of order 0 and 1 only. Smoother functions can be obtained allowing a larger (possibly infinite) number of coefficients to be different from zero. For instance, setting

$$a_k = \frac{1}{(2k!)}$$

and remembering that $\sum_{m=0}^{\infty} \frac{\omega^{2m}}{(2k!)} = \cosh(\omega)$ we obtain $G(x) = \frac{1}{\cosh(x)}$, which is a very smooth, bell-shaped function.

Another interesting choice (Yuille and Grzywacz, 1988) is:

$$a_m = \frac{\sigma^{2m}}{m!2^m}$$

which gives as Green's function a multidimensional Gaussian of variance σ. The regularized solution is then a linear superposition of Gaussians centered on the data points $\mathbf{x}_i$. Its physical interpretation is simple: regarding σ as "time" the solution satisfies the heat equation:

$$\frac{\partial f(\mathbf{x},\sigma)}{\partial \sigma} = \nabla^2 f(\mathbf{x},\sigma)$$

with boundary conditions $f(\mathbf{x}_i,\sigma) = y_i$. The regularized solution for $\lambda = 0$ can then be regarded as the pattern of temperature of a conducting bar which is in contact, at the points $\mathbf{x}_i$, with infinite heat sources at temperature y_i. The value of σ is then related to the diffusion time.

Example 2

A widely used class of stabilizers is given by the functionals considered by Duchon (1976, 1977) and Meinguet (1979, 1979a) in their variational approach to multivariate interpolation, that is one of the possible generalizations of spline theory from one to many dimensions. In particular they considered *rotationally invariant* functionals of the form

$$H_m[f] = \sum_{i_1 \ldots i_m}^{n} \int_{R^n} d\mathbf{x} \|\partial_{i_1 \ldots i_m} f(\mathbf{x})\|^2$$

where $\partial_{i_1 \ldots i_m} = \frac{\partial^m}{\partial x_{i_1} \ldots \partial x_{i_m}}$ and $1 \leq m$. The solution to this variational problem is of the form

$$F(\mathbf{x}) = \sum_{i=1}^{N} c_i h^m(\|\mathbf{x} - \mathbf{x}_i\|) + \sum_{i=1}^{k} d_i p_i(\mathbf{x}) \tag{18}$$

which is exactly the same as equation (3). Here m is a measure of the degree of smoothness of the solution, and is related to the maximum polynomial precision that can be obtained by the relation: $k \leq m$. Since the stabilizers are rotationally invariant, the corresponding Green's functions h^m are radial, and for each fixed value of m they turn out to be $h^m(r) = r^{2m-n} \ln r$ if $n \leq 2m$ for n even, and $h^m(r) = r^{2m-n}$ otherwise.

As an example we show the case $n = m = 2$. The functional to be minimized is

$$H_2[f] = \int_{R^2} dx dy \left[\left(\frac{\partial^2 f}{\partial x^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x \partial y} \right)^2 + \left(\frac{\partial^2 f}{\partial y^2} \right)^2 \right]$$

and the Green's function h is the well known "thin plate spline" $h(r) = r^2 \ln r$. In this case a linear term appears as the second term of the right hand side of equation (18). Thin plate splines have been introduced by engineers for aeroelastic calculations (Harder and Desmareis, 1972), their name coming from the fact that H_2 is the bending energy of a thin plate of infinite extent. The results obtained with thin plate splines are comparable with those obtained with Hardy's multiquadrics. Another radial basis function that has been extensively used is r^3, with a linear term in the expansion as well, wich is equivalent to cubic splines in the one dimensional case.

5.2 Extending to Movable Centers: GRBF

In the previous section we have shown that the function minimizing the functional H is specified by N coefficients, where N is the number of data points. Therefore, when N becomes large, the regularization approach suffers from the same drawbacks of the RBF method that have been pointed out in a previous section. To reduce the computational complexity of the representation, we then write an approximate solution f^* of the regularization problem as an expansion involving a fewer number of centers, as done by Broomhead and Lowe (1988), that do not necessarily coincide with some of the data points $\mathbf{x}_i$, that is:

$$f^*(\mathbf{x}) = \sum_{\alpha=1}^{n} c_\alpha G(\mathbf{x}; \mathbf{t}_\alpha) \tag{19}$$

where the coefficients c_α and the centers $\mathbf{t}_\alpha$ are unknown. We now have to face the problem of finding the n coefficients c_α and the $d \cdot n$ coordinates of the centers $\mathbf{t}_\alpha$ so that the expansion (19) is optimal. In this case we dispose of a natural definition of optimality, given by the functional H. We then impose the condition that the set $\{c_\alpha, \mathbf{t}_\alpha | \alpha = 1, ..., n\}$ must be such that it minimizes $H[f^*]$, and the following equations must be satisfied:

$$\frac{\partial H[f^*]}{\partial c_\alpha} = 0 \ , \qquad \frac{\partial H[f^*]}{\partial \mathbf{t}_\alpha} = 0, \quad \alpha = 1, ..., n \ . \tag{20}$$

We call an expansion of the type of equation (19) with the coefficients satisfying equation (20) a *Generalized Radial Basis Function (GRBF) expansion.*

The explicit form of equation (20) depends on the specific constraint operator that has been used. We perform here the computations for the constraint operator $\|P_1 f^*\|^2$ considered in the Example 1 of the previous section, with boundary conditions such the function f and all its derivatives vanish on the border of the integration domain. The main difficulty is to find the explicit expression of $\|P_1 f\|^2$ as a function of c_α and $\mathbf{t}_\alpha$. To accomplish this task we notice that, by using Green's formulas (Courant and Hilbert, 1962), which are the multidimensional analogue of integration by parts, and using our boundary conditions, the m-th term of P_1 can be written as

$$\int_{R^n} d\mathbf{x} (P^m f(\mathbf{x}))^2 = (-1)^m \int_{R^n} d\mathbf{x} f(\mathbf{x}) P^{2m} f(\mathbf{x}) \ . \tag{21}$$

Substituting equation (21) in P_1, and using definition of the differential operator $\hat{P}P$, we obtain

$$\|P_1 f\|^2 = \int_{R^n} d\mathbf{x} f(\mathbf{x}) \hat{P}_1 P_1 \ f(\mathbf{x}) \ . \tag{22}$$

When f^* is substituted in equation (22) each term containing $G(\mathbf{x})$ gives a delta function and the integral disappears, yielding:

$$\|P_1 f^*\|^2 = \sum_{\alpha,\beta=1}^{n} c_\alpha c_\beta G(\mathbf{t}_\alpha; \mathbf{t}_\beta) \ .$$

Defining a rectangular $N \times n$ matrix G as $(G)_{i\alpha} = G(\mathbf{x}_i; \mathbf{t}_\alpha)$ and a symmetric $n \times n$ square matrix as $(g)_{\alpha\beta} = G(\mathbf{t}_\alpha; \mathbf{t}_\beta)$, $H[f^*]$ can finally be written in the following simple form:

$$H[f^*] = \mathbf{c}(G^T \ G + \lambda g)\mathbf{c} - 2\mathbf{c}G^T\mathbf{y} + \mathbf{y} \cdot \mathbf{y} \ . \tag{23}$$

Notice that equation (23) is a quadratic form in the coefficients c_α, so that minimization with respect to them is easily done. For each fixed set of centers $\mathbf{t}_\alpha$ the optimal vector $\mathbf{c}$ is then given by

$$\mathbf{c} = (G^T\ G + \lambda g)^{-1} G^T \mathbf{y}\ . \tag{24}$$

Notice that if we let λ go to zero and G is radial (and centers are fixed) the approximate method of Broomhead and Lowe is recovered. Their method, however, lacks the capability of looking for the optimal set of knots of the expansion, as required by regularization theory. The possibility of moving the knots, by a procedure of the gradient descent type, could noticeably improve the quality of approximation. It has been mentioned by Broomhead and Lowe (1988), and has been used by Moody and Darken (1989) (their method is a heuristic version of RBF with moving centers, see section 6.4). Notice that from the point of view of approximation theory this is a nonlinear problem that reminds us of the splines with free knots, that are splines whose knots are allowed to vary with the function being approximated (De Vore and Popov, 1987; Braess, 1986).

Clearly this approximated solution does not satisfy equation (10) anymore. However an explicit computation shows that equation (10) is satisfied at the centers, that is

$$f^*(\mathbf{t}_\alpha) = \sum_{i=1}^{N} \frac{y_i - f^*(\mathbf{x}_i)}{\lambda} G(\mathbf{t}_\alpha; \mathbf{x}_i)\ . \tag{25}$$

The converse is also true: if one fixes the set of knots and requires that equation (25) hold for each α, equation (24) is easily recovered.

5.3 GRBF and Gradient Descent

In the previous section we introduced the GRBF method to approximate the regularized solution. It requires the minimization of the multivariate function $H[f^*]$, which is not convex in general. Gradient-descent is probably the simplest approach for attempting to find the solution to this problem, though, of course, it is not guaranteed to converge. Several other iterative methods, such as versions of conjugate gradient and simulated annealing (Kirkpatrick et al., 1983) may be better than gradient descent and should be used in practice: in the following we will consider for simplicity (stochastic) gradient descent. It is straightforward to extend our equations to other methods. In the gradient descent method the values of c_α and $\mathbf{t}_\alpha$ that minimize $H[f^*]$ are regarded as the coordinates of the stable fixed point of the following dynamical system:

$$\dot{c_\alpha} = -\omega \frac{\partial H[f^*]}{\partial c_\alpha}, \quad \alpha = 1, \ldots K$$

$$\dot{\mathbf{t}_\alpha} = -\omega \frac{\partial H[f^*]}{\partial \mathbf{t}_\alpha}, \quad \alpha = 1, \ldots K$$

where ω is a parameter determining the microscopic timescale of the problem and is related to the rate of convergence to the fixed point. The gradient descent updating rules are given by the discretization (in time) of the previous equations. Clearly, since the function $H[f^*]$

is not convex, more than one fixed point could exist, corresponding to local, suboptimal minima. To overcome this problem one could use "stochastic" gradient descent by adding a random term to the gradient descent equations. They become then stochastic equations of the Langevin type, often used to model the relaxation of a physical system toward the equilibrium in the presence of noise (Wax, 1954; Ma, 1976; Parisi, 1988). In this case the learning process is governed by the following stochastic equations

$$\dot{c_\alpha} = -\omega \frac{\partial H[f^*]}{\partial c_\alpha} + \eta_\alpha(t), \quad \alpha = 1, ...K \tag{26}$$

$$\dot{\mathbf{t}_\alpha} = -\omega \frac{\partial H[f^*]}{\partial \mathbf{t}_\alpha} + \epsilon_\alpha(t), \quad \alpha = 1, ...K \tag{27}$$

where η_α and ϵ_α are white noise of zero mean and variance

$$< \eta_\alpha(t)\eta_\beta(t') > \; = \; < \epsilon_\alpha(t)\epsilon_\beta(t') > \; = \; 2T\delta_{\alpha\beta}\delta(t-t')$$

where T measures the power of the noise, analog to temperature. Solving these equations is similar to using Montecarlo methods of the Metropolis type (Metropolis at al., 1953) (decreasing the variance of the noise term during the relaxation is similar to performing stochastic annealing).

We now consider for simplicity the case in which the Green's function is radial ($G(\mathbf{x};\mathbf{t}) = h(\|\mathbf{x}-\mathbf{t}\|^2)$) and λ is set to zero. Defining the interpolation error as

$$\Delta_i = y_i - y_i^* \quad ,$$

where $y_i^* = f^*(\mathbf{x}_i)$ is the response of the network to the i-th example, we can write the gradient terms as

$$\frac{\partial H}{\partial c_\alpha} = -2 \sum_{i=1}^{N} \Delta_i h(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2) \quad , \tag{28}$$

$$\frac{\partial H}{\partial \mathbf{t}_\alpha} = 4c_\alpha \sum_{i=1}^{N} \Delta_i h'(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2)(\mathbf{x}_i - \mathbf{t}_\alpha) \tag{29}$$

where h' is the first derivatives of h. Equating $\frac{\partial H}{\partial \mathbf{t}_\alpha}$ to zero we notice that at the fixed point the knot vectors $\mathbf{t}_\alpha$ satisfy the following set of nonlinear equations:

$$\mathbf{t}_\alpha = \frac{\sum_i P_i^\alpha \mathbf{x}_i}{\sum_i P_i^\alpha} \quad \alpha = 1, \ldots, K$$

where $P_i^\alpha = \Delta_i h'(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2)$. The optimal knots are then a weighted sum of the data points. The weight P_i^α of the data point i for a given knot α is high if the interpolation error Δ_i is high there *and* the radial basis function centered on that knot changes quickly in a neighbor of the data point. This observation could suggest faster methods for finding a quasi-optimal set of knots. Notice that if the radial basis function h depends on a parameter ϵ, that is if $h = h(r, \epsilon)$, the functional $H[f^*]$ must be minimized even with respect to this parameter. The following equation must then be added to equations (28) and (29):

$$\frac{\partial H}{\partial \epsilon} = -2 \sum_{i,\ \alpha} \Delta_i c_\alpha \frac{\partial}{\partial \epsilon} h(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2, \epsilon). \tag{30}$$

As we mentioned before, an invertible nonlinearity, such a sigmoid, may be present at the output of the network. In this case

$$f^*(\mathbf{x}) = \sigma(\sum_{\alpha=1}^{N} c_\alpha h(\|\mathbf{x} - \mathbf{t}_\alpha\|^2)) \ . \tag{31}$$

holds and equations (28) and (29) are modified in the following way:

$$\frac{\partial H}{\partial c_\alpha} = -2 \sum_{i=1}^{N} \sigma'(y_i^*) \Delta_i h(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2) \quad , \tag{32}$$

$$\frac{\partial H}{\partial \mathbf{t}_\alpha} = 4 c_\alpha \sum_{i=1}^{N} \sigma'(y_i^*) \Delta_i h'(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2)(\mathbf{x}_i - \mathbf{t}_\alpha) \tag{33}$$

In the gradient descent equations nothing forbids that two or more centers may move towards each other until they coincide. Clearly, this should be avoided (it corresponds to a degeneracy of the solution) in an efficient algorithm. A formal way to ensure that centers do never overlap is to add a term to the functional that it is minimized of the form $\sum_{\alpha \neq \beta} \Psi(\|\mathbf{t}_\alpha - \mathbf{t}_\beta\|)$, where Ψ is an appropriate repulsive potential, such as $\Psi(y) = \frac{1}{y^2}$. Equations (28) and (29) can be easily modified to reflect this additional term (see Girosi and Poggio, 1989a). In practice, it may be sufficient to have a criterion that forbids to any two centers to move too close to each other.

