January 1996

Smoothing regularizers for projective basis function networks

John Moody

Thorsteinn S. Ragnvaldson

OGI Technical Report CSE 96-002

Smoothing Regularizers for Projective Basis Function Networks

John E. Moody and **Thorsteinn S. Rögnvaldsson**
Department of Computer Science, Oregon Graduate Institute
PO Box 91000, Portland, OR 97291
moody@cse.ogi.edu denni@cse.ogi.edu

Abstract

Smoothing regularizers for radial basis functions have been studied extensively, but no general smoothing regularizers for *projective basis functions* (PBFs), such as the widely-used sigmoidal PBFs, have heretofore been proposed. We derive new classes of algebraically-simple m^{th}-order smoothing regularizers for networks of projective basis functions $f(W,x) = \sum_{j=1}^{N} u_j g\left[x^T v_j + v_{j0}\right] + u_0$, with general transfer functions $g[\cdot]$. These regularizers are:

$$R_G(W,m) = \sum_{j=1}^{N} u_j^2 \|v_j\|^{2m-1} \quad \text{Global Form}$$

$$R_L(W,m) = \sum_{j=1}^{N} u_j^2 \|v_j\|^{2m} \quad \text{Local Form} \quad .$$

With appropriate constant factors, these regularizers bound the corresponding m^{th}-order smoothing integral

$$S(W,m) = \int d^D x \Omega(x) \left\| \frac{\partial^m f(W,x)}{\partial x^m} \right\|^2 \quad .$$

In the above expressions, W denotes all the network weights $\{u_j, u_0, v_j, v_0\}$, and $\Omega(x)$ is a weighting function (not necessarily the input density) on the D-dimensional input space. The global and local cases are distinguished by different choices of $\Omega(x)$.

These simple algebraic forms $R(W,m)$ enable the direct enforcement of smoothness without the need for costly Monte-Carlo integrations of $S(W,m)$. The regularizers are tested on illustrative sample problems and compared to quadratic weight decay. The new regularizers are shown to yield better generalization errors than weight decay when the implicit assumptions in the latter are wrong. Unlike weight decay, the new regularizers distinguish between the roles of the input and output weights and capture the interactions between them.

1 Introduction: What are the right biases?

Regularization is a technique for reducing prediction risk by balancing model bias and model variance. A regularizer $R(W)$ imposes prior constraints on the network parameters W. Using squared error as the most common example,

the objective functional that is minimized during training is

$$E = \frac{1}{2M}\sum_{i=1}^{M}[y^{(i)} - f(W, \boldsymbol{x}^{(i)})]^2 + \lambda R(W) \ , \tag{1}$$

where $y^{(i)}$ are target values corresponding to the inputs $\boldsymbol{x}^{(i)}$, M is the number of training patterns, and the regularization parameter λ controls the importance of the prior constraints relative to the fit to the data. Several approaches can be applied to estimate λ (see for example Eubank (1988) or Wahba (1990)) in order to minimize the prediction risk by optimizing the bias/variance tradeoff.

Regularization reduces model variance at the cost of introducing some model bias. An important question arises: What are the right biases? (Geman, Bienenstock & Doursat 1992). A good choice of the regularizer $R(W)$ will result in lower expected prediction error than will a poor choice.

Weight decay is often used effectively, but it is an *ad hoc* technique that controls weight values rather than the fit to the data directly. It is thus not necessarily optimal. Weight decay is not appropriate for arbitrary function parameterizations, since it will give very different results, depending upon whether a function is parameterized, for example, as $f(w, x)$ or as $f(w^{-1}, x)$.

