TR-H-209

Receptive Field Weighted Regression.

Stefan SCHAAL and Christopher G. ATKESON
(Georgia Inst. Tech.)

1997.1.29

ATR人間情報通信研究所

〒619-02　京都府相楽群精華町光台2-2　TEL: 0774-95-1011

ATR Human Information Processing Research Laboratories

2-2, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-02, Japan
Telephone: +81-774-95-1011
Fax　　　: +81-774-95-1008

Schaal, S. & Atkeson, C. G. (1997). "Receptive Field Weighted Regression." Technical Report TR-H-209, ATR Human Information Processing Laboratories, 2-2 Hikaridai, Seika-cho, Soraku-gu, Kyoto 619-02, Japan.

Receptive Field Weighted Regression

Stefan Schaal‡*

sschaal@cc.gatech.edu
http://www.cc.gatech.edu/fac/Stefan.Schaal

Christopher G. Atkeson‡

cga@cc.gatech.edu
http://www.cc.gatech.edu/fac/Chris.Atkeson

‡College of Computing, Georgia Institute of Technology, Atlanta, GA 30332
*ATR Human Information Processing Laboratories, 2-2 Hikaridai, Seika-cho, Soraku-gun, 619-02 Kyoto

Abstract

We introduce a constructive, incremental learning system for regression problems that models data by means of spatially localized linear models. In contrast to other approaches, the size and shape of the receptive field of each locally linear model as well as the parameters of the locally linear model itself are learned independently, i.e., without the need for competition or any other kind of communication. This characteristic is accomplished by incrementally minimizing a weighted penalized local cross validation error. As a result, we obtain a learning system that can allocate resources as needed while dealing with the bias-variance dilemma in a principled way. The spatial localization of the linear models increases robustness towards negative interference. Our learning system can be interpreted as a nonparametric adaptive bandwidth smoother, as a mixture of experts where the experts are trained in isolation, and as a learning system which profits from combining independent expert knowledge on the same problem. It illustrates the potential learning capabilities of purely local learning and offers an interesting and powerful approach to learning with receptive fields.

1 Introduction

Learning with spatially localized basis functions has become a popular paradigm in machine learning and neurobiological modeling. In the context of radial basis function networks (Moody & Darken, 1988; Poggio & Girosi, 1990), it was demonstrated that these learning methods offer an alternative to learning with global basis functions, such as sigmoidal neural networks, and that their theoretical foundation can be solidly grounded in approximation theory (Powell, 1987). In neurophysiological studies, the concept of localized information processing in the form of receptive fields has been known since at least the work of Mountcastle (1957) and Hubel and Wiesel (1959). Since then, a wealth of experimental evidence has been accumulated which suggests that information processing based on local receptive fields is a ubiquitous organizational principle in neurobiology that offers interesting computational opportunities (e.g., Zipser & Anderson, 1988; Lee, Rohrer, & Sparks, 1988; Georgopoulos, 1991; Field, 1994; Olshausen & Field, 1996; Daugman & Downing, 1995).

In this paper we explore the computational power of local, receptive field-based incremental learning with the goal of approximating unknown functional relationships between an incoming stream of input and output data. By incremental learning we do not just mean that the parameters of the learning system are updated incrementally. We want to address a learning scenario in which after a new data point is incorporated in the learning system it is discarded and cannot be re-used, in which input and output distributions of the data are unknown, and in which these distribution may change over time. This situation resembles the learning of sensory and sensorimotor transformations in biology, and it also applies to a variety of artificial domains, ranging from autonomous robotic systems to process control.

Given these constraints on incremental learning, two major problems need to be addressed. The first one is how to allocate the appropriate number of resources, e.g., receptive fields, in order to deal with the tradeoff between overfitting and oversmoothing, called the bias-variance dilemma (e.g., Geman, Bienenstock, & Doursat, 1992). The second problem of incremental learning comes from negative interference, the forgetting of useful knowledge while focusing on learning from new data. Methods to prevent these undesirable effects require either validation data sets, memorizing of all training data, or strong prior knowledge about the learning problem. However, none of these alternatives are available in the setting we have described as we want to avoid storing data and do not have knowledge about the structure of the learning task.

In order to address the problems of incremental learning, we will make use of nonparametric regression statistics (e.g., Scott, 1992; Hastie & Tibshirani, 1990). Nearest neighbor algorithms for pattern recognition and Parzen windows for density estimation are the best known methods out of this field (e.g., Duda & Hart, 1973). It is interesting to note that many nonparametric regression methods are essentially receptive field-based: predictions are made from data out of a restricted local neighborhood around the query point. The size of the neighborhood can be irregular, as typically is the case in nearest neighbor approaches, or it can be a symmetric bell-shaped weighting function as in Parzen windows. Receptive fields in nonparametric regression are most often built on the fly, and they are discarded right after the prediction—a paradigm that has been termed lazy learning (Aha, in press). Necessarily, such nonparametric methods need to store training data. Another important characteristic is that predictions made are usually based on a single receptive field. Early on, this inspired the field of nonparametric regression to pursue more complex models in a receptive field, for instance, low order polynomials (e.g., Cleveland, 1979; Cleveland & Loader, 1995), while many neural network learning algorithms focused on combining the activation strengths of many receptive fields to optimize predictions, as in radial basis function networks.

In this paper we will demonstrate how a nonparametric regression approach can be used to build a receptive field-based learning system for incremental function approximation without the need to store the training and without discarding receptive fields after using them. A locally linear model will be fitted incrementally within each receptive field such that local function approximation is accomplished in the spirit of a Taylor series expansion. The new property of this learning approach is that each receptive field is trained entirely independently of all other receptive fields, whereby it adjusts the parameters of its locally linear model, the size and shape of its receptive field, as well as the bias on the relevance on its individual input dimensions. New receptive fields are allocated as needed. The resulting algorithm, Receptive Field Weighted Regression (RFWR), achieves robust incremental learning. It also has some interesting relations to previously suggested learning methods. It can be interpreted as a mixture of experts system (Jacobs, Jordan, Nowlan, & Hinton, 1991; Jordan & Jacobs, 1994) where the experts are trained in isolation. It can also be interpreted as system where a set of experts is trained independently on the same problem, and which profits from combining these experts for making predictions (e.g., Perrone & Cooper, 1993). And finally, RFWR can be interpreted as a nonparametric memory-based learner (Atkeson, Moore, & Schaal, in press) which only stores data that are surprising.

In the following section, we will first give some motivation of how we are going to attack the problems of incremental learning. In Section 3, we describe the details of our nonparametric incremental learning system. Section 4 provides a theoretical assessment of the statistical characteristics of our learning method, and Section 5 gives a variety of empirical evaluations.

2 Incremental Learning

2.1 Statistical Assumptions

The assumed underlying statistical model of our problems is the standard regression model:

$$\mathbf{y} = f(\mathbf{x}) + \varepsilon \tag{1}$$

where $\mathbf{x} \in \Re^n$ denotes the n-dimensional vector of input variables, $\mathbf{y} \in \Re^m$ the m-dimensional vector of output variables, and $f(\cdot)$ a deterministic vector valued function mapping the input $\mathbf{x}$ to the output $\mathbf{y}$. The additive random noise ε is assumed to be independently distributed, $E\{\varepsilon_i \varepsilon_j\} = 0$ for $i \neq j$, and mean zero, $E\{\varepsilon \mid \mathbf{x}\} = 0$, but otherwise of unknown distribution ($E\{\cdot\}$ denotes the expectation operator). The input data is distributed according to the density $p(\mathbf{x})$.

2.2 Localizing Interference

Interference in learning is a natural side-effect of the ability to generalize, i.e., to interpolate or extrapolate an output for an unseen input from previously learned data. Generalization is accomplished by allowing changes to the parameters of the learning system to have non-local effects. If these effects reduce the overall correctness of predictions to a larger extent than they improve them, interference is called negative or even catastrophic. Incremental learning is particularly endangered by negative interference because there is no direct way to balance the amount of positive interference (i.e., generalization) with the amount of negative interference: any parameter update is usually greedy; its only concern is with the reduction of the error of the current piece of training data. To see the statistical causes of interference, consider using the mean squared error criterion J to select a model $\hat{f}(\cdot)$ to approximate the true function $f(\cdot)$:

$$J = E\left\{\left\|\mathbf{y} - \hat{f}(\mathbf{x})\right\|^2\right\} = \int_{-\infty}^{+\infty} \left\|\mathbf{y} - \hat{f}(\mathbf{x})\right\|^2 p(\mathbf{x},\mathbf{y})\, d\mathbf{x}\, d\mathbf{y} = \int_{-\infty}^{+\infty} \left\|\mathbf{y} - \hat{f}(\mathbf{x})\right\|^2 p(\mathbf{y}|\mathbf{x}) p(\mathbf{x})\, d\mathbf{x}\, d\mathbf{y} \tag{2}$$

If there is an infinite amount of training data, the result for $\hat{f}(\cdot)$ will asymptotically only depend on the conditional distribution $p(\mathbf{y} \mid \mathbf{x})$ (Papoulis, 1991):

$$\hat{f}(\mathbf{x}) = E\{\mathbf{y}|\mathbf{x}\} = \int_{-\infty}^{+\infty} \mathbf{y}\, p(\mathbf{y}|\mathbf{x})\, d\mathbf{y} \tag{3}$$

For a finite amount of training data, however, the estimate $\hat{f}(\cdot)$ does depend on *both* the conditional distribution $p(\mathbf{y} \mid \mathbf{x})$ and the input distribution $p(\mathbf{x})$ (Fan & Gijbels, 1996). Thus, a stable model $\hat{f}(\cdot)$ can only be obtained if neither of these distributions changes during learning.

These considerations point towards the two major causes for negative interference. If $p(\mathbf{y} \mid \mathbf{x})$ changes, i.e., the functional relationship between $\mathbf{x}$ and $\mathbf{y}$ is non stationary, the parameters in a learning system may have to change. Analogously, if the data for learning are not sampled from a fixed input distribution $p(\mathbf{x})$, the parameters of the learning system may also change. It is particularly a change of the input distribution $p(\mathbf{x})$ which is likely to happen in incremental learning. Imagine a robot learning the dynamics model of its arm, a model which maps joint positions, joint velocities, and joint accelerations to corresponding joint torques. Whenever the robot moves, it will receive valid data about this functional relationship. But, since the robot is fulfilling different tasks at different times, the sampled data will come from quite different input distributions—for example, think of the difference between movements for cooking and movements for playing tennis.

One of the interesting properties of learning with localized receptive fields lies in their potential robustness towards interference. If learning is truely spatially localized, i.e., it is guaranteed that an update of the parameters of one receptive field has no effect on the parameters of another receptive field, interference will be spatially localized as well. This is illustrated in the example of Figure 1. Using a synthetic data set suggested by Fan and Gijbels (1995), we trained a 3-layer sigmoidal feedforward neural network (6 hidden units, using backpropagation

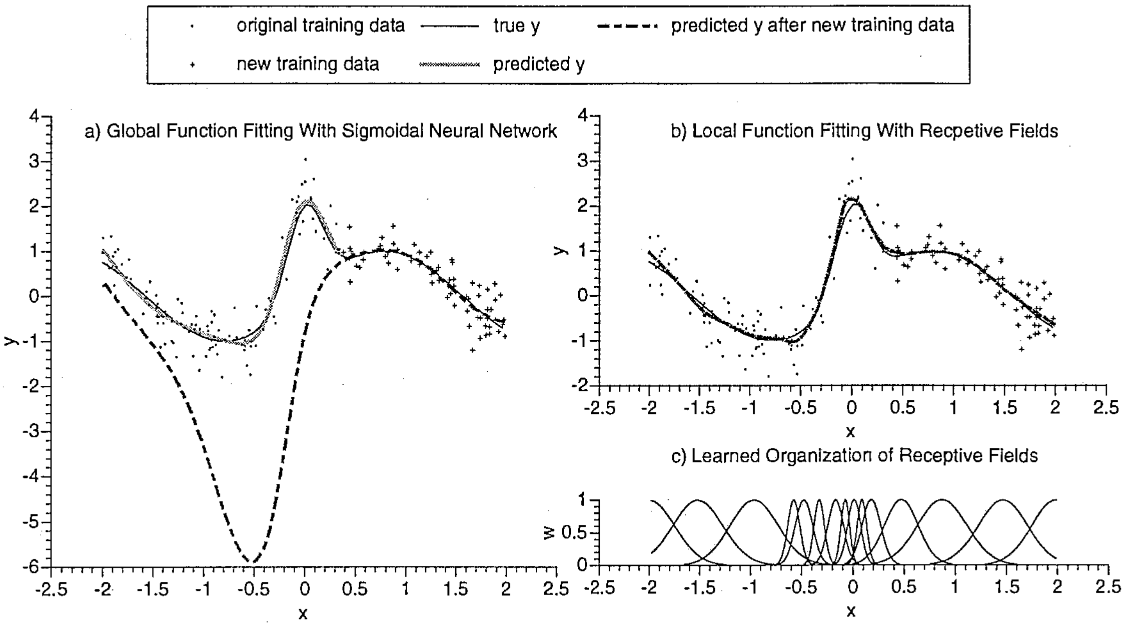

Figure 1: a) Results of function approximation of the function y=sin(2x)+2exp(-16x^2)+N(0,0.16) with a sigmoidal neural network, b) results of function approximation by a local receptive field-based algorithm, fitting locally linear models in each receptive field (note that the data traces "true y", "predicted y", and "predicted y after new training data" largely coincide), c) the organization of the (Gaussian) receptive fields of b) after training.

with momentum) on 130 noisy data points uniformly distributed in $x \in [-2.0, 0.5]$ ("•" in Figure 1). The function fitting result obtained is shown by the "predicted y" trace in Figure 1a. Then we continued training the network on 70 new data points ("+" in Figure 1) drawn from the same function but with a changed input distribution $x \in [0.5, 2.0]$. The network learned to accommodate these new data points, but by doing so, it also significantly changed its predictions for the previously learned data, although this data is largely far away from the new training data. This effect is due to the non-local nature of sigmoidal basis functions, and is prone to lead to catastrophic interference, as shown in Figure 1a.

