A Recursive Local Polynomial Approximation Method using Dirichlet Clouds and Radial Basis Functions

Arta A. Jamshidi and Warren B. Powell
Department of Operations Research and Financial Engineering
Princeton University,
Princeton, NJ 08544.
e-mail:{arta,powell}@princeton.edu

Abstract

We present a recursive function approximation technique that does not require the storage of the arrival data stream. Our work is motivated by algorithms in stochastic optimization which require approximating functions in a recursive setting such as a stochastic approximation algorithm. The unique collection of these features in this technique is essential for nonlinear modeling of large data sets where the storage of the data becomes prohibitively expensive and in circumstances where our knowledge about a given query point increases as new information arrives. The algorithm presented here provides locally adaptive parametric models (such as linear models). The local models are updated using recursive least squares and only stores the statistical representative of the local approximations. The resulting scheme is very fast and memory efficient without compromising accuracy in comparison to the standard and some advanced techniques used for functional data analysis in the literature. We motivate the algorithm using synthetic data and illustrate the algorithm on several real data sets.

1 Introduction

There are three major classes of function approximation methods: look-up tables, parametric models (linear or non-linear), and nonparametric models. Parametric regression techniques (such as linear regression [34]) assume that the underlying structure of the data is known a priori and is in the span of the regressor function. Due to the simplicity of this approach it is commonly used for regression. Nonparametric models,[14, 36, 55], do not assume a specific structure underlying the data. Nonparametric methods use the raw data to build local approximations of the function, producing a flexible but data-intensive representation. Although nonparametric models are generally data hungry, the resulting approximation may be more accurate, [19, 26, 53] and is less sensitive to structural errors arising from a parametric model. Most nonparametric models require keeping track of all observed data points, which make function evaluations increasingly expensive as the algorithm progresses, a serious problem in stochastic search. Regression based on least squares, Lasso regression, ridge regression, and least absolute deviation all have the same underlying structure as ordinary least squares, except the objective/penalty terms are slightly modified. In this work we formulate a problem that is a balance between parametric and nonparametric models to benefit from the advantages of both modeling strategies. Semi-parametric models are discussed in [51].

Our work is motivated by the need to approximate functions within stochastic search algorithms where new observations arrive recursively. The most classical representation of the problem is given by

$$\min_x \mathbb{E}\, F(x,\nu),$$

where $\mathbb{E}$ denotes the expectation operator, x is a deterministic parameter and ν is a random variable. Other applications include approximate dynamic programming, where we need to approximate expectations of value functions. Since we are unable to compute the expectation directly, we depend on the use of Monte Carlo samples. We are interested in the class of algorithms which replaces $\mathbb{E}\, F(x,\nu)$ with an approximation $\overline{F}^n(x)$ which can be quickly optimized. As we obtain new information from each iteration, we need a fast and flexible method for updating the approximation $\overline{F}^n(x)$. We require an approximation method that is more flexible than classical parametric models will offer, but we need a fast, compact representation to minimize computational overhead.

Bayesian techniques for function approximation or regression are computationally intense and require storage of all the data points [19]. Updating the model requires revisiting all the data points in the history. These limitations make these methods impractical in the circumstances where there is a need to update the model as a stream of new information arrives and in situations where there is a need for fast algorithms with limited storage space. In addition these methods have multiple tunable parameters. There is a rich literature on nonparametric Baysian methods for regression [24]. In spirit, our work is closest to [52] with a proposed model that mixes over both the covariates and response, and the response is drawn from a multinomial logistic model. This work handles various response shapes. Dirichlet process mixtures of generalized linear models (DP-GLM) proposed in [24] is a generalization of this idea for various response types. Other statistical techniques that are widely used for regression include regression trees [4], where data is divided into a fixed, tree-based partitioning and a regression model is fit to data in each leaf of the tree, and Gaussian processes, which assume the observations arise from a Gaussian process model with known covariance function. This method does not handle variations in the variance of the response unless a Dirichlet process mixtures of GP is assumed [46].

Locally Weighted Scatterplot Smoothing (LOWESS), [11, 12, 13], does not require a global function to represent the underlying function. This procedure produces a model based on segments of data based on nearest neighborhood. At each point this method assigns a low-degree polynomial using weighted least squares to the segment of data that are closest to the point of interest. The weighted least squares assigns more weight to the points that are closer to the query point. This method is computationally intensive and keeps track of all the data points. In addition, it does not incorporate an updating method as new data points arrive.

Locally linear models [15, 16], is a weighted locally linear regression method that has advantages over regular kernel smoothing regression methods such as [56] and [38]. This technique builds linear models around each observation and keeps track of all the data points. In addition this method does not provide a global mathematical formula for the regression function. This method performs better than spline methods [15].

The multilayer-perceptron and the associated back-propagation algorithm,[57, 50], and radial basis function [5, 42, 41] have received considerable attention for function approximation. Back-propagation networks require a lot of training data and training and as a result are quite slow. One of the main attractions of the radial basis functions (RBFs) is that the resulting optimization problem can be broken efficiently into linear and nonlinear subproblems. An automatic function approximation technique using RBF that does not need tuning ad-hoc parameters is proposed in [29]; the multivariate extension to this technique is proposed in [31]. For a comprehensive treatment on various growing RBF techniques, see [29] and references therein. The issue of selecting the number of basis functions with growing and pruning algorithms from a Bayesian prospective is described in [27]. In [1], a hierarchical full Bayesian model for radial basis functions (RBFs) is proposed. Normalized RBFs are presented in [32] which perform well for approximation for less number of training data.

We seek a general purpose approximation method that can approximate a wide range of functions using a compact representation which can be updated recursively with fast function evaluations. Toward this goal, we propose a novel recursive and fast approximation scheme for modeling nonlinear data streams that does not require storage of the data history. The covariates may be continuous or categorical, but we assume that the response function is continuous, although not necessarily differentiable. The new method is robust in the presence of homoscedastic and heteroscedastic noise. Our method automatically determines the model order for a given data set and produces an analytical formulation describing the underlying function. Unlike similar algorithms in the literature, our method has one tunable parameter. Our proposed scheme introduces a cover over the input space to define local regions. This scheme only stores a statistical representation of data in local regions and locally approximates the data with a low order polynomial such as a linear model. The local model parameters are quickly updated using recursive least squares. A nonlinear weighting system is associated with each local model which determines the contribution of this local model to the overall model output. The combined effect of the local approximations and the weights associated to them produces a nonlinear function approximation tool. On the data sets considered in this study, our new algorithm provides superior computational efficiency, both in terms of computational time and memory requirements compared to the existing nonparametric regression methods without loss of accuracy. The test results on various synthesized and real multivariate data sets are carried out and compared with nonparametric regression methods in the literature. In this work we have made an attempt to bridge various fields in statistics, approximation and numerical analysis on function approximation.

The organization of this paper is as follows: Section 2 provides an overview of radial basis functions and motivates the use of normalized Radial Basis Functions with polynomial modulated response terms. Section 3 introduces various aspects of the proposed algorithm including the notion of a cloud cover for the input space, the model building procedure and the updating roles. This section also provides the stopping criterion for the algorithm and the procedure to measure the goodness of fit. Section 4 shows the convergence properties of the proposed algorithm in finite time and asymptotically. Section 5 demonstrates the performance and robustness of the algorithm using various synthetic and real data sets with various input dimensions and noise levels. This section provides a comparison of the new method and the available algorithms in the literature. Section 6 provides concluding remarks and discusses future work.

2 Normalized Radial Basis Functions

Radial Basis Functions are powerful tools for function approximation, [5, 7, 43]. Over the years RBFs have been used successfully to solve a wide range of function approximation problems (see e.g., [3]). RBFs have also been used for global optimization (see e.g., [48, 49, 23, 28]). An RBF expansion is a linear summation of special nonlinear basis functions. In general, an RBF is a mapping $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ that is represented by

$$f(x) = \sum_{i=1}^{N_c} \alpha_i \phi(\|x - c_i\|_{W_i}), \tag{1}$$

where x is an input pattern, ϕ is the radial basis function centered at location c_i, and α_i denotes the weight for the ith RBF. N_c denotes the total number of RBFs. The term W_i denotes the parameters in the weighted inner product

$$\|x\|_W = \sqrt{x^T W x}.$$

The dimensions of the input n and output m are specified by the dimensions of the input-output pairs. Universal approximation properties of RBFs are established in [39, 40].

Normalized RBFs of the following form were proposed by Moody and Darken, [35],

$$f(x) = \frac{\sum_{i=1}^{N_c} \alpha_i \phi(\|x - c_i\|_{W_i})}{\sum_{i=1}^{N_c} \phi(\|x - c_i\|_{W_i})}. \tag{2}$$

Normalized RBFs appear to have advantages over regular RBF, especially in the domain of pattern classification, see e.g., [6]. In this work it is reported that using normalized RBFs reduced the order of the model and the robustness of generalization. It has also been reported that normalized RBFs require less data when training models of dynamical systems, [32].

The polynomial modulation of the normalized RBF expansion leads to the following expansion,

$$f(x) = \frac{\sum_{i=1}^{N_c} p_i(x) \phi(\|x - c_i\|_{W_k})}{\sum_{i=1}^{N_c} \phi(\|x - c_i\|_{W_i})}, \tag{3}$$

where $p_i(x)$ is assumed to be a low order polynomial such as a linear function ($p_i(x) = \alpha_i x + \beta_i$).

The above notion leads us to the idea of normalized radial basis functions that have a polynomial response terms. In what follows we provide a fast algorithm that recursively updates the response terms and the weights associated to them as new data points arrive. During the training phase the unknown parameters in the model need to be calculated including the number of the RBFs in the function expansion.

3 On-line Learning Scheme

The goal of our proposed algorithm is to find a mapping f from $x \in \mathbb{R}^n$, a vector in n-dimensional space, to $\mathbb{R}$ such that $y^k = f(x^k)$ as the input-output pair $\{(x^k, y^k)\}$ become available. We expect to have a total of K arrival data points where $k \in \{1, ..., K\}$ with $\mathcal{X} = \{x^k\}_{k=1}^K$ and $\mathcal{Y} = \{y^k\}_{k=1}^K$ representing the domain and range observable values. We assume that data arrives as a stream. Throughout the process we would like to find the parameters associated with the model provided in Equation (3) to find an accurate model for the data. The observed data might be noisy and we would like to have a model that has good generalization ability. In this regression framework, the problem is to minimize the cost function

$$E(\psi) = \frac{1}{2} \sum_{k=1}^{K} \| f(x^k, \psi) - y^k \|^2,$$

given the available data points. ψ contains all the parameters, $\alpha_i, \beta_i, c_i, W_i, N_c$, that are used to define $f(x)$ as shown in Equation (3). In this work, the Euclidean inner product is used for the metric $\|\cdot\|$ and we use the Gaussian kernel,

$$\phi(r) = \exp(-r^2).$$

Note that other kernels could be used. For a comprehensive review of RBF kernels used in the literature and novel skew and compactly supported expansions (see [30]). In what follows the description of an algorithm in this framework is presented that does not require the storage of the data stream and locally approximates the underlying functional behavior of the data. The response model parameters are updated quickly using recursive least squares and the weights associated to each linear response is updated via statistical techniques. Note that the model order is also determined in the algorithm.

3.1 A Data Driven Space Cover

The algorithm proposed here works by forming a cover for the domain of the underlying function f upon arrival of new data points, x^k. We define a cover

$$C = \{U_i : i \in \Delta\}$$

where U_i denotes an indexed family of sets, indexed with the elements of set Δ. C is a cover of $\mathcal{X}$ if $\mathcal{X