Since many real world problems are intrinsically smooth, we propose that in many cases, an appropriate bias to impose is to favor solutions with low m^{th}-order curvature. Direct penalization of curvature is a parametrization-independent approach. The desired regularizer is the standard D dimensional curvature functional of order m:

$$S(W, m) = \int d^D x \boldsymbol{\Omega}(\boldsymbol{x}) \left\| \frac{\partial^m f(W, \boldsymbol{x})}{\partial \boldsymbol{x}^m} \right\|^2 \ . \tag{2}$$

Here $\| \ \|$ denotes the ordinary euclidean tensor norm and $\partial^m / \partial \boldsymbol{x}^m$ denotes the m^{th} order differential operator. The weighting function $\boldsymbol{\Omega}(\boldsymbol{x})$ ensures that the integral converges and determines the region over which we require the function to be smooth. $\boldsymbol{\Omega}(\boldsymbol{x})$ is not required to be equal to the input density $p(\boldsymbol{x})$, and will most often be different.

The use of smoothing functionals like (2) has been extensively studied for smoothing splines (Eubank 1988, Hastie & Tibshirani 1990, Wahba 1990) and for radial basis function (RBF) networks (Powell 1987, Poggio & Girosi 1990, Girosi, Jones & Poggio 1995). However, no general class smoothing regularizers that directly enforce smoothness $S(W, m)$ for *projective basis functions* (PBFs), such as the widely used sigmoidal PBFs, has been previously proposed.

Since explicit enforcement of smoothness using (2) requires costly, impractical Monte-Carlo integrations,[1] we derive algebraically-simple regularizers $R(W, m)$ that tightly bound $S(W, m)$.

2 Derivation of Simple Regularizers from Smoothing Functionals

We consider single hidden layer networks with D input variables, N_h nonlinear hidden units, and N_o linear output units. For clarity, we set $N_o = 1$, and drop the subscript on N_h (the derivation is trivially extended to the case $N_o > 1$). Thus, our network function is

$$f(\boldsymbol{x}) = \sum_{j=1}^{N} u_j g[\theta_j, \boldsymbol{x}] + u_0 \tag{3}$$

where $g[\cdot]$ are the nonlinear transfer functions of the internal hidden units, $\boldsymbol{x} \in R^D$ is the input vector[2] , θ_j are the parameters associated with internal unit j, and W denotes all parameters in the network.

For regularizers $R(W)$, we will derive *strict upper bounds* for $S(W, m)$. We desire the regularizers to be as general as possible so that they can easily be applied to different network models. Without making any assumptions about $\boldsymbol{\Omega}(\boldsymbol{x})$ or $g(\cdot)$, we have the upper bound

$$S(W, m) \leq N \sum_{j=1}^{N} u_j^2 \int d^D x \boldsymbol{\Omega}(\boldsymbol{x}) \left\| \frac{\partial^m g[\theta_j, \boldsymbol{x}]}{\partial \boldsymbol{x}^m} \right\|^2 , \tag{4}$$

[1] Note that (2) is not just one integral, but actually $\mathcal{O}(D^m)$ integrals, since the norm of the operator $\partial^m / \partial x^m$ has $\mathcal{O}(D^m)$ terms. This is extremely expensive to compute for large D or large m.

[2] Throughout, we use small letter boldface to denote vector quantities.

which follows from the inequality $\left(\sum_{i=1}^{N} a_i\right)^2 \leq N \sum_{i=1}^{N} a_i^2$. We consider two possible options for the weighting function $\Omega(\boldsymbol{x})$. One is to require *global* smoothness, in which case $\Omega(\boldsymbol{x})$ is a very wide function that covers all relevant parts of the input space (e.g. a very wide gaussian distribution or a constant distribution). The other option is to require *local* smoothness, in which case $\Omega(\boldsymbol{x})$ approaches zero outside small regions around some reference points (e.g. the training data).

In a longer paper (Moody & Rögnvaldsson 1996), we consider two general families of transfer functions $g[\cdot]$, namely *projective basis* and *radial basis* representations. In this paper, we focus on projective basis functions.

2.1 Projective Basis Representations

Projective basis functions (PBFs) are of the form $g[\theta_j, \boldsymbol{x}] = g\left[\boldsymbol{x}^T \boldsymbol{v}_j + v_{j0}\right]$, where $\theta_j = \{\boldsymbol{v}_j, v_{j0}\}$, $\boldsymbol{v}_j = (v_{j1}, v_{j2}, \ldots, v_{jD})$ is the vector of weights connecting hidden unit j to the inputs, and v_{j0} is the bias, offset, or threshold.