We repeated the same experiment with our receptive field-based learning system, RFWR, which generates locally linear models in each receptive field (see also Figure 2a) and blends them for predictions (Figure 1b). On the original training data, RFWR achieves comparable result to that of the sigmoidal neural network. After training on the new data, however, no interference is apparent. The original fit in the left part of the graph was not visibly altered, in contrast to the neural network. Looking at the size and distribution of the receptive fields in Figure 1c, it is clear that this learned receptive field structure is unlikely to propagate interference to a large spatial extent. Robustness towards negative interference is accomplished by localizing interference—the best we can do since interference cannot be eliminated for finite data samples.

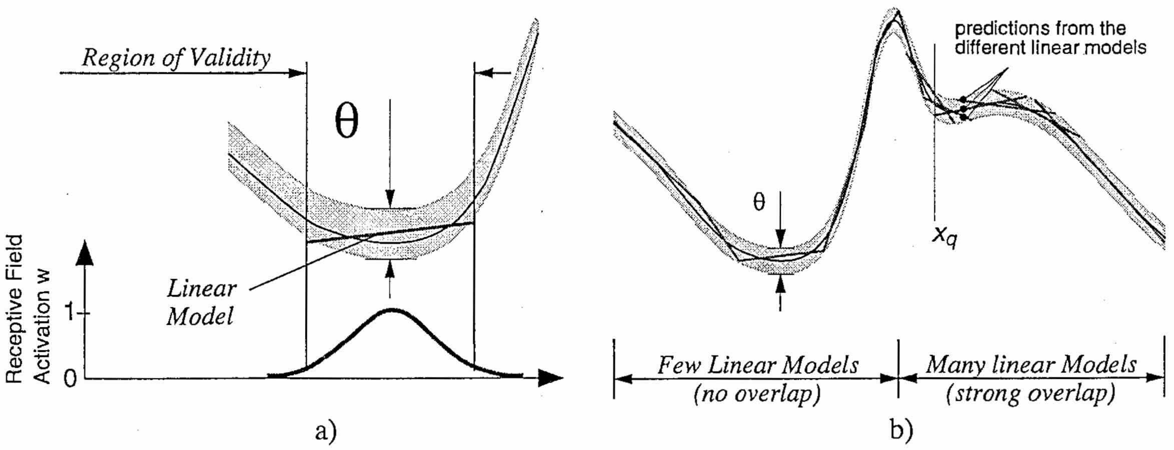
Region of Validity
θ
Linear Model
Receptive Field Activation w
1
0
a)
predictions from the different linear models
θ
x_q
Few Linear Models (no overlap)
Many linear Models (strong overlap)
b)

Figure 2: a) Region of validity of a linear model given a permitted approximation error θ.; b) Function approximation with piecewise linear models.

2.3 Defeating Resource Allocation

Due to the bias-variance tradeoff (Geman et al., 1992), every learning algorithm has to consider a model selection phase in order to find an appropriate compromise between oversmoothing and overfitting. Usually, this is accomplished by setting certain meta parameters, for instance, the number of hidden units in a neural network, according to some model selection criterion, e.g., cross validation (Stone, 1974). Thus, the question most frequently asked in model selection is: "How many free parameters should be allocated in order to achieve a good bias-variance tradeoff?" However, another approach can be pursued: "Given a *fixed* number of free parameters, how should a given data set be *spatially limited* in order to achieve a good bias-variance tradeoff for the remaining data?"—instead of adapting the complexity of the learning system, one can also adapt the complexity of the region the data is drawn from. For general nonlinear function approximators, it is unclear how to answer this question. For spatially localized function fitting, however, this question translates into: "How should the extent of a receptive field be changed in order to make its associated fixed parametric model fit the data appropriately." Figure 2 illustrates this idea for the case of locally linear models. In the spirit of a Taylor series expansion, let us assume that we know how to learn the region of validity, i.e., the size of the receptive field, of a locally linear model such that its approximation error is at a pre-set value θ (Figure 2a). Note that the approximation error should be larger than the error at the center of the receptive field, and that it exactly matches the pre-set value θ : if the error were less, the receptive field would be expanded, and vice versa. This assures that every receptive field deals with the bias-variance dilemma individually: the bias is prescribed a priori, and the variance follows automatically. In order to approximate the entire nonlinear function, we have to cover the input space with sufficiently many locally linear models. Importantly, it does *not* matter whether we allocate too many locally linear models: an average of the outputs of all linear models at a query point x_q, each with an approximation error of θ, cannot have a larger error than θ. The example in Figure 2b demonstrates this effect: at any query point x_q, the average of the individual predictions of the different linear models must lie inside the θ bound. What is required is that every data point is handled by at least one locally linear model, and, due to averaging, the more overlapping linear models exist, the better a function estimate can be expected. Hence, this procedure is capable of addressing the resource allocation problem while *avoiding* the tendency to overfitting as each linear model covers as much space as possible within the θ bound.

2.4 Summary

Given the discussion of the last two sections, a promising route to robust incremental learning seems to be a local receptive field-based system that can also adjust the extent of its receptive fields. However, care must be taken how one goes about accomplishing this goal. Learning methods based on competitive learning cannot achieve the properties described in the previous section. In competitive learning, the size of a receptive field results from a global competition process of all local models to account for the training data. Therefore, changing the number of local models causes a change of the extent of *all* receptive fields and, thus, makes the approximation threshold θ a function of the number of local models, and subsequently, the number of local models a critical choice for the bias-variance tradeoff—exactly what we would like to avoid. The next section will explain how an alternative approach based on nonparametric statistics offers a route to achieve our goals without resorting to competitive learning.

3 Receptive Field Weighted Regression

The goal of RFWR is to construct a system of receptive fields for incremental function approximation. A prediction $\hat{\mathbf{y}}$ for a query point $\mathbf{x}$ is built from the normalized weighted sum of the individual predictions $\hat{\mathbf{y}}_k$ of all receptive fields:

$$\hat{\mathbf{y}} = \frac{\sum_{k=1}^{K} w_k \hat{\mathbf{y}}_k}{\sum_{k=1}^{K} w_k} \tag{4}$$

The weights w_k correspond to the activation strengths of the corresponding receptive fields. They are determined from the size and shape of each receptive field, characterized by a kernel function. A variety of possible kernels have been suggested (e.g., Atkeson et al., in press). Smooth approximations are the easiest accomplished by smooth symmetric bell-shaped kernels. For analytical convenience, we use a Gaussian kernel:

$$w_k = \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{c}_k)^T \mathbf{D}_k (\mathbf{x}-\mathbf{c}_k)\right), \quad \text{where} \quad \mathbf{D}_k = \mathbf{M}_k^T \mathbf{M}_k \tag{5}$$

which parameterizes the receptive field by its location in input space, $\mathbf{c}_k \in \Re^n$, and a positive definite distance metric $\mathbf{D}_k$, determining size and shape of the receptive field. For algorithmic reasons, it is convenient to represent $\mathbf{D}_k$ as an upper diagonal matrix $\mathbf{M}_k$. Any choice of $\mathbf{M}_k$ ensures the positive definiteness of $\mathbf{D}_k$.

Within each receptive field, a simple parametric function models the relationship between input and output data. Local polynomials of low order have found widespread use in nonparametric statistics (Nadaraya, 1964; Watson, 1964; Wahba & Wold, 1975; Cleveland, 1979; Cleveland & Devlin, 1988). We will focus on locally linear models, as they accomplish a favorable compromise between computational complexity and quality of result (Hastie & Loader, 1993):

$$\hat{\mathbf{y}}_k = (\mathbf{x}-\mathbf{c}_k)^T \mathbf{b}_k + b_{0,k} = \tilde{\mathbf{x}}^T \beta_k, \qquad \tilde{\mathbf{x}} = \left((\mathbf{x}-\mathbf{c}_k)^T, 1\right)^T \tag{6}$$

where β_k denotes the parameters of the locally linear model.

To clarify the elements and parameters of RFWR, Figure 3 gives a network-like illustration for a single output system. The inputs are routed to all receptive fields, each of which consists of a linear and a Gaussian unit. The learning algorithm of RFWR determines the parameters $\mathbf{c}_k$, $\mathbf{M}_k$, and β_k for each receptive field *independently*, i.e., without any information about the other receptive fields, in contrast to competitive learning. RFWR adds and prunes receptive fields as needed, such that the number of receptive fields, K, will automatically adjust to the learning problem at hand. A one dimensional example of function fitting with RFWR was already shown in Figure 1b,c. It should be noted that the size of each receptive field adapted according to the local curvature of the function, that there is a certain amount of overlap between the receptive fields, and that the center locations have not been chosen with respect to any explicit optimization criterion.

3.1 Learning With RFWR

Three ingredients of the algorithm need to be discussed: the update of the linear model parameters β_k, the distance metric $\mathbf{M}_k$, and when and where to add and prune receptive fields. The centers $\mathbf{c}_k$ are not changed after they are allocated. For the sake of clarity, we will drop the index k when possible from now on since each receptive field is updated in the same way.

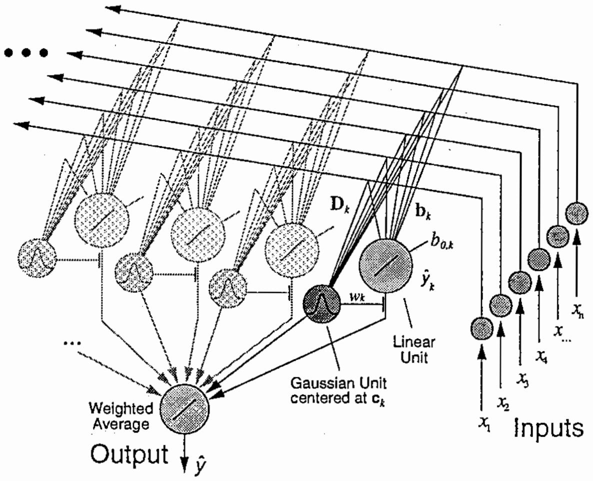

Figure 3: A network illustration of Receptive Field Weighted Regression

3.1.1 *Learning the Linear Model*

Learning of β is straightforward since the problem is linear. It will be useful to leave the incremental learning framework for a moment and think in terms of a batch update. If we summarize the input part of all p training data points in the rows of the matrix $\mathbf{X} = (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_p)^T$, the corresponding output part in the rows of the matrix $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_p)^T$, and the corresponding weights in the diagonal matrix $\mathbf{W} = \mathrm{diag}(\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_\mathrm{p})$, the parameter vector β can be calculated from a weighted regression:

$$\beta = \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{W}\mathbf{Y} = \mathbf{P}\mathbf{X}^T\mathbf{W}\mathbf{Y} \tag{7}$$

This kind of locally weighted regression has found extensive application in nonparametric statistics (Cleveland, 1979; Cleveland & Loader, 1995), in time series prediction (Farmer & Sidorowich, 1987, 1988), and in regression learning problems (Atkeson, 1989; Moore, 1991; Schaal & Atkeson, 1994; Atkeson et al., in press). The result for β in Equation (7) is *exactly* the same when β is calculated by recursive least squares from one sequential sweep through the training data (Ljung & Söderström, 1986). Given a training point $(\mathbf{x}, \mathbf{y})$, the incremental update of β yields:

$$\beta^{n+1} = \beta^n + w\mathbf{P}^{n+1}\tilde{\mathbf{x}}\mathbf{e}_{cv}^T \tag{8}$$

$$\text{where} \quad \mathbf{P}^{n+1} = \frac{1}{\lambda}\left(\mathbf{P}^n - \frac{\mathbf{P}^n\tilde{\mathbf{x}}\tilde{\mathbf{x}}^T\mathbf{P}^n}{\frac{\lambda}{w} + \tilde{\mathbf{x}}^T\mathbf{P}^n\tilde{\mathbf{x}}}\right) \quad \text{and} \quad \mathbf{e}_{cv} = \left(\mathbf{y} - \beta^{n^T}\tilde{\mathbf{x}}\right)$$

This update is employed by RFWR. It is useful to note that recursive least squares corresponds to a Newton training method with guaranteed convergence to the global minimum of, in our case, a weighted squared error criterion (Atkeson et al., in press). Furthermore, the recursive update avoids an explicit matrix inversion. Differing from the batch update in Equation (7), Equation (8) also includes a forgetting factor λ. As the distance metric $\mathbf{M}$ will change during learning (see below), so will the weight w for every data point. For this reason, it is necessary to include λ in (8) in order to gradually cancel the contributions from previous data points where $\mathbf{M}$ was yet not learned properly (Ljung & Söderström, 1986).

3.1.2 Learning the Shape and Size of the Receptive Field

Adjusting the shape and size of the receptive field is accomplished by adjusting the distance metric $\mathbf{M}$. At the first glance, one might hope that this can be done by gradient descent in the weighted mean squared error criterion:

$$J = \frac{1}{W}\sum_{i=1}^{p} w_i \left\| \mathbf{y}_i - \hat{\mathbf{y}}_i \right\|^2 \quad \text{where} \quad W = \sum_{i=1}^{p} w_i \tag{9}$$

which is the basis of the solution of locally weighted regression in Equation (7) (Atkeson et al, in press). Unfortunately, minimizing (9) may result in a quite inappropriate solution. If for each training point one receptive field is centered right on this point, and the corresponding $\mathbf{M}$ is chosen such that the receptive field is so narrow that it is only activated by this data point, the corresponding linear model can fit this one data point with zero error. The function approximation result would strongly tend towards overfitting. It is this property that has made learning algorithms resort to competitive learning with a fixed number of local receptive fields: the global competitive process will prevent receptive fields from modeling just one data point (assuming there are more data points than receptive fields) (e.g., Moody & Darken, 1988; Jordan & Jacobs, 1994). But allowing for such a global competitive process takes away the property of being a local learner, even if the receptive fields are actually spatially localized.

