Networks for Approximation and Learning

TOMASO POGGIO, ASSOCIATE MEMBER, IEEE, AND FEDERICO GIROSI

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 multidimensional 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 problem of the approximation of nonlinear mappings—especially continuous mappings. We develop a theoretical framework for approximation based on regularization techniques that leads to a class of three-layer networks that we call regularization networks and include as a special case the well-known Radial Basis Functions method. Regularization 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. This paper generalizes the theory of regularization networks to a formulation that turns out to include task-dependent clustering and dimensionality reduction. We also discuss briefly some intriguing analogies with neurobiological data.

I. Learning as Approximation

The problem of learning a mapping between an input and an output space is 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).

Manuscript received August 15, 1989; revised March 20, 1990. This work was supported in part by a grant from the Office of Naval Research (ONR), Cognitive and Neural Sciences Division, by the Artificial Intelligence Center of Hughes Aircraft Corporation, and by the NATO Scientific Affairs Division (0403/87). Support for the Artificial Intelligence 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 under Office of Naval Research contract N00014-85-K-0124. The work of T. Poggio is supported by the Uncas and Ellen Whitaker chair.

The authors are with the Artificial Intelligence Laboratory and Center for Biological Information Processing, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.

IEEE Log Number 9038827.

A suggestive point of view on networks and classical approximation methods is provided by Omohundro [1] and an interesting review of networks, statistical inference, and estimation techniques has been given by Barron and Barron [2]. Learning from the point of view of approximation has been also considered among others by Schwartz [3], Poggio *et al.* [4], [5], Aloimonos [6], Moody and Darken [7], and Poggio [8]. A related area of research, concerned with learning of Boolean functions, has been developing rapidly since the seminal work of Valiant [9].

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 belonging to some set P (X and W are real vectors $X = (x_1, x_2, \cdots, x_n)$ and $W = (w_1, w_2, \cdots, w_m)$). 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 distinguish 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.
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. It is especially focused on the question of a good representation for learning continuous functions.

A. Networks and Approximation Schemes

Almost all approximation schemes can be mapped into some kind of network that can be dubbed as a "neural network." 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 represented by the composition of many basic functions. To see how the approx-

imation problem maps into such 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 then be stated formally as:

Approximation problem: *If $f(X)$ is a continuous function defined on set 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*

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

for all W in the set P.

A solution to this problem, if it exists, is said to be a *best approximation*. The existence of a best approximation depends ultimately on the class of functions to whom $F(W, X)$ belongs [10].

With these definitions we can consider a few examples of approximating functions $F(W, X)$: $R^n \rightarrow R$, that correspond to multilayer networks (see [11]):

- the classical linear case is

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

  where W and X are n-dimensional vectors. It corresponds to a network without hidden units;
- the classical approximation scheme is linear in a suitable basis $\{\Phi_i\}_{i=1}^m$ of functions of the original inputs X, that is

  $$F(W, X) = \sum_{i=1}^{m} W_i \Phi_i(X)$$

  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, F is a polynomial.
- the nested sigmoids scheme (of the type used with the backpropagation learning scheme, see [12]) can be written as

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

  where σ is a sigmoid function. It corresponds to a multilayer network of units that sum their inputs with "weights" $W = \{w_n, v_i, u_j, \cdots\}$ 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

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

  with σ being a linear threshold function, can represent all Boolean functions (any mapping S: $\{0, 1\}^N \rightarrow \{0, 1\}$ 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 arbitrarily well any continuous multivariate functions [13], [14] (Cybenko [15] and Moore and Poggio [16], 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 the network version of the problem of learning can be approached 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.

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 sums or products 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 [17], could be used to justify the use of multilayer networks [2] (see also [18]). Unfortunately, the claim is not warranted, as revealed by an analysis of Kolmogorov's result [19] to which we refer the reader.

Thus, exact representations with the required properties do not exist. Good and general approximating representations, however, may exist. The next section discusses the formulation of the problem of learning from examples as the problem of approximation of mappings. From this point of view, regularization techniques used for surface reconstruction are a natural framework for the problem of learning. This leads to the following problem: is there a connection between regularization techniques and feedforward, multilayer networks? Sections III and IV provide a solution to this problem by showing that regularization leads to an approximation scheme which is general, powerful, and maps into a class of networks with one layer of hidden units that we call *regularization networks*. We show that regularization networks are strictly related to the well-known interpolation method of Radial Basis Functions (RBF). A subset of regularization networks consists of Radial Basis Functions (though not all of them). The Appendix reviews some of the existing results about RBF. Section IV also provides powerful extensions of the basic regularization networks. In this paper we refer to the most powerful and general of the regularization networks as Hyper Basis Functions (HyperBF). The possible relevance of the work to neurophysiology is then briefly outlined in section V, together with a number of properties of Gaussian radial basis functions. Section VI sketches some of the applications of the technique, while the last section mentions several classical algorithms that can be regarded as special cases of HyperBF. We conclude with some comments on the crucial problem

of dimensionality faced by this and almost any other learning or approximation technique.

II. 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 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.

B. 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 [20], [21] 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, any number of examples for 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. Smoothness can be measured in a number of different ways. As we will explain later, our measure of deviation from smoothness is some functional containing derivatives of the function considered. The results of Stone [22] (see section VII-C) suggest that, if nothing else is known about a high dimensional function to be approximated, the only option may be to assume a high degree of smoothness. Otherwise, the number of examples required would be totally unpractical.

B. 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 [21], [23] (the first to introduce this technique in computer vision was Grimson, in 1981 [24]). 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. Thus, 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 [25]. A large body of results in fitting and approximating with splines may be therefore exploited.

C. 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 [11]). 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 nosie:

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

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 [11]). Under these conditions, the first term $-\Sigma_i (z_i - d_i)^2$—in the regularization principle of equation (1) corresponds to $C(d|z)$, whereas the second term—$\|Pz\|^2$—corresponds to the prior $C(z)$ [26].

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. Notice that

in practice as much a priori information as possible 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 curse of dimensionality (see [18]).

As pointed out by Rivest (in preparation), one can reverse the relationship between prior probabilities and complexity (see (2)). Instead of determining the complexity $C(z)$ in (2) from the prior, one may measure the *complexity* of the *a priori* hypotheses to determine the prior probabilities. Rissanen [27] 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 [28]). They connect data compression and coding with Bayesian inference, regularization, hypersurface reconstruction, and learning.

D. From Hypersurface Reconstruction to Networks

In the section above we have sketched the strict relations between learning, Bayes estimation, 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 and others have proposed that MRFs can be implemented in terms of hybrid networks of coupled analog and digital elements [26]. Standard regularization can be implemented by resistive grids, and has been implemented on an analog VLSI chip [29]. It is then natural to ask whether 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 Regularization Networks. Regularization Networks are closely related to an interpolation technique called Radial Basis Functions (RBF), which has recent theoretical foundations (see the review of Powell [30]) and has been used with very promising results [31]–[36].

III. Regularization Theory and Regularization Networks

In this section we apply regularization theory to the approximation/interpolation problem and we show the equivalence between regularization and a class of three-layer networks that we call regularization networks. These networks are not only equivalent to generalized splines, but are also closely related to the classical Radial Basis Functions used for interpolation tasks, which are discussed in some detail in Appendix A.

A. Regularization Theory

Let $S = \{(\boldsymbol{x}_i, y_i) \in R^n \times R | i = 1, \cdots N\}$ be a set of data that we want to approximate by means of a function f. The regularization approach [37], [21], [38], [23] determines the function f that minimizes the functional

$$H[f] = \sum_{i=1}^{N} (y_i - f(\boldsymbol{x}_i))^2 + \lambda \|Pf\|^2 \tag{3}$$

where P is a constraint operator (usually a differential operator), $\|\cdot\|$ is a norm on the function space to which Pf 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 [39], that in this case can always be written as

$$\hat{P}Pf(\boldsymbol{x}) = \frac{1}{\lambda} \sum_{i=1}^{N} (y_i - f(\boldsymbol{x}))\delta(\boldsymbol{x} - \boldsymbol{x}_i) \tag{4}$$

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 (4) is a partial differential equation, and it is well known that its solution can be written as the integral transformation 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:

$$\hat{P}PG(\boldsymbol{x}; \boldsymbol{y}) = \delta(\boldsymbol{x} - \boldsymbol{y}).$$

Because of the delta functions appearing in (4) the integral transformation becomes a discrete sum and f can then be written as

$$f(\boldsymbol{x}) = \frac{1}{\lambda} \sum_{i=1}^{N} (y_i - f(\boldsymbol{x}_i))G(\boldsymbol{x}; \boldsymbol{x}_i). \tag{5}$$

Equation (5) says that the solution of the regularization problem lies in an N-dimensional subspace of the space of smooth functions. A basis for this subspace is given by the N functions $G(\boldsymbol{x}; \boldsymbol{x}_i)$. In the following we will refer to $G(\boldsymbol{x}; \boldsymbol{x}_i)$ as to the Green's function "centered" at the point $\boldsymbol{x}_i$, and to the points $\boldsymbol{x}_i$ as to the "centers" of the expansion. The reason for this lies in the fact that usually the Green's function is translationally invariant, that is $G = G(\boldsymbol{x} - \boldsymbol{x}_i)$, and in this case $G(\boldsymbol{x})$ and $G(\boldsymbol{x} - \boldsymbol{x}_i)$ are equivalent modulo a coordinates translation that maps $\boldsymbol{x}_i$ in the origin.