For PBFs, expression (4) simplifies to

$$S(W,m) \leq N \sum_{j=1}^{N} u_j^2 \|\boldsymbol{v}_j\|^{2m} I_j(W,m), \qquad \text{with} \qquad I_j(W,m) \equiv \int d^D x \Omega(\boldsymbol{x}) \left(\frac{d^m g[z_j(\boldsymbol{x})]}{dz_j^m}\right)^2 \tag{5}$$

and $z_j(\boldsymbol{x}) \equiv \boldsymbol{x}^T \boldsymbol{v}_j + v_{j0}$.

Although the most commonly used $g[\cdot]'$s are sigmoids, our analysis applies to many other forms, for example flexible fourier units, polynomials, and rational functions.[3] The classes of PBF transfer functions $g[\cdot]$ that are applicable (as determined by $\Omega(x)$) are those for which the integral (5) is finite and well-defined.

2.2 Global weighting

For the global case, we select a gaussian form for the weighting function

$$\Omega_G(\boldsymbol{x}) = (\sqrt{2\pi}\sigma)^{-D} \exp\left[\frac{-\|\boldsymbol{x}\|^2}{2\sigma^2}\right] \tag{6}$$

and require σ to be large. The gaussian simplifies evaluation of the smoothing integral considerably, since it is both separable and spherically symmetric. Integrating out all dimensions, except the one associated with the projection vector $\boldsymbol{v}_j$, we are left with

$$I_j(W,m) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} dx e^{-x^2/2\sigma^2} \left(\frac{d^m g[z_j(x)]}{dz_j^m}\right)^2 . \tag{7}$$

If $(d^m g[z]/dz^m)^2$ is integrable and approaches zero outside a region that is small compared to σ, we can accurately bound (7) by setting the exponential in the integrand equal to unity. This implies

$$I_j(W,m) \leq \frac{I(m)}{\|\boldsymbol{v}_j\|} \quad \text{with} \quad I(m) \equiv \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} dz \left(\frac{d^m g[z]}{dz^m}\right)^2 . \tag{8}$$

Defining the global regularizer to be

$$R_G(W,m) \equiv \sum_{j=1}^{N} u_j^2 \|\boldsymbol{v}_j\|^{2m-1} , \tag{9}$$

the bound of equation (5) becomes

$$S(W,m) \leq N I(m) R_G(W,m) , \tag{10}$$

where the subscript G emphasizes the fact that this is an upper bound in the global limit of large σ. Since λ absorbs all constant multiplicative factors, we need only weigh expression (9) into the training objective function.

[3] See for example Moody & Yarvin (1992).

2.2.1 Local weighting

For the local case, we consider weighting functions of the general form

$$\Omega_L(\boldsymbol{x}) = \frac{1}{M}\sum_{i=1}^{M}\Omega(\boldsymbol{x}^{(i)},\sigma) \tag{11}$$

where $\boldsymbol{x}^{(i)}$ are a set of points, and $\Omega(\boldsymbol{x}^{(i)},\sigma)$ is a function that decays rapidly for large $\|\boldsymbol{x}-\boldsymbol{x}^{(i)}\|$, such as $(\sqrt{2\pi}\sigma)^{-D}\exp\left[-\|\boldsymbol{x}-\boldsymbol{x}^{(i)}\|^2/2\sigma^2\right]$. We require that $\lim_{\sigma\to 0}\Omega(\boldsymbol{x}^{(i)},\sigma) = \delta(\boldsymbol{x}-\boldsymbol{x}^{(i)})$, where $\delta(\cdot)$ is the delta function. Thus, when the $\boldsymbol{x}^{(i)}$ are the training data points, the limiting distribution of (11) is the *empirical distribution*.