An alternative way to avoid this overfitting effect is to use leave-one-out cross validation. The cost function to be minimized changes from Equation (9) to

$$J = \frac{1}{W}\sum_{i=1}^{p} w_i \left\| \mathbf{y}_i - \hat{\mathbf{y}}_{i,-i} \right\|^2 \tag{10}$$

The notation $\mathbf{y}_{i,-i}$ denotes that the prediction of the i-th data point is calculated from training the learning system with the i-th data point excluded from the training set. Thus, it becomes inappropriate for a receptive field to just focus on one training point since the error measure is calculated from data which did not exist in the training set. Leave-one-out cross validation, however, is usually computationally very expensive since a p-fold training of the learning system is required, for p data points in the training set. Furthermore, for example for a sigmoidal neural network, it might be unclear how to combine the resultant p different solutions to the learning parameters to a final solution. However, for linear regression problems, there is an exception rendering these concerns irrelevant. Due to the Sherman-Morrison-Woodbury Theorem (e.g., Belsley, Kuh, & Welsh, 1980), Equation (10) can be re-written as:

$$J = \frac{1}{W}\sum_{i=1}^{p} w_i \left\|\mathbf{y}_i - \hat{\mathbf{y}}_{i,-i}\right\|^2 = \frac{1}{W}\sum_{i=1}^{p} \frac{w_i \left\|\mathbf{y}_i - \hat{\mathbf{y}}_i\right\|^2}{\left(1 - w_i \tilde{\mathbf{x}}_i^T \mathbf{P} \tilde{\mathbf{x}}_i\right)^2} \tag{11}$$

This equation states that the leave-one-out cross validation error can be obtained without p-fold training of the learning system, but rather by an adjustment of the weighted mean squared error with the help of the inverted covariance matrix **P** (cf. Equation (7)). Equation (11) corresponds to a weighted version of the PRESS residual error in standard linear regression techniques (Myers, 1990). Neglecting for a moment how this cost function can be minimized incrementally, we have obtained a criterion which can be used to adjust **M** (Schaal & Atkeson, 1994).

Unfortunately, there is still a point of concern with Equation (11). Minimizing the locally weighted leave-one-out cross validation error results in a consistent learning system, i.e., with an increasing number of training data, the receptive fields will shrink to a very small size. The advantage of this behavior is that function approximation becomes asymptotically unbiased, i.e., consistent, but as a disadvantage, an ever increasing number of receptive fields will be required to represent the approximated function. This property can be avoided by introducing a penalty term in (11):

$$J = \frac{1}{W}\sum_{i=1}^{p} \frac{w_i \left\|\mathbf{y}_i - \hat{\mathbf{y}}_i\right\|^2}{\left(1 - w_i \tilde{\mathbf{x}}_i^T \mathbf{P} \tilde{\mathbf{x}}_i\right)^2} + \gamma \sum_{i,j=1}^{n} D_{ij}^2 \tag{12}$$

where the scalar γ determines the strength of the penalty. By penalizing the sum of squared coefficients of the distance metric **D**, we are essentially penalizing the second derivatives of the function at the site of a receptive field. This is similar to approaches taken in spline fitting (deBoor, 1978; Wahba, 1990) and acts like a low-pass filter: the higher the second derivatives, the more smoothing (and thus bias) will be introduced locally. Another positive effect of the penalty term is that the introduction of bias reduces the variance of the function estimate, a problem usually associated with local function fitting methods (Friedman, 1984). Section 4 will analyze the exact properties of (12) in more detail.

What remains is how to minimize (12) incrementally by adjusting **M** by gradient descent with learning rate α:

$$\mathbf{M}^{n+1} = \mathbf{M}^n - \alpha \frac{\partial J}{\partial \mathbf{M}} \tag{13}$$

Applying the chain rule, the derivative of (13) can be written as

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{M}} &= \frac{\partial}{\partial \mathbf{M}}\left(\sum_{i=1}^{p}\frac{w_i\left\|\mathbf{y}_i-\hat{\mathbf{y}}_i\right\|^2}{W\left(1-w_i\mathbf{x}_i^T\mathbf{P}\mathbf{x}_i\right)^2}+\gamma\sum_{i,j=1}^{n}D_{ij}^2\right) \\ &= \frac{\partial}{\partial \mathbf{M}}\left(\sum_{i=1}^{p}J_{1,i}+J_2\right)=\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{\partial J_{1,i}}{\partial w_j}\frac{\partial w_j}{\partial \mathbf{M}}+\frac{\partial J_2}{\partial \mathbf{M}}\end{aligned} \tag{14}$$

Without storing data in incremental learning, we cannot use cross validation and, thus, cannot obtain the true gradient in (14). The usual approach to derive a stochastic gradient would be to drop the two sums in (14). However, such a gradient would be quite inaccurate since the first term of (14) would always be positive: shrinking the receptive field reduces the weight of a data point and thus its contribution to the weighted error. It turns out that we are able to derive a much better stochastic approximation. Given one training point $(\mathbf{x},\mathbf{y})$ and its associated weight w from (5), the derivative for this point can be approximated as:

$$\frac{\partial J}{\partial \mathbf{M}} \approx \sum_{i=1}^{p}\frac{\partial J_{1,i}}{\partial w}\frac{\partial w}{\partial \mathbf{M}}+\frac{w}{W}\frac{\partial J_2}{\partial \mathbf{M}}=\frac{\partial w}{\partial \mathbf{M}}\sum_{i=1}^{p}\frac{\partial J_{1,i}}{\partial w}+\frac{w}{W}\frac{\partial J_2}{\partial \mathbf{M}} \tag{15}$$

Summing (15) over all data points and recalling that W stands for the sum of weights (cf. Equation (9)), Equation (15) can be verified to result in Equation (14). Despite the term $J_{1,i}$, it is now possible to obtain an incremental version of the stochastic derivative in (15) by introducing the "memory traces" W, E, $\mathbf{H}$, and $\mathbf{R}$ (cf. notation in (8)):

$$\begin{aligned} W^{n+1} &= \lambda W^n + w \\ E^{n+1} &= \lambda E^n + w\,\mathbf{e}_{cv}^T\mathbf{e}_{cv} \\ \mathbf{H}^{n+1} &= \lambda \mathbf{H}^n + \frac{w\,\tilde{\mathbf{x}}\,\mathbf{e}_{cv}^T}{1-h}, \quad \text{where} \quad h = w\,\tilde{\mathbf{x}}^T\mathbf{P}^{n+1}\,\tilde{\mathbf{x}} \\ \mathbf{R}^{n+1} &= \lambda \mathbf{R}^n + \frac{w^2\,\mathbf{e}_{cv}^T\mathbf{e}_{cv}\tilde{\mathbf{x}}\,\tilde{\mathbf{x}}^T}{1-h} \end{aligned} \tag{16}$$

The resulting incremental version of the derivative (15) becomes:

$$\frac{\partial J}{\partial \mathbf{M}} \approx \frac{\partial w}{\partial \mathbf{M}}\sum_{i=1}^{p}\frac{\partial J_{1,i}}{\partial w}+\frac{w}{W^{n+1}}\frac{\partial J_2}{\partial \mathbf{M}} \tag{17}$$

where :

$$\frac{\partial w}{\partial M_{rl}} = -\frac{1}{2}w(\mathbf{x}-\mathbf{c})^T\frac{\partial \mathbf{D}}{\partial M_{rl}}(\mathbf{x}-\mathbf{c}), \quad \frac{\partial J_2}{\partial M_{rl}} = 2\gamma\sum_{i,j=1}^{n}D_{ij}\frac{\partial D_{ij}}{\partial M_{rl}}$$

$$\frac{\partial D_{ij}}{\partial M_{rl}} = M_{rj}\delta_{il} + M_{ir}\delta_{jl} \quad (\delta \text{ is the Kronecker operator})$$

$$\sum_{i=1}^{p} \frac{\partial J_{1,i}}{\partial w} \approx -\frac{E^{n+1}}{\left(W^{n+1}\right)^2} +$$
$$\frac{1}{W^{n+1}}\left(\mathbf{e}_{cv}^T \mathbf{e}_{cv} - \left(2\,\mathbf{P}^{n+1}\,\tilde{\mathbf{x}}\left(\mathbf{y} - \tilde{\mathbf{x}}^T \beta^{n+1}\right)^T\right) \otimes \mathbf{H}^n - \left(2\,\mathbf{P}^{n+1}\,\tilde{\mathbf{x}}\,\tilde{\mathbf{x}}^T \mathbf{P}^{n+1}\right) \otimes \mathbf{R}^n\right)$$

Deriving this derivative is possible due to the fact that an application of the Sherman-Morrison-Woodbury theorem allows us to take derivatives through the inverted covariance matrix **P** (Belsley et al., 1980; Atkeson & Schaal, 1995), and that a sum of the form $\Sigma \mathbf{v}_i^T \mathbf{Q}\, \mathbf{v}_i$ can be written as $\Sigma \mathbf{v}_i^T \mathbf{Q}\, \mathbf{v}_i = \mathbf{Q} \otimes \Sigma\, \mathbf{v}_i \mathbf{v}_i^T$, where the operator $\otimes$ denotes a element-wise multiplication of two homomorphic matrices or vectors with a subsequent summation of all coefficients, $\mathbf{Q} \otimes \mathbf{V} = \Sigma Q_{ij} V_{ij}$. It is interesting to note that the stochastic derivative (17) is not just concerned with reducing the error of the current training point as in many other learning algorithms, but rather that it takes into account the previously encountered training data, too, through the memory traces (16). Thus, both the β and **M** update in RFWR are not greedy with respect to the current training sample, a characteristic which will contribute favorably to speed and robustness of incremental learning.

3.1.3 Adding Receptive Fields and Automatic Bias Adjustment

A new receptive field is created if a training sample $(\mathbf{x}, \mathbf{y})$ does not activate any of the existing receptive field by more than a threshold w_{gen}. The center of the new receptive field becomes $\mathbf{c} = \mathbf{x}$, **M** is set to a manually chosen default value, $\mathbf{M}_{def}$, and all other parameters are initialized to zero, except the matrix **P**. **P** corresponds to an inverted covariance matrix of the weighted inputs (treating the constant input "1" as the *(n+1)*-th input). A suitable initialization of **P** is as a diagonal matrix, the diagonal elements set to $P_{ii} = 1/r_i^2$, where the coefficients r_i are usually small quantities, e.g., 0.001 (Ljung & Söderström, 1986). We summarize all r_i in the *(n+1)*-dimensional vector $\mathbf{r} = \left(r_1, r_2, \ldots, r_{n+1}\right)^T$.

The parameters **r** have an interesting statistical interpretation: they introduce bias in the regression coefficients β, and correspond to one of the common forms of biased regression, ridge regression (Belsley et al., 1980). From a probabilistic point of view, they are Bayesian priors that the coefficients of β are zero. From an algorithmic perspective, they are fake data points of the form $[\mathbf{x}_r = (0, \ldots, r_i^2, 0, \ldots)^T, \mathbf{y}_r = 0]$ (Atkeson et al., in press). Under normal circumstances, the sizes of the coefficients of **r** are too small to introduce noticeable bias. However, ridge regression parameters have to be larger if the input data is locally rank deficient, i.e., the matrix inversion in (7) is close to singular. For high dimensional input spaces, it is quite common to have locally rank deficient input data. Although RFWR does not explicitly require matrix inversions, the rank deficiency affects the incremental update in (8) by generating estimates of β with very large variances, causing unreliable predictions. For this reason, we include the ridge regression parameters as an automatically adjustable quantity in RFWR. As for the distance metric, the update rule of r is gradient descent in the cost (12):

$$\mathbf{r}^{n+1} = \mathbf{r}^{n} - \alpha_r \frac{\partial J}{\partial \mathbf{r}} \tag{18}$$

After each update of $\mathbf{P}$, the change in r is added to $\mathbf{P}$. Additionally, it is necessary to add back the fraction of $\mathbf{r}$ which was lost due to the forgetting factor λ—bias should not to be forgotten over time. These two computations can be performed together and are surprisingly simple. Appendix 9.1 details this update and the stochastic approximation of $\partial J/\partial \mathbf{r}$, which is analogous to the derivation of (17).

3.1.4 *Pruning Receptive Fields*

The last element in RFWR is a pruning facility. A receptive field is pruned if it overlaps too much with another receptive field. This effect is detected by a training sample activating two receptive fields simultaneously more than w_{prune}. The receptive field with the larger determinant of the distance metric $\mathbf{D}$ is pruned. For computational convenience, det($\mathbf{D}$) can be approximated by ΣD_{ii}^2 (Deco & Obradovic, 1996). It should be noted that pruning due to overlap aims primarily at computational efficiency, since, as discussed in Section 2.3, overlap does not degrade the approximation quality.

The second cause for pruning is if the bias-adjusted weighted mean squared error

$$wMSE = \frac{E^n}{W^n} - \gamma \sum_{i,j=1}^{n} D_{ij}^2 \tag{19}$$

of the linear model of a unit is excessively large in comparison to other units—the bias adjustment term will be explained in Section 4. Empirically, there are usually two ways to adjust $\mathbf{M}$ in order to minimize (12). The one we normally want to avoid is $\mathbf{M}$=$\mathbf{0}$, i.e., the zero matrix. It indicates that the receptive field performs global regression instead of locally weighted regression. Global linear regression for a nonlinear function has a large *wMSE*. A simple outlier detection test among the *wMSE* of all receptive fields suffices to deal with such behavior. The receptive field is then reinitialized with randomized values. Normally, pruning takes place rarely, and if it happens, it is mostly due to an inappropriate initialization of RFWR.

3.1.5 *Summary of RFWR*

In sum, each RFWR subnet has three sets of adjustable parameters: β for the locally linear model, $\mathbf{M}$ for the size and shape of the receptive fields, and $\mathbf{r}$ for the bias. The linear model parameters are updated by a Newton method, while the other parameters are updated by gradient descent. A compact pseudo-code overview of RFWR is shown below.

```
Initialize the RFWR with no receptive field (RF);
For every new training sample (x,y):
    a)  For k=1 to #RF:
            –   calculate the activation from (5)
            –   update the receptive field parameters according to (13), and (18)
        end;
    b)  If no subnet was activated by more than $w_{gen}$;
            –   create a new RF with $\mathbf{c}=\mathbf{x}$, $\mathbf{M}=\mathbf{M}_{def}$
        end;
    c)  If two RFs are activated more than $w_{prune}$;
            –   erase the RF with the larger det(D)
        end;
    d)  calculate the $m=E\{wMSE\}$ and $std=E\{(wMSE-m)^2\}^{0.5}$ of all RFs;
    e)  For k=1 to #RF:
            If $|wMSE-m| > \varphi\, std$,
                –   reinitialize receptive field with $\mathbf{M} = \varepsilon\, \mathbf{M}_{def}$
            end;
        end;
```

The scalar φ is a (positive) outlier removal threshold, e.g., φ=3.17, and the scalar ε is a random value $\varepsilon=1+|N(0,1)|$. This choice of ε ensures that the new distance metric will result in a smaller receptive field which is less likely to converge to a **M**=0 solution.