A set of equations for the unknown coefficients $c_i = y_i - f(\boldsymbol{x}_i)/\lambda$ is easily obtained by evaluating equation (5) at the N data points $\boldsymbol{x}_i$. A straightforward calculation yields the following linear system:

$$(G + \lambda I)\mathbf{c} = \boldsymbol{y} \tag{6}$$

where I is the identity matrix, and we have defined

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

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

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} c_i G(\boldsymbol{x}; \boldsymbol{x}_i) \tag{7}$$

where the coefficients satisfy the linear system (6).

We notice however that this expression is not the complete solution of the minimization problem. In fact all the functions that lie in the null space of the operator P are "invisible" to the smoothing term in the functional (3), so that the previous expansion is the solution *modulo a* term that lies in the null space of P. The form of this term depends on the stabilizer that has been chosen and on the boundary conditions, and therefore on the particular problem that

has to be solved. For this reason, and since its inclusion does not modify the main conclusions, we will disregard it in the following. We just mention that for a stabilizer that is a homogeneous, rotationally invariant operator of degree n, the null space is the space of polynomials of degree $2n - 1$. This and other aspects of the minimization problem (3) can be found in the book of Wahba [40], where a result similar to the one of (7) is derived in a rigorous way by means of the technique of reproducing kernels.

Since the operator $\hat{P}P$ in equation (4) is self-adjoint, its Green's function is symmetric: $G(\mathbf{x}; \mathbf{y}) = G(\mathbf{y}; \mathbf{x})$. As a consequence the matrix G of equation (6) is symmetric and its eigenvalues are real numbers. The matrix $G + \lambda I$ is then of full rank (unless $-\lambda$ is equal to one of its eigenvalues) and the linear system (6) always has a solution. The existence of a solution in the case of $\lambda = 0$, that corresponds to pure interpolation, depends on the properties of the Green's function G. If the Green's function is positive definite this limit always exists, and an expansion of the type (7) interpolates the data without any null space term. If the Green's function is conditionally positive definite of some order (see Appendix A), well known results of approximation theory [41] guarantee that the addition to the expansion (7) of a polynomial of appropriate degree, that is the polynomial that lies in the null space of P, makes it always possible to interpolate the data points. Conditionally positive definiteness of the Green's function, as well as other properties, derives from the structure of the stabilizer P (see the next section for some examples). If the operator P is translationally invariant, G will depend on the difference of its arguments ($G = G(\mathbf{x} - \mathbf{y})$) and if it is rotationally and translationally invariant G will be a radial function: $G = G(\|\mathbf{x} - \mathbf{y}\|)$. In the last case the regularized solution is given by the following expansion:

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i G(\|\mathbf{x} - \mathbf{x}_i\|) \tag{8}$$

and the method of Radial Basis Functions may be recovered (see Appendix A). There are strict connections between the Radial Basis Function method and variational principles, some of which are sketched in the following subsection. Interesting results can be found in the paper of Dyn [42], where the work of Madych and Nelson [43] on semi-reproducing kernels and variational principles is discussed.

Notice that the requirement of rotational and translational invariance on P 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. An example involving non-radial stabilizers is the case of tensor product splines. Tensor product splines correspond to non-radial stabilizing operators that are the product of "one-dimensional" operators. In two dimensions, for example, they correspond to stabilizers of the form $P = P_x P_y$, where P_x (P_y) is a differential operator involving only derivatives with respect to x (y). The Green's function associated to $P_x P_y$ is the product of the Green's functions associated to P_x and P_y. The two dimensional problem is then regarded as the "tensor product" of two one-dimensional problems.

We now give some examples of stabilizers P and of their properties.

1) Examples

Multidimensional Splines: A widely used class of stabilizers is given by the functionals considered by Duchon [44] and Meinguet [45] in their variational approach to multivariate interpolation. In particular they considered functionals of the form

$$\|O^m f\|^2 = \sum_{i_1 \cdots i_m}^{n} \int_{R^n} d\mathbf{x} (\partial_{i_1 \cdots i_m} f(\mathbf{x}))^2$$

where $\partial_{i_1 \cdots i_m} = \partial^m / \partial x_{i_1} \cdots \partial x_{i_m}$ and $m \gtrsim 1$. Stabilizers of this type are invariant under rotations and translations. Moreover, since the differential operator involved is homogeneous, a scale transformation of the variables affects this functional multiplying it by a constant, implying that the operations of finding the solution and scaling the data commute.

The Green's function associated to this stabilizer is radial, translation invariant and satisfies the following differential equation (in the sense of the distributions):

$$(-1)^m \nabla 2^m G(\mathbf{x}) = \delta(\mathbf{x})$$

where ∇^{2m} is the m-iterated Laplacian in n dimensions. The solution of this differential equation can be found using the method of generalized Fourier transforms, and it is shown to be (see Gelfand and Vilenkin, pp. 202, 1964)

$$G(\mathbf{x}) = \begin{cases} \|\mathbf{x}\|^{2m-n} \ln \|\mathbf{x}\| & \text{if } 2m > n \text{ and } n \text{ is even} \\ \|\mathbf{x}\|^{2m-n} & \text{otherwise.} \end{cases} \tag{9}$$

It is clear from equation (9) that the constraint $2m > n$ has to be imposed on the degree of smoothness m in order to obtain a Green's function that is not singular in the origin. Suppose now that the condition $2m > n$ is fulfilled: it is well known from spline theory that if the stabilizer is of order m then the Green's function is conditionally positive definite of the same order. This means that, given m, in order to interpolate the set $\mathcal{S}$ of data, $\mathcal{S} = \{(\mathbf{x}_i, y_i) \in R^n \times R | i = 1, \cdots N\}$, the following function can be used:

$$f(\mathbf{x}) = \sum_{k=1}^{N} c_i G(\mathbf{x} - \mathbf{x}_i) + p_{m-1}(\mathbf{x})$$

where $p_{m-1}(\mathbf{x})$ is a polynomial of degree $m - 1$.

In the case $n = m = 2$ the functional to be minimized is

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

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 (9). Thin plate splines have been introduced by engineers for aeroelastic calculations [46], their name coming from the fact that $\|O^2 f\|^2$ is the bending energy of a thin plate of infinite extent.

A Generalization of Multidimensional Splines: We now consider the following generalization of the class of stabilizers previously shown:

$$\|P^M f\|^2 = \sum_{n=0}^{M} a_n \|O^n f\|^2 \tag{10}$$

The stabilizer is rotationally and translationally invariant and the Green's function satisfies the distributional dif-

ferential equation:

$$\sum_{m=0}^{M} (-1)^m a_m \nabla^{2m} G(\boldsymbol{x} - \boldsymbol{y}) = \delta(\boldsymbol{x} - \boldsymbol{y}). \quad (11)$$

By Fourier transforming both sides of equation (11) we obtain:

$$\sum_{m=0}^{M} a_m (\boldsymbol{s} \cdot \boldsymbol{s})^m \, G(\boldsymbol{s}) = 1$$

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

$$G(\boldsymbol{x}) = \int_{R^n} d\boldsymbol{s} \, \frac{e^{i\boldsymbol{s}\cdot\boldsymbol{x}}}{\sum_{m=0}^{M} a_m (\boldsymbol{s} \cdot \boldsymbol{s})^m} = \int_{R^n} d\boldsymbol{s} e^{i\boldsymbol{s}\cdot\boldsymbol{x}} \, dV(\boldsymbol{s}) \quad (12)$$

where $V(\boldsymbol{s})$ is a bounded non-decreasing function if $a_0 \neq 0$. Now we can apply Bochner's theorem [47], which states that a function is positive definite if and only if it can be written in the form (12), to conclude that $G(\boldsymbol{x})$ is positive definite. Notice that the condition $a_0 > 0$ is crucial in this particular derivation, and, as it has been pointed out by Yuille and Grzywacz [48], *it is a necessary and sufficient condition for the Green's function to fall asymptotically to zero.* If a_0 is zero a similar result holds, the Green's function being *conditionally* positive definite of some order, as in the case of the spline functions previously shown.