In the limit $\sigma \to 0$, equation (5) becomes

$$S(W,m) \le \frac{N}{M}\sum_{j=1}^{N} u_j^2\|\boldsymbol{v}_j\|^{2m}\sum_{i=1}^{M}\left(\frac{d^m g[z_j(\boldsymbol{x}^{(i)})]}{dz_j^m}\right)^2 . \tag{12}$$

In theory, we could compute the expression within parenthesis in (12) for each input pattern $\boldsymbol{x}^{(i)}$ during training and use it as our regularization cost. However, this requires explicit design for each transfer function form and also becomes increasingly complicated as we go to higher m. To construct a simpler and more general form, we instead assume that the m^{th} derivative of the transfer function is bounded and define the constant

$$C_L(m) \equiv \max_z\left(\frac{d^m g[z]}{dz^m}\right)^2 , \tag{13}$$

and the local smoothing regularizer

$$R_L(W,m) \equiv \sum_{j=1}^{N} u_j^2\|\boldsymbol{v}_j\|^{2m} \quad . \tag{14}$$

This gives the bound

$$S(W,m) \le NC_L(m)R_L(W,m) \tag{15}$$

for the maximum local curvature of the function (the subscript L emphasizes that it is an upper bound in the local limit).

3 Empirical Example

We have done extensive simulation studies that demonstrate the efficacy of our new regularizers for PBF networks on a variety of problems. A full account will be given in Moody & Rögnvaldsson (1996). Here, we demonstrate the value of using smoothing regularizers on a simple problem which illustrates a key difference between smoothing and quadratic weight decay, the two dimensional bilinear function

$$t(x_1,x_2) = x_1x_2. \tag{16}$$

This example was used by Friedman & Stuetzle (1981) to demonstrate projection pursuit regression. It is the simplest example of a function that has interactions between the input variables. The function can be well-fitted by a one hidden layer network with four sigmoidal hidden units by expressing the function in the form $t(x_1,x_2) = 0.5(x_1+x_2)^2 - 0.5(x_1-x_2)^2$ and approximating the quadratic functions with a superposition of two sigmoids.

We have run nine different versions of this experiment with training sets of sizes $M \in \{20, 40, 100\}$, randomly sampled from the space $-1 \le \{x_1, x_2\} \le 1$, and additive gaussian noise with standard deviation $s \in \{0.1, 0.2, 0.5\}$, which corresponds to signal-to-noise ratios (SNR) of $\{3.33, 1.67, 0.67\}$. The student networks have 8 hidden $\tanh[\cdot]$ units and one linear output. Figure 1 illustrates, for the special case of $s = 0.2$ and a training set with 40 data points, how the generalization performance improves when higher order smoothing regularizers ($m = 2$ and $m = 3$) are used instead of weight decay or first order smoothers, which yield inferior solutions.

Over the nine sample size and SNR cases, we find that both the global and local smoothing regularizers with ($m = 2$ and $m = 3$) outperform weight decay (whether bias weights are included or not), that the local $m = 1$ case performs similarly to weight decay. Only in the global $m = 1$ case does weight decay have the edge. This is not surprising, though, since the target function is quadratic. Weight decay performs poorly relative to the $m = 2$ and $m = 3$ smoothing regularizers, because it lacks any form of interaction between the input layer and output layer weights $\boldsymbol{v}_j$ and u_j.