3.2 Second Order Gradient Descent

With little extra computation, it is possible to replace the gradient descent update of **M** in (13) by second order gradient descent to gain learning speed. In what follows, we adopt Sutton's (1992a,b) Incremental Delta-Bar-Delta (IDBD) algorithm. The derivation of the algorithm remains as demonstrated in Sutton (1992a,b), only that his standard least squares criterion is replaced by our cost function (12), and that we apply IDBD to updating a distance metric. The idea is to replace the learning rate α in (13) by an individual learning rate for each coefficient of **M** of the following form:

$$M_{ij}^{n+1} = M_{ij}^{n} - \alpha_{ij}^{n+1}\frac{\partial J}{\partial M_{ij}}, \quad \text{where} \quad \alpha_{ij}^{n+1} = \exp\left(\beta_{ij}^{n+1}\right) \quad \text{and} \quad \beta_{ij}^{n+1} = \beta_{ij}^{n} - \theta\frac{\partial J}{\partial M_{ij}}h_{ij}^{n} \tag{20}$$

Thus, the learning rates α_{ij} are changed in geometric steps by gradient descent in the meta parameter β_{ij} with meta learning rate θ. The term h_{ij} is updated as

$$h_{ij}^{n+1} = h_{ij}^{n}\left[1-\alpha_{ij}^{n+1}\frac{\partial^2 J}{\partial M_{ij}^2}\right]^{+} - \alpha_{ij}^{n+1}\frac{\partial J}{M_{ij}}, \quad \text{where we define} \quad [z]^{+} = \begin{cases} z & \text{if} \quad z>0 \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

h_{ij} is initialized to zero when a receptive field is created. It corresponds to a memory term which stores a decaying trace of the cumulative sum of recent changes to M_{ij}. For more details see Sutton (1992a,b). In order to apply this second order update, it is necessary to store the parameters α_{ij}, β_{ij}, and h_{ij}, and to compute the second derivative in (21). Appendix 9.2 gives an incremental approximation of this derivative which turns out to be quite simple. It is also possible to apply second order learning to the ridge regression update (18). Empirically, however, we did not find any significant improvements of doing so and, hence, only incorporated second order updates for the distance metric in RFWR.

4 Theoretical Assessment of RFWR

4.1 Asymptotic Properties of RFWR

For the linear model β and the distance metric $\mathbf{D}$ asymptotic approximations can be derived. Assuming that the number of training data points p goes to infinity, that the variance of the noise σ^2 is locally constant, and that the input distribution is locally uniform, the expected value of (12) can be written as

$$E\{J\} = E\left\{\frac{1}{W}\sum_{i=1}^{p}\frac{w_i\left\|\mathbf{y}_i-\hat{\mathbf{y}}_i\right\|^2}{\left(1-w_i\mathbf{x}_i^T\mathbf{P}\mathbf{x}_i\right)^2}+\gamma\sum_{i,j=1}^{n}D_{ij}^2\right\} \tag{22}$$

$$\xrightarrow{p\to\infty}\frac{\int_{-\infty}^{+\infty} w\left\|\mathbf{y}-\hat{\mathbf{y}}\right\|^2 p(\mathbf{y}|\mathbf{x})p(x)\,d\mathbf{y}\,d\mathbf{x}}{\int_{-\infty}^{+\infty} wp(x)\,d\mathbf{x}}+\gamma\sum_{i,j=1}^{n}D_{ij}^2=\frac{\int_{-\infty}^{+\infty} w\left\|f(\mathbf{x})-\hat{\mathbf{y}}\right\|^2 d\mathbf{x}}{\int_{-\infty}^{+\infty} w\,d\mathbf{x}}+\gamma\sum_{i,j=1}^{n}D_{ij}^2+\sigma^2$$

Next, the real function $f(\mathbf{x})$ is represented as a Taylor series expansion at the center of the receptive field. Without loss of generality, the center is assumed to be at the origin in input space. We furthermore assume that the size and shape of the receptive field are such that terms higher than quadratic are negligible, and that for notational simplicity the output is one dimensional. Thus, Equation (22) can be re-written as

$$E\{J\}\approx\frac{\int_{-\infty}^{+\infty} w\left(f_0+\mathbf{f}^T\mathbf{x}+\frac{1}{2}\mathbf{x}^T\mathbf{F}\mathbf{x}-b_0-\mathbf{b}^T\mathbf{x}\right)^2 d\mathbf{x}}{\int_{-\infty}^{+\infty} w\,d\mathbf{x}}+\gamma\sum_{i,j=1}^{n}D_{ij}^2+\sigma^2 \tag{23}$$

where f_0, $\mathbf{f}$, and $\mathbf{F}$ denote the constant, linear, and quadratic terms (Hessian) of the Taylor series expansion, respectively. The value of the integral in (23) remains invariant for any volume preserving rotation of the input space. Thus, in order to solve (23), we assume that the input space has been rotated about the origin by the orthonormal matrix $\mathbf{N}$, $\mathbf{x} \rightarrow \mathbf{Nx}$, such that $\mathbf{D}$ is diagonal. After inserting (5) into (23), we obtain:

$$E\{J\} \approx (f_0 - b_0)^2 + \sum_{i=1}^{n} \left[\frac{(b_i - f_i)^2}{D_{ii}} + (f_0 - b_0)\frac{F_{ii}}{D_{ii}} + \frac{3}{4}\left(\frac{F_{ii}}{D_{ii}}\right)^2 \right] \tag{24}$$

$$+\frac{1}{2}\sum_{i=1}^{n}\sum_{j=i+1}^{n} \frac{2F_{ij}^2 + F_{ii}F_{jj}}{D_{ii}D_{jj}} + \gamma \sum_{i,j=1}^{n} D_{ij}^2 + \sigma^2$$

By taking the partial derivatives with respect to all unknown parameters, we can derive the following asymptotic results:

- *The eigenvectors of* $\mathbf{D}$ *align with the eigenvectors of* $\mathbf{F}$: If, for a moment, we assume that the (diagonal) distance metric $\mathbf{D}$ is given, and $\mathbf{F}$ is to be determined such that (24) is minimized, the partial derivatives $\partial E\{J\}/\partial F_{ij}$ require that $F_{ij} = 0$ for $i \neq j$, i.e., $\mathbf{F}$ be diagonal. Thus, the converse result must hold that for a given diagonal $\mathbf{F}$ the distance metric $\mathbf{D}$ must be diagonal in order to minimize (24).

- *The estimated locally linear model* $\mathbf{b}$ *is asymptotically unbiased*: This follows from the fact that minimization of (24) with respect to $\mathbf{b}$ results in $\mathbf{b}$=$\mathbf{f}$. Note that this result holds even if $\mathbf{F}$ is non diagonal and we only estimate a diagonal $\mathbf{D}$, as it might be the case in high dimensional spaces where estimating a full $\mathbf{D}$ becomes computationally too expensive or would require too much data.

- *The penalty term introduces non vanishing bias*: The expected bias at the center of the receptive field becomes a function of the penalty factor and the eigenvalues of $\mathbf{F}$, denoted as F'_{ii} :

$$bias \leq \frac{\gamma^{0.25}}{2^{0.75}} \sum \operatorname{sgn}(F'_{ii}) \sqrt{|F'_{ii}|} \tag{25}$$

  The equality holds iff $\mathbf{D}$ and $\mathbf{F}$ have aligned eigenvectors. If we use a diagonal $\mathbf{D}$ for a non diagonal $\mathbf{F}$, the receptive field become smaller, which subsequently reduces the expected bias, but will require a larger number of receptive fields in the learning system. Note that, due to the square root in (25), the bias tends to be larger with larger eigenvalues F'_{ii}, i.e., in areas with high curvature. This acts like a low pass filter in the function fitting process.

- *The distance metric* $\mathbf{D}$ *will be a scaled image of the Hessian* $\mathbf{F}$: The expected coefficients of the distance metric become

$$D_{ii} \geq \frac{\sqrt{|F'_{ii}|}}{(2\gamma)^{0.25}} \tag{26}$$

As above, equality holds iff **D** and **F** have aligned eigenvectors. From this equation it is also obvious that without the penalty term (i.e., for $\gamma \rightarrow 0$), the coefficients of **D** would asymptotically tend to infinity, as mentioned in Section 3.1.2.

In sum, these asymptotic results confirm that the penalty term in the cost function (12) has the desired characteristics: receptive fields cannot shrink to zero size, and a controlled amount of bias was introduced in the sense of low pass filtering. It is interesting that the estimated locally linear model tends to become unbiased (under the assumption that $O(2)$ errors of the Taylor series are negligible). This implies that applications requiring a gradient estimate from the function approximator can expect reliable results. The calculation of the gradient estimate is a natural by-product of every lookup in RFWR.

4.2 Some Helpful Statistical Estimates

The asymptotic results above can be used to derive several statistical quantities which help monitoring and initializing RFWR:

- *Penalty selection*: From a maximal permissible bias and an estimate of the maximal eigenvalues anticipated in a specific learning problem, the required penalty factor can be estimated as

$$\gamma = \frac{8\, bias_{\max}^4}{\left(\sum_{i=1}^{n} \operatorname{sgn}\left(F'_{ii,\max}\right)\sqrt{\left|F'_{ii,\max}\right|}\right)^4} \tag{27}$$

- *Bias adjusted weighted mean squared error*: Another derivation from (24) is the expected value of the cost function (12)

$$E\{J\} \leq 2\gamma \sum_{i,j=1}^{n} D_{ij}^2 + \sigma^2 \tag{28}$$

$$wMSE = E\{J\} - \gamma \sum_{i,j=1}^{n} D_{ij}^2 \leq \gamma \sum_{i,j=1}^{n} D_{ij}^2 + \sigma^2 = \frac{E}{W}$$

where again the equality holds iff **D** and **F** have aligned eigenvectors. As shown in (28), the expected value for the weighted mean squared error (*wMSE*) can thus be formulated which is composed of the noise variance and a term due to the non vanishing bias in the local model. Thus, a bias adjusted weighted mean squared error, *wMSE*, can be formulated, as already given in Equation (19).

- *Real bias and prediction intervals*: From (25) and (26), a pessimistic estimate of the real bias of a prediction becomes

$$bias \le \sqrt{0.5\gamma} \sum_{i=1}^{n} D'_{ii} \tag{29}$$

where D'_{ii} denotes the eigenvalues of $\mathbf{D}$. Based on this estimate, bias adjusted prediction intervals—a variant of confidence intervals (e.g., Myers, 1990)—can be approximated similarly as in Schaal and Atkeson (1994):

$$I = \hat{y} \pm \left(bias + t_{\alpha/2, W^n - dof^n} s^n \sqrt{1 + \tilde{\mathbf{x}}^T \mathbf{P}^n \tilde{\mathbf{x}}} \right) \tag{30}$$

$$\text{where} \quad s^n = \sqrt{wMSE}, \quad dof^{n+1} = \lambda\, dof^n + w^2 \tilde{\mathbf{x}}^T \mathbf{P}^n \tilde{\mathbf{x}}$$

The variable dof denotes the local degrees of freedom used by the locally linear model (Schaal & Atkeson, 1994; Atkeson et al., in press), and we gave its recursive estimation formula in (30) in analogue with Equation (16). s^n is a bias adjusted estimate of the local standard deviation of the error. $t_{\alpha/2, W^n - dof^n}$ is Student's t-value with $(W^n - dof^n)$ degrees of freedom for a $100 * (1 - \alpha)\%$ confidence bound. These prediction intervals assume a locally normal error distribution.

5 Simulation Results

5.1 Basic Function Approximation with RFWR

First, we will establish that RFWR is capable of competing with state-of-the-art supervised learning techniques on a fixed training set. A sufficiently complex learning task that still can be illustrated nicely is to approximate the function

$$z = \max\left\{ e^{-10x^2}, e^{-50y^2}, 1.25 e^{-5(x^2+y^2)} \right\} + N(0, 0.01) \tag{31}$$

from a sample of 500 points, drawn uniformly from the unit square. This function consists of a narrow and a wide ridge which are perpendicular to each other, and a Gaussian bump at the origin (Figure 4a). Training data is drawn uniformly from the training set without replacement; training time is measured in epochs, i.e., multiples of 500 training samples. The test set consists of 1681 data points corresponding to the vertices of a 41x41 grid over the unit square; the corresponding output values are the exact function values. The approximation error is measured as a normalized mean squared error, *nMSE*, i.e., the *MSE* on the test set normalized by the variance of the outputs of the test set. RFWR's initial parameters are set to $\mathbf{M}_{def} = 5.0\mathbf{I}$ ($\mathbf{I}$ is the identity matrix), $\gamma = 10^{-7}, w_{gen} = 0.1$, and $w_{prune} = 0.9$. The pruning and generation thresholds are of minor importance;

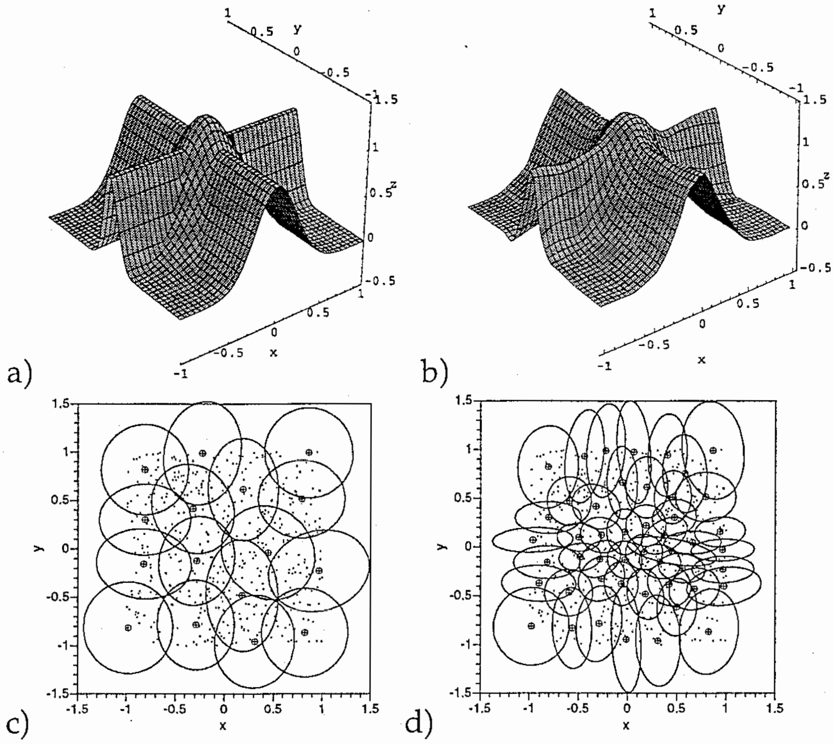

Figure 4: a) target function to be approximated; b) approximated function after 50 epochs of training; c) receptive fields in input space after 1 epoch, given by contour lines of 0.1 isoactivation and a ⊕ mark for the centers (the training data is displayed by small dots); d) receptive fields after 50 epochs of training.

they just determine the overlap of the receptive fields. The choice for the penalty term was computed from (27) to tolerate a maximal bias of about 0.1. The default value for the distance metric was determined manually such that an initial receptive field covered a significant portion of the input space. Ridge regression parameters did not play any role in this example and were omitted.