One simple one-dimensional example is provided by the following choice of the coefficients:

$$a_n = \begin{cases} q^2 & q > 0, \text{ if } n = 0 \\ 1 & \text{if } n = 1 \\ 0 & \text{otherwise.} \end{cases}$$

The Green's function for $q \neq 0$ is shown to be [49]:

$$G_q(x - y) = \frac{1}{2q} e^{-q|x-y|}.$$

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 number of coefficients to be different from zero. Here we give an example of a stabilizer of the type (10) in which the degree, and then the number M of terms, is let go to infinity. In this case the differential operator defining the Green's function ceases to be a differential operator in the standard sense, and it is called a pseudo-differential operator. As an example we consider the choice $a_m = \sigma^{2m}/m!2^m$, that leads to the pseudodifferential equation:

$$\sum_{n=0}^{\infty} (-1)^n \frac{\sigma^{2m}}{m!2^m} \nabla^{2n} G(\boldsymbol{x}) = \delta(\boldsymbol{x}).$$

Standard Fourier techniques yield the Green's function

$$G(\boldsymbol{x}) = Ae^{-(x^2/2\sigma^2)}$$

where A is a normalization constant. The regularized solution is then a linear superposition of Gaussians centered on the data points $\boldsymbol{x}_i$, and has interesting properties that will be shown in section (7).

B. Regularization Networks

An obvious property of this technique is that it can be implemented by a simple network with just one layer of hidden units, as shown in Fig. 1. The first layer of this network, that we call *regularization network* consists of "input" units whose number is equivalent to the number of independent variables of the problem. The second layer is composed of nonlinear "hidden" units fully connected to the first layer. There is one hidden unit for each data point $\boldsymbol{x}_i \equiv (x_i, y_i, z_i, \cdots)$, and the connections between the ith hidden unit and the input units are given by the coordinates $(x_i, y_i, z_i, \cdots)$ of the ith data point. The "activation function" of the hidden units is the Green's function G, so that the output of the ith hidden unit is $G(\boldsymbol{x}; \boldsymbol{x}_i)$. 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 expansion (7). It is straightforward to see that gradient descent, that minimizes the interpolation error on the data points, can be used to solve the system (6) with λ set to zero. If the Green's function is positive definite this solution will be the "optimal" interpolant, that is the interpolant that minimizes the functional $\|Pf\|^2$, even without the polynomial terms. If the Green's function is conditionally positive definite some appropriate polynomial units should be added to the network in order to obtain the optimal interpolant (Fig. 1 shows the case of a polynomial of order 1, i.e., linear).

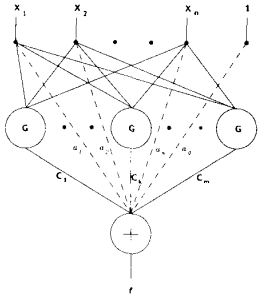


Fig. 1. The regularization network used to approximate a mapping between $x_1, x_2, \cdots, x_n$ and y, given a set of sparse, noisy data. In addition to the linear combination of Green's functions constant and linear terms are shown here as direct connections from the input to the output with weights $a_0, a_1, a_2, \cdots, a_n$. Constant, linear and higher order polynomials may be needed, depending on the stabilizer P.

Notice that the architecture of the regularization network is completely determined by the learning problem, and that, unlike most of the current "neural" networks, all weights between the input and the hidden layer are known. From the point of view of approximation theory the regularization network has three desirable properties:

1. It has been shown [50] that a regularization network can approximate arbitrarily well any multivariate continuous function on a compact domain, given a sufficiently high number of units. This property is shared by algebraic and trigonometric polynomials, as is shown by the classical Weierstrass Theorem, and by a large class of networks with one layer of hidden units [13], [15], [14], [16], [51].
2. Since the approximation scheme derived from regularization theory is linear in the unknown coefficients, it is easy to prove that it has the so called *best-approximation property* [50]. This means that

given a function f, there always exists a choice of coefficients that approximates f better than all other possible choices. This property, which is important mainly from the theoretical point of view, is shared by all the classical approximating schemes, such as approximation by polynomial and splines with fixed knots, in which the approximating solution depends *linearly* on the unknown parameters. It can be shown [50], that multilayer feedforward networks, of the type usually considered for backpropagation schemes, do not have this property.

3. The solution computed by the regularization network is "optimal" in the sense that it minimizes a functional that measures how much it oscillates. This eliminates solutions that perfectly interpolate the data points but badly oscillate where there are no data. This property is typical of the spline interpolation method, but it is not shared, for example, by the polynomial interpolation scheme.

Notice that in the regularization networks output units may also compute a fixed, nonlinear, invertible function σ ([11]), as already observed by Broomhead and Lowe [34]. This is useful for instance in the case of classification tasks, the function σ being naturally chosen to be a sigmoid function. Clearly a similar nonlinear function could be applied to each of the inputs. It seems possible that in some cases suitable input and output processing of this type may be advantageous. Poggio (see [18]) following Resnikoff [52], 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 these 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.

IV. Extensions of the Regularization Approach

In this section we extend the theory by defining a more general form of regularization networks, that can perform task-dependent clustering and dimensionality reduction and that we call Hyper Basis Functions. The extensions we propose are two:

1. The network associated with equation (7) has a complexity (number of units) that is independent of the dimensionality of the input space but is on the order of the dimensionality of the training set (number of examples), which is usually high. We show how to justify an approximation of equation (7) in which the number of units is much smaller than the number of examples and the positions of the "centers" of the expansion are modified during learning [11]. This scheme can be further extended by considering in equation (7) the superposition of different types of functions G, such as Gaussians of different scales.
2. The norm $\|\boldsymbol{x} - \boldsymbol{x}_i\|$ may be considered as a *weighted norm*

$$\|\boldsymbol{x} - \boldsymbol{x}_i\|_{\boldsymbol{W}}^2 = (\boldsymbol{x} - \boldsymbol{x}_i)^T \boldsymbol{W}^T \boldsymbol{W} (\boldsymbol{x} - \boldsymbol{x}_i)$$

where $\boldsymbol{W}$ is a square matrix and the superscript T indicates the transpose. In the simple case of diagonal $\boldsymbol{W}$ the diagonal elements w_{ii} assign a specific weight to each input coordinate, and the standard Euclidean norm is obtained when $\boldsymbol{W}$ is set to the identity matrix. They play a critical role whenever different types of inputs are present. We will show how the weighted norm idea can be derived rigorously from a slightly more general regularization principle than equation (3).

In the following we will introduce these two extensions and show that the *moving centers* are related to clustering techniques and that the *norm-weights* correspond to dimensionality reduction. We also mentioned two further extensions: learning in the presence of unreliable examples and learning from positive *and* negative examples.

A. Moving Centers: An Approximation to the Regularization Solution

The solution given by standard regularization theory to the approximation problem can be very expensive in computational terms when the number of examples is very high. The computation of the coefficients of the 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 addition, 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) [53]. In this section, we show how to reduce the complexity of the problem, introducing an approximation to the regularized solution.

A standard technique that has been used to find approximate solutions of variational problems is to expand the solution on a finite basis. The approximated solution $f^*(\boldsymbol{x})$ has then the following form:

$$f^*(\boldsymbol{x}) = \sum_{i=1}^{n} c_i \phi_i(\boldsymbol{x}) \tag{13}$$

where $\{\phi_i\}_{i=1}^n$ is a set of linearly independent functions [54]. The coefficients c_i are usually found according to some rule that guarantees a minimum deviation from the true solution. In the case of standard regularization, when the functional to minimize is given by equation (3), this method gives the *exact* solution if n is equal to the number of data points N, and $\{\phi_i\}_{i=1}^n = \{G(\boldsymbol{x}; \boldsymbol{x}_i)\}_{i=1}^N$, where G is the Green's function of the operator $\hat{P}P$. In this case the unknown coefficients of the expansion (13) can be obtained in a simple way by substituting expansion (13) in the regularization functional (3), that becomes a *function* $H[f^*] = H^*(c_1, \cdots, c_N)$, and then by minimizing $H[f^*]$ with respect to the coefficients, that is by setting:

$$\frac{\partial H[f^*]}{\partial c_i} = 0 \qquad i = 1, \cdots, N. \tag{14}$$

It can be easily shown [11] that if the Green's function vanishes on the boundary of the region that is considered

the set of equations (14) is equivalent to the linear system (6). In more general cases the basis $\{\phi_i\}_{i=1}^n$ should be enlarged, to include terms that generate the null space of P, in order to obtain the correct solution. For simplicity, we disregard these terms in the following, since they do not change the main conclusions. A natural approximation to the exact solution will then be of the form:

$$f^*(\boldsymbol{x}) = \sum_{\alpha=1}^{n} c_\alpha G(\boldsymbol{x}; \boldsymbol{t}_\alpha) \tag{15}$$

where the parameters $\boldsymbol{t}_\alpha$, that we call "centers", and the coefficients c_α are unknown, and are in general fewer than the data points ($n \leq N$). This form of solution has the desirable property to be a universal approximator for continuous functions [50] and to be the only choice that guarantees that in the case of $n = N$ and $\{\boldsymbol{t}_\alpha\}_{\alpha=1}^n = \{\boldsymbol{x}_i\}_{i=1}^n$ the correct solution (of equation (3)) is consistently recovered. We will see later how to find the unknown parameters of this expansion.