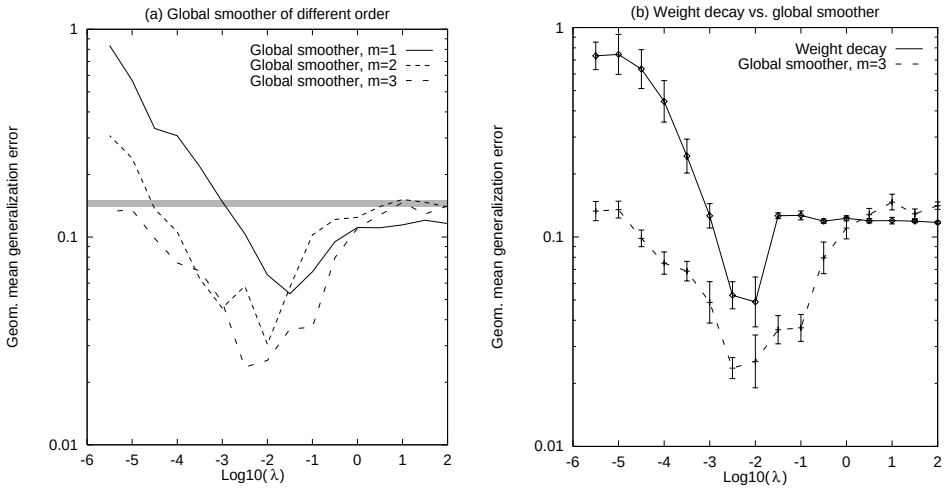

Figure 1: (**a**) Generalization errors on the x_1x_2 problem, with 40 training data points and a signal-to-noise ratio of 2/3, for different values of the regularization parameter and different orders of the smoothing regularizer. For each value of λ, 10 networks have been trained and averaged (geometric average). The best generalization error decreases with increasing order of the smoothing regularizer. The shaded area shows the 95% confidence bands for the average performance of a linear model on the same problem. (**b**) Similar plot for the $m = 3$ smoother compared to the standard weight decay method. Error bars mark the estimated standard deviation of the mean generalization error of the 10 networks.

4 Quality of the Regularizers: Approximations *vs* Bounds

With appropriate multiplicative factors, eqs. (9) and (14), are strict upper bounds to the smoothness functional $S(W, m)$, eq. (2), in the global and local limits, $\sigma \to \infty$ and $\sigma \to 0$. However, questions arise as to how tight the bounds are and how well the regularizers $R(W, m)$ track the curvature functional $S(W, m)$. If the bounds are not sufficiently tight, then penalizing $R(W, m)$ might not have the effect of penalizing $S(W, m)$.

In this section, we present approximations to $S(W, m)$ that are proportional to the bounds, and present empirical results that demonstrate that penalizing $R(W, m)$ does in fact have the effect of penalizing $S(W, m)$ for networks of PBFs. Note that for the proposed regularizers $R(W, m)$ to be effective in penalizing $S(W, m)$, we need only have an approximate monotonic relationship between them. In fact, we argue and demonstrate empirically that an approximate linear relationship between R and S holds.

We first observe that the bound (4) should not introduce significant error for problems in which the m^{th} derivatives of the internal unit activities are *uncorrelated*. Under the uncorrelated internal unit assumption, therefore, the bounds of equations (10) and (15) can be replaced by the approximations:

$$S_G(W, m) \approx I_G(m) R_G(W, m) \tag{17}$$

$$S_L(W, m) \approx C_L(m) R_L(W, m) \ , \tag{18}$$

using $\left(\sum_{i=1}^{N} a_i\right)^2 \approx \sum_{i=1}^{N} a_i^2$. Note that the right hand sides differ from those in equations (10) and (15) only by a factor of N, so these approximations are proportional to the bounds.

For our regularizers, the constant factor N doesn't matter, since it can be absorbed into the regularization parameter λ (along with the values of the factors $I_G(m)$ or $C_L(m)$). Since λ is selected on a case by case basis anyway, such constant factors are irrelevant. In practical terms then, there is no difference between using the upper bounds (10) and (15) or the uncorrelated approximations (17) and (18).

Our empirical results presented below indicate the uncorrelated hidden unit assumption yields a good approximation for both the global and the local cases, especially when the dimensionality of the input space gets large. The probability of having significant correlation between two internal units decreases exponentially with the input space dimension, and is very small already for moderate numbers of variables. Furthermore, even in low dimensions, the possible positive