A first qualitative evaluation of Figure 4 confirms that RFWR fulfills our expectations. The initially large receptive fields (Figure 4c) adjust during learning according to the local curvature of the function: they become narrow and elongated in the region of the ridges, and they remain large in the flat parts of the function ((Figure 4d). The number of the receptive fields increased from 16 after one training epoch to 48, and the final approximation result was *nMSE*=0.02.

We compared the learning results of RFWR with 3 other algorithms: standard global linear regression and a sigmoidal 3-layer backpropagation neural network as baseline comparisons, and the mixture of experts algorithm as a state-of-the-art comparison (Jacobs et. al, 1991; Jordan & Jacobs, 1994; Xu, Jordan, & Hinton, 1995). Standard linear regression cannot accomplish a better result than *nMSE*=1.0 on this example—the function has no linear trend in the chosen region of input space. The sigmoidal network was trained by backpropagation with momentum in a variety of configurations using 20 to 100 units in the hidden layer (the output layer had one linear unit). These networks did not accomplish results better than *nMSE*=0.1 within 20000 training epochs. Doubling the number of training samples and reducing the noise level to N(0,0.0001) finally resulted in *nMSE*=0.02 for a 100 hidden unit net after about 15000 epochs. By using the Cascade Correlation algorithm (Fahlman & Lebiere, 1990) to fit our original 500 data point training set we confirmed that the function (31) seems to be a difficult learning task for sigmoidal networks: Cascade Correlation did not converge when confined to using only sigmoidal hidden units, while it achieve good function fitting (*nMSE*=0.02) when it was allowed to use Gaussian hidden units.

A more natural and interesting comparison is with the mixture of experts (ME) system, particularly as suggested in Xu et al. (1995). In Xu et al. (1995), in contrast to the softmax gating network of Jordan and Jacobs (1994), the experts use a mixture of Gaussians as the gating network, and both the gating net and the locally linear models in each leaf of the gating net can be updated by an analytical version of the Expectation–Maximation (EM) algorithm (Dempster, Laird, & Rubin, 1977). Thus, the basic elements of this form of ME are the same as in RFWR—locally linear models and Gaussian receptive fields—while the training methods of the two systems differ significantly—competitive parametric likelihood maximization vs. local nonparametric learning. As ME is not a constructive algorithm, the performance determining parameters are how many experts are allocated and how the system is initialized. The algorithm was tested with 25, 50, 75, and 100 experts. Initially, the experts were placed uniformly distributed in the input space with an initial covariance matrix of the Gaussians comparable to the initialization of RFWR's distance metric. We conducted a similar test with RFWR, setting its determining parameter, the penalty γ, to 10^{-6}, 10^{-7}, 10^{-8}, and 10^{-10}.

Figure 5 summarizes the results. Each learning curve is the average of 10 learning trials for each condition of the corresponding algorithm; the training data was randomly regenerated for each trial. Both algorithms achieve a *nMSE*=0.12 after only one training epoch—a typical signature of the fast recursive least squares updating of the linear models employed by both algorithms—which is about what the sigmoidal neural network had achieved after 10000 to 20000 epochs. Both algorithms converge after about 100 epochs. By adding more experts, the mixture of experts improves its performance to a best average value of *nMSE*=0.04 with a slight trend to overfitting for 75 experts. RFWR accomplishes consistently a result of *nMSE*=0.02 for all but the $\gamma=10^{-6}$ runs, with a slight tendency to overfitting for $\gamma=10^{-10}$. One standard deviation error bars are indicated by the black bars at the beginning and end of each learning curve.

It was surprising that ME did not achieve the same ultimate fit accuracy as RFWR. This behavior was due to a) the relative small training set, b) the relatively low signal to noise ratio of the training data, and c) the way the gating network assigns training samples to each expert. By significantly increasing the

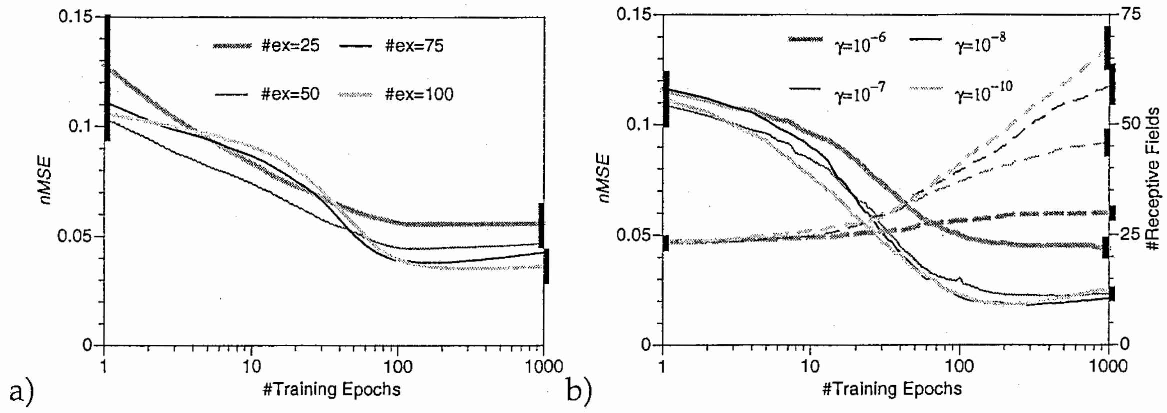

Figure 5: Average learning curves (solid lines) for a) ME, and b) RFWR. The black bars indicate one standard deviation error bars at the beginning and at the end of learning; for overlapping traces having approximately the same standard deviation, only one bar is shown. For RFWR (b), the increase of the number of receptive fields over time (dashed lines) is indicated as well.

amount of training data and/or lowering the noise, the results of both algorithms become indistinguishable. It seems to be the method of credit assignment which makes a significant difference. The expectation step in ME uses normalized weights (i.e., posterior probabilities) to assign training data to the experts. Normalized weights create much sharper decision boundaries between the experts than unnormalized weights as in RFWR. Thus, in the case of noise and not too much training data, the ME algorithm tends to establish too sharp decision boundaries between the experts and starts fitting noise. Given the underlying assumption of ME that the world was generate by a mixture of linear models, this behavior may be expected. Since in our test cases, the world is actually a continuous function and not a mixture of linear models, the assumptions of ME are only an approximation, which explains why the algorithm does not perform entirely appropriately.

The assumptions of RFWR are quite different: every receptive fields tries to find a region of validity which allows it to approximate the tangent plane in this region with some remaining bias. In the spirit of a low order Taylor series expansion, this is a reasonable way to proceed. Thus, RFWR achieves consistent results with low variance (Figure 5b). It is also interesting to see how the number of receptive fields of RFWR grows as a function of the penalty factor (Figure 5b). As expected from the derivation of the cost function (12), a very small penalty parameter causes the receptive fields to keep on shrinking and entails a continuous growth of the number of receptive fields. Nevertheless, the tendency towards overfitting remained low, as can be seen in the $\gamma = 10^{-10}$ traces in Figure 5b. When continuing learning until 10000 epochs, the *nMSE* saturated close to the current values for all penalty factors. The local cross validation term in (12) is responsible for this desirable behavior—when cross validation was not used, overfitting was

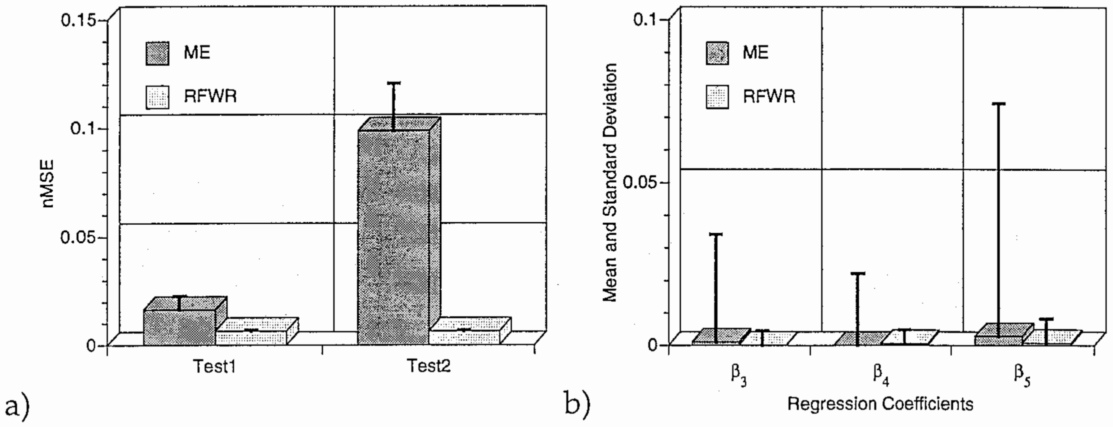

Figure 6: a) average *nMSE* of ME and RFWR after 1000 training epochs (see text for further explanations); b) mean and standard deviation of the regression coefficients of the irrelevant inputs.

significantly more pronounced and the *nMSE* continued increasing for very small penalty factors.

5.2 Dealing With Irrelevant Inputs

In order to establish the usefulness of the ridge regression parameters, we conducted a further comparison with ME. In sensorimotor control, it is unlikely that all variables given to the learning system are really relevant to the task. One can distinguish between three kinds of irrelevant inputs: a) constant inputs, b) changing inputs which are meaningless, and c) copies and linear combinations of other inputs. Ideally, one would like an autonomous learning system to be robust towards such signals. To explore the behavior of ME and RFWR in such cases, three additional inputs were added to the function (31): a) one input of N(0.1,0.001), b) one input with a Brownian walk in the interval [-0.1,0.1], and c) one input which was a copy of x with added Gaussian noise N(0,0.0025). Otherwise, training data was generated uniformly by the function (31), but with reduced additive noise of N(0,0.0025) to improve the signal to noise ratio. For these tests, the ridge regression coefficients were initialized to 0.25 for each input.

Figure 6 summarizes the average results of 10 trials for each algorithm. In Figure 6a, we show the mean *nMSE* and its standard deviation on two test sets. In Test1, the predictions were generated by using only the regression coefficients of the relevant inputs, i.e., $\beta_0, \beta_1, \beta_2$, on the same 1681 point test set as in the experiment of Section 5.1. This was to establish whether these coefficients adjusted correctly to model the target function. Both algorithms achieved good learning results on this test (Figure 6a). In Test2, we probed the robustness of the learned model towards the irrelevant inputs: we added the noisy constant, the Brownian, and the noisy x-copy input to the test set, but we also added an offset of 0.1 to each of these signals. If the algorithm learned that these inputs were irrelevant,

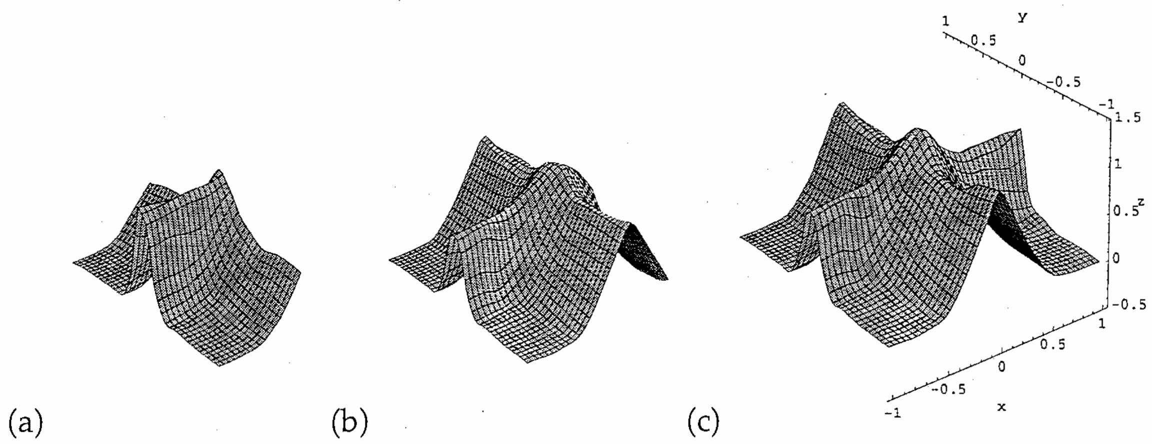

Figure 7: Reconstructed function after training on (a) T_1, (b) then T_2, (c) and finally T_3.

this change should not matter. However, if the irrelevant inputs were mistakenly employed as signal to improve the *nMSE* on the training data, the predictions should deteriorate. Figure 6a demonstrates that the results of RFWR remained virtually unaltered by this test, while those of ME became significantly worse. This outcome can be explained by looking at the standard deviations of the regression coefficients of all the locally linear models (Figure 6b). In contrast to ME, RFWR set the regression coefficients of the irrelevant inputs ($\beta_3, \beta_4, \beta_5$) very close to zero, thus achieving the desired robustness. Such behavior was due to an adjustment of the corresponding ridge regression parameters: they increased for the irrelevant inputs and decreased to zero for the relevant inputs. As a note, we should point out that ME was not designed to deal with learning problems with irrelevant inputs, and that there are ways to improve its performance in such cases. However, this experiment clearly illustrates that it is necessary to deal with the problem of irrelevant inputs, and that local bias adjustment by means of ridge regression is one possible way to do so.