B. Different Types of Basis Functions and Multiple Scales

This scheme can be further extended by considering in equation (15) the superposition of different types of functions G, such as Gaussians at different scales. The function f to be approximated is regarded as the sum of p components f^m, $m = 1, \cdots, p$, 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

$$H[f] = \sum_{i=1}^{N} \left(\sum_{m=1}^{p} f^m(\boldsymbol{x}_i) - y_i \right)^2 + \sum_{m=1}^{p} \lambda_m \|P^m f^m\|^2. \tag{16}$$

Analyzing the structure of the Euler-Lagrange equations associated to equation (16) it can be shown that the function $F(\boldsymbol{x})$ that minimizes the functional (16) is a *linear superposition of linear superpositions* of the Green's functions G^m corresponding to the stabilizers P^m. Exactly as in the previous section, an approximated solution f^* to the variational problem is sought of the following form:

$$f^*(\boldsymbol{x}) = \sum_{m=1}^{p} \sum_{\alpha=1}^{K_m} c_\alpha^m G^m(\boldsymbol{x}; \boldsymbol{t}_\alpha^m) \tag{17}$$

where $K_m < N$ and the coefficients c_α^m and the *centers* $\boldsymbol{t}_\alpha^m$ are to be found.

This method leads in particular 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 $\sigma_1, \cdots, \sigma_p$: we can use this information to choose a set of p stabilizers whose Green's functions are, for example, Gaussians of variance $\sigma_1, \cdots, \sigma_p$. 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.

This 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.

C. Weighted Norm and Regularization

The norm in equation (15) is usually intended as an Euclidean norm. If the components of $\boldsymbol{x}$ are of different types, it is natural to consider a *weighted norm* defined as $\|\boldsymbol{x}\|_{\boldsymbol{W}}^2 = \boldsymbol{x}^T \boldsymbol{W}^T \boldsymbol{W} \boldsymbol{x}$, since the relative scale of the components is otherwise arbitrary. The case in which the matrix $\boldsymbol{W}$ is known (from prior information) does not present any difficulty. It is interesting, however, to see what it means in terms of the underlying regularization principle.

The regularization principle consists of finding the f that minimizes the functional:

$$H_{\boldsymbol{W}}[f] = \sum_{i=1}^{N} (y_i - f(\boldsymbol{x}_i))^2 + \lambda \|Pf\|_{\boldsymbol{y}}^2 \tag{18}$$

where we assume that P is radially symmetric in the variable $\boldsymbol{y}$ and that $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ (i.e. $\boldsymbol{y}$ is a known linear transformation of $\boldsymbol{x}$ that depends on the parameters $\boldsymbol{W}$). This means that the smoothness constraint is given in a space that is an affine transformation of the original $\boldsymbol{x}$ space. The Green's function associated with equation (18) is $G(\|\boldsymbol{y}\|^2) = G(\|\boldsymbol{x}\|_{\boldsymbol{W}}^2)$. If this formulation is used together with the moving center scheme, the approximated solution of the regularization problem has the form:

$$f^*(\boldsymbol{x}) = \sum_{\alpha=1}^{n} c_\alpha G(\|\boldsymbol{x} - \boldsymbol{t}_\alpha\|_{\boldsymbol{W}}^2) \tag{19}$$

Suppose now that the parameters $\boldsymbol{W}$ are unknown. We can formulate the problem of finding f and $\boldsymbol{W}$ that minimize the functional $H_{\boldsymbol{W}}(f)$. Thus finding the optimal $\boldsymbol{W}$ corresponds to finding the best stabilizer among those that are expressed in a coordinate system which is a linear transformation of the original one.

The simplest case is the case of $\boldsymbol{W}$ diagonal and $G(x) = e^{-x^2}$. In this case

$$G(\|\boldsymbol{x}\|_{\boldsymbol{W}}^2) = e^{-x_1^2 w_1^2} e^{-x_2^2 w_2^2} \cdots e^{-x_n^2 w_n^2}$$

and thus the diagonal elements w_i of $\boldsymbol{W}$ are equivalent to the inverse of the variance σ of each component of the multidimensional Gaussian.

D. Learning with Unreliable and Negative Examples

In the standard regularization approach, as well as in the extensions shown above, the set of data g is fixed, and all the data points are used in order to obtain a solution. It is possible to modify the functional (3) to take in account the possibility of excluding unreliable data, that are "spurious" or "too noisy". An analysis, similar to the one performed by Geiger and Girosi [55] on the problem of reconstructing piecewise smooth surfaces, shows that in the presence of unreliable data the functional (3) has to be replaced with the functional [56]

$$H'[f] = \sum_{i=1}^{N} V(\Delta_i) + \lambda \|Pf\|^2. \tag{20}$$

Here we have defined the "effective potential" V as the function

$$V(x) = x^2 - \frac{1}{\beta} \ln (1 + e^{-\beta(\epsilon - x^2)})$$

where ϵ is a positive number and β is a positive parameter, that is usually let go to infinity. In the case of β going to infinity the meaning of the effective potential is the following: closeness to the ith data point is enforced only if the interpolation error Δ_i is smaller that $\sqrt{\epsilon}$ (in this region $V(x) = x^2$ as in the standard regularization case). If the interpolation error is larger than $\sqrt{\epsilon}$ it is very likely that the datum is spurious, so that there is no need to enforce the function to go through that datum ($V(x) = \text{constant} = \epsilon$, while in the standard regularization case it is still quadratic).

The effective potential can be used also to deal with the problem of *negative examples*. Suppose we know that the function at the points $\{\boldsymbol{t}_\alpha\}_{\alpha=1}^K$ has to assume values that are *far* from the values $\{y_\alpha\}_{\alpha=1}^K$. This *a priori* knowledge can be introduced in the standard regularization functional by adding an appropriate term, that is, by minimizing

$$H''[f] = \sum_{i=1}^{N} \Delta_i^2 - \sum_{\alpha=1}^{N} V(\Delta_\alpha) + \lambda \|Pf\|^2. \tag{21}$$

where γ is a positive parameter. Due to the minus sign in equation (21) the interpolation error Δ_α at the points $\{\boldsymbol{t}_\alpha\}_{\alpha=1}^K$ is enforced to be larger than $\sqrt{\gamma}$. Notice that the solutions of the minimization problems (20) and (21) still have the form of linear superposition of Green's functions, and then can be implemented by a regularization network whose "weights" are found using a gradient-descent procedure.

E. How to Learn Centers' Positions and Norm Weights

Suppose that we look for an approximated solution of the regularization problem of the form (19). We now have the problem of finding the n coefficients c_α, the $d \times n$ coordinates of the centers $\boldsymbol{t}_\alpha$ and the d^2 elements of the matrix $\boldsymbol{W}$ so that the expansion (19) is optimal. In this case we can make use of a natural definition of optimality, given by the functional H. We then impose the condition that the set $\{c_\alpha, \boldsymbol{t}_\alpha | \alpha = 1, \cdots, n\}$ and the matrix $\boldsymbol{W}$ must be such that they minimizes $H[f^*]$, and the following equations must be satisfied:

$$\frac{\partial H[f^*]}{\partial c_\alpha} = 0, \qquad \frac{\partial H[f^*]}{\partial \boldsymbol{t}_\alpha} = 0, \qquad \frac{\partial H[f^*]}{\partial \boldsymbol{W}} = 0,$$
$$\alpha = 1, \cdots, n.$$

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 [57] or variations of the Metropolis algorithm (Caprile and Girosi, in preparation) may be better than gradient descent and should be used in practice. Since the function $H[f^*]$ to minimize is in general non-convex, a stochastic term in the gradient descent equations may be advisable to avoid local minima. In the gradient descent method the values of c_α, $\boldsymbol{t}_\alpha$, and $\boldsymbol{W}$ that minimize $H[f^*]$ are regarded as the coordinates of the stable fixed point of the following dynamical system:

$$\dot{c}_\alpha = -\omega \frac{\partial H[f^*]}{\partial c_\alpha}, \qquad \dot{\boldsymbol{t}}_\alpha = -\omega \frac{\partial H[f^*]}{\partial \boldsymbol{t}_\alpha}, \qquad \dot{\boldsymbol{W}} = -\omega \frac{\partial H[f^*]}{\partial \boldsymbol{W}},$$
$$\alpha = 1, \cdots, n$$

where ω is a parameter determining the microscopic timescale of the problem and is related to the rate of convergence to the fixed point. Defining

$$\Delta_i \equiv y_i - f^*(\boldsymbol{x}) = y_i - \sum_{\alpha=1}^{n} c_\alpha G(\|\boldsymbol{x}_i - \boldsymbol{t}_\alpha\|_W^2)$$

and setting $\lambda = 0$ for simplicity (the more general case can be approached in a similar way) in equation (3) we obtain

$$H[f^*] = H_{c,W,t} = \sum_{i=1}^{N} (\Delta_i)^2.$$

The important quantities—that can be used in more efficient schemes than gradient descent—are:

- for the c_α
$$\frac{\partial H[f^*]}{\partial c_\alpha} = -2 \sum_{i=1}^{N} \Delta_i G(\|\boldsymbol{x}_i - \boldsymbol{t}_\alpha\|_W^2); \tag{22}$$
- for the centers t_α
$$\frac{\partial H[f^*]}{\partial c_\alpha} = 4c_\alpha \sum_{i=1}^{N} \Delta_i G'(\|\boldsymbol{x}_i - \boldsymbol{t}_\alpha\|_W^2)\, \boldsymbol{W}^T\boldsymbol{W}(\boldsymbol{x}_i - \boldsymbol{t}_\alpha) \tag{23}$$
- and for $\boldsymbol{W}$
$$\frac{\partial H[f^*]}{\partial \boldsymbol{W}} = -4\boldsymbol{W} \sum_{\alpha=1}^{N} c_\alpha \sum_{i=1}^{N} \Delta_i G'(\|\boldsymbol{x}_i - \boldsymbol{t}_\alpha\|_W^2) Q_{i,\alpha} \tag{24}$$
where $Q_{i,\alpha} = (\boldsymbol{x}_i - \boldsymbol{t}_\alpha)(\boldsymbol{x}_i - \boldsymbol{t}_\alpha)^T$ is a dyadic product and G' is the first derivative of G. Instead of $\boldsymbol{W}$ we have also used $\boldsymbol{M} = \boldsymbol{W}^T\boldsymbol{W}$ with the appropriate equivalent of equation (24).

Remarks

1. Equation (22) has a simple interpretation: the correction is equal to the sum over the examples of the products between the error on that example and the "activity" of the "unit" that represents with its center that example. Notice that $H[f^*]$ is quadratic in the coefficients c_α, and if the centers and the matrix $\boldsymbol{W}$ are kept fixed, it can be shown [11] that the optimal coefficients are given by
$$\boldsymbol{c} = (G^TG + \lambda g)^{-1}G^T\boldsymbol{y} \tag{25}$$
where we have defined $(\boldsymbol{y})_i = y_i$, $(\boldsymbol{c})_\alpha = c_\alpha$, $(G)_{i\alpha} = G(\boldsymbol{x}_i; \boldsymbol{t}_\alpha)$ and $(g)_{\alpha\beta} = G(\boldsymbol{t}_\alpha; \boldsymbol{t}_\beta)$. If λ is let go to zero the matrix on the right side of equation (25) converges to the pseudoinverse of G [58], and if the Green's function is radial the approximation method of Broomhead and Lowe [34] is recovered.
2. Equation (23) is similar to task-dependent clustering [11]. This can be best seen by assuming that Δ_i are constant: then the gradient descent updating rule makes the centers move towards the majority of the data, to find the position of the cluster. Equating $\partial H[f^*]/\partial \boldsymbol{t}_\alpha$ to zero we notice that, when the matrix $\boldsymbol{W}$ is set to the identity matrix, the optimal centers $\boldsymbol{t}_\alpha$ satisfy the following set of nonlinear equations:
$$\boldsymbol{t}_\alpha = \frac{\sum_i P_i^\alpha \boldsymbol{x}_i}{\sum_i P_i^\alpha} \qquad \alpha = 1, \cdots, n$$
where $P_i^\alpha = \Delta_i G'(\|\boldsymbol{x}_i - \boldsymbol{t}_\alpha\|^2)$. The optimal centers are then a weighted sum of the data points. The weight P_i^α of the data point i for a given center $\boldsymbol{t}_\alpha$ is high if the interpolation error Δ_i is high there *and* the radial basis function centered on that knot changes quickly in a neighborhood of the data

point. This observation could suggest faster methods for finding a quasi-optimal set of knots [7].

3. Equation (24) contains the quantity $\Sigma_{i=1}^{N} Q_{i,\alpha}$ which is an estimate of the correlation matrix of all the examples relative to $\boldsymbol{t}_\alpha$ (modulus a normalization factor). Notice that $\Sigma_{i=1}^{N} Q_{i,\alpha}$ can be written as the product of a matrix containing N columns, each being one example (minus the quantity $\boldsymbol{t}_\alpha$) times the transpose of the same matrix: it is therefore a $d \times d$ matrix (d being the number of components of $\boldsymbol{x}$). It can be shown [59] that equation (24), under strong simplifying conditions, converges to a $\boldsymbol{W}$ with rows that are close to the eigenvectors of Q with the smallest eigenvalues. In other words, the equation would then converge to rows of $\boldsymbol{W}$ that span the space orthogonal to the space spanned by the principal components of the input examples (i.e. the eigenvectors of Q with the largest eigenvalues).
4. Equation (24) is similar to an operation of (task-dependent) dimensionality reduction [60] whereas equation (23) is similar to a clustering process. It is conceivable that learning the weights of the norm is even more important than learning the centers and that in many cases it may be preferable to set the centers to a representative subset of the data and to keep them fixed thereafter.
5. A specific matrix $\boldsymbol{W}$ corresponds to a specific metric in the multidimensional input space: $\boldsymbol{W}$ projects the input vector into the subspace spanned by its rows. In the case of the rows of $\boldsymbol{W}$ spanning the space orthogonal to the principal components of the inputs, $\boldsymbol{W}$ assigns a metric ellipsoid with the larges axes (corresponding to a large σ, if the Green's function is a Gaussian) along the principal components and the small axis (corresponding to a small σ in the Gaussian) orthogonal to it: thus even vectors that are far away (in the ordinary Euclidean metric) are close in this metric if they lie in the hyperplane of the principal components and even close vectors (in the ordinary metric) are far away in the metric induced by $\boldsymbol{W}$ if they are orthogonal to the principal components.
6. In the case of N examples, $n = N$ fixed centers and $\boldsymbol{W} = I$, there are enough data to constrain the N c_α to be found. Moving centers add another $n \times d$ parameters (d is the number of input components) and the matrix $\boldsymbol{W}^T\boldsymbol{W}$ another d^2 parameters. Thus the number of examples N must be sufficiently large to constrain adequately the free parameters—n d-dimensional centers, n coefficients c_α and d^2 entries of the matrix $\boldsymbol{W}$ (notice that only $(d^2 + d)/2$ entries will be independent). Thus the condition $N > K + Kd + d^2$ should be satisfied.
7. In the case of Gaussian basis functions, learning the entries of a diagonal $\boldsymbol{W}$ is equivalent to learn the σ of each two-dimensional (or one-dimensional) Gaussian receptive field for each center.
8. 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, for example adding to the functional (3) a term of the form $\Sigma_{\alpha \neq \beta} \Psi(\|\boldsymbol{t}_\alpha - \boldsymbol{t}_\beta\|)$, where Ψ is an appropriate repulsive potential. The gradient descent equation can be easily modified to reflect this additional term.

F. A Practical Algorithm

It seems natural to try to find a reasonable initial value for the parameters c_α, $\boldsymbol{t}_\alpha$, and $\boldsymbol{W}$ to start the minimization process. In absence of more specific prior information the following heuristics seems reasonable.

- Set the number of centers n and set the centers positions to a subset of the examples;
- Set the rows of $\boldsymbol{W}$ to be vectors orthogonal to the eigenvectors of $\Sigma_\alpha \Sigma_i Q_{i,\alpha}$ with largest eigenvalues;
- Use matrix pseudo-inversion to find the $\mathbf{c}_\alpha$;
- Use the $\boldsymbol{t}_\alpha$, $\boldsymbol{W}$ and c_α found so far as initial values for gradient descent equations.

As discussed in [59] even more general strategies may make sense.

V. Gaussian Basis Functions and Science-Fiction Neurobiology

In this section we point out some remarkable properties of Gaussian Basis Functions, that may have significant implications for neurobiology and, to a lesser extent, for VLSI circuit implementations.

A. Factorizable Radial Basis Functions

The synthesis of radial basis functions in many 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* (tensor product splines correspond to factorizable Green functions which are not radial). A multidimensional Gaussian function can be represented as the product of lower dimensional Gaussians. For instance a 2D Gaussian radial function centered in $\boldsymbol{t}$ can be written as:

$$G(\|\boldsymbol{x} - \boldsymbol{t}\|^2) \equiv e^{-\|\boldsymbol{x}-\boldsymbol{t}\|^2} = e^{-(x-t_x)^2} e^{-(y-t_y)^2}. \qquad (26)$$

This dimensionality factorization is especially attractive from the physiological point of view, since it is difficult to imagine how neurons could compute $G(\|\boldsymbol{x} - \boldsymbol{t}_\alpha\|^2)$ in a simple way for dimensions higher than two. The scheme of Fig. 2, 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).