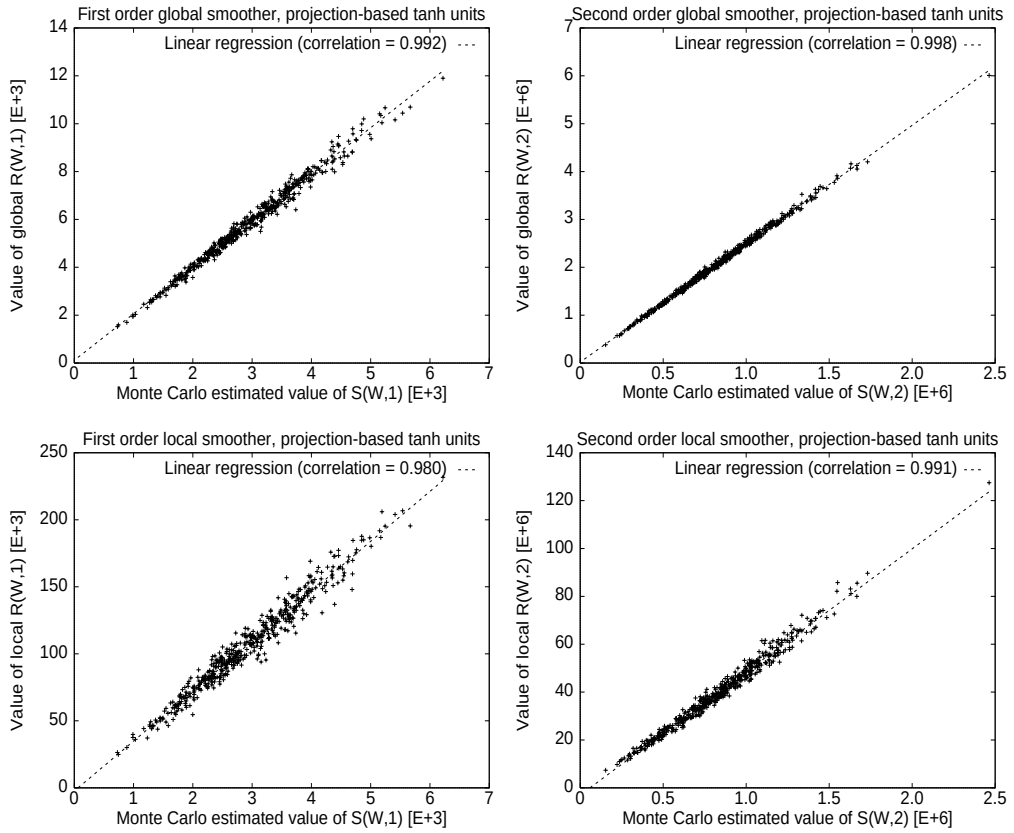

Figure 2: Linear correlation between $S(W, m)$ and $R(W, m)$ for a neural network with 10 input units, 10 internal tanh$[\cdot]$ PBF units, and one linear output. The left column shows results for first order smoothing $(m = 1)$ and the right column shows results for second order smoothing $(m = 2)$. The top row shows the global form of the regularizer $R_G(W, m)$ and the bottom row shows results for the local form $R_L(W, m)$. Note that the correlation coefficients are very close to 1.0, confirming that penalizing $R(W, m)$ effectively penalizes the curvature functional $S(W, m)$.

overlap between internal units decreases for many transfer functions (e.g. sigmoids) with increasing order of the derivative. The accuracies of the approximation thus improves with increasing input dimension, and with increasing m. This is a very nice effect, since many real problems deal with many (10 or more) input variables.

4.1 Empirical Comparisons of $R(W, m)$ vs $S(W, m)$

For the regularizers $R(W, m)$ to be effective in penalizing $S(W, m)$, an approximate monotonically-increasing relationship must hold between them. The uncorrelated internal unit assumption implies that this relationship is linear.

To test for such a linear scaling, we generated a large number of randomly selected networks. For each such network, we computed the values of $R(W, m)$ and performed Monte Carlo integrations to compute $S(W, m)$. For each experiment, we fit a linear model $R(W, m) = \alpha + \beta S(W, m)$ to the data, estimating the parameters α and β. The accuracy of the linear scaling is measured by the linear correlation $\langle RS \rangle / \sqrt{(\langle R^2 \rangle \langle S^2 \rangle)}$. Under the assumption of a linear relationship, the quality of the regularizers can thus be measured. If the linear correlation is high, using the regularizer $R(W, m)$ effectively penalizes the smoothing functional $S(W, m)$.