5.3 Shifting Input Distributions

As mentioned in the Introduction, it is easy to conceive of learning tasks where the input distribution of the training data changes over time. To test RFWR's performance on such problems, we designed the following experiment. In three sequential episodes training data for learning (31) was uniformly drawn from three slightly overlapping input regions in the unit cube: $T_1 = \{(x, y, z) \mid -1.0 < x < -0.2\}$, $T_2 = \{(x, y, z) \mid -0.4 < x < 0.4\}$, and $T_3 = \{(x, y, z) \mid 0.2 < x < 1.0\}$. First the algorithm was trained on T_1 for 50,000 iterations and tested on T_1, then trained on T_2 for 50,000 iterations and tested on T_1 and T_2, and finally trained on T_3 for 50,000 iterations and tested on test data from all regions. Figure 7 gives an example of how learning proceeded. This test probes how much of the previously learned

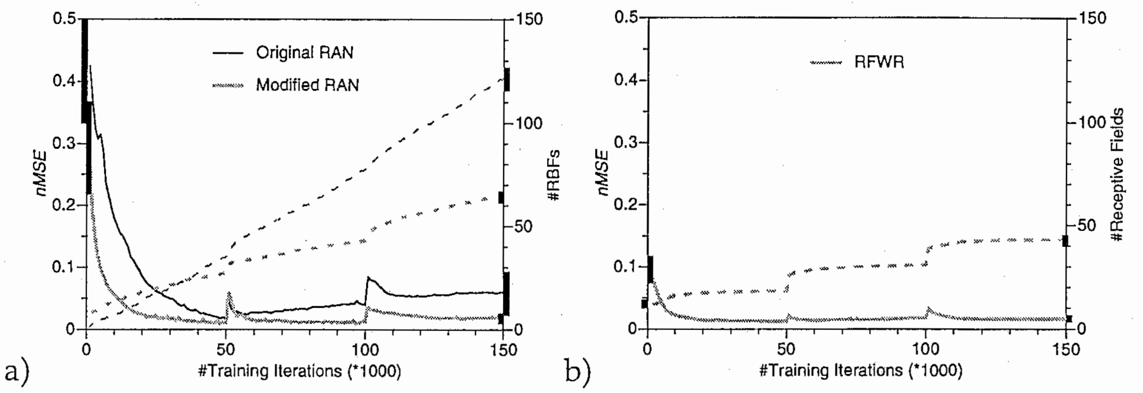

Figure 8: Average learning curves (solid lines) and average number of receptive field/radial basis functions (dashed lines) for a) RAN, and b) RFWR. The black bars give the one standard deviation at the beginning and the end of learning.

competence is forgotten when the input distribution shifts. All parameters of RFWR were chosen as in 5.1 except for $\mathbf{M}_{def}$ which was set to a slightly larger value of $\mathbf{M}_{def} = 6.0\mathbf{I}$.

As the ME algorithm is not constructive and thus not well suited for learning with strongly shifting input distributions, we chose the Resource Allocating Network (RAN) of Platt (1991) for a comparison, a learning algorithm which is constructive, which has no competitive learning component, and which has inspired a variety of other algorithms. RAN is a radial basis function (RBF) network that adds RBFs at the site of a training sample according to two criteria: a) when the approximation of the training sample error is too large, and b) when no RBF is activated by the training sample more than a threshold ξ value. Both criteria have to be fulfilled simultaneously to create a new RBF. The spherical width of the RBF is chosen according to its distance to the nearest neighboring RBF. By using gradient descent with momentum, the RBF centers are adjusted to reduce the mean squared approximation error, as are the weights of the linear regression network in the second layer of the RBF net. The strategy of RAN is to start initially with very wide RBFs and to increase the threshold ξ over time until a prechosen upper limit is reached, causing the creation of ever smaller RBFs at sites with large error. As in RFWR, we used Gaussians (Equation (5)) as the parametric structure of a RBF.

Figure 8 summarizes the average of 10 learning trials for each algorithm. RFWR shows large robustness towards the shift of input distribution: there is only a minor increase of *nMSE* due to interference in the overlapping parts of the training data. In contrast, as can be seen in the "original RAN" trace of Figure 8a, RAN significantly increases the *nMSE* during the second and third training episode. Since RAN starts out with initial RBFs which cover the entire input space, interference is not properly localized, which explains the observed behavior.

Note that we already have excluded the constant term in the linear regression layer of RAN (Platt, 1991), a term that is globally active and would decrease the performance in Figure 8 even more.

From the experience with RFWR, three possible improvements of RAN come to mind. First, instead of starting with very large RBFs initially, we can limit the maximal initial size as in RFWR to $\mathbf{M}_{def}$. Second, we can employ the hyper radial basis function technique of Poggio and Girosi (1990) to also adjust the width $\mathbf{M}$ of the RBFs by gradient descent as in RFWR (Furlanello, Giuliani, & Trentin, 1995). And third, instead of having the time varying threshold ξ a global variable, we can define it as an individual variable for each RBF, thus removing the explicit dependency on global training time. By initializing RAN with $\mathbf{M}_{def} = 6.0\mathbf{I}$ as in RFWR, these modification resulted in a significant improvement of robustness of RAN as shown in Figure 8a. Note that this version of RAN requires only half as many RBFs, converges more quickly, and achieves very low final approximation errors. As in RFWR, a localizing of the learning parameters lead to a significant improvement of robustness of incremental learning.

5.4 Sensorimotor Learning

As a last evaluation, we use a traditional example of sensorimotor learning, the approximation of the inverse dynamics of a two-joint arm (Atkeson, 1989). The state of the arm is given by two joint angles, θ_1 and θ_2 (Figure 9a). The inverse dynamics model is the map from two joint angles, two joint velocities, and two joint accelerations to the corresponding torques necessary to achieve the joint acceleration in a given state. We assume that the arm controller makes use of a low gain feedback PID controller whose performance is enhanced by feedforward commands from the learned inverse dynamics (An, Atkeson, & Hollerbach, 1988). The torques for the shoulder and elbow joint are learned by separate networks as there is no reason to believe that a receptive field for the elbow torque should have the same shape as for the shoulder torque—for RFWR this would mean that both outputs have the same Hessian which is definitely not the case. The task goal is to draw a figure "8" in two parts of the work space. Figure 9a shows the desired and the initial performance without the feedforward commands. Training proceeded in two steps: first, the arm performed sinusoidal movements with varying frequency content in the area of the upper "8". A total of 45,000 training points, sampled at 100Hz, was used for training—each training sample was only used once in the sequential order it was generated. The learning results are shown in the top part of Figure 9b for RFWR, and Figure 9c for the modified RAN. Both algorithms were able to track the figure "8" properly.

Next, the algorithms were trained in an analogous fashion on 45,000 samples around the lower figure "8". The bottom parts of Figure 9b,c show the corresponding good learning results. However, when returning to performing the upper figure "8", RAN showed significant interference (Figure 9c), although both

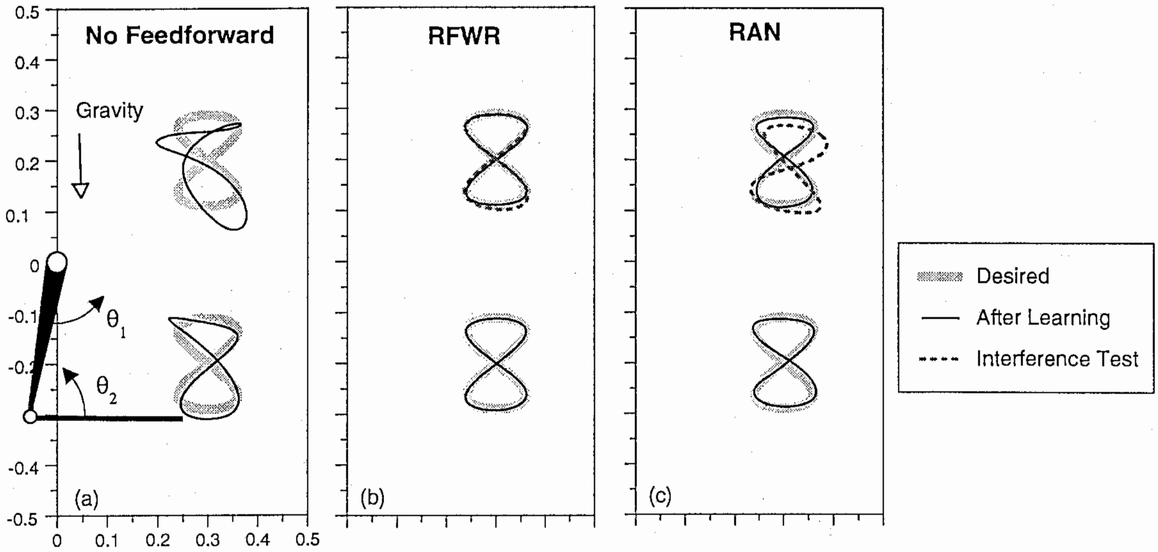

Figure 9: a) Initial performance of the two joint arm when drawing the figure "8" without feedforward control signals; b) performance of RFWR after learning; c) performance of RAN after learning.

algorithms were initiated with the same $\mathbf{M}_{def} = 6.0\mathbf{I}$ (note that position, velocity, and acceleration inputs were normalized prior to learning to compensate for the differences in units). This effect highlights the difference between the learning strategy of RBF networks in comparison to the nonparametric statistics approach to modeling with locally linear model. RBF networks need a sufficient overlap of the radial basis functions to achieve good learning results—one RBF by itself has only limited function approximation capabilities, an effect discussed in the context of hyperacuity (e.g., Churchland & Sejnowski, 1992). Gradient descent on the shape parameter $\mathbf{M}$ of the Gaussian RBFs quickly decreased $\mathbf{M}$ in our example to achieve an appropriately large overlap. This overlap, however, encourages negative interference, as is evident in Figure 9c. The 6-dimensional input space of this example emphasized the need for large overlap, while the 2-dimensional example of the previous section did not highlight this problem. Experiments which used a fixed as in the original RAN algorithm did not achieve better learning results within a reasonable training time. Clearly there is always the unattractive solution of adding thousands of quite narrow overlapping RBFs. In the results of Figure 9, both algorithms allocated less than 100 receptive fields.

6 Related Work

The field which contributes the most to the development of RFWR is nonparametric statistics. Cleveland (1979) introduced the idea of employing locally linear models for memory-based function approximation, called locally weighted regression (LWR). In a series of subsequent papers, he and his colleagues extended the statistical framework of LWR to include multi-dimensional function approximation and local approximation techniques with higher order polynomials (e.g., Cleveland, Devlin, & Gross, 1988; Clevland & Devlin, 1988). Cleveland and Loader (1995) suggested local C_p-tests and local PRESS for choosing the degree of local mixing of different order polynomials as well as local bandwidth adjustment and reviewed a large body of literature on the history of LWR. Hastie and Tibshirani (1990, 1994) give related overviews of nonparametric regression methods. Hastie and Loader (1993) discuss the usefulness of local polynomial regression and show that locally linear and locally quadratic function fitting have appealing properties in terms of the bias/variance trade-off. Friedman (1984) proposed a variable bandwidth smoother for one dimensional regression problems. Using different statistical techniques, Fan and Gijbels (1992, 1995) suggested several adaptive bandwidth smoothers for LWR and provided detailed analyses of the asymptotic properties of their algorithms.

For the purpose of time series prediction, LWR was first used by Farmer and Siderowich (1987, 1988). Atkeson (1989) introduced the LWR framework for supervised learning in robot control. Moore (1991) employed LWR for learning control based on learning forward models. In the context of learning complex manipulation tasks with a robot, Schaal and Atkeson (1994a,b) demonstrated how LWR can be extended to allow for local bandwidth adaptation by employing local cross validation and local confidence criteria. Schaal and Atkeson (1996) introduced the first non memory-based version of LWR. Schaal (in press) applied RFWR for value function approximation in reinforcement learning. Locally weighted learning for classification problems can be found, e.g., in Lowe (1995). Aha (in press) compiled a series of papers on nonparametric local classification and regression learning, among which Atkeson, Moore, and Schaal (a,b, in press) give an extended survey on locally weighted learning and locally weighted learning applied to control.

Besides nonparametric statistics, RFWR is related to work on constructive learning algorithms, local function approximation based on radial basis functions (RBF), and Kohonen-like self-organizing maps (SOM). A RBF function approximator with a locally linear model in each RBF was suggested by Millington (1991) for reinforcement learning. Platt (1991) suggested a constructive RBF-based learning system. Furlanello et al. (1995b) and Furlanello and Giuliani (1995a) extended Platt's method by using Poggio and Girosi's (1990) hyper radial basis functions and local principal component analysis. For learning control, Cannon and Slotine (1995) derived a constructive radial basis function network which used wavelet-like RBFs to adapt to spatial frequency; this is similar to local bandwidth adaptation in nonparametric statistics and the adjustable receptive fields in RFWR. Orr (1995) discussed recursive least squares methods and ridge regression for learning with radial basis function networks. He also suggests several other methods, e.g., generalized cross validation, for regularizing ill-conditioned regression.

One of the most established constructive learning systems is Cascade Correlation (Fahlman & Lebiere, 1990), a system sharing ideas with projection pursuit regression (Friedman, 1981). Related to this line of research is the Upstart algorithm of Frean (1990), the SOM based cascading system of Littman and Ritter (1993), and the work of Jutton and Chentouf (1995). The first usage of locally linear models for regression problems in the context of SOMs was by Ritter and Schulten (1986) who extended Kohonen maps to fit locally linear models (LLM) within each of the units of the SOM. Related to this work is Smagt and Groen's (1995) algorithm which extended LLM to a hierarchical approximation in which each Kohonen unit itself can contain another LLM network. Fritzke (1994, 1995) demonstrated how SOMs can constructively add units, both in the context of RBF and LLM regression problems. Bruske and Sommer (1995) combined Fritzke's ideas with Martinetz and Schulten's (1994) Neural Gas algorithm to accomplish a more flexible topographic representation as in the original SOM work. A large body of literature on constructive learning stems from fitting high order global polynomials to data, for instance, as given in Sanger (1991), Sanger, Sutton, and Matheus (1992), and Shin and Ghosh (1995). Due to the global character of these learning methods, the danger of negative interference is quite large. Additional references on constructive learning for regression can be found in the survey by Kwok and Yeung (1995).