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 Gaussian Radial Basis Function 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.

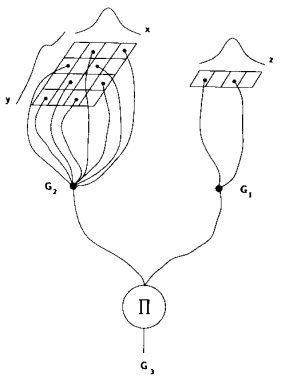


Fig. 2. A three-dimensional radial Gaussian implemented by multiplying two-dimensional Gaussian and one-dimensional Gaussian receptive fields. The latter two functions are synthesized directly by appropriately weighted connections from the sensor arrays, as neural receptive fields are usually thought to arise. Notice that they transduce the implicit position of stimuli in the sensor array into a number (the activity of the unit). They 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.

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 network may be one of the inputs to another network.

B. Style of Computation and Physiological Predictions

The multiplication operation required by the previous interpretation of Gaussian networks 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, as discussed in more detail by [11], directly on the dendritic tree of the neuron representing the corresponding radial function.

The 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 (22) or a somewhat similar, quasi-hebbian scheme is not too unlikely and may require only a small amount of plausible neural circuitry. Equations (23) seem more difficult to implement for a network of real neurons. 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 (23), such as, for instance, competitive learning, and that are biologically more plausible. The same can be said about the process that determines the weights in the norm.

Regularization networks with a Gaussian Green's function suggest an intriguing metaphor for a computational strategy that the brain may use. 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 obvious from the regularization formulation. The view is in the spirit of the key role that the concept of receptive field has always played in neurophysiology. 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.

VI. Some Applications

Many problems in several different fields such as system estimation, computer vision, speech understanding, statistical estimation, analysis of time series, signal processing can be formulated as problems of approximating multivariate functions from sparse data or, equivalently, as problems of learning from examples. In all these cases, especially when the problem involves continuous output values rather than binary (as in classification tasks), regularization networks can be used. This section sketches just three applications.

A. Recognizing a 3D Object from its Perspective Views

Consider the problem of recognizing a wire-frame 3D object from any of its perspective views. A view of the object is represented, for instance, as a $2N$ vector $x_1, y_1, x_2, y_2, \cdots, x_N, y_N$ of the coordinates on the image plane of N labeled and visible points on the object. Additional different types of features can also be used, such as angles between vertices. The network learns to map any view of the object into a classification function. The results with

images generated with computer graphics tools are encouraging, using a small number of training views [61]. Other encoding schemes are roughly equivalent, such as angles between the edges or segments lengths. A similar network with the same centers but different c can be used to provide the attitude of the object in space for any of its views [62].

B. Learning Dynamical Systems

HyperBF can be used to "learn" a dynamical system from the time course of its output. In fact, RBF have been often suggested as a good technique for this problem and have been successfully tested in some cases [34, 36]. 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). 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 [63] and the backpropagation algorithm of Lapedes and Farber [64]. The task of learning filters, especially recursive filters, for signal processing applications, is a closely related problem.

C. Learning Perceptual and Motor Tasks

Regularization networks have a good chance of being capable of synthesizing several vision algorithms from examples, since several problems in vision have satisfactory solutions in terms of regularization. The use of regularization networks is not restricted to sensory processes and they may also be used to learn motor tasks 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 [65]. Regularization 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.

VII. Conclusions

In this final section we discuss the structure of HyperBF, their relation to classical techniques, some general points about the most crucial problem of learning, the "curse of dimensionality", and its relation with the key assumption underlying regularization and regularization networks—the assumption of smoothness.

A. How Regularization Networks Really Work

Regularization networks 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.

To have a feeling of how regularization networks work let us consider a specific, extreme case, in which we consider a regularization network as a classifier, something the formal theory does not actually allow. Imagine using a regularization scheme to classify patterns, such as handwritten digits, in different classes. Assume that the input is a binary 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 regularization networks of increasing generality:

1. Each of the unit (its center corresponds to an example) is an hypersphere 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. The network is equivalent here 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 (as many as examples) and the output. The c can then be found by the pseudoinverse techniques (or gradient descent) and are guaranteed to be optimal in the L_2 sense.
4. We now allow a number of (movable) centers, which is less than the number of examples. Moving a center is equivalent to modifying the corresponding template. Thus equation (23) attempts to develop better templates by modifying during training the existing ones. In our case, this means changing the pixel values in the arrays representing the digits.
5. We allow an arbitrary weighted norm, with the weights to be found during the learning stage. This corresponds to finding which new features, synthesized as linear combinations of the inputs components, optimally capture the information in the input set, necessary for the task. Irrelevant features, in our example irrelevant pixels, will be assigned negligible weights.
6. Finally the most general network, in addition to the above features, also contains radial units, for instance of the Gaussian type, of different scale (i.e. σ), together with non-radial units associated to appropriate stabilizers and units that may receive only subsets of the inputs. This is the HyperBF scheme.

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 HyperBF suggest that, in a sense, HyperBF 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 into a powerful approximation scheme equivalent to generalized splines, which are probably the most powerful approximation method known. From another perspective, Gaussian HyperBF can be regarded as *disjunction of conjunctions* and seem therefore a natural and satisfying way to connect the representation of Boolean functions with the approximation of continuous, smooth multivariate functions.

B. Relations with Other Methods

Many existing schemes for networks that learn are encompassed by the framework of regularization networks. In this section, we will mention briefly some of the most obvious connections with existing methods. Because of space limitations, we refer the reader to the appropriate references and to [11] for a detailed discussion of the methods we consider.

Regularization networks are the feedforward network versions of regularization, and are therefore equivalent to generalized splines. They are similar to the architecture used for backpropagation, being multilayer networks with one hidden layer and two or even three sets of adjustable parameters. Their Boolean limit version carves the input space into hyperspheres, each corresponding to a center: a radial unit is active if the input vector is within a certain radius of its center and is otherwise silent. The Boolean limit of backpropagation carves the space with hyperplanes. With an arbitrary number of units each network can approximate the other, since each network can approximate arbitrarily well continuous functions on a limited interval [13, 50]. Multilayer networks with sigmoid units do not have, however, the best approximation property that regularization networks have [50]. The Boolean limit of regularization networks is almost identical to Kanerva's associative memory algorithm [66], which is itself closely related to vector quantization. Parzen windows, potential techniques in pattern recognition and, more in general, kernel estimation methods can be regarded as special cases of our method. Close analogies between Kanerva's model and Marr's [67] and Albus' [68] models of the cerebellum also exist [69, 11]. The update equation that controls the evolution of the centers $\boldsymbol{t}_\alpha$ is also similar to Kohonen's topology preserving algorithm [70, 11] (which is also similar to the k-means algorithm [71]) and can be interpreted as a learning scheme in which the centers of the radial functions move to find centers of clusters of input vectors. Coarse coding techniques can be interpreted within the HyperBF framework (for the special case of Gaussian Radial Basis functions) [11]. Regularization networks have also similarities with the class of Memory-Based Reasoning methods, recently used by D. Waltz and coworkers [72] on massively parallel machines, since in their simplest version (as many centers as examples) they are essentially look-up tables that find those past instances that are sufficiently close to the new input. In fact, regularization networks can be regarded as a powerful and simple extension of Memory-Based Reasoning that makes it equivalent to generalized splines.

C. Networks and Learning: The Pervasive Problem of Dimensionality

Our main result shows that for the learning problem regularization theory yields naturally a class of feedforward multilayer networks. 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* [22], [2]? 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 [73], [22], [74]. This problem has been extensively studied and some fundamental results have been obtained by Stone [22]. 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 $\epsilon_n = n^{-(p/2p+d)}$ 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 *exponentially with the ratio between the dimensionality d and its degree of smoothness p*.

Other interesting results have been obtained by Baum and Haussler on the statistical reliability of networks for binary classification [75], [76], whereas another approach to dimensionality reduction has been pursued by J. Schwartz [3] (similar to [77]). 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 [78], [79] and even without a priori information on the dimensionality of the problem, turned out to be effective in computer simulations [3].

D. 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 neural networks.

In this paper, we developed a theoretical framework based on regularization techniques that leads to a class of three-layer networks, useful for approximation, that we call regularization networks. The most general form of them is called Hyper Basis Functions, since they are 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. Regularization networks have a feedforward, multilayer network

architecture with good theoretical foundations. They may provide the best framework within which we can study general issues for learning techniques of the neural network type.

A Radial Basis Functions: A Review

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 $\{x_i \in R^n | i = 1, \cdots N\}$ and N real numbers $\{y_i \in R | i = 1, \cdots N\}$ find a function F from R^n to R satisfying the interpolation conditions:*

$$F(\boldsymbol{x}_i) = y_i \qquad i = 1, \cdots, N.$$

The RBF approach consists in choosing a function F of the following form:

$$F(\boldsymbol{x}) = \sum_{i=1}^{N} c_i h(\|\boldsymbol{x} - \boldsymbol{x}_i\|) + \sum_{i=1}^{m} d_i p_i(\boldsymbol{x}) \qquad m \leq n \qquad (27)$$

where h is a continuous function from R^+ to R, usually called the *radial basis function*, $\|\cdot\|$ is the Euclidean norm on R^n, $\{p_i | i = 1, \cdots, m\}$ 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 (27), so that the remaining degrees of freedom are fixed by imposing the following constraints:

$$\sum_{i=1}^{N} c_i p_j(\boldsymbol{x}_i) = 0, \qquad j = 1, \cdots, m.$$

In order to discuss the solvability of the interpolation problem by means of this representation we need the following definition [80, 41]:

Definition A.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 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in R^n$ and scalars $c_1, \cdots, c_N$ such that $\sum_{i=1}^{N} c_i p(\boldsymbol{x}_i) = 0$ for all $p \in \pi_{k-1}(R^n)$, the quadratic form $\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j f(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$ 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 $\sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j f(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$ is (positive) nonnegative [81].

Well known results of approximation theory assert that a sufficient condition for the existence of a solution of the form (27) to the interpolation problem is that $h \in \mathcal{P}_k(R^n)$, where we have defined $\mathcal{P}_k(R^n)$ as the set of conditionally positive definite functions of order k. It is then an important problem to give a full characterization of this class. In particular it is important to characterize the set of functions that are conditionally positive definite of order k over any R^n, that we define as simply $\mathcal{P}_k$.

An interesting characterization of $\mathcal{P}_k$ has been recently obtained by C. A. Micchelli [41]. Before stating his result we first give the following:

Definition A.2 *A function f is said to be completely monotonic on $(0, \infty)$ provided that it is $C^\infty(0, \infty)$ and $(-1)^l(\partial^l f / \partial \boldsymbol{x}^l)(x) \geq 0$, $\forall x \in (0, \infty)$, $\forall l \in \mathfrak{N}$, where $\mathfrak{N}$ is the set of natural numbers.*

We define $\mathfrak{M}_k$ the set of all the functions whose kth derivative is completely monotonic on $(0, \infty)$. Micchelli showed that there is a deep connection between $\mathfrak{M}_k$ and $\mathcal{P}_k$. In fact he proved the following theorem:

Theorem A.1 (Micchelli, 1986) *For every natural number k, $h(r^2) \in \mathcal{P}_k$ whenever $h(r)$ is continuous on $[0, \infty)$ and $(-1)^k(\partial^k h(r)/\partial r^k)$ is completely monotonic on $(0, \infty)$.*

To our extents the practical implication of this theorem is the following: if the kth derivative of $h(r)$ is completely monotonic the expansion (27) can be used to solve the interpolation problem. It has been noticed [41, 30] that this theorem encompasses the results obtained by Duchon [44] and Meinguet [45] in their variational approach to splines. For instance the functions $h(r) = r^{3/2}$ and $g(r) = \frac{1}{2} r \log \sqrt{r}$ belong to $\mathfrak{M}_2$, and by theorem A.1 the functions $h(r^2) = r^3$ and $g(r^2) = r^2 \log r$ ("thin plate splines") belong to $\mathcal{P}_2$: therefore it is possible to interpolate any set of data points using $h(r^2)$ and $g(r^2)$ as radial basis functions in the expansion (27), where the polynomial is of degree one. This corresponds exactly to the result derived by Duchon and Meinguet, but without some of their limitations (see Example 2 in section V-A-2). Since this method has been shown to embody natural spline interpolation in one dimension [30], can then be considered as an extension of natural splines to multivariable interpolation.

We notice that when $k = 0$ the theorem of Micchelli gives, as a particular case, a well known theorem of Schoenberg on the positive definite functions [81, 82]. In this case the radial basis functions expansion (27) becomes

$$F(\boldsymbol{x}) = \sum_{i=1}^{N} c_i h(\|\boldsymbol{x} - \boldsymbol{x}_i\|). \qquad (28)$$

The unknown coefficients c_i can be recovered by imposing the interpolation conditions $F(\boldsymbol{x}_j) = y_j (j = 1, \cdots N)$, that substituted in equation (28) yields the linear system

$$H\boldsymbol{c} = \boldsymbol{y}. \qquad (29)$$

where we have defined $(\boldsymbol{y})_j = y_j$, $(\boldsymbol{c})_i = c_i$, $(H)_{ij} = h(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)$. The theorem of Micchelli ensures that the solution of system (29) always exists, since the matrix H can be inverted, being strictly positive definite.

From equation (29) it turns out that a necessary and sufficient condition to solve the interpolation problem is the invertibility of the matrix H. Theorem (A.1), however, gives only a sufficient condition, so that many other functions could be used as radial basis functions without being strictly positive definite. Another sufficient condition has been given by Micchelli, who proved the following theorem [41]:

Theorem A.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 $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N \in R^n$*

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

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 theo-

rem (A.2), so that the choice $(c^2 + r^2)^\alpha$ is possible for the function h in (28).

A list of functions that can be used in practice for data interpolation by means of the RBF expansion (28) is given below, and their use is justified by the results of Micchelli:

$$h(r) = e^{-(r/c)^2} \quad \text{(Gaussian)}$$

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

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

$$h(r) = r \quad \text{(linear)}$$

Notice that the linear case corresponds, in one dimension, to piecewise linear interpolation, that is the simplest case of spline interpolation. In the case $\beta = \frac{1}{2}$ the radial basis function corresponds to the "Hardy's multiquadric" [31], that has been extensively used in surface interpolation with very good results [32], [33]. Some of the functions listed above have been used in practice.

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 [32] 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 [41]. 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 have seen in section III.

Acknowledgment

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

References

[1] S. Omohundro, "Efficient algorithms with neural network behavior," *Complex Systems*, vol. 1, p. 273, 1987.

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

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

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

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

[6] 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.

[7] J. Moody and C. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, vol. 1, no. 2, pp. 281–294, 1989.

[8] T. Poggio, "On optimal nonlinear associative recall," *Biological Cybernetics*, vol. 19, pp. 201–209, 1975.

[9] L. G. Valiant, "A theory of learnable," *Proc. of the 1984 STOC*, pp. 436–445, 1984.

[10] J. R. Rice, *The Approximation of Functions*, Vol. 1. Addison-Wesley, Reading, MA, 1964.

[11] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," A.I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

[12] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing*, ch. 8, pp. 318–362. MIT Press, Cambridge, MA, 1986.

[13] G. Cybenko, "Approximation by superposition of a sigmoidal function," *Math. Control Systems Signals*, in press, 1989.

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

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

[16] B. Moore and T. Poggio, "Representations properties of multilayer feedforward networks," in *Abstracts of the First Annual INNS Meeting*, p. 502, New York, 1988, Pergamon Press.

[17] 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*, vol. 114, pp. 953–956, 1957.

[18] T. Poggio, "Visual algorithms," in O. J. Braddick and A. C. Sleigh, editors, *Physical and Biological Processing of Images*, pp. 128–153, Springer-Verlag, Berlin, 1982.

[19] F. Girosi and T. Poggio, "Representation properties of networks: Kolmogorov's theorem is irrelevant," *Neural Computation*, vol. 1, no. 4, pp. 465–469, 1989.

[20] J. Hadamard, *La theorie des equations aux derivees partielles*. Editions Scientifiques, Pekin, 1964.

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

[22] C. J. Stone, "Optimal global rates of convergence for nonparametric regression," *Ann. Stat.*, vol. 10, pp. 1040–1053, 1982.

[23] M. Bertero, "Regularizations methods for linear inverse problems," in C. G. Talenti, editor, *Inverse Problems*. Springer-Verlag, Berlin, 1986.

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

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

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

[27] J. Rissanen, "Modeling by shortest data description," *Automatica*, vol. 14, pp. 465–471, 1978.

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

[29] 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.

[30] 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.

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

[32] R. Franke, "Scattered data interpolation: Tests of some method," *Math. Comp.*, vol. 38(5), pp. 181–200, 1982.

[33] E. J. Kansa, "Multiquadrics—a scattered data approximation scheme with applications to computational fluid dynamics—I," *Computers Math. Appl.*, vol. 19, no. 8/9, pp. 127–145, 1990.

[34] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," *Complex Systems*, vol. 2, pp. 321–355, 1988.

[35] S. Renals and R. Rohwer, "Phoneme classification experiments using radial basis functions," in *Proceedings of the International Joint Conference on Neural Networks*, pp. I-461–

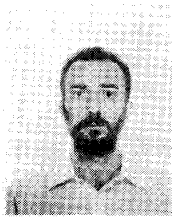
I-467, Washington, D.C., June 1989, IEEE TAB Neural Network Committee.
[36] M. Casdagli, "Nonlinear prediction of chaotic time-series," *Physica D*, vol. 35, pp. 335-356, 1989.
[37] A. N. Tikhonov, "Solution of incorrectly formulated problems and the regularization method," *Soviet Math. Dokl.*, vol. 4, pp. 1035-1038, 1963.
[38] V. A. Morozov, *Methods for Solving Incorrectly Posed Problems*. Springer-Verlag, Berlin, 1984.
[39] R. Courant and D. Hilbert, *Methods of Mathematical Physics*, Vol. 2. Interscience, London, England, 1962.
[40] G. Wahba, *Splines Models for Observational Data*, Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.
[41] C. A. Micchelli, "Interpolation of scattered data: Distance matrices and conditionally positive definite functions," *Constr. Approx.*, vol. 2, pp. 11-22, 1986.
[42] N. Dyn, "Interpolation of scattered data by radial functions," in C. K. Chui, L. L. Schumaker, and F. I. Utreras, editors, *Topics in Multivariate Approximation*. Academic Press, New York, 1987.
[43] W. R. Madych and S. A. Nelson, "Multivariate interpolation and conditionally positive definite functions, II," *Mathematics of Computation*, vol. 54, no. 189, pp. 211-230, Jan. 1990.
[44] J. Duchon, "Spline minimizing rotation-invariant semi-norms in Sobolev spaces," in W. Schempp and K. Zeller, editors, *Constructive Theory of Functions of Several Variables*, Lecture Notes in Mathematics, 571. Springer-Verlag, Berlin, 1977.
[45] J. Meinguet, "Multivariate interpolation at arbitrary points made simple," *J. Appl. Math. Phys.*, vol. 30, pp. 292-304, 1979.
[46] R. L. Harder and R. M. Desmarais, "Interpolation using surface splines," *J. Aircraft*, vol. 9, pp. 189-191, 1972.
[47] S. Bochner, "Vorlesungen ueber Fouriersche Integrale," in *Akademische Verlagsgesellschaft*, Leipzig, 1932.
[48] A. Yuille and N. Grzywacz, "The motion coherence theory," in *Proceedings of the International Conference on Computer Vision*, pp. 344-354, Washington, D.C., December 1988, IEEE Computer Society Press.
[49] I. Stakgold, *Green's Functions and Boundary Problems*. John Wiley and Sons, New York, 1979.
[50] F. Girosi and T. Poggio, "Networks and the best approximation property," *Biological Cybernetics*, vol. 63, pp. 169-176, 1990.
[51] M. Stinchcombe and H. White, "Universal approximation using feedforward networks with non-sigmoid hidden layer activation functions," in *Proceedings of the International Joint Conference on Neural Networks*, pp. I-607-I-611, Washington, D.C., June 1989, IEEE TAB Neural Network Committee.
[52] H. L. Resnikoff, "On the psychophysical function," *J. Math. Biol.*, vol. 2, pp. 265-276, 1975.
[53] J. Demmel, "The geometry of ill-conditioning," *J. Complexity*, vol. 3, pp. 201-229, 1987.
[54] S. G. Mikhlin, *The Problem of the Minimum of a Quadratic Functional*. Holden-Day, San Francisco, CA, 1965.
[55] D. Geiger and F. Girosi, "Parallel and deterministic algorithms for MRFs: surface reconstruction and integration," in *Lecture Notes in Computer Science*, Vol. 427: Computer Vision—ECCV 90, O. Faugeras, Ed. Berlin: Springer-Verlag, 1990.
[56] F. Girosi, T. Poggio, and B. Caprile, "Extensions of a theory of networks for approximation and learning: Outliers and negative examples," A.I. Memo 1220, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.
[57] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 219-227, 1983.
[58] A. Albert, *Regression and the Moore-Penrose Pseudoinverse*. Academic Press, New York, 1972.
[59] T. Poggio and F. Girosi, "Extension of a theory of networks for approximation and learning: Dimensionality reduction and clustering," A.I. Memo 1167, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.
[60] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Wiley, New York, 1973.
[61] T. Poggio and S. Edelman, "A network that learns to recognize 3D objects," *Nature*, vol. 343, pp. 263-266, 1990.
[62] S. Edelman and T. Poggio, "Bringing the grandmother back into the picture: A memory-based view of object recognition," A.I. Memo 1181, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1990.
[63] 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.
[64] 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.
[65] T. Flash and N. Hogan, "The coordination of arm movements: An experiment confirmed mathematical model," *The Journal of Neuroscience*, vol. 5(7), pp. 1688-1703, 1985.
[66] P. Kanerva, *Sparse Distributed Memory*, MIT Press, Cambridge, MA, 1988.
[67] D. Marr, "A theory of cerebellar cortex," *J. Physiology*, vol. 202, pp. 437-470, 1969.
[68] J. S. Albus, "A theory of cerebellar functions," *Math. Bio.*, vol. 10, pp. 25-61, 1971.
[69] J. D. Keeler, "Comparison between Kanerva's SDM and Hopfield-type neural networks," *Cognitive Science*, vol. 12, pp. 299-329, 1988.
[70] T. Kohonen, "Self organized formation of topologically correct feature maps in the brain," *Biological Cybernetics*, vol. 43, pp. 59-69, 1982.
[71] 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.*, p. 281. U. California Press, Berkeley, CA, 1967.
[72] C. Stanfill and D. Waltz, "Towards memory-based reasoning," *Comm. of the ACM*, vol. 29, pp. 1213-1228, 1986.
[73] G. G. Lorentz, *Approximation of Functions*. Chelsea Publishing Co., New York, 1986.
[74] C. J. Stone, "Additive regression and other nonparametric models," *Ann. Stat.*, vol. 13, pp. 689-705, 1985.
[75] E. B. Baum, "On the capabilities of multilayer perceptrons," *J. Complexity*, vol. 4, pp. 193-215, 1988.
[76] E. B. Baum and D. Haussler, "What size net gives valid generalization?" in D. S. Touretzky, editor, *Advances in Neural Information Processing Systems I*, pp. 81-90. Morgan Kaufmann Publishers, Carnegie Mellon University, 1989.
[77] F. Girosi and T. Poggio, "Networks for learning: A view from the theory of approximation of functions," in *Proc. of the Genoa Summer School on Neural Networks and Their Applications*, Genoa, Italy, June 1989, Prentice-Hall (in press).
[78] G. Wahba, "Practical approximate solutions to linear operator equations when the data are noisy," *SIAM J. Numer. Anal.*, p. 14, 1977.
[79] 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*, vol. 31, pp. 377-403, 1979.
[80] I. M. Gelfand and N. Ya. Vilenkin, *Generalized Functions. Vol. 4: Applications of Harmonic Analysis*, Academic Press, New York, 1964.
[81] I. J. Schoenberg, "Metric spaces and positive definite function," *Ann. of Math.*, vol. 44, pp. 522-536, 1938.
[82] J. Stewart, "Positive definite functions and generalizations, an historical survey," *Rocky Mountain J. Math.*, vol. 6, pp. 409-434, 1976.

Tomaso A. Poggio (Associate Member, IEEE) was born in Geona, Italy, in 1947. He received the Ph.D. degree in theoretical physics from the University of Genoa in 1970.

From 1971 to 1982, he was Wissenschaftlicher Assistant at the Max Planck Institut für Biologische Kybernetic in Tübingen, West Germany. Since 1982, he has been a Professor at the Artificial Intelligence Laboratory at the Massachusetts

Institute of Technology (MIT) in Cambridge, MA. In 1984, he was additionally appointed Professor at MIT's Whitaker College of Health Sciences and Technology, and was named the first director of the Center for Biological Information Processing. In 1988, he was named to the Uncas and Helen Whitaker Professorship.

Professor Poggio has authored papers in areas ranging from psychophysics and biophysics to information processing in man and machine, artificial intelligence, and machine vision. He is on the editorial boards of several interdisciplinary journals, and is a Corporate Fellow of Thinking Machines Corporation.



Federico Girosi was born in Alassio, Italy, on February 27, 1962. He received the Ph.D degree in theoretical physics from the University of Genoa in 1986.

From 1986 to 1988 he worked on various topics in computer vision at the University of Genoa, focusing his attention on optical flow computation and interpretation. Beginning in 1988, he was awarded a Fairchild fellowship at the Brain Sciences Department and the Artificial Intelligence Laboratory, Massachusetts Institute of Technology (MIT), Cambridge, MA, where he works with Prof. Tomaso Poggio on problems related to machine learning, networks, and approximation theory.