Figure 2 shows the correlation between the value of the true functional (2) and our regularizers for networks with 10 input units (D) and 10 internal units (N). The value of $S(W, m)$ is estimated through Monte Carlo integration. That is, we sample 10^5 input data patterns from a gaussian distribution with zero mean and unit variance, and replace the

integration with a summation over these points. This is repeated 500 times, picking new random weights each time but keeping the network architecture constant.

The correlation is very high for both the global and local forms, although the global form is slightly better. To verify that this finding is not spurious, we repeat our Monte Carlo simulations for many different network architectures with varying D and N, using the same method for sampling weights. These results (presented in Moody & Rögnvaldsson (1996)) show that the same conclusions hold as the number of hidden or input units increase or decrease. As anticipated, the regularizers are better estimates of $S(W, m)$ when the number of inputs grows or when the order m is increased.

5 Discussion

Our regularizers $R(W, m)$ are the first general class of m^{th}–order smoothing regularizers to be proposed for projective basis function networks. The forms of the regularizers differ fundamentally from quadratic weight decay, in that they distinguish the roles of the input weights $\boldsymbol{v}_j$ and output weights u_j, and capture the interactions between them. Our regularizers apply to PBFs with large classes of transfer functions $g[\cdot]$, including sigmoids.

Our approach differs from that developed for smoothing splines and smoothing radial basis functions, in that we derive smoothing regularizers for given classes of units $g[\theta, x]$, rather than derive the forms of the units $g[\cdot]$ by requiring them to be Greens functions of the smoothing operator $S(\cdot)$. Our approach thus has the advantage that it can be applied to the types of networks most often used in practice.

In a longer paper (Moody & Rögnvaldsson 1996), we present the application of our approach to radial basis function networks and present extensive simulation results for both PBFs and RBFs.

Acknowledgements

Both authors thank Steve Rehfuss and Dr. Lizhong Wu for stimulating input. John Moody thanks Volker Tresp for a provocative discussion at a 1991 Neural Networks Workshop sponsored by the Deutsche Informatik Akademie. We gratefully acknowledge support for this work from ARPA and ONR (grant N00014-92-J-4062), NSF (grant CDA-9503968), the Swedish Institute, and the Swedish Research Council for Engineering Sciences (contract TFR-282-95-847).

References

Eubank, R. L. (1988), *Spline Smoothing and Nonparametric Regression*, Marcel Dekker, Inc.

Friedman, J. H. & Stuetzle, W. (1981), 'Projection pursuit regression', *J. Amer. Stat. Assoc.* **76**(376), 817–823.

Geman, S., Bienenstock, E. & Doursat, R. (1992), 'Neural networks and the bias/variance dilemma', *Neural Computation* **4**(1), 1–58.

Girosi, F., Jones, M. & Poggio, T. (1995), 'Regularization theory and neural networks architectures', *Neural Computation* **7**, 219–269.

Hastie, T. J. & Tibshirani, R. J. (1990), *Generalized Additive Models*, Vol. 43 of *Monographs on Statistics and Applied Probability*, Chapman and Hall.

Moody, J. E. & Yarvin, N. (1992), Networks with learned unit response functions, *in* J. E. Moody, S. J. Hanson & R. P. Lippmann, eds, 'Advances in Neural Information Processing Systems 4', Morgan Kaufmann Publishers, San Mateo, CA, pp. 1048–55.

Moody, J. & Rögnvaldsson, T. (1996), Smoothing regularizers for projective and radial basis function networks, Manuscript in preparation.

Poggio, T. & Girosi, F. (1990), 'Networks for approximation and learning', *IEEE Proceedings* **78**(9).

Powell, M. (1987), Radial basis functions for multivariable interpolation: a review., *in* J. Mason & M. Cox, eds, 'Algorithms for Approximation', Clarendon Press, Oxford.

Wahba, G. (1990), *Spline models for observational data*, CBMS-NSF Regional Conference Series in Applied Mathematics.