The idea of the mixture of experts in Jacobs et al. (1991) and hierarchical mixtures of experts in Jordan and Jacobs (1994) is related to RFWR as the mixture of experts approach looks for similar partitions of the input space, particularly in the version of Xu et al. (1995). Ormeneit and Tresp (1995) suggested methods to improve the generalization of mixture models when fit with the EM algorithm (Dempster et al, 1977) by introducing Bayesian priors. Closely related to the hierarchical mixture of experts are nonparametric decision-tree techniques, in which the seminal work of Breiman, Friedman, Olshen, and Stone introduced classification and regression trees (CART), and Friedman (1991) proposed the MARS algorithm, a CART derivative particularly targeted at smooth function approximation for regression problems.

Finally, adaptive receptive fields and the way receptive fields are created in RFWR resemble in part the classification algorithms of Reilly, Cooper, and Elbaum (1982) and Carpenter and Grossberg (1987).

7 Discussion

This paper aims at emphasizing two major points. First, truly local learning—i.e., learning without competition, without gating nets, without global regression on top of the local receptive fields—is a feasible approach to learning, and, moreover, it can compete with state-of-the-art learning systems. Second, truly incremental learning—i.e., learning without knowledge about the input and conditional distributions, learning that must cope with continuously incoming data with many partially redundant and/or partially irrelevant inputs—needs to have a variety of mechanisms to make sure that incremental learning is robust. A carefully designed truly local learning system can accomplish this robustness.

In order to be a truly local learning system, RFWR borrowed in particular from work in nonparametric statistics. Following the definition of Hájek (1969), the term "nonparametric" indicates that the function to be modeled potentially consists of very large families of distributions which cannot be indexed by a finite-dimensional parameter vector in a natural way. This view summarizes the basic assumptions of our learning system, with the addition of prior knowledge about smoothness. It should be stressed that, if more prior knowledge is available for a particular problem, it should be incorporated in the learning system. It is unlikely that a nonparametric learner outperforms problem-tailored parametric learning—e.g., fitting sinusoidal data with a sinusoid is the best one can do. The examples given throughout this paper were to highlight when local nonparametric learning can be advantageous, but there is no claim that it is generally superior over other learning systems. On the other hand, when it comes to learning without having strong prior knowledge about the problem, nonparametric methods can be quite beneficial. For instance, Quartz and Sejnowski (submitted) claim that constructive nonparametric learning might be one of the key issues to understand the development of the organization of brains.

In order to achieve its properties, RFWR had to make use of several new algorithmic features. We introduced a stochastic approximation to leave-one-out local cross validation, i.e., cross validation which does not need a validation set anymore. This technique can potentially be useful for many other domains as it only requires that the (local) parameters to be estimated are linear in the inputs. By employing a novel penalized local cross validation criterion, we were able to derive locally adaptive multidimensional distance metrics. These distance metrics can be interpreted as local approximations of the Hessians of the function to be modeled. In order to speed up learning of the distance metric, we derived a second order gradient descent method. Finally, the penalized local cross validation criterion could also be employed to achieve automatic local bias adjustment on the relevance of input dimensions, obtained by local ridge regression. Using all these features, the constructive process of RFWR only needs to monitor the activation strength of all receptive fields in order to decide when to create a new receptive field—most constructive learning system need to monitor an approximation error criterion as well, which can easily lead to an unfavorable bias-variance tradeoff.

Despite the merits of RFWR, several issues have not been addressed in this paper and are left to future research. RFWR makes use of gradient-based learning which requires a proper choice of learning rates. Even though we incorporated second order learning derived from Sutton (1992a,b), it is still necessary to do some experimentation with the choice of the learning rates in order to achieve close to optimal learning speed without entering unstable domains. It is also necessary to choose an appropriate initial distance metric **D** (cf. Equation (5)), characterizing the initial size of a receptive field. Too large an initial receptive field has the danger that the receptive field grows to span the entire input domain: the initial receptive field has to be such that structure in the data cannot be mistaken with high variance noise. As a positive side-effect of truly local learning, however, these open parameters can be explored by allowing just a small number of receptive fields on an initial data set and monitoring their learning behavior—each receptive field learns independently and there is no need to do parameter exploration with a large number of receptive fields.

A last algorithmic point concerns computational complexity. Recursive least squares is an $O(n^2)$ process, i.e., quadratic in the number of inputs, and the update of a full distance metric is worse than $O(n^2)$. If the dimensionality of the inputs goes beyond about 10, a learning task with many receptive fields will run fairly slowly on a serial computer. Fitting only diagonal distance metrics alleviates this effect and might be necessary anyway since the number of open parameters in the learning system might become too large compared to the number of training data points.

This discussion naturally leads to the long standing question of how local learning methods can deal with high dimensional input spaces at all. As nicely described in Scott (1992), the curse of dimensionality has adverse effects on all systems which make use of neighboring points in the Euclidean sense, since the concept of "neighborhood" becomes gradually more counterintuitive when growing beyond 10 input dimensions, and it pretty much vanishes beyond 20 dimensions: every point is about the same distance from every other point. In such domains, the parametric model chosen for learning—be it local or global—becomes the key to success, essentially meaning that any learning system requires strong biases in high-dimensional worlds. However, it still remains unclear whether high dimensional input spaces have *locally* high dimensional distributions. Our experience in sensorimotor learning is that this may not be true for many interesting problems, as physical systems do not realize arbitrary distributions. For instance, a seven degree-of-freedom anthropomorphic robot arm, whose inverse dynamics model requires learning in a 21-dimensional input space, seems to realize locally not more than 4-8 dimensional input distributions. Thus, a future research goal will be to incorporate local dimensionality reduction as a preprocessing step in every receptive field (Vijayakumar & Schaal, submitted).

As a last point, one might wonder in how far a local learning system like RFWR could have any parallels with neurobiological information processing.

Particularly inspired by work on the visual cortex, one of the mainstream assumptions about receptive field-based learning in the brain is that receptive fields are broadly tuned and widely overlapping, and that the size of the receptive fields does not seem to be a free parameter in normal learning (as opposed to developmental and reorganizational processes after lesions, e.g., Merzenich, Kaas, Nelson, Sur, & Felleman, 1983). This view emphasizes that accuracy of encoding must be achieved by subsequent postprocessing steps. In contrast, RFWR suggest overlapping but much more finely tuned receptive fields, such that accuracy can be achieved directly by one or several overlapping units. Fine tuning can be achieved not only by a change of the size of the receptive field, but also by "plug-in" approaches where several receptive fields tuned for different spatial frequencies contribute to learning (Cannon & Slotine, 1995). To distinguish between those two principles, experiments that test for interference and generalization during learning can provide valuable insights into the macroscopic organization of learning. In motor control, the experiments by Shadmehr and Mussa-Ivalidi (1994), Imamizu, Uno, and Kawato (1995), and Shadmehr, Brashers-Krug, and Mussa-Ivaldi (1995) are examples of such investigations.

Whether the learning principles of RFWR are biologically relevant or not remains speculative. What we have demonstrated, however, is that there are alternative and powerful methods to accomplish incremental constructive learning based on local receptive fields, and it might be interesting to look out for cases where such learning systems might be applied. Receptive field-based local learning is an interesting research topic for neural computation, and truly local learning methods are just starting to demonstrate their potential.

8 Acknowledgments

This research was partly funded by the ATR Human Information Processing Research Laboratories. Additional support for S. Schaal was provided by the German Research Association, the German Scholarship Foundation, and the Alexander von Humboldt Foundation. Support for C. Atkeson was provided under Air Force Office of Scientific Research grant F49-6209410362, and by a National Science Foundation Presidential Young Investigator Award.

9 Appendix

9.1 Ridge Regression Derivatives

Each ridge regression parameter can be conceived of as a weighted data point of the form $[\mathbf{x}_r = r_i^2(0,\ldots,1,0,\ldots)^T, \mathbf{y}_r = 0]$ which was incorporated in the regression by the recursive least squares update (8). Thus, the derivative of the cost function (12) is a simplified version of the derivative (17):

$$\frac{\partial J}{\partial r_i} = \frac{2r_i}{W^{n+1}}\left(-\left(2\mathbf{P}^{n+1}\mathbf{x}_r\left(\mathbf{y}_r - \mathbf{x}_r^T\beta^{n+1}\right)^T\right)\otimes\mathbf{H}^{n+1} - \left(2\mathbf{P}^{n+1}\mathbf{x}_r\mathbf{x}_r^T\mathbf{P}^{n+1}\right)\otimes\mathbf{R}^{n+1}\right) \tag{32}$$

By taking advantage of the many zero elements of the ridge "data points", the actual computation of this derivative is greatly sped up.

There are several ways to incorporate the update of the ridge regression parameters in the matrix **P**, and it should be noted that we also need to add back the fraction of the ridge parameters which was forgotten due to the forgetting factor λ in each update of **P** (Equation (8)). It turns out, that there is a quite efficient way to perform this update. At every update of a receptive field, the forgetting factor effectively reduces the contribution of each ridge parameter by:

$$\Delta_{\lambda,i} = (1-\lambda)r_i^2 \tag{33}$$

The update due to gradient descent is:

$$\Delta_{grad,i} = \left(r_i + \Delta r_i\right)^2 - r_i^2 \tag{34}$$

and the total increment becomes:

$$\Delta_i = \Delta_{\lambda,i} + \Delta_{grad,i} = (1-\lambda)r_i^2 + \left(r_i + \Delta r_i\right)^2 - r_i^2 = -\lambda r_i^2 + \left(r_i + \Delta r_i\right)^2 \tag{35}$$

Due to the fact that the ridge vectors are all unit vectors, it is possible to update **P** by just executing a recursive least squares update for the *increment*, i.e., to add a ridge data point of the form $[\mathbf{x}_r = \Delta_i(0,\ldots,1,0,\ldots)^T, \mathbf{y}_r = 0]$ for every ridge parameter by using Equation (8). This update can be accelerated by taking into account the zeros in the ridge points. An additional speed up can be obtained by not updating **P** every iteration but rather by accumulating the increments until they exceed a manually chosen threshold.

9.2 Second Derivatives of Distance Metric Update

The second derivative of the cost function (12) with respect to the coefficients of the distance metric is:

$$\frac{\partial J}{\partial \mathbf{M}} \approx \frac{\partial w}{\partial \mathbf{M}}\sum_{i=1}^{p}\frac{\partial J_{1,i}}{\partial w} + \frac{w}{W^{n+1}}\frac{\partial J_2}{\partial \mathbf{M}} \tag{36}$$

$$\frac{\partial^2 J}{\partial \mathbf{M}^2} \approx \frac{\partial^2 w}{\partial \mathbf{M}^2}\sum_{i=1}^{p}\frac{\partial J_{1,i}}{\partial w} + \frac{\partial w}{\partial \mathbf{M}}\sum_{i=1}^{p}\frac{\partial^2 J_{1,i}}{\partial w^2}\frac{\partial w}{\partial \mathbf{M}} + \frac{w}{W^{n+1}}\frac{\partial^2 J_2}{\partial \mathbf{M}^2}$$

where :

$$\frac{\partial^2 w}{\partial M_{rl}^2} = \frac{1}{w}\left(\frac{\partial w}{\partial \mathbf{M}}\right)^2 - w\left(x_l - c_l\right)^2, \quad \frac{\partial^2 J_2}{\partial M_{rl}^2} = 2\gamma\left(2D_{ll} + \sum_{i,j=1}^{n}\left(\frac{\partial D_{ij}}{\partial M_{rl}}\right)^2\right)$$

$$
\begin{aligned}
\sum_{i=1}^{p} \frac{\partial^2 J_{1,i}}{\partial w^2} \approx & -\frac{\mathbf{e}_{cv}^T \mathbf{e}_{cv}}{\left(W^{n+1}\right)^2} \\
& -\frac{2}{W^{n+1}} \left(\left(-\frac{\mathbf{I}}{W^{n+1}} - 2\mathbf{P}^{n+1} \tilde{\mathbf{x}} \tilde{\mathbf{x}}^T \right) \mathbf{P}^{n+1} \tilde{\mathbf{x}} \left(\mathbf{y} - \tilde{\mathbf{x}}^T \beta^{n+1} \right)^T \right) \otimes \mathbf{H}^n \\
& + \frac{2}{W^{n+1}} \frac{\left(\mathbf{y} - \tilde{\mathbf{x}}^T \beta^{n+1} \right)^T \left(\mathbf{y} - \tilde{\mathbf{x}}^T \beta^{n+1} \right) h}{w} \\
& -\frac{1}{\left(W^{n+1}\right)^2} \left(\mathbf{e}_{cv}^T \mathbf{e}_{cv} - 2\left(\mathbf{P}^{n+1} \tilde{\mathbf{x}} \left(\mathbf{y} - \tilde{\mathbf{x}}^T \beta^{n+1} \right)^T \right) \otimes \mathbf{H}^n \right) + 2 \frac{E^{n+1}}{\left(W^{n+1}\right)^3}
\end{aligned}
$$

This equation makes use of notation and results derived in (16) and (17).

10 References

Aha, D. (in press). "Lazy Learning." *Artificial Intelligence Review.*

An, C. H., Atkeson, C. G., & Hollerbach, J. M. (1988). *Model-based control of a robot manipulator.*Cambridge, MA: MIT Press.

Atkeson, C. G., Moore, A. W., & Schaal, S. (in press). "Locally weighted learning." *Artificial Intelligence Review.*

Atkeson, C. G., Moore, A. W., & Schaal, S. (in press). "Locally weighted learning for control." *Artificial Intelligence Review.*

Atkeson, C. G. (1989a). "Learning arm kinematics and dynamics." *Annual Review Neuroscience,* 12, pp.157-83.

Atkeson, C. G. (1989b). "Using local models to control movement." In: Touretzky, D. (Ed.), *Advances in Neural Information Processing Systems 1,* pp.79-86 San Mateo, CA: Morgan Kaufmann.

Atkeson, C. G., & Schaal, S. (1995). "Memory-based neural networks for robot learning." *Neurocomputing,* 9, pp.243-269.

Belsley, D. A., Kuh, E., & Welsch, R. E. (1980). *Regression diagnostics: Identifying influential data and sources of collinearity.* New York: Wiley.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees.* Belmont, CA: Wadsworth International Group.