In terms of networks the GRBF method has the same implementation of RBF. The only difference is that the parameters of the hidden layer are now allowed to vary, being the coordinates of the knots of the GRBF expansion. From this point of view the GRBF network is similar to backpropagation in the sense that there are two layers of weights to be modified by a gradient descent method, and there are in principle local minima. figure 3 shows the network that should be used in practice for finding the parameters. Notice that a constant, a linear term (and possibly higher order polynomials) should be added to the radial basis representation (depending on the stabilizer, polynomials may be in the null space of the regularized solution, as for thin-plate splines but not for gaussian radial basis functions). Equations (28) and (29) should be modified appropriately. Figure 3 makes this clear.

5.3.1 A semiheuristic algorithm

The previous discussion and some of the analogies discussed later (with the k-means algorithm and with Moody's approach) suggest the following heuristic algorithm for GRBFs.

1. set the number of centers and use the k-means algorithm or a comparable one (or even equation 29 with $\Delta_i = constant$) to find the initial positions $\mathbf{t}_\alpha$ of the centers. Alternatively, and more simply, set the initial value of the centers to a subset of the examples

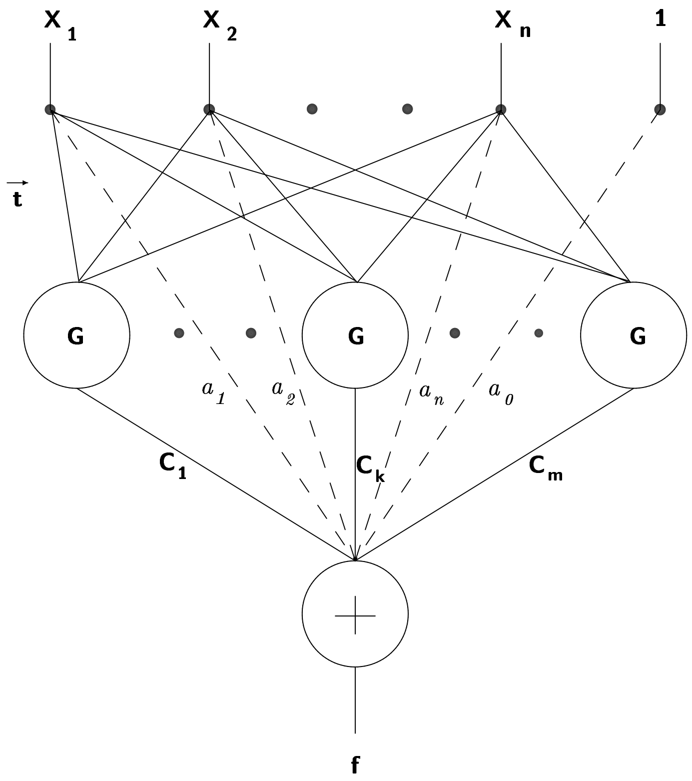

Figure 3: The GRBF network used to approximate a mapping between $x_1, x_2, ..., x_n$ and y, given a set of sparse, noisy data. In addition to the linear combination of radial basis functions, the network shows other terms that contribute to the output: constant and linear terms are shown here as direct connections from the input to the output with weights $a_0, a_1, a_2, \dot{a}_n$. Constant, linear and even higher order polynomials may be needed, depending on the stabilizer P. Gaussian radial basis functions, on the other hand, may not need additional terms.

2. use the pseudoinverse technique to find the values of the coefficients c_α (see equation 24)

3. use the $\mathbf{t}_\alpha$ and c_α found so far as initial values for equations (29) and (28)

4. explore how performance changes by changing incrementally the number of centers.

6 Relations with other methods

We have been impressed by the generality of this formulation and by how many existing schemes can be understood within this framework. In this section, we will mention briefly some of the most obvious connections with existing methods. In the next sections we will discuss possible extensions of the method, its relation to a specific style of computation that has biological undertones, its meaning from a Bayes point of view, and finally, some general points about the most crucial problem of learning, the "curse of dimensionality".

6.1 GRBF and Classification Tasks

RBF and GRBF are an interpolation and approximation method for continuous, in fact smooth, functions, as shown by the fact that they are generalized splines. It is quite natural to ask whether the method can be modified to deal with piecewise constant functions, i.e., with classification tasks, and in particular boolean functions. More precisely, we have so far considered GRBFs as a technique to solve the problem of approximating real valued functions $f : R^n \to R^m$; we ask now whether they can be specialized to deal with the problem of approximating functions $h : R^n \to \{0,1\}$ (the classification problem) and $t : \{0,1\}^n \to \{0,1\}$ (the boolean learning problem).

Computer experiments show that without any modification, Gaussian radial basis functions can be used to learn successfully XOR (Broomhead and Lowe, 1988). A simple form of RBF was shown to perform well on the classification task of NetTalk (Wolpert, 1988). We expect therefore that classification tasks could be performed by the GRBF method described earlier using a basis of smooth functions. In a similar way, backpropagation networks with smooth sigmoid nonlinearities have been used in several classification tasks. Thus, it seems that the method can be used, as it stands, for classification tasks and for learning boolean functions. The question remains, however, whether a special basis of radial functions could be used advantageously. Consider the task of learning boolean functions. In backpropagation networks, the boolean limit corresponds to the smooth sigmoid nonlinearities becoming linear threshold functions. The obvious boolean limit of radial basis functions is a basis of hyperspheres:

$$S_{\alpha,d}(x) = 1 \quad for \;\; ||x - x_\alpha|| \leq d \;\; else = 0 \tag{34}$$

where $S_{\alpha,d}$ is the hypersphere of radius d centered in x_α, x, d are boolean vectors and $||\cdot||$ is the Hamming distance. The use of such a basis may be similar to closest match classification (which can also be used for continuous mappings by using Euclidean metric instead of Hamming distance). Of course, this basis does not satisfy Micchelli's condition, and cannot be derived from regularization. It may be tempting to conjecture that arbitrarily close smooth approximations exist that do. The obvious case of a difference of sigmoids, however, does not satisfy Micchelli's condition, since its first derivative is not completely monotonic:

$$S(x) = 1 - \frac{1}{1 + \alpha e^{x}} - \frac{1}{1 + \alpha e^{-x}} = \frac{\alpha^2 - 1}{(1 + \alpha^2) + 2\alpha \cosh(x)}.$$

In the boolean limit of backpropagation, one knows that a network with one hidden layer can represent any boolean function given a sufficient number of hidden units (because it is well known that any boolean function can be written as a threshold of linear combinations of threshold functions). The representation may require in general a large number of hidden units because it amounts to a disjunctive normal form of the boolean function. In a similar way, a basis of hyperspheres can be used to represent any boolean function by having a sufficient number of "centers," one for each term in the disjunctive normal form of the function.

Seen in a more geometrical way (consider the case of a binary classification problem on R), the boolean limit of RBFs carves the n-dimensional input space into hyperspheres, whereas the linear threshold limit for BP carves the space into regions bounded by hyperplanes. It seems clear that each of the partitions can be made to approximate the other arbitrarily well, given a sufficient number of hidden units and/or centers.

6.2 GRBF and Backpropagation

GRBF are similar, but not identical, to backpropagation networks with one hidden layer, since: a) they also have one hidden layer of smooth differentiable functions; b) they are updated by a gradient descent method (as backpropagation) that operates on two "layers" of parameters, the c_i and the $\mathbf{t}_\alpha$; c) the update moves the "centers" of those blurred hyperspheres (in the Gaussian case) and their weight in the final layer, whereas in backpropagation "blurred" hyperplanes are shifted during learning.

From the point of view of approximation and regularization theory GRBFs have solid theoretical foundations, a property that is not (yet) shared by backpropagation. However, some results on the approximating power of backpropagation networks have been recently obtained (Carrol and Dickinson, 1989; Cybenko, 1989; Funahashi 1989; Arai, 1989). Their essence is that a network with one hidden layer of sigmoid units can synthesize arbitrary well any continuous function, but may require a very large number of hidden units. Among other advantages relative to backpropagation networks, the simplest version of our scheme – a radial basis function network with centers fixed and centered at the examples – is guaranteed to perform quite well, to have an efficient implementation (there are no local minima in the error functional) and to be equivalent to a powerful approximation technique, that is generalized splines. Interestingly, this network may represent the initial step in a gradient descent procedure for synthesizing a more powerful GRBF network (compare section 5.3.1).

6.3 GRBF and Vector Quantization

The classification limit of GRBF (with the basis of equation (34) and $||\cdot||$ the euclidean distance, say) is clearly related to vector quantization (Gersho, 1982). Vector quantization of a signal vector f involves subdivision of the n-dimensional vector space into J decision regions D_j, each enclosing one of the J reconstruction values. The signal vector f is quantized to the reconstruction vector r_j, if f lies in the decision region D_j. In terms of equation 34, this mean that $S_j(f) = 1$, and the domains in which the S_i are nonzero are disjoint.

6.4 GRBF and Kohonen's Algorithm

Kohonen (1982) suggested an algorithm to establish a topology conserving and dimensionality reduction map from a set of inputs in a high dimensional space. The algorithm has been suggested in order to describe maps similar to those that form between cortical areas. It has also been used for several other tasks, such as learning motor movements (Ritter and Schulten, 1986, 1987). Kohonen's algorithm can be regarded as a special form of the k-means method (MacQueen, 1967) for finding the centers of n clusters in a set of inputs. It turns out that this is what the update equation in the $\mathbf{t}_\alpha$ does, i.e.

$$\frac{\partial E}{\partial \mathbf{t}_\alpha} = 4c_\alpha \sum_{i=1}^{N} \Delta_i h'(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2)(\mathbf{x}_i - \mathbf{t}_\alpha) \tag{35}$$

with Δ_i = constant. The differences with respect to Kohonen's algorithm are that a) each center is affected by all data points and not only by the ones that "belong" to it b) h' depends on $||\mathbf{x}_i - \mathbf{t}_\alpha||$ rather than on the distance between the "locations" i and α. Notice that the addition of a "repulsive" terms in the functional to avoid overlapping centers make the analogy with Kohonen's algorithm even closer.

Intuitively, the last equation with $\Delta_i = \mathit{constant}$ adjusts the $\mathbf{t}_\alpha$ to the centers of the cluster of the data. This analogy suggests a heuristic scheme to improve convergence of the gradient descent method: first find the centers of the clusters with an algorithm like equation (35) with $\Delta_i = \mathit{constant}$, then use the full scheme (equations (28) and (29)). The heuristic should help the method to avoid local minima.

Moody and Darken (1989) propose a similar heuristic as the core of their method; they first find the position of the "centers" (they do not use, however, strict Radial Basis Functions) with the k-means algorithm and then find the coefficients c_i of the expansion. The k-means algorithm and Kohonen's are also related to vector quantization (see above). Other so-called competitive learning algorithms are similar to Kohonen's algorithm.

6.5 GRBF, Kanerva's Model and Marr's Model of the Cerebellum

Kanerva introduced in 1984 a memory model called Sparse Distributed Memory (SDM). Keeler (1988) has shown that the SDM can be regarded as a three-layer network with one layer of hidden units (see figure 4). In this description, the first layer consists of n boolean units that represent input vectors (binary addresses) $\mathbf{a}$. The hidden layer consists of m binary units s (the select vectors) with $m >> n$. The weights between the input layer and the hidden layer are given by the matrix A, with rows that correspond to the storage locations. Each output unit (in SDM there are n output units) is also binary with connection weights to the hidden units given by the matrix C, which is updated during learning by a Hebbian learning rule. Thus given an input address $\mathbf{a}$, the selected locations are:

$$\mathbf{s} = S_{0,d}(A\mathbf{a}),$$

where $S_{0,d}$ was defined in section 6.1 (we assume $x_0 = 0$). C is set to be the sum of the outer products of desired outputs $\mathbf{d}$ and selected locations, i.e. $C = \sum_j \mathbf{d}_j \mathbf{s}_j^T$.

Clearly the analogy with the boolean limit of RBF is complete: the matrix A contains the locations of the centers (the selected locations) and C corresponds to the c of RBF. In the SDM, the center locations are fixed. The Hebbian one-step update rule can be regarded as a zero-order approximation of the gradient descent scheme, which is equivalent, for fixed centers, to the pseudoinverse (see Appendix B.2).

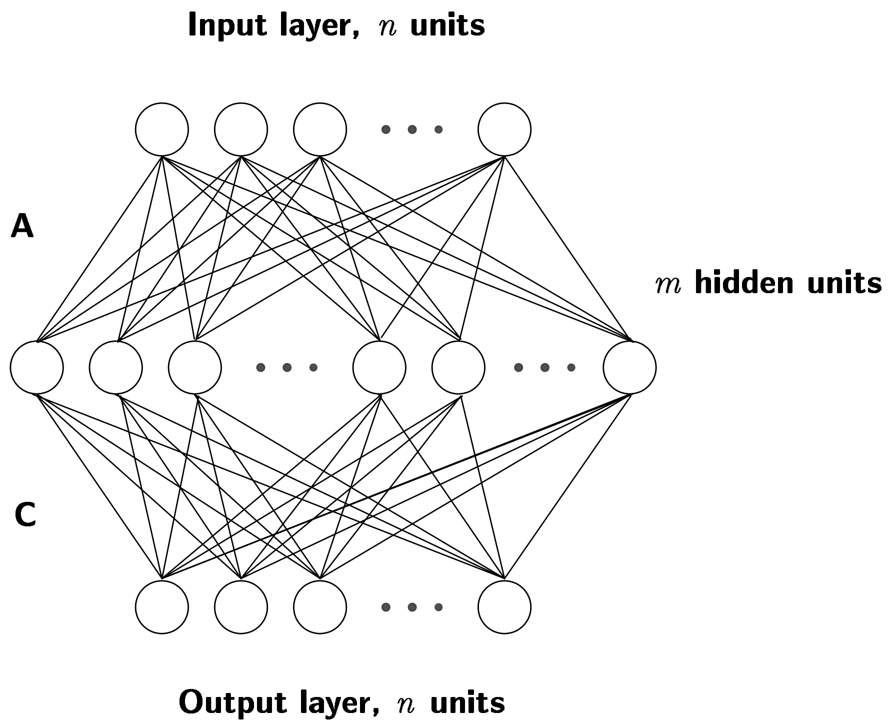

Figure 4: Kanerva's SDM represented as a three-layer network. The matrix A of connections between the first layer and the hidden layer contains the locations of the m centers. The matrix C of modifiable connections between the hidden layer and the output layer corresponds to the coefficients c in the RBF formulation (redrawn from Keeler, 1988).

Keeler has discussed the similarities between the SDM model and the Hopfield model. He has also pointed out that Kanerva's SDM is very similar in mathematical form to a model of the cerebellum introduced by Marr (1969) and Albus (1971). In our language, the mossy fibers are the input lines, the granule cells correspond to the centers (the granule cells are the most populous neurons in the brain), and the Purkinje cells correspond to the output units (the summation). Other cells, according to Marr and Albus, are involved in what we would call control functions; the basket cells that receive another input may inhibit the Purkinje cells, whereas the stellate cells could change the gain of the Purkinje cells or their threshold. The Golgi cells receive inputs from the granule cells (the centers), and feedback into the mossy fiber - granule cell connections. It may be interesting to consider in more detail whether the circuitry of the cerebellum may have anything to do with the more general continuous update scheme described by equations (28) and (29).

Keeler also suggests modifying the SDM to optimize the addresses A_i according to $A_i^{new} = A_i^{old} - \lambda(A_i^{old} - x)$. This is about the same as Kohonen's algorithm and reduces (for $d = 1$) to the unary representation of Baum, Moody and Wilczek (1988). Keeler provides interesting estimates of the performance of SDM. In conclusion, Kanerva's algorithm can be regarded as a special case of the boolean limit of GRBF, which again provides a more general framework and a connection with continuous approximation.

6.6 Other Methods

The GRBF formulation seems to contain as special cases two well-known schemes in the field of pattern recognition. One is *Parzen windows* (Parzen, 1962), which is an approximation scheme typically used to estimate probability distributions, and which is remarkably similar to a simple form of RBF (Duda and Hart, 1973). The other scheme is the method of *potential functions* for determining discriminant functions (Duda and Hart, 1973). The method was originally suggested by the idea that if the samples were thought of as points in a multidimensional space and if electrical charges were placed at these points, the electrostatic potential would serve as a useful discriminant function $g(\mathbf{x}) = \sum_i q_i K(\mathbf{x}, \mathbf{x}_i)$, with K being typically a radial function (as classical electrostatic potentials are). Potentials such as the gaussian have also been used. GRBF (and RBF) may be used to give a more rigorous foundation to these two rather heuristic methods.

GRBF has also similarities with the class of Memory-Based Reasoning methods, recently used by D. Waltz and coworkers on massively parallel machines, since in its simplest version (as many centers as examples) it is essentially a look-up table that finds those past instances that are sufficiently close to the new input.

7 Gaussian GRBFs and science-fiction neurobiology

In this section we point out some remarkable properties of gaussian Generalized Radial Basis Functions, that may have implications for neurobiology, for VLSI hardware implementations and, from a conceptual point of view, for extending in interesting directions the GRBF approach.

7.1 Factorizable Radial Basis Functions

The synthesis of radial basis functions in high dimensions may be easier if they are factorizable. It can be easily proven that *the only radial basis function which is factorizable is the Gaussian.* A multidimensional gaussian function can be represented as the product of lower dimensional gaussians. For instance a 2D gaussian radial function centered in $\mathbf{t}$ can be written as:

$$G(\|\mathbf{x} - \mathbf{t}\|^2) \equiv e^{-\|\mathbf{x}-\mathbf{t}\|^2} = e^{-(x-t_x)^2} e^{-(y-t_y)^2} \ . \tag{36}$$

This dimensionality factorization is especially attractive from the physiological point of view, since it is difficult to imagine how neurons could compute $h(\|\mathbf{x} - \mathbf{t}_\alpha\|^2)$ in a simple way for dimensions higher than two. The scheme of figure 5, on the other hand, is physiologically plausible. Gaussian radial functions in one and two dimensions can be readily implemented as *receptive fields* by weighted connections from the sensor arrays (or some retinotopic array of units representing with their activity the position of features).

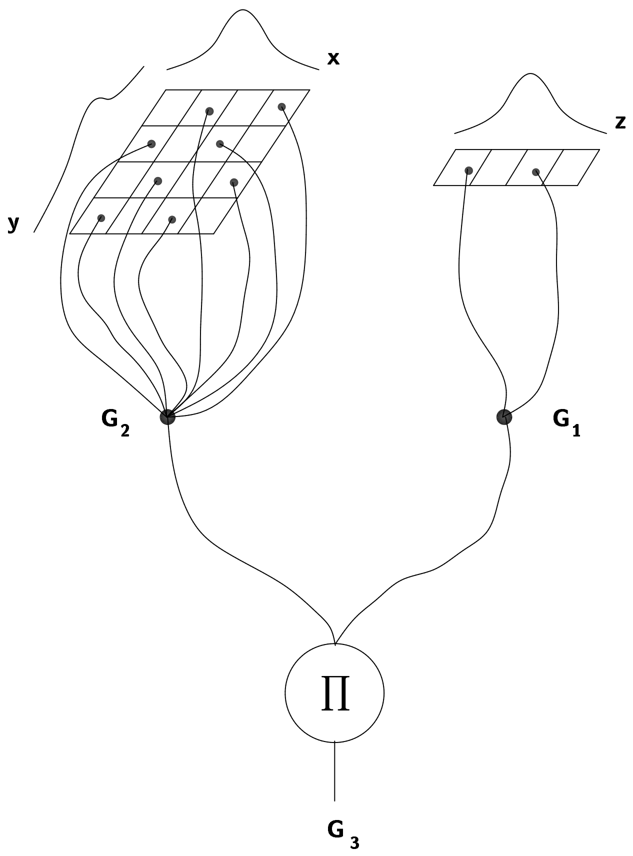

Figure 5: *A three-dimensional radial gaussian implemented by multiplying two-dimensional gaussian and one-dimensional gaussian receptive fields. The latter two functions are synthesized directly by appropriately weighted connections from the sensor arrays, as neural receptive fields are usually thought to arise. Notice that they transduce the implicit position of stimuli in the sensor array into a number (the activity of the unit). They thus serve the dual purpose of providing the required "number" representation from the activity of the sensor array and of computing a gaussian function. 2D gaussians acting on a retinotopic map can be regarded as representing 2D "features", while the radial basis function represents the "template" resulting from the conjunction of those lower-dimensional features.*

Physiological speculations aside, this scheme has three interesting features from the point of view of a hardware implementation and also in purely conceptual terms. Consider the example of a GRBF network operating on images:

1. the multidimensional radial functions are synthesized directly by appropriately weighted connections from the sensor arrays, without any need of an explicit computation of the norm and the exponential.

2. 2D gaussians operating on the sensor array or on a retinotopic array of features extracted by some preprocessing transduce the implicit position of features in the array into a number (the activity of the unit). They thus serve the purpose of providing the required "number" representation from the "array" representation.

3. 2D gaussians acting on a retinotopic map can be regarded as representing 2D "features", while each radial basis function represents the "template" resulting from the conjunction of those lower-dimensional features. Notice that in this analogy the radial basis function is the AND of several features and could also include the negation of certain features, that is the AND NOT of them. The scheme is also hierarchical, in the sense that a multidimensional gaussian "template" unit may be a "feature" input for another radial function (again because of the factorization property of the gaussian). Of course a whole GRBF network may be one of the inputs to another GRBF network.

7.2 Style of Computation and Physiological Predictions

The multiplication operation required by the previous interpretation of gaussian GRBFs to perform the "conjunction" of gaussian receptive fields is not too implausible from a biophysical point of view. It could be performed by several biophysical mechanisms (see Koch and Poggio, 1987). Here we mention three mechanisms:

1. inhibition of the silent type and related circuitry (see Torre and Poggio, 1978; Poggio and Torre, 1978)

2. the AND-like mechanism of NMDA receptors

3. a logarithmic transformation, followed by summation, followed by exponentiation. The logarithmic and exponential characteristic could be implemented in appropriate ranges by the sigmoid-like pre-to-postsynaptic voltage transduction of many synapses.

If the first or the second mechanism are used, the product of figure 5 can be performed directly on the dendritic tree of the neuron representing the corresponding radial function (alternatively, each dendritic tree may perform pairwise products only, in which case a logarithmic number of cells would be required). The GRBF scheme also requires a certain amount of memory per basis unit, in order to store the center vector. In the gaussian case the center vector is effectively stored in the position of the 2D (or 1D) receptive fields and in their connections to the product unit(s). This is plausible physiologically. The update equations are probably not. Equation (28) or a somewhat similar, quasi-hebbian scheme is not too unlikely and may require only a small amount of plausible neural circuitry. Equation (29) seems more difficult to implement for a network of real neurons (as an aside, notice that in the gaussian case the term $h'(\|\mathbf{x}_i - \mathbf{t}_\alpha\|^2)$ has an interesting form !). It should be stressed, however, that the centers may be moved in other ways – or not at all! In the gaussian case, with basis functions synthesized through the product of gaussian receptive fields, moving the centers means establishing or erasing connections to the product unit. This can be done on the basis of rules that are different from the full equation 29, such as, for instance, competitive learning, and that are biologically more plausible. Computer experiments are needed to assess the efficacy of different strategies to learn the optimal position of the centers.

The GRBF method with the Gaussian suggests an intriguing metaphor for a computational strategy that the brain may use, in some cases. Computation, in the sense of generalization from examples, would be done by superposition of receptive fields in a multidimensional input space. In the case of Gaussian radial basis functions, the multidimensional receptive fields could be synthesized by combining lower dimensional receptive fields, possibly in multiple stages. From this point of view, some cells would correspond to radial functions with centers in a high dimensional input space, somewhat similar to prototypes or coarse "grandmother cells", a picture that seems superficially consistent with physiological evidence. They could be synthesized as the conjunction of gaussian weighted positive and negative features in 2D retinotopic arrays.

Notice that from this perspective the computation is performed by *gaussian receptive fields* and their combination (through some approximation to multiplication), rather than by threshold functions. The basis units may not even need to be all radial, as we discuss in the next section. The view is in the spirit of the key role that the concept of receptive field has always played in neurophsyiology. It predicts the existence of low-dimensional feature-like cells and multidimensional Gaussian-like receptive fields, somewhat similar to template-like cells, a fact that could be tested experimentally on cortical cells. [3]

7.3 An example: recognizing a 3D object from its perspective views

To illustrate the previous remarks let us consider the following specific task in model-based visual recognition.

A set of visible points on a 3D object $P_1, P_2, \cdots, P_n$ maps under perspective projection onto corresponding points on the image plane, whose coordinates will be indicated by $(x_1, y_1),\ (x_2, y_2), \cdots (x_n, y_n)$. The set of these coordinates represents a *view*. To each view j, obtained by a $3D$ rigid transformation of the object, we associate the vector $\mathbf{v}^j = (x_1^j,\ y_1^j, x_2^j,\ y_2^j,\ \cdots, x_n^j,\ y_n^j)$ in a $2n$-dimensional space. Following Basri and Ullman (1989), who had considered the simpler case of orthographic projection plus scaling (they prove the surprising result that any view can be obtained as the *linear* combination of a small number of other appropriate views), we would like to synthesize a system that takes any view of the specific object as input and provides a *standard* view $\mathbf{v}_0$ as output. The view of a different object would lead to something different from $\mathbf{v}_0$. Is the task feasible? The following argument shows that it is. So called structure-from-motion theorems prove that for a moving rigid object as few as 2 perspective views and a small number of corresponding

[3] We conjecture that Andersen's data, modeled by Zipser and Andersen (1988) with a backpropagation network, may be better accounted for by a GRBF network. In the latter scheme (see previous discussion) the radial functions are the product of 2D gaussians representing the visual receptive fields and 1D gaussians representing the eye position. This accounts immediately for the multiplicative property of the V7 cells found by Andersen. Their activities are then superimposed to obtain the desired mapping into head-centered coordinates.

points are sufficient for recovering the motion and the 3D structure of the object (the available results, without a numerical stability analysis and under rather weak conditions, are: 8 points (Longuet-Higgins, 1981) , 5 points (quoted by Longuet-Higgins, 1981), 7 points (Tsai and Huang, 1982)). Thus a small number of perspective views of an object contains the necessary information for computing any perspective view of the same object (provided the points of interest are always visible).

We can use GRBF to achieve this goal, in the following way. We have (see figure 5) $2n$ inputs to accommodate the vectors $\mathbf{v}_j$ and, say K radial basis functions initially centered each on one of a subset of the M views used to "learn" the system ($M \geq K$). Instead of one output only as shown in figure 5 the system will have $2n$ outputs corresponding to the components of the standard view $\mathbf{v}_0$. Thus for each of the M inputs in the training set the desired output is $\mathbf{v}_0$. In the simple case of "fixed centers" the network must attempt to satisfy

$$V_0 = GC$$

where V_0 is the $(M, 2N)$ matrix whose rows are all equal to the standard view, C is the $(K, 2N)$ matrix of the coefficients c and G is the (M, K) matrix $G(\|\mathbf{v}_j - \mathbf{t}_\alpha\|^2)$. The best solution in the least square sense is

$$C = G^+ V_0.$$

Computer simulations show that the GRBF network generalizes successfully to views that are not part of the training set (Poggio and Edelman, 1990). We are presently exploring several issues, such as how performance degrades with decreasing number of views and of radial basis functions.

In general it is better to have as few centers as possible and move them by using equations 26 and 27. Notice that each center corresponds to a view or to some "intermediate" view. Also notice that the required multidimensional gaussians can be synthesized by the product of two-dimensional gaussian receptive fields, looking in the retinotopic space of the features corresponding to $P_1, P_2, \cdots, P_n$. There would be a two-dimensional gaussian receptive field for each view in the training set and for each feature P_i, centered at P_i (in the simple case of fixed centers), for a total of N two-dimensional gaussian receptive fields for each of the M centers. We have synthesized a *yes/no* detector for a specific object, irrespectively of its 3D position, by subtracting from the output of the GRBF network the selected standard view, then taking the euclidean norm of the resulting error vector and thresholding it appropriately. All of this assumes that the correspondence problem between views is solved, a quite difficult proposition!

Finally, we cannot resist the temptation of mentioning that the GRBF recognition scheme we have outlined has intriguing similarities with some of the data about visual neurons responding to faces obtained by Perrett and coworkers (see Perrett et al., 1987, Poggio and Edelman, 1989).

7.4 Gaussian GRBFs, Coarse Coding and Product Units

A popular approach in the neural network literature is *coarse coding*. This technique assigns a receptive field to each computing unit. If the dimensions of the receptive fields are properly chosen, a point in the input space generally belongs to a number of different receptive fields, and is then encoded in a distributed way. An example of coarse coding is the "scalarized" representation adopted by Saund (1987) to perform dimensionality reduction by means of a network with one hidden layer. In his work a scalar value is represented by the pattern of activity over a set of units. The pattern of activity is determined by sampling a gaussian-like function G centered on the scalar value itself. If $t_1, ..., t_n$ are the points at which the function G is sampled the "scalarized" representation of the value x is then the following:

$$x \Rightarrow \{G(x-t_1),\ G(x-t_2),\ ...,\ G(x-t_n)\}\ .$$

Once the input is represented in such a way, it is further processed by the hidden and the output layer, for instance of a BP network. If the input consists of more than one variable, each variable is "scalarized" separately.

Clearly, the previous sections show that a gaussian GRBF network (with fixed centers) is equivalent to coarse coding followed by product units (also called Sigma-Pi units, see Rumelhart et al. 1986; Mel, 1988). An example is shown in figure 6 for a two dimensional input. The first layer represents the coarse coding stage with two-dimensional gaussian receptive fields. The second layer consist of product units that synthesize the gaussian radial basis functions. The output of these units is then summed with the weights c_1, c_2 and c_3 to give the value of the GRBF expansion.

In this example the output of a product unit with inputs $(x_1, x_2, ..., x_n)$ was simply $y = \prod_{i=1}^n x_i$. This is a particular case of the Product Units introduced by Durbin and Rumelhart (1989). The output of a Product Unit is in general $y = \prod_{i=1}^n x_i^{p_i}$ where the p_i are exponents that have to be chosen. Substituting the units in the hidden layer with Products Units one obtains

$$\left(e^{-(x-t_x^i)^2}\right)^{p_1}\left(e^{-(y-t_y^i)^2}\right)^{p_2} = e^{-[p_1(x-t_x^i)^2+p_2(y-t_y^i)^2]}\ , i = 1, 2, 3\ . \tag{37}$$

This gives an expansion on a set of *non radial basis functions* that can also be derived from regularization (see section 8.2).

8 Some Future Extensions and Applications

This section sketches some of the extensions and applications of GRBFs that will be discussed in detail in a forthcoming paper.

8.1 The Bayes Approach to GRBFs

Earlier in the paper we have shown that the regularization principle is strictly related to the prior assumptions on the mapping to be learned and to the noise distribution in the examples (section 3.3). Because of the "equivalence" between GRBFs and regularization, it follows that the form of the radial basis function – its shape and its parameters, such as scale

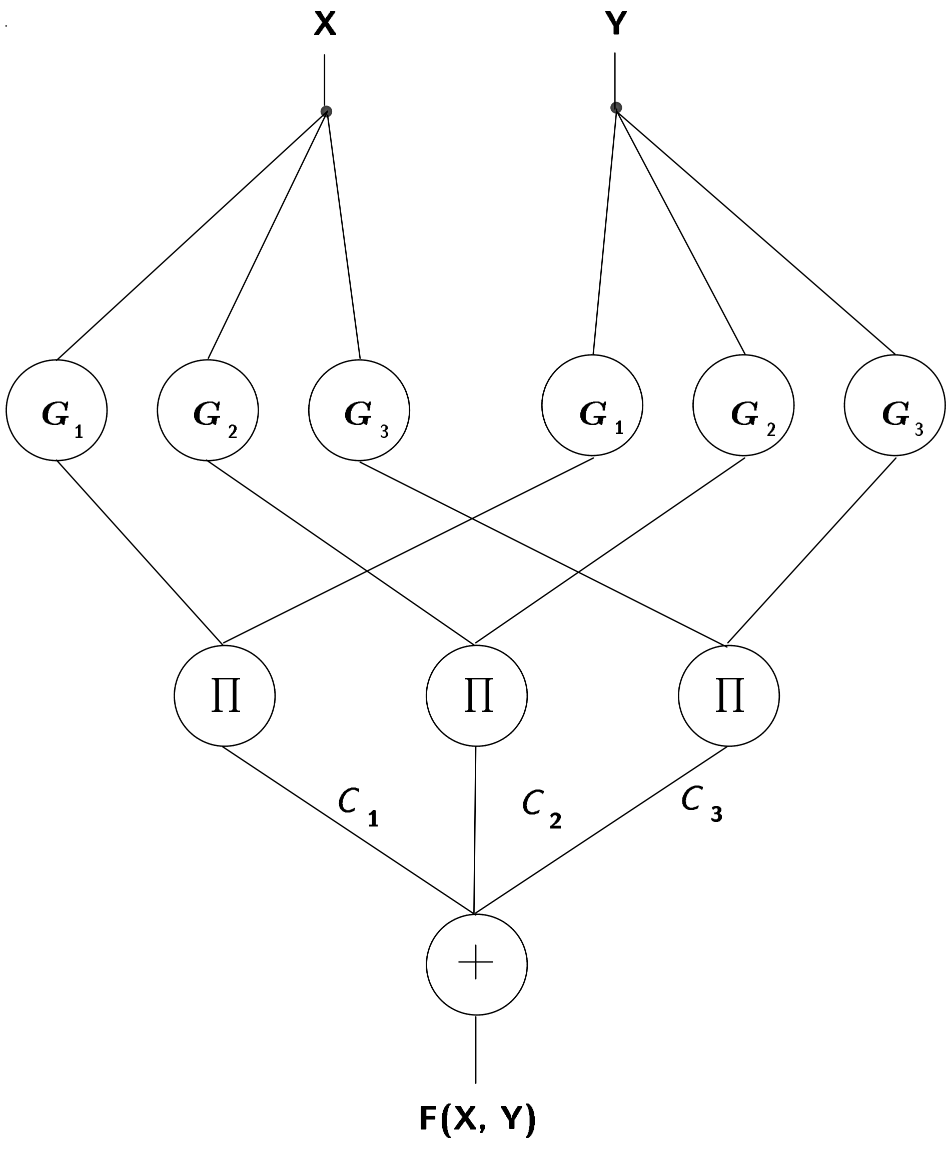

Figure 6: *A GRBF network with three knots, which can be used for the reconstruction of a two dimensional function. The first layer consists of Gaussian units, and implements one dimensional receptive fields. The second layer consists of product units and is used to recover the two-dimensional receptive field. The output is a weighted sum of the outputs of the product units.*

– depends on the *prior*, that is on the *a priori* knowledge of the smoothness properties of the mapping to be learned. The prior is in turn related to the complexity of the hypothesis. This point of view seems quite powerful both as an interpretation of GRBFs and as an approach for generalizing the GRBF scheme in several directions including, for instance, dimensionality regularization (Girosi and Poggio, 1989). Another important generalization has to do with the type of *a priori* knowledge that is assumed. The approach of this paper deals only with assumptions about smoothness of the mapping to be learned. Assumptions of this type are quite weak – in terms of required a priori knowledge – but powerful – in terms of their implications (see Stone, 1982). It is natural to ask whether it is possible to extend the approach of this paper to other types of *prior* assumptions.

8.2 Generalizing GRBFs to HyperBFs

The GRBF network that we have derived consists of radial function all of the same type. In the gaussian case all units have the same σ which is set by the stabilizer, that is by the *prior* assumptions about the function to be learned. It is natural to attempt to write a more general expansion than equation (19) in terms of, say, gaussians with σ depending on $\mathbf{t}_\alpha$, replacing σ by σ_α. Gradient descent could then be performed on σ_α, in addition to $\mathbf{t}_\alpha$ and c_α. Moody and Darken (1989) have reported successful testing of a method somewhat similar to radial basis functions in which the σ of the gaussian units is determined from the data and is different from unit to unit. Somewhat surprisingly, it seems very difficult to derive rigorously from regularization an expansion of this type. [4]

We found, however, a different solution to the multiscale problem that is consistent with regularization and, at the same time, generalizes the GRBF scheme beyond radial functions. We call the resulting expansions and the associated networks *Hyper Basis Functions*. In this section we sketch the main results.

The main idea is to consider the mapping to be approximated as the sum of several functions, each one with its own prior, that is stabilizer. The corresponding regularization principle then yields a superposition of different Green's functions, in particular gaussians with different σ. Thus, instead of doing gradient descent on σ_α the network has a "repertoire" of different σ at the same positions and chooses which ones to use through the corresponding coefficient. In more detail, the function f to be approximated is regarded as the sum of p components f^m, $m = 1, \ldots, p$, each component having a different prior probability. Therefore the functional $H[f]$ to minimize will contain p stabilizers P^m and will be written as

$$H[f] = \sum_{i=1}^{N}(\sum_{m=1}^{p} f^m(\mathbf{x}_i) - y_i)^2 + \sum_{m=1}^{p} \lambda_m \|P^m f^m\|^2 \quad . \tag{38}$$

From the structure of equation (38) it is clear that the exact solution will be a *linear superposition of linear superpositions* of the Green's functions G^m corresponding to the stabilizers P^m. The approach of section 5.2 – of using less radial functions than examples (with centers

[4]In principle, optimal $\sigma(x)$ could be found as the solution to a variational problem in σ that contains the standard regularization problem: find the σ, and then the stabilizer, such that the solution of the regularization problem is optimal. The problem is related to the optimization problems involving MRFs with line processes (Geiger and Girosi, 1989), and to cross-validation techniques (Craven and Wahba, 1979; Wahba, 1977).

generally different from the examples) – can be naturally extended to deal with this class of functionals (Girosi and Poggio, 1989a). The approximated solution to the minimization problem defined by equation (38) is written as (the *HyperBF expansion*)

$$F(\mathbf{x}) = \sum_{m=1}^{p} \sum_{\alpha=1}^{K} c_{\alpha}^{m} G^{m}(\mathbf{x}; \mathbf{t}_{\alpha}^{m}) \ . \tag{39}$$

In particular, this method leads to radial basis functions of multiple scales for the reconstruction of the function f. Suppose we know *a priori* that the function to be approximated has components on a number p of scales $\sigma_1, \ldots, \sigma_p$: we can use this information to choose a set of p stabilizers whose Green's functions are, for example, Gaussians of variance $\sigma_1, \ldots, \sigma_p$. According to example 1 of section (5.1.2) we have:

$$\|P^m f^m\|^2 = \sum_{k=0}^{\infty} a_k^m \int_{R^n} d\mathbf{x} (D^m f^m(\mathbf{x}))^2 \tag{40}$$

where $a_k^m = \frac{\sigma_m^{2k}}{k! 2^k}$. As a result, the solution will be a *superposition of superpositions* of gaussians of different variance. Of course, the gaussians with large σ should be preset, depending on the nature of the problem, to be fewer and therefore on a sparser grid, than the gaussians with a small σ.

The method yields also non-radial Green's functions – by using appropriate stabilizers – and also Green's functions with a lower dimensionality – by using the associated f^m and P^m in a suitable lower-dimensional subspace. Again this reflects *a priori* information that may be available about the nature of the mapping to be learned. In the latter case the information is that the mapping is of lower dimensionality or has lower dimensional components (see the problem of Saund, 1987).

Algorithms to perform gradient descent on the expansion equation (39) are the same that can be used for GRBFs (see for instance equations (26) and (27)). The additional "repulsive" term in the functional $H[f]$ to be minimized, introduced earlier, should now consist of the sum of pairwise interactions among centers of *the same type* only, for instance among gaussians with the same σ. In equation 39 two identical basis functions with the same center are clearly redundant, but two different basis functions in the same position, for instance two gaussians with a different scale, may actually make sense. This amounts to a form of an *exclusion principle* for basis units of the same type (the index m in equation (40), see Girosi and Poggio, 1989a).

Notice that an efficient heuristic scheme may want to *create* basis units in regions of the input space where the network is not performing well and *annihilate* basis units that are not activated much. This common sense heuristics fits perfectly within the formal framework: creation of a new unit means moving the center of an existing, remote one to an useful initial position; annihilation is equivalent to a 0 value for the associated coefficient c.

One class of more efficient algorithms than simple gradient descent are multigrid techniques. Multiresolution techniques have been used in many fields such as numerical analysis (Brandt, 1977) and computer vision (Rosenfeld and Kak, 1982). The idea is to find a solution at a coarse scale, then use this coarse solution as a starting point for higher resolution descriptions that can then be used to refine the coarse levels in an iterative procedure that proceeds from coarse to fine and back. Multigrid methods for solving partial differential equations work in this general way. The same basic idea could be used with multiple-scales HyperBF: a coarse description is first obtained by doing gradient descent only on the few gaussian functions with large σ. A finer description is then derived by approximating the difference between the data and the coarse approximation with more and smaller gaussian functions (the finer description may be used only in a subregion of the input space, somewhat similarly to a "fovea"). The whole process is then iterated again in a sequence of error correction steps, from coarse to fine and from fine to coarse. The process should be equivalent to doing gradient descent on the all set of basis functions at once, but it could be computationally more efficient and even more capable of avoiding local minima (depending on the problem). Ideas of a similar flavor have been recently suggested by Moody (1989).

8.3 Nonlinear input and output coding

We have mentioned already that the HyperBF scheme when used for classification tasks could include an invertible nonlinear function σ of the output without affecting the basic technique. Clearly a similar nonlinear function σ could be applied to each of the inputs: the HyperBF approximation would be then performed on the transformed inputs to yield the σ^{-1} transformation of the given output. It seems possible that in some cases suitable input and output processing of this type may be advantageous. Is there a general reason for it? Poggio (1982), following a forceful argument by Resnikoff (1975), has argued that the input and the output of the mapping to be approximated should be processed by a nonlinear function in order to match the domain and the range of the approximating function. Resnikoff had proposed as nonlinear functions for this processing the birational functions, the exponential function, the logarithmic function and the composition of this functions, since they achieve the necessary conversion of domain and range with minimal disruption of the algebraic structure of the input and output spaces. Input and output coding of this type tries to linearize the approximation as much as possible by exploiting *a priori* information about the range and the domain of the mapping to be approximated. Interestingly, the sigmoid function used at the output of many neural networks can be derived from the composition of a rational function and an exponential and matches the range of functions used for binary classification.

8.4 Learning Dynamical Systems

HyperBF can be used to "learn" a dynamical system from the time course of its output. In fact, RBF have been suggested often as a good technique for this problem and have been successfully tested in some cases (Broomhead and Lowe, 1988; Casdagli, 1989). The technique involves the approximation of the "iterated map" underlying the dynamical system (the crucial problem is, of course, the estimation of the dimension of the attractor and the choice of the input variables, see Farmer et al., (1983) for a good review). We have every reason to believe that HyperBF will perform on this problem at least as well as the linear techniques of Farmer and Sidorowich (1988) and the backpropagation algorithm of Lapedes and Farber (1988).

8.5 Learning the Dynamical System Underlying a Complex Mapping

Dynamical systems (in particular discrete time systems, that is difference equations) are Turing universal (the game "Life" is an example that has been demonstrated to be Turing universal). Thus a dynamical system such as the feedback system shown in figure 7, where $(x_0, x_1, \ldots, x_n)$ are the signal values at time $t_0, t_1, \ldots, t_n$, is equivalent to a finite state machine. Clearly the mapping $f(x_0; x_1 \ldots x_n)$ which is iterated by the feedback system can be approximated by HyperBFs. This offers the possibility of "learning" a computation more complex than a boolean function, if examples with sufficient information are provided.

In the following, consider the simpler case of learning a mapping F between an input vector x and an output vector $y = F(x)$ that belong to the same n-dimensional space. The input and the output can be thought as the asymptotic states of the discrete dynamical system obtained iterating some map f. In several cases, the dynamical system that asymptotically performs the mapping may have a much simpler structure than the direct mapping F. In other words, it is possible that the mapping f such that

$$\lim_{n\to\infty} f^{(n)}(x) = F(x) \tag{41}$$

is much simpler than the mapping $y = F(x)$ (here $f^{(n)}(x)$ means the n-th iterate of the map f).

A concrete example is the cooperative stereo algorithm (Marr and Poggio, 1976) that maps a 3D set of possible matches into the 3D set of true matches. The HyperBF technique can then be used to approximate f rather than F. Each update is done after iterating equation (41) with a "large enough" n, by using the obvious generalization of equation (28) and (29) (see Girosi and Poggio, 1989a).

8.6 Multilayer networks

In a somewhat similar spirit to the previous section, multilayer radial basis function networks (with more than one hidden layer) may be used for functions that happen to be well representable as the composition of functions that have a simple radial basis functions expansion. There is no reason to believe, however, that such "multilayer" functions represent a large and interesting class.

Especially with backpropagation networks, researchers have often argued for several layers of hidden units. From the point of view of HyperBF, one layer is needed (the basis functions themselves), but there is no obvious sense in additional layers. We believe that this is true in general for *single networks*.

On the other hand, there is a very good reason for parallel and hierarchical *systems* consisting, for instance, of several HyperBF modules connected together: the first network, for instance, may synthesize features that are used by one or more other modules for different, specific tasks.

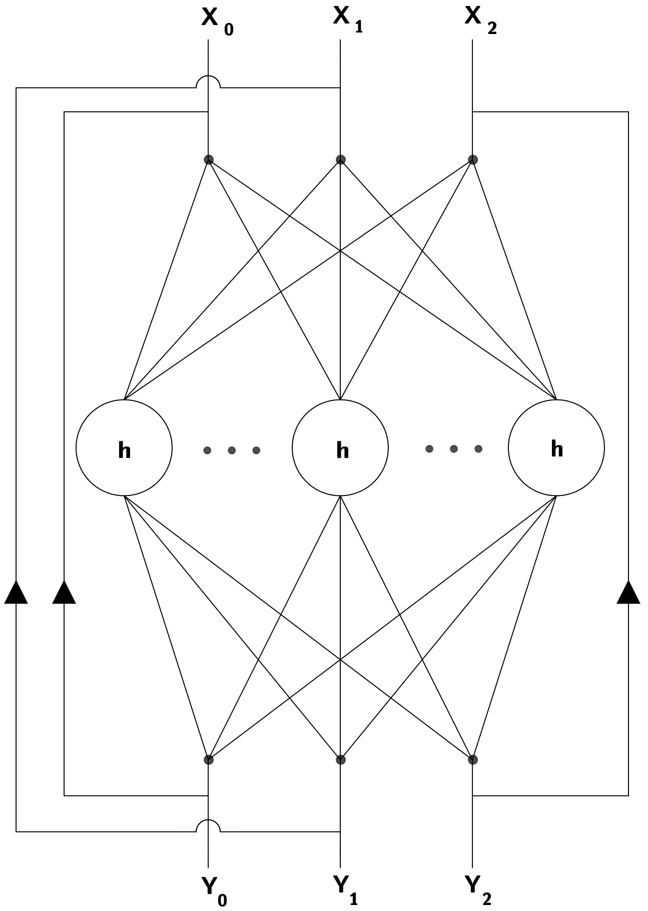

Figure 7: A HyperBF network used to approximate a mapping from (x_0, x_1, x_2) to (y_0, y_1, y_2) by approximating the dynamical system that maps asymptotically the input into the output. The input is given and the feedback network is iterated a sufficient number of times. Then the HyperBF update equations are used. The same procedure is used for new examples. The goal is to approximate the dynamical system that gives asymptotically the desired input-output relation.

8.7 Learning Perceptual and Motor tasks

HyperBFs have a good chance of being capable of synthesizing several vision algorithms from examples, since (a) it is known that several problems in vision have satisfactory solutions in terms of regularization and (b) HyperBF networks are essentially equivalent to regularization. We suggest that the use of HyperBFs is not restricted to sensory processes and that they may also be used to learn motor tasks (Mel, 1988 has demonstrated a scheme for learning a simple motor task which can be considered a special case of GRBFs) and even to model biological motor control. In support of this latter point, notice that simple biological trajectory control seems to be well explained by variational formulations of the regularization type (Flash and Hogan, 1985). HyperBF networks are equivalent to regularization *and* may have attractive neural interpretations: basis functions, possibly radial, may correspond to motor units with a multidimensional motor field, whereas their sum may be implicitly performed by the whole mechanical system, say a multijoint arm. HyperBFs may also be used for simple forms of trajectory planning. Potential methods, which associate to obstacles appropriate repulsive forces and obtain in this way an overall potential field that can be used in driving the trajectory, have been used with some success (for instance by Ahuja, 1989). These methods are the motor analog of the potential methods used for pattern recognition (see section 6.6). Clearly, HyperBFs seem naturally suited to learn these fields from sets of examples. In any case, computer experiments will have to be performed to assess the performance of the HyperBF approach.

8.8 Learning Input Features

The theoretical framework we have developed in this paper does not say anything about what is probably the most difficult problem in learning a mapping, that is the choice of the input representation. In the HyperBF scheme, with gaussian basis functions, gradient descent on the centers positions and on the coefficients c_α can be regarded as equivalent to learning which features to combine in prototypes by selecting the useful combinations. The problem of the elementary features, however, is outside the regularization framework, which is at the basis of HyperBF. Substantial extensions of our approach, for instance using the Bayes point of view, or different approaches, such as genetic-like algorithms, are clearly required.

9 Conclusions

9.1 How HyperBFs really work

HyperBF have a rather simple structure that seems to capture some of the main lessons that are becoming evident in the fields of statistics and neural networks. HyperBF can be regarded as a process involving three coupled stages:

- a stage which finds centers of clusters of training examples in the associated n-dimensional space
- the coding of the inputs by the K basis functions

- a stage that performs an optimal linear mapping from the K-dimensional space of basis functions into the desired output vector

The first stage can be regarded as similar to simple and efficient clustering algorithms. The third stage is classical: optimal linear mappings by themselves work well in many situations. Our approach to HyperBF shows that this sequence of stages is not just attractive heuristics but derives rigorously from regularization and is thereby solidly grounded in approximation theory. The theory says that the basis functions provide a sufficient nonlinear coding of the input for linear approximation to work (see Poggio, 1982). In addition, it says how to couple these various stages in a single procedure.

To have a feeling of how HyperBFs work let us consider a specific, extreme case, in which we consider a HyperBFs network as a classifier, something the formal theory does not actually allow. Imagine using a HyperBF scheme to classify patterns, such as handwritten digits, in different classes. Assume that the input is a binary a 8-bit vector of length N and each of the basis functions is initially centered on the point in the N-dimensional input space that corresponds to one of the training examples (fixed centers case). The system has several outputs, each corresponding to one of the digit classes. Let us consider a series of special cases of HyperBF of increasing generality:

1. Each of the unit (its center corresponds to an example) is an hypersphere (see equation (34)) and is connected, with weight 1, to its output class only. Classification is done by reading out the class with maximum output. In this case, the system is performing a Parzen window estimate of the posterior probability and then using a MAP criterion. The Parzen-window approach is similar (and asymptotically equivalent) to the k_n nearest-neighbor estimation, of which the nearest-neighbor rule is a special case. In this special case the network is equivalent to a hypersphere classifier.

2. We now replace the hypersphere by a multidimensional gaussian that is an allowed radial basis function (the hypersphere does not satisfy Micchelli's condition and cannot be derived from regularization). At least for the task of approximating smooth functions the network should perform better than in the non-gaussian case. The centers of the radial basis functions may be regarded as representing "templates" against which the input vectors are matched (think, for instance of a radial gaussian with small σ, centered on its center, which is a point in the n-dimensional space of inputs).

3. We may do even better by allowing arbitrary c values between the radial units and the output. The c can then be found by the pseudoinverse technique (or gradient descent) and are guaranteed to be optimal in the l_2 sense.

4. We now allow a number of (movable) centers with radial basis functions. This is the GRBF scheme. Moving a center is equivalent to modifying the corresponding template. Thus equation (29) attempts to develop better templates by modifying during training the existing ones. In our example, this means changing the pixel values in the arrays representing the digits.

5. Finally the most general network contains radial units of the gaussian type of different scale (i.e. σ), together with non-radial units associated to appropriate stabilizers and units that receive only a subset of the inputs.

This list shows that the HyperBF scheme is an extension of some of the simplest and most efficient approximation and learning algorithms which can be regarded as special cases of it. In addition, it illuminates a few interesting aspects of the HyperBF algorithm, such as its massive parallelism and its use of prototypes. The network is massively parallel in the sense that it may in general require a large number of basis units. While this property could have been regarded quite negatively a few years ago, this is not so anymore. The advent of parallel machines such as the Connection Machine with about 65,000 processors and of special purpose parallel hardware has changed the perspective towards massive parallelism. The use of prototypes by HyperBFs suggest that, in a sense, HyperBFs networks are an extension of massive template matchers or look-up tables. We believe that this property makes them intriguingly attractive: after all, if memory is cheap, look-up tables are a good starting point. The HyperBF scheme says how to extend look-up tables to do as well as possible in terms of approximating a multidimensional function about which very little is known.

9.2 Networks and Learning: The Pervasive Problem of Dimensionality

Our main result shows that the class of feedforward multilayer networks identical to HyperBF are equivalent to generalized splines, and are capable of well approximating a smooth function. This is highly satisfactory from a theoretical point of view, but in practice another fundamental question must also be addressed: *how many samples are needed to achieve a given degree of accuracy* (Stone, 1982; Barron and Barron, 1988)? It is well known that the answer depends on the dimensionality d and on the degree of smoothness p of the class of functions that has to be approximated (Lorentz, 1966, 1986; Stone, 1982, 1985). This problem has been extensively studied and some fundamental results have been obtained by Stone (1982). He considered a class of nonparametric estimation problems, like surface approximation, and computed the optimal rate of convergence ϵ_n, that is a measure of how accurately a function can be approximated knowing n samples of its graph. He showed that using a local polynomial regression the optimal rate of convergence $\epsilon_n = n^{-\frac{p}{2p+d}}$ can be achieved, generalizing previous results based on local averages. This means that the number of examples needed to approximate a function reasonably well grows enormously with the dimension d of the space on which it is defined, although this effect is mitigated by a high degree of smoothness p (in fact ϵ_n depends only on the ratio $\frac{d}{p}$). For instance in the case of a twice differentiable function of two variables, 8000 examples are needed to obtain $\epsilon_n = 0.05$, but if the function depends on 10 variables the number of examples necessary to obtain the same rate of convergence grows up to 10^9. However, if a function of 10 variables is 10 times differentiable 8000 examples will be enough to obtain $\epsilon_n = 0.05$.

Interestingly these results lead to an *a posteriori* justification of the smoothness assumption that plays a central role in standard regularization. In fact, when the number of dimensions becomes larger than 3 or 4, one is forced to assume a high degree of smoothness: otherwise the number of examples required will be so large that the approximation task becomes hopeless. This justifies the use of high-order stabilizers such as the gaussian (the stabilizer is equivalent to the prior, see earlier) for high dimensional spaces, whenever no other type of prior information about the function to be approximated is available. Notice that in the case of an infinite degree of smoothness the optimal rate of convergence tends to an asymptotic value. In fact it is straightforward to check that $\lim_{p\to\infty} \epsilon_n = n^{-\frac{1}{2}}$. In this case it is possible to obtain $\epsilon_n = 0.05$ with just 400 examples. It must be noticed however that the set of functions with infinite degree of smoothness is a very small subset of the space of the continuous functions.

The results stated above are optimal, but it is not guaranteed that they are achievable with HyperBF. The similarity of HyperBF with splines is encouraging, however, since splines has been proved to be optimal in this sense (Cox, 1984). Our results also suggest that most of the networks proposed in the recent literature, since they are similar to the HyperBF nets, are likely to have the same fundamental limitations, perhaps even more strongly, from the point of view of sample complexity.

Other interesting results have been obtained by Baum and Haussler on the statistical reliability of networks for binary classification (Baum, 1988; Baum and Haussler, 1989). They use the concept of Vapnik-Chervonenkis dimension (Vapnik and Chervonenkis, 1971) in the network context to give the probability of error on new data given the error on the training data. This approach is different from the one pursued in approximation and regression theory, since they do not estimate a priori the accuracy of the network. These results do not directly apply to our case, but, since the concept of Vapnik-Chervonenkis dimension has been shown to be very powerful, we think it will also be relevant in the context of the HyperBF method.

Of course this analysis requires that the true dimension of the data set is known, which is not always the case. Especially when the number of dimensions is large, not all of them are relevant: the problem of "dimensionality reduction" is how to find the relevant dimensions. A solution to the problem consists in considering the dimension as another random variable that has to be estimated from data (see Girosi and Poggio, 1989). A priori knowledge about the number of dimensions can be embedded in the MAP estimator, to make, for instance, low dimensionality solutions more likely than others.

Another approach to dimensionality reduction has been pursued by J. Schwartz (1988), and has been shown to be not very different from ours (Girosi and Poggio, 1989). He solves the learning problem for many data sets, obtained from the original one dropping some dimensions, and then selects the one that gives the best result. This method is more similar to Generalized Cross Validation (Wahba, 1977; Craven and Wahba, 1979) and even without a priori information on the dimensionality of the problem, turned out to be effective in computer simulations (Schwartz, 1988).

9.3 Summary

Approaching the problem of learning in networks from the point of view of approximation theory provides several useful insights. It illuminates what network architectures are doing; it suggests more principled ways of obtaining the same results and ways of extending further the approach; and finally, it suggests fundamental limitations of all approximation methods, including so called neural networks.

In this paper, we developed a theoretical framework based on regularization techniques that led to a class of three-layer networks, useful for approximation, that we call Generalized Radial Basis Functions (GRBF), since they are closely related to the well-known Radial

Basis Functions, mainly used for strict interpolation tasks. We have introduced several new extensions of the method and its connections with splines, regularization, Bayes formulation and clustering. GRBFs have a direct interpretation in terms of a feedforward, multilayer network architecture. Unlike neural networks they have good theoretical foundations. They may provide the best framework within which we can study general issues for learning techniques of the neural network type.

Acknowledgments We are grateful to C. Atkeson, S. Edelman, E. Hildreth, D. Hillis, A. Hurlbert, E. Grimson, B. Kahle, A. Singer, L. Tucker, D. Weinshall for useful discussions and suggestions.

A Kolmogorov's and Vitushkin's Theorems

In this section we discuss the problem of the exact representation of continuous multivariate functions by superposition of univariate ones. Some results on this topic will be reported as well their interpretation in the framework of the multilayer networks.

This problem is the thirteenth of the 23 problems that Hilbert formulated in his famous lecture at the International Conference of Mathematicians in Paris, 1900. Although his original formulation dealt with properties of the solution of a seventh degree algebraic equation this problem can be restated as follows:

prove that there are continuous functions of three variables, not representable as superpositions of continuous functions of two variables

The definition of superposition of functions can be found in Vitushkin (1954), but is quite cumbersome and to our aims is sufficient to define it as the usual composition. For example the function xy is the superposition of the functions $g(\cdot) = exp(\cdot)$ and $h(\cdot) = log(\cdot)$, since we can write

$$xy = e^{(logx+logy)} = g(h(x) + h(y)).$$

In 1957 Kolmogorov and Arnol'd showed the conjecture of Hilbert to be false: by Kolmogorov's theorem any continuous function of several variables can represented by means of a superposition of continuous functions of a single variable and the operation of addition. The original statement of Kolmogorov (1957) is the following:

Theorem A.1 *There exist fixed increasing continuous functions $h_{pq}(x)$, on $I = [0, 1]$ so that each continuous function f on I^n can be written in the form*

$$f(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g_q(\sum_{p=1}^{n} h_{pq}(x_p)),$$

where g_q are properly chosen continuous functions of one variable.

Since the original formulation of Kolmogorov many authors have improved the representation of theorem A.1. The main results are concerned with the possibility of replacing the functions g_q by a single function g (Lorentz, 1962) and of writing h_{pq} as $l_p h_q$ (Sprecher, 1964). Here we report a formulation due to Kahane (1975) whose proof does not need the construction of the functions h_q, relying instead on the Baire's theorem. We first give some definitions. We will say that a statement is true for quasi every element of a complete metric space M if it is true on the intersection of a countable family of open sets which are everywhere dense in M. Let H be the space with uniform norm consisting of all functions continuous and non decreasing on the segment I and $H^k = H \times ... \times H$ the k-th power of the space H.The following theorem then holds:

Theorem A.2 *Let $l_p (p = 1, ..., n)$ be a collection of rationally independent constants. Then for quasi every collection $\{h_1, ..., h_{2n+1}\} \in H^{2n+1}$ it is true that any function $f \in C(I^n)$ can be represented on I^n in the form*

$$f(x_1, ..., x_n) = \sum_{q=1}^{2n+1} g(\sum_{p=1}^{n} l_p h_q(x_p)),$$

where g is a continuous function.

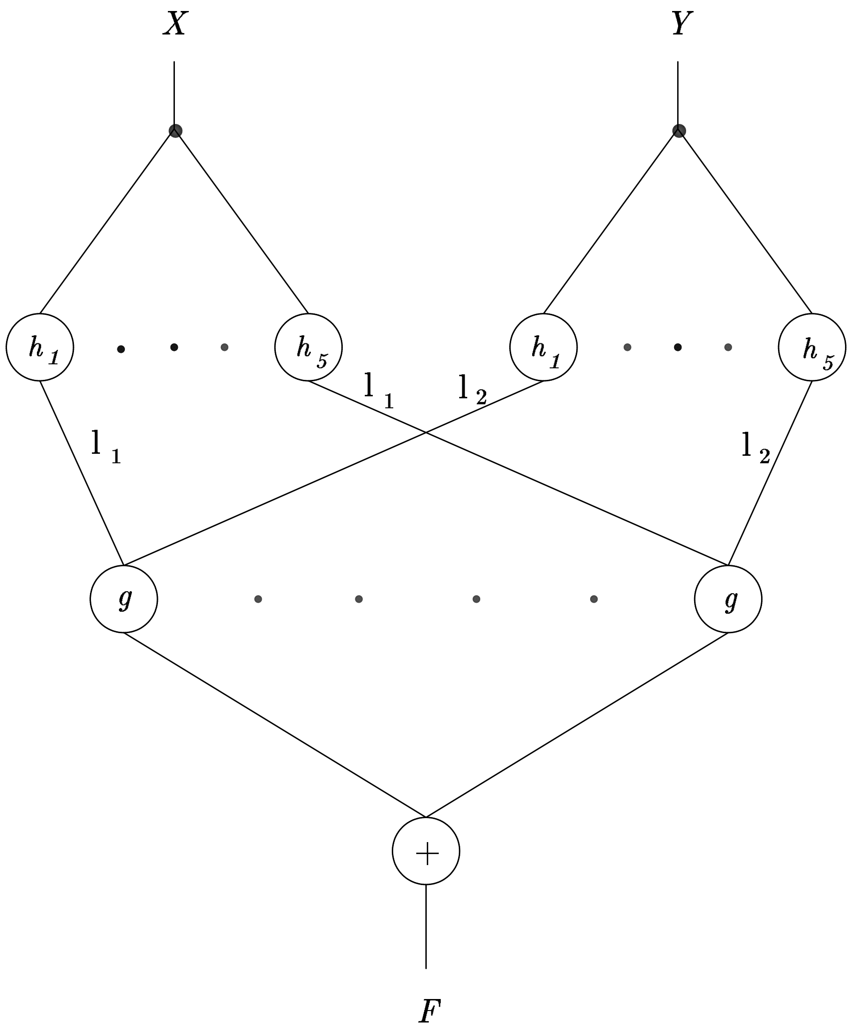

Figure 8: *The network implementation of Kahane's version of Kolmogorov's theorem*

The representation of this formula has been graphically depicted in figure 8: it is evident the connection between this superposition scheme and a multilayer network architecture. In this framework this result could seem very appealing : the representation of a function requires a fixed number of nodes, smoothly increasing with the dimension of the input space. Unfortunately further studies on this subject showed that these results are somewhat pathological and their practical implications very limited. The problem lies in the inner functions of Kolmogorov's formula: although they are continuous, some results of Henkin and Vitushkin (1967) prove that they must be highly non-smooth. Since the number of binary digits required to code a function with a given accuracy is inversally proportional to its degree of smoothness, as a result a very poor accuracy will be obtained implementing Kolmogorov's formula with finite precision. One could ask if it is possible to find a superposition scheme in which the functions involved are smooth. The answer is negative, even for two variable functions, and was given by Vitushkin with the following theorem (1954):

Theorem A.3 *There are r ($r = 1, 2, ...$) times continuously differentiable functions of $n \geq 2$ variables, not representable by superposition of r times continuously differentiable functions of less than n variables; there are r times continuously differentiable functions of two variables which are not representable by sums and continuously differentiable functions of one variable.*

We notice that the intuition underlying Hilbert's conjecture and theorem A.3 is the same: not all the functions with a given degree of complexity can be represented in simple way by means of functions with a lower degree of complexity. The reason for the failing of Hilbert's conjecture is a "wrong" definition of complexity: Kolmogorov's theorem shows that the number of variables is not sufficient to characterize the complexity of a function. Vitushkin showed that such a characterization is possible and gave an explicit formula. Let f be a r times continuously differentiable function defined on I^n with all its partial derivatives of order r belonging to the class $Lip[0,1]^\alpha$. Vitushkin puts

$$\chi = \frac{r + \alpha}{n}$$

and shows that it can be used to measure the inverse of the complexity of a class of functions. In fact he succeded in proving the following:

Theorem A.4 *Not all functions of a given characteristic $\chi_0 = \frac{q_0}{k_0} > 0$ can be represented by superpositions of functions of characteristic $\chi = \frac{q}{k} > \chi_0, q \geq 1$.*

Theorem A.3 is easily derived from this result.

B Linear Approximation (1-Layer Networks)

Let us assume that the mapping between a set of input vectors y and a set of output vectors z is linear (for an example, see Hurlbert and Poggio, 1988). How can we estimate the linear mapping from a set of examples? We start by arranging the sets of vectors in two matrices Y and Z. The problem of synthesizing the linear operator L is then equivalent to "solving" the following equation for L:

$$Z = LY \tag{42}$$

A general solution to this problem is given by

$$L = ZY^+, \tag{43}$$

where Y^+ is the pseudoinverse of Y. This is the solution which is most robust against errors, if equation (42) admits several solutions and it is the optimal solution in the least-squares sense, if no exact solution of equation (42) exists. This latter case is the one of interest to us: in order to overconstrain the problem, and so avoid look-up table solutions, we require that the number of examples (columns of Y) be larger than the rank of the matrix L. In this case, there is no exact solution of equation (42) and the matrix L is chosen instead to minimize the expression

$$M = \|LY - Z\|^2. \tag{44}$$

L may be computed directly by minimizing (44), which yields

$$L = ZY^T(YY^T)^{-1} \tag{45}$$

In practice, we compute L using equation (45), but first regularize it by adding a stabilizing functional to obviate problems of numerical stability (Tikhonov and Arsenin, 1977).

B.1 Recursive Estimation of L

It is of particular importance for practical applications that the pseudoinverse can be computed in an adaptive way by updating it when new data become available (Albert, 1972). Consider again equation (45). Assume that the matrix Y consists of $n-1$ input vectors and Z of the corresponding *correct* outputs. We rewrite equation (45) as

$$L_{n-1} = Z_{n-1}Y^+_{n-1} \tag{46}$$

If another input-output pair y_n and z_n becomes available, we can compute L_n recursively by

$$L_n = L_{n-1} + (z_n - L_{n-1}y_n)t_n^T, \tag{47}$$

where

$$t_n^T = \frac{y_n^T(Y_{n-1}Y_{n-1}^T)^{-1}}{1 + y_n^T(Y_{n-1}Y_{n-1}^T)^{-1}y_n}, \tag{48}$$

provided that $(Y_{n-1}Y_{n-1}^T)^{-1}$ exists (i.e., that the number of columns in Y is greater than or equal to the dimension of y). The case in which $(Y_{n-1}Y_{n-1}^T)^{-1}$ does not exist is discussed together with more general results in Albert (1972). Note that $(z_n - L_{n-1}y_n)$ in the updating equation (47) is the error between the desired output and the predicted one, in terms of the current L. The coefficient t_n is the weight of the correction: with the value given by equation (48) the correction is optimal and cannot be improved by any iteration without new data. A different value of the coefficient is suboptimal but may be used to converge to the optimal solution by successive iterations of equation (48) using the same data.

B.2 Optimal Linear Estimation, Regression and Bayesian Estimation

The optimal linear estimation scheme we have described is closely related to a special case of Bayesian estimation in which the best linear unbiased estimator (BLUE) is found. Consider equation (42): the problem is to construct an operator L that provides the best estimation Ly of z. We assume that the vectors y and z are sample sequences of gaussian stochastic processes with, for simplicity, zero mean. Under these conditions the processes are fully specified by their correlation functions

$$E[yy^T] = C_{yy}, \quad E[zy^T] = C_{zy} \tag{49}$$

where E indicates the expected value. The BLUE of z (see Albert, 1972) is, given y,

$$z^{est} = C_{zy}C_{yy}^{-1}y, \tag{50}$$

which is to be compared with the regression equation

$$Ly = ZY^T(YY^T)^{-1}y. \tag{51}$$

The quantities ZY^T and YY^T are approximations to C_{zy} and C_{yy}, respectively, since the quantities are estimated over a finite number of observations (the training examples). Thus there is a direct relation between BLUEs and optimal linear estimation. The learned operator captures the stochastic regularities of the input and output signals. Note that if the input vectors y are orthonormal, then $L = ZY^T$ and the problem reduces to constructing a simple correlation memory of the holographic type (see Poggio, 1975). Under no restrictions on the vectors y, the correlation matrix ZY^T may still be considered as a low-order approximation to the optimal operator (see Kohonen, 1978).

C Polynomial Approximation

A natural extension of linear estimation is polynomial estimation. The Weierstrass-Stone theorem suggests that polynomial mappings, of which linear mappings are a special case, can approximate arbitrarily well all continuous real functions. There are function space equivalents of Weierstrass' theorem. In our case, defining $X \cdot X \cdot \because X$ (n times) as the set of linear combinations of the monomials of degree n in the components $(X_1, \ldots, X_k)$ of X, we simply look for the multivariate polynomial given by $L(X) = L_0 + L_1X + L_2X \cdot X + L_3X \cdot X \cdot X + \ldots$ that minimizes $\|y - L(X)\|$. The problem is equivalent to finding the optimal linear estimator on the "expanded" space consisting of all the products of the components of X. The monomials such as X_1, X_1X_2, or $X_1^2X_2^3X_3\cdots$ become the new features that are input to an optimal linear estimator. The resulting polynomial estimator can approximate any continuous mapping. In the case of optimal polynomial estimation, the crucial problem is computational complexity, which translates into the computation of the pseudoinverse of very large matrices y_t (see equation (43)). As a (small) step toward simplifying this problem, one can think of computing the optimal L_0, then the optimal L_1 connection, and so on in a sequence, up to L_n. Poggio (1975) proposes that the optimal polynomial estimator of order n can be found by an iterative procedure that first find the optimal linear estimator, then the optimal second-order correction up to the optimal n-order correction and then back again to the linear correction and so on. A theorem (Poggio, 1975) guarantees convergence of the procedure to the optimal n-th order estimator.

Recently, it has been demonstrated that a multilayer network can represent exactly any polynomial mapping, given polynomial output functions of at least order two for each unit and enough hidden units and layers (Moore and Poggio, 1988).

D The Relation between Regularization and Bayes Estimation

The problem of hypersurface reconstruction – and therefore of learning – can be formulated as a Bayesian estimation problem. The goal is to estimate the *a posteriori* probability $P_{z/d}$ of the solution z given the data d and use the estimate according to a chosen performance criterion . Bayes theorem yields

$$P(z/d) \propto P(z)P(d/z)$$

where $P(z)$ is the *prior* probability density of the process associated with z and $P(d/z)$ is the conditional probability density of the data d given the hypersurface z.

We consider now the special case of z (or equivalently the result of the action of any differential operator P on it) being a gaussian process. In this case, the *a priori* probability distribution of z is

$$P(z) \propto e^{-\frac{1}{2}(z,\hat{P}Pz)}$$

.

where $(\cdot,\cdot)$ is a scalar product in the space to whom z belongs and $\hat{P}$ is the adjoint of the operator P. Let us assume that the noise process affecting the data d taken from z is additive, white and gaussian with variance σ^2. Then the conditional probability $P(d/z)$ can be written as

$$P(d/z) \propto e^{-\frac{1}{2\sigma^2}\|z-d\|^2}.$$

where $\|\cdot\|$ is the norm induced by the scalar product $(\cdot,\cdot)$. Depending on the optimality criterion there are now several ways of obtaining the best estimate of z given the data d. A commonly used estimate is the *Maximum A Posteriori* (MAP) estimate

$$P(z_{\text{best}}/d) = \max\{P(z/d)|z \in Z\}$$

In our case the following holds

$$P(z_{\text{best}}/d) = \max e^{-\frac{1}{2\sigma^2}\|z-d\|^2 - \frac{1}{2}z\hat{P}Pz}.$$

Since by definition $(z, \hat{P}Pz) = \|Pz\|^2$, this is equivalent to finding the solution z that minimizes the functional

$$\|z - d\|^2 + \lambda\|Pz\|^2$$

that, in the case of sparse data, becomes the functional of equation (1).

This sketch of the relation between the Bayesian approach and regularization shows that the first term in equation 1, $\sum_i \|z_i - d_i\|^2$, is $-logP_{d/z}$, in other words it represents the known model of the measurement process or, equivalently, the model of the noise. The second term, $\|Pz\|^2$, is $-logP_z$ and therefore is dictated by the *prior*, that is the *a priori* knowledge about the solution, such as its smoothness properties.

References

[1] H. Abelson. Towards a theory of local and global in computation. *Theoretical Computer Science*, 6:41–67, 1978.

[2] F.P. Agterberg. *Geomathematics.* Elsevier, Amsterdam, 1974.

[3] A. Albert. *Regression and the Moore-Penrose Pseudoinverse.* Academic Press, New York, 1972.

[4] J.S. Albus. A theory of cerebellar functions. *Math. Bio.*, 10:25–61, 1971.

[5] J.Y. Aloimonos. Unification and integration of visual modules: an extension of the Marr paradigm. In *Proceedings Image Understanding Workshop*, pages 507–551, Palo Alto, CA, May 1989. Morgan Kaufmann, San Mateo, CA.

[6] W. Arai. Mapping abilities of three-layer networks. In *Proceedings of the International Joint Conference on Neural Networks*, pages I–419–I–423, Washington D.C., June 1989. IEEE TAB Neural Network Committee.

[7] V.I. Arnol'd. On functions of three variables. *Dokl. Akad. Nauk SSSR*, 114:679–681, 1957.

[8] A. R. Barron and Barron R. L. Statistical learning networks: a unifying view. In *Symposium on the Interface: Statistics and Computing Science*, Reston, Virginia, April 1988.

[9] R. Basri and S. Ullman. Recognition by linear combinations of models. Technical Report CS89-11, Weizman Institute of Science, 1989.

[10] E. B. Baum. On the capabilities of multilayer perceptrons. *J. Complexity*, 4:193–215, 1988.

[11] E. B. Baum and D. Haussler. What size net gives valid generalization? In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems I*, pages 81–90. Morgan Kaufmann Publishers, Carnegie Mellon University, 1989.

[12] E.B. Baum, J. Moody, and F. Wilczek. Internal representations for associative memory. *Biological Cybernetics*, 59:217–228, 1988.

[13] A. Ben-Israel and T.N.E. Greville. *Generalized inverses: theory and applications.* Wiley, New York, 1974.

[14] M. Bertero. Regularization methods for linear inverse problems. In C. G. Talenti, editor, *Inverse Problems.* Springer-Verlag, Berlin, 1986.

[15] M. Bertero, T. Poggio, and V. Torre. Ill-posed problems in early vision. *Proceedings of the IEEE*, 76:869–889, 1988.

[16] S. Bochner. Vorlesungen ueber Fouriersche Integrale. In *Akademische Verlagsgesellschaft*, Leipzig, 1932.

[17] S. Bochner. *Lectures on Fourier integral.* Princeton Univ. Press, Princeton, New Jersey, 1959.

[18] D. Braess. *Nonlinear Approximation Theory.* Springer-Verlag, Berlin, 1986.

[19] A. Brandt. Multilevel adaptive solutions to boundary-value problems. *Math. Comp.*, 31:333–390, 1977.

[20] D.S. Broomhead and D. Lowe. Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2:321–355, 1988.

[21] S.M. Carrol and B.W. Dickinson. Construction of neural nets using the Radon transform. In *Proceedings of the International Joint Conference on Neural Networks*, pages I–607–I–611, Washington D.C., June 1989. IEEE TAB Neural Network Committee.

[22] M. Casdagli. Nonlinear prediction of chaotic time-series. *Physica D*, 35:335–356, 1989.

[23] R. Courant and D. Hilbert. *Methods of mathematical physics. Vol. 2.* Interscience, London, England, 1962.

[24] D.D. Cox. Multivariate smoothing spline functions. *SIAM J. Numer. Anal.*, 21:789–813, 1984.

[25] P. Craven and G. Wahba. Smoothing noisy data with spline functions: estimating the correct degree of smoothing by the method of generalized cross validation. *Numer. Math*, 31:377–403, 1979.

[26] G. Cybenko. Continuous valued neural networks with two hidden layers are sufficient. Technical report, Dept. of Computer Sciences, Tufts Univ., Medford, MA, 1988.

[27] G. Cybenko. Approximation by superposition of a sigmoidal function. *Math. Control Systems Signals*, 2(4):303–314, 1989.

[28] J. Demmel. The geometry of ill-conditioning. *J. Complexity*, 3:201–229, 1987.

[29] R. A. DeVore and V. A. Popov. Free multivariate splines. *Constructive Approximation*, 3:239–248, 1987.

[30] J. Duchon. Interpolation des fonctions de deux variables suivant le principle de la flexion des plaques minces. *R.A.I.R.O. Anal. Numer.*, 10:5–12, 1976.

[31] J. Duchon. Spline minimizing rotation-invariant semi-norms in Sobolev spaces. In W. Schempp and K. Zeller, editors, *Constructive theory of functions os several variables, Lecture Notes in Mathematics, 571.* Springer-Verlag, Berlin, 1977.

[32] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* Wiley, New York, 1973.

[33] R. Durbin and D.E. Rumelhart. Product units: a computationally powerful and biologically plausible extension to backpropagation networks. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems I.* Morgan Kaufmann Publishers, Carnegie Mellon University, 1989.

[34] J.D. Farmer, E. Ott, and J.A. Yorke. The dimension of chaotic attractors. *Physica D*, 7:153–180, 1983.

[35] J.D. Farmer and J.J. Sidorowich. Exploiting chaos to predict the future and reduce noise. Technical report, Los Alamos National Laboratory, New Mexico, 1988.

[36] T. Flash and N. Hogan. The coordination of arm movements: an experiment confirmed mathematical model. *The Journal of Neuroscience*, 5(7):1688–1703, 1985.

[37] R. Franke. Scattered data interpolation: tests of some method. *Math. Comp.*, 38(5):181–200, 1982.

[38] K. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2:183–192, 1989.

[39] D. Geiger and F. Girosi. Parallel and deterministic algorithms for MRFs: surface reconstruction and integration. A.I. Memo No. 1114, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

[40] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6:721–741, 1984.

[41] A. Gersho. On the structure of vector quantizers. *IEEE Transactions on Information Theory*, 28(2):157–166, 1982.

[42] F. Girosi and T. Poggio. Networks for learning: a view from the theory of approximation of functions. In P. Antognetti and V. Milutinovicc̀, editors, *Neural Networks: Concepts, Applications, and Implementations, Vol. I*, chapter 6, pages 110–155. Prentice Hall, Englewood Cliffs, New Jersey, 1991.

[43] W. E. L. Grimson. *From Images to Surfaces.* MIT Press, Cambridge, MA, 1981.

[44] R.L. Harder and R.M. Desmarais. Interpolation using surface splines. *J. Aircraft*, 9:189–191, 1972.

[45] R.L. Hardy. Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res*, 76:1905–1915, 1971.

[46] J. G. Harris. An analog VLSI chip for thin-plate surface interpolation. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems I*. Morgan Kaufmann Publishers, Carnegie Mellon University, 1989.

[47] R. Hecht-Nielsen. Kolmogorov's mapping neural network existence theorem. In *Proc. 1987 IEEE International Conference on Neural Networks*, pages III–593–III–605, New-York, 1987. IEEE Press.

[48] D. Hilbert. Mathematische probleme. *Nachr. Akad. Wiss. Göttingen*, pages 290–329, 1900.

[49] A. Hurlbert and T. Poggio. Synthetizing a color algorithm from examples. *Science*, 239:482–485, 1988.

[50] I. R. Jackson. Convergence properties of radial basis functions. *Constructive approximation*, 4:243–264, 1988.

[51] J.P. Kahane. Sur le theoreme de superposition de Kolmogorov. *Journal of Approximation Theory*, 13:229–234, 1975.

[52] P. Kanerva. *Sparse distributed memory.* M.I.T. Press, Cambridge, MA, 1988.

[53] J.D. Keeler. Comparison between Kanervàs SDM and Hopfield-type neural networks. *Cognitive Science*, 12:299–329, 1988.

[54] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:219–227, 1983.

[55] R.W. Klopfenstein and R. Sverdlove. Approximation by uniformly spaced gaussian functions. In E.W. Cheney, editor, *Approximation Theory IV*, pages 575–580. Academic Press, New York, 1983.

[56] C. Koch and T. Poggio. Biophysics of computational systems: Neurons, synapses, and membranes. In G.M. Edelman, W.E. Gall, and W.M. Cowan, editors, *Synaptic Function*, pages 637–697. John Wiley and Sons, New York, 1987.

[57] T. Kohonen. *Associative Memory: A System Theoretic Approach.* Springer-Verlag, Berlin, 1978.

[58] A. N. Kolmogorov. On the representation of continuous functions of several variables by superposition of continuous functions of one variable and addition. *Dokl. Akad. Nauk SSSR*, 114:953–956, 1957.

[59] A. Lapedes and R. Farber. Nonlinear signal processing using neural networks: prediction and system modelling. Los Alamos National Laboratory LA-UR-87-2662, 1987. Submitted to Proc. IEEE.

[60] H.C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projecions. *Nature*, 293:133–135, 1981.

[61] G. G. Lorentz. Metric entropy, widths, and superposition of functions. *Amer. Math. Monthly*, 69:469–485, 1962.

[62] G. G. Lorentz. Metric entropy and approximation. *Bull. Amer. Math. Soc*, 72:903–937, 1966.

[63] G. G. Lorentz. On the 13-th problem of Hilbert. In *Proceedings of Symposia in Pure Mathematics*, pages 419–429, Providence, RI, 1976. Americam Mathematical Society.

[64] G. G. Lorentz. *Approximation of Functions*. Chelsea Publishing Co., New York, 1986.

[65] S.-K. Ma. *Modern Theory of Critical Phenomena*. W.A. Benjamin Inc., New York, 1976.

[66] J. MacQueen. Some methods of classification and analysis of multivariate observations. In L.M. LeCam and J. Neyman, editors, *Proc. 5th Berkeley Symposium on Math., Stat., and Prob.*, page 281. U. California Press, Berkeley, CA, 1967.

[67] D. Marr. A theory of cerebellar cortex. *J. Physiology*, 202:437–470, 1969.

[68] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 194:283–287, 1976.

[69] J. L. Marroquin, S. Mitter, and T. Poggio. Probabilistic solution of ill-posed problems in computational vision. *J. Amer. Stat. Assoc.*, 82:76–89, 1987.

[70] J. Meinguet. An intrinsic approach to multivariate spline interpolation at arbitrary points. In B. H. Sahney, editor, *Polynomial and Spline Approximation*. Reidal, Dordrecht, 1979.

[71] J. Meinguet. Multivariate interpolation at arbitrary points made simple. *J. Appl. Math. Phys.*, 30:292–304, 1979.

[72] B. W. Mel. MURPHY: a robot that learns by doing. In D. Z. Anderson, editor, *Neural Information Processing Systems*. American Institute of Physics, University of Colorado, Denver, 1987.

[73] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *J. Phys. Chem*, 21:1087, 1953.

[74] C. A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2:11–22, 1986.

[75] M. L. Minsky and S. Papert. *Perceptrons.* MIT Press, Cambridge, MA, 1969.

[76] J. Moody. Fast learning in multi resolution hierarchies. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems I*, pages 29–39. Morgan Kaufmann Publishers, Carnegie Mellon University, 1989.

[77] J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1(2):281–294, 1989.

[78] B. Moore and T. Poggio. Representations properties of multilayer feedforward networks. In *Abstracts of the first annual INNS meeting*, page 502, New York, 1988. Pergamon Press.

[79] V.A. Morozov. *Methods for solving incorrectly posed problems.* Springer-Verlag, Berlin, 1984.

[80] S. Omohundro. Efficient algorithms with neural network behaviour. *Complex Systems*, 1:273, 1987.

[81] G. Parisi. *Statistical Field Theory.* Addison-Wesley, Reading, Massachusets, 1988.

[82] E. Parzen. On estimation of a probability density function and mode. *Ann. Math. Statis.*, 33:1065–1076, 1962.

[83] R. Penrose. A generalized inverse for matrices. *Proc. Cambridge Philos. Soc.*, 51:406–413, 1955.

[84] D. I. Perrett, A.J. Mistlin, and A.J. Chitty. Visual neurones responsive to faces. *Trends in Neuroscience*, 10(9):358–364, 1987.

[85] T. Poggio. On optimal nonlinear associative recall. *Biological Cybernetics*, 19:201–209, 1975.

[86] T. Poggio. Visual algorithms. In O. J. Braddick and A. C. Sleigh, editors, *Physical and Biological Processing of Images*, pages 128–153. Springer–Verlag, Berlin, 1982.

[87] T. Poggio and S. Edelman. A network that learns to recognize 3D objects. *Nature*, 343:263–266, 1990.

[88] T. Poggio and the staff. MIT progress in understanding images. In *Proceedings Image Understanding Workshop*, Cambridge, MA, April 1988. Morgan Kaufmann, San Mateo, CA.

[89] T. Poggio and the staff. M.I.T. progress in understanding images. In *Proceedings Image Understanding Workshop*, pages 56–74, Palo Alto, CA, May 1989. Morgan Kaufmann, San Mateo, CA.

[90] T. Poggio and V. Torre. A theory of synaptic interactions. In W.E. Reichardt and T. Poggio, editors, *Theoretical approaches in neurobiology*, pages 28–38. The M.I.T Press, Cambridge, MA, 1978.

[91] M. J. D. Powell. Radial basis functions for multivariable interpolation: a review. In J. C. Mason and M. G. Cox, editors, *Algorithms for Approximation*. Clarendon Press, Oxford, 1987.

[92] S. Renals and R. Rohwer. Phoneme classification experiments using radial basis functions. In *Proceedings of the International Joint Conference on Neural Networks*, pages I–461–I–467, Washington D.C., June 1989. IEEE TAB Neural Network Committee.

[93] H.L. Resnikoff. On the psychophysical function. *J. Math. Biol.*, 2:265–276, 1975.

[94] S. Rippa. Interpolation and smoothing of scattered data by radial basis functions. M.sc. thesis, Tel Aviv University, 1984.

[95] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[96] H. Ritter and K. Schulten. Topology conserving mappings for learning motor tasks. In *Proc. AIP Conference: Neural networks of computing*, pages 376–380, Snowbird, Utah, 1986. J.S. Denker.

[97] H. Ritter and K. Schulten. Extending Kohonen's self-organizing mapping algorithm to learn ballistic movements. In R. Eckmiller and C. von der Malsburg, editors, *Neural Computers*, pages 393–406. Springer, Berlin, 1987.

[98] H. Ritter and K. Schulten. Convergence properties of Kohonen's topology conserving maps: fluctuations, stability, and dimension selection. *Biological Cybernetics*, 60:59–71, 1988.

[99] A. Rosenfeld and A.C. Kak. *Digital Picture Processing*. Academic Press, New York, 1982.

[100] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing*, chapter 8, pages 318–362. MIT Press, Cambridge, MA, 1986.

[101] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(9):533–536, October 1986.

[102] E. Saund. Dimensionality-reduction using connectionist networks. A.I. Memo No. 941, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1987.

[103] I.P. Schagen. Interpolation in two dimensions – a new technique. *J. Inst Math. Appl.*, 23:53–59, 1979.

[104] I.J. Schoenberg. Metric spaces and completely monotone functions. *Ann. of Math.*, 39:811–841, 1938.

[105] I.J. Schoenberg. Metric spaces and positive definite functions. *Ann. of Math.*, 44:522–536, 1938.

[106] J. Schwartz. On the effectiveness of backpropagation learning in trainable N^2 nets, and of other related form of discrimination learning. Preprint, 1988.

[107] T. J. Sejnowski and C. R. Rosenberg. Parallel networks that learn to pronounce english text. *Complex Systems*, 1:145–168, 1987.

[108] R.J. Solomonoff. Complexity-based induction systems: comparison and convergence theorems. *IEEE Transactions on Information Theory*, 24, 1978.

[109] D.A. Sprecher. On the structure of continuous functions of several variables. *Trans. Amer. Math. Soc.*, 115:340–355, 1965.

[110] J. Stewart. Positive definite functions and generalizations, an historical survey. *Rocky Mountain J. Math.*, 6:409–434, 1976.

[111] C. J. Stone. Optimal global rates of convergence for nonparametric regression. *The Annals of Statistics*, 10:1040–1053, 1982.

[112] C. J. Stone. Additive regression and other nonparametric models. *The Annals of Statistics*, 13:689–705, 1985.

[113] A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math. Dokl.*, 4:1035–1038, 1963.

[114] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems.* W. H. Winston, Washington, D.C., 1977.

[115] V. Torre and T. Poggio. An application: a synaptic mechanism possibly underlying motion detection. In W.E. Reichardt and T. Poggio, editors, *Theoretical approaches in neurobiology*, pages 39–46. The M.I.T Press, Cambridge, MA, 1978.

[116] R.Y. Tsai and T.S. Huang. Uniqueness and estimation of 3-D motion parameters of a rigid planar patch from three perspective views. In *Proc. of IEEE International Conference on ASSP*, Paris, France, May 3-5 1982.

[117] V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequences of events to their probabilities. *Th. Prob. and its Applications*, 17(2):264–280, 1971.

[118] A. G. Vitushkin. On Hilbert's thirteenth problem. *Dokl. Akad. Nauk SSSR*, 95:701–704, 1954.

[119] A. G. Vitushkin and G.M. Henkin. Linear superposition of functions. *Russian Math. Surveys*, 22:77–125, 1967.

[120] G. Wahba. Practical approximate solutions to linear operator equations when the data are noisy. *SIAM J. Numer. Anal.*, 14, 1977.

[121] N. Wax. *Selected papers on noise and stochastic processes.* Dover Publications, New York, 1954.

[122] D. Wolpert. Alternative generalizers to neural nets. In *Abstracts of the first annual INNS meeting*, New York, 1988. Pergamon Press.

[123] A. Yuille and N. Grzywacz. The motion coherence theory. In *Proceedings of the International Conference on Computer Vision*, pages 344–354, Washington, D.C., December 1988. IEEE Computer Society Press.

[124] D. Zipser and R.A. Andersen. A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. *Nature*, 331:679–684, 1988.