Bruske, J., & Sommer, G. (1995). "Dynamic cell structure learns perfectly topology preserving map." *Neural Computation,* 7, pp.845-865.

Cannon, M., & Slotine, J. E. (1995). "Space-frequency localized basis function networks for nonlinear system estimation and control." *Neurocomputing,* 9, 3, pp.293-342.

Carpenter, G. A., & Grossberg, S. (1987b). "A massively parallel architecture for a self-organizing neural pattern recognition machine." *Computer Vision, Graphics, and Image Processing,* 37, pp.54-115.

Churchland, P. S., & Sejnowski, T. J. (1992). *The computational brain.* Boston, MA: MIT Press.

Cleveland, W. S. (1979). "Robust locally weighted regression and smoothing scatterplots." *Journal of the American Statistical Association,* 74, pp.829-836.

Cleveland, W. S., Devlin, S. J., & Grosse, E. (1988a). "Regression by local fitting: Methods, properties, and computational algorithms." *Journal of Econometrics,* 37, pp.87-114.

Cleveland, W. S., & Devlin, S. J. (1988b). "Locally weighted regression: An approach to regression analysis by local fitting." *Journal of the American Statistical Association*, **83**, pp.596-610.

Cleveland, W. S., & Loader, C. (1995a). "Smoothing by local regression: Principles and methods." Technical Report, AT&T Bell Laboratories, Murray Hill, NY.

Daugman, J., & Downing, C. (1995). "Gabor wavelets for statistical pattern recognition." In: Arbib, M. A. (Ed.), *The Handbook of Brain Theory and Neural Networks*, pp.414-420. Cambridge, MA: MIT Press.

de Boor, C. (1978). *A practical guide to splines*. New York: Springer.

Deco, G., & Obradovic, D. (1996). *An information-theoretic approach to neural computation*. New York: Springer.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). "Maximum likelihood from incomplete data via the EM algorithm." *Journal of the Royal Statistical Society B*, **39**, pp.1-38.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis*. New York: Wiley.

Fan, J., & Gijbels, I. (1992). "Variable bandwidth and local linear regression smoothers." *The Annals of Statistics*, **20**, 4, pp.2008-2036.

Fan, J., & Gijbels, I. (1995). "Data-driven bandwidth selection in local polynomial fitting: Variable bandwidth and spatial adaptation." *Journal of the Royal Statistical Society B*, **57**, pp.371-395.

Fan, J., & Gijbels, I. (1996). *Local polynomical modelling and its applications*. London: Chapman & Hall.

Farmer, J. D., & Sidorowich (1987). "Predicting chaotic time series." *Phys. Rev. Lett.*, **59 (8)**, pp.845-848.

Farmer, J. D., & Sidorowich (1988b). "Exploiting chaos to predict the future and reduce noise." In: Lee, Y. C. (Ed.), *Evolution, Learning, and Cognition*, p.27. Singapore: World Scientific.

Field, D. J. (1994). "What is the goal of sensory coding?." *Neural Computation*, **6**, pp.559-601.

Frean, M. (1990). "The upstart algorithm: A method for constructing and training feedforward neural networks." *Neural Computation*, **2**, pp.198-209.

Friedman, J. H., & StŸtzle, W. (1981b). "Projection pursuit regression." *Journal of the American Statistical Association, Theory and Models*, **76**, 376, pp.817-823.

Friedman, J.H. (1984). "A variable span smoother." Technical Report No.5, Department of Statistics, Stanford University.

Friedman, J. H. (1991). "Multivariate adaptive regression splines." *The Annals of Statistics*, **19**, pp.1-141.

Fritzke, B. (1994b). "Growing cell structures _ A self-organizing network of unsupervised and supervised learning." *Neural Networks*, **7**, 9, pp.1441-1460.

Fritzke, B. (1995). "Incremental learning of locally linear mappings." In: *Proceedings of the International Conference on Artificial Neural Networks*, Paris, France, Oct.9-13.

Furlanello, C., & Giuliani, D. (1995a). "Combining local PCA and radial basis function networks for speaker normalization." In: Girosi, F., Makhoul, J., Manolakas, E., & Wilson, E. (Eds.), *Proceedings of the 1995 IEEE Workshop on Neural Networks for Signal Processing V*, pp.233-242. New York: IEEE.

Furlanello, C., Giuliani, D., & Trentin, E. (1995b). "Connectionist speaker normalization with generalized resource allocating networks." In: Tesauro, D., Touretzky, D. S., & Leen, T. K. (Eds.), *Advances in Neural Information Processing Systems 7*, pp.867-874. Cambridge, MA: MIT Press.

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

Georgopoulos, A. P. (1991). "Higher order motor control." *Annual Review of Neuroscience,* **14**, pp.361-377.

Hájek, J. (1969). *A course in nonparametric statistics*. San Francisco, CA: Holden-Day.

Hastie, T. J., & Tibshirani, R. J. (1990). *Generalized additive models*. London: Chapman and Hall.

Hastie, T., & Loader, C. (1993). "Local regression: Automatic kernel carpentry." *Statistical Science,* 8, pp.120-143.

Hastie, T. J., & Tibshirani, R. J. (1994c). "Nonparametric regression and classification: Part I: Nonparametric regression." In: Cherkassky, V., Friedman, J. H., & Wechsler, H. (Eds.), *From Statistics to Neural Networks: Theory and Pattern Recognition Applications. ASI Proceedings, subseries F, Computer and Systems Sciences*. Springer.

Hubel, D. H., & Wiesel, T. N. (1959). "Receptive fields of of single neurons in the cat's striate cortex." *Journal of Neurophysiology,* **148**, 574-591.

Imamizu, H., Uno, Y., & Kawato, M. (1995). "Internal representations of the motor apparatus: Implications from generalization in visuomotor learning." *Journal of Experimental Psychology,* **21**, 5, pp.1174-1198.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). "Adaptive mixtures of local experts." *Neural Computation,* **3**, pp.79-87.

Jordan, M. I., & Jacobs, R. (1994). "Hierarchical mixtures of experts and the EM algorithm." *Neural Computation,* **6**, 2, pp.181-214.

Jutten, C., & Chentouf, R. (1995). "A new scheme for incremental learning." *Neural Processing Letters,* **2**, 1, pp.1-4.

Kwok, T.-Y., & Yeung, D.-Y. (1995). "Constructive feedforward neural networks for regression problems: A survey." Technical Report HKUST-CS95-43, Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.

Lee, C., Rohrer, W. R., & Sparks, D. L. (1988). "Population coding of saccadic eye movement by neurons in the superior colliculus." *Nature,* **332**, pp.357-360.

Littmann, E., & Ritter, H. (1993). "Generalization abilities of cascade network architectures." In: Hanson, S. J., Cowan, J. , & Giles, C. L (Eds.), *Advances in Neural Information Processing Systems 5*, pp.188-195. Morgan Kaufmann.

Ljung, L., & Söderström, T. (1986). *Theory and practice of recursive identification*. Cambridge, MIT Press.

Lowe, D. G. (1995). "Similarity metric learning for a variable-kernel classifier." Neural Computation, .

Martinetz, T., & Schulten, K. (1994). "Topology representing networks." *Neural Networks,* 7, 3, pp.507-522.

Merzenich, M. M., Kaas, J. H., Nelson, R. J., Sur, M., & Felleman, D. (1983). "Topographic reorganization of somatosensory cortical areas 3b and 1 in adult monkeys following restricted deafferentation." *Neuroscience,* **8**, pp.33-55.

Millington, P. J. (1991). "Associative reinforcement learning for optimal control." Master Thesis CSDL-T-1070, Massachusetts Institute of Technology, Cambridge, MA.

Moody, J., & Darken, C. (1988). "Learning with localized receptive fields." In: Touretzky, D., Hinton, G., & Sejnowski, T. (Eds.), *Proceedings of the 1988 Connectionist Summer School,* pp.133-143. San Mateo, CA: Morgan Kaufmann.

Moore, A. (1991a). "Fast, robust adaptive control by learning only forward models." In: Moody, J. E., Hanson, S. J., & and Lippmann, R. P. (Eds.), *Advances in Neural Information Processing Systems 4*. San Mateo, CA: Morgan Kaufmann.

Mountcastle, V. B. (1957). "Modality and topographic properties of single neurons of cat's somatic sensory cortex." *Journal of Neurophysiology,* **20**, pp.408-434.

Myers, R. H. (1990). *Classical and modern regression with applications*. Boston, MA: PWS-KENT.

Nadaraya, E. A. (1964). "On estimating regression." *Theor. Prob. Appl.,* **9**, pp.141-142.

Olshausen, B. A., & Field, D. J. (1996). "Emergence of simple-cell receptive field properties by learning a sparse code for natural images." *Nature,* **381**, pp.607-609.

Ormoneit, D., & Tresp, V. (1995). "Improved Gaussian mixture density estimates using Bayesian penalty terms and network averaging." Technical Report FKI-205-95, Theoretical Computer Science and Foundations of Artificial Intelligence, Technische Universität München, Munich, Germnay.

Orr, M. J. L. (1995). "Regularization in the selection of radial basis function centers." *Neural Computation,* **7**, pp.606-623.

Papoulis, A. (1991). *Probability, random variables, and stochastic processes*. New York: McGraw-Hill.

Perrone, M. P., & Cooper, L. N. (1993). "When networks disagree: Ensemble methods for hybrid neural networks." In: Mammone, R. J. (Ed.), *Neural Networks for Speech and Image processing*. Chapman-Hall.

Platt, J. (1991). "A resource-allocating network for function interpolation." *Neural Computation,* **3**, pp.213-225.

Poggio, R., & Girosi, F. (1990). "Regularization algorithms for learning that are equivalent to multilayer networks." *Science,* **247**, 4945, pp.978-982.

Powell, M. J. D. (1987). "Radial basis functions for multivariate interpolation: A review." In: Mason, J. C., & Cox. M. G. (Eds.), *Algorithms for Approximation*, pp.143-167. Oxford: Clarendon Press.

Quartz, S. R., & Sejnowski, T. J. (submitted). "The neural basis of cognitive development: A constructivist manifesto." *Journal of Brain and Behavioral Sciences.*

Reilly, D. L., Cooper, L. N., & Elbaum, C. (1982). "A neural model for category learning." *Biological Cybernetics,* **45**, pp.35-41.

Ritter, H., & Schulten, K. (1986). "Topology conserving mappings for learning motor tasks." In: Denker, J. S. (Ed.), *Neural Networks for Computing,* **151**, pp.376-380. AIP Conference Proceedings, Snowbird, Utah.

Sanger, T. D. (1991). "A tree-structured adaptive network for function approximation in high-dimensional spaces." *IEEE Transactions on Neural Networks,* **2**, 2, pp.285-293.

Sanger, T. D., Sutton, R. S., & Matheus, C. J. (1992). "Iterative construction of sparse polynomial approximations." In: Hanson, S. J., Moody, J. E., & Lippmann, R. P. (Eds.), *Advances in Neural Information Processing Systems 4*, pp.1064-1071. San Mateo, CA: Morgan-Kaufmann.

Schaal, S. (in press). "Learning from demonstration." In: *Advances in Neural Information Processing Systems 9.*

Schaal, S., & Atkeson, C. G. (1994a). "Robot juggling: An implementation of memory-based learning." *Control Systems Magazine,* **14**, 1, pp.57-71.

Schaal, S., & Atkeson, C. G. (1994b). "Assessing the quality of learned local models." In: Cowan, J. , Tesauro, G., & Alspector, J. (Eds.), *Advances in Neural Information Processing Systems 6*. San Mateo, CA: Morgan Kaufmann.

Schaal, S., & Atkeson, C. G. (1996). "From isolation to cooperation: An alternative of a system of experts." In: Touretzky, D. S., Mozer, M. C., & Hasselmo, M. E. (Eds.), *Advances in Neural Information Processing Systems 8*, pp.605-611. Cambridge, MA: MIT Press.

Scott, D. W. (1992). *Multivariate Density Estimation.* New York: Wiley.

Shadmehr, R., Brashers-Krug, T., & Mussa-Ivaldi, F. A. (1995). "Interference in learning internal models of inverse dynamics in humans." In: Tesauro, G., Touretzky, D. S., & Leen, K. T. (Eds.), *Advances in Neural Information Processing Systems 7.*.

Shadmehr, R., & Mussa-Ivaldi, F. A. (1994b). "Adaptive representation of dynamics during learning of a motor task." *Journal of Neuroscience,* **14**, 5, pp.3208-3224.

Shin, Y., & Ghosh, J. (1995). "Ridge polynomial networks." *IEEE Transactions on Neural Networks,* **6**, 2. pp.610-622.

Stone, M (1974). "Cross-validatory choice and assessment of statistical predictors." *Journal of the Royal Statistical Society,* **B36**, 111-147.

Sutton, R. S. (1992a). "Gain adaptation beats least squares." In: *Proceedings of Seventh Yale Workshop on Adaptive and Learning Systems,* pp.161-166, New Haven, CT.

Sutton, R. S. (1992b). "Adapting bias by gradient descent: An incremental version of Delta-Bar-Delta." In: *Proceedings of the Tenth National Conference on Artificial Intelligence,* pp.171-176, Cambridge, MA: MIT Press.

van der Smagt, P., & Groen, F. (1995). "Approximation with neural networks: Between local and global approximation." In: *Proceedings of the 1995 International Conference on Neural Networks,* **II**, pp.1060-1064, Perth, Austrialia.

Vijayakumar, S., & Schaal, S. (submitted). "Local dimensionality reduction for locally weighted learning."

Wahba, G., & Wold, S. (1975). "A completely automatic french curve: Fitting spline functions by cross-validation." *Communications in Statistics,* **4 (1)**, .

Wahba, G. (1990). *Spline models for observational data.* Philadelphia, PA: Society for Industrial and Applied Mathematics.

Watson, G. S. (1964). "Smooth regression analysis." *Sankhaya: The Indian Journal of Statistics A,* **26**, pp.359-372.

Xu, L., Jordan, M. I., & Hinton, G. E. (1995). "An alternative model for mixture of experts." In: Tesauro, G., Touretzky, D. S., & Leen, T. K. (Eds.), , pp.633-640. Cambridge, MA: MIT Press.

Zipser, D., & Anderson, R. A (1988). "A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons." *Nature,* **331**, 6158, pp. 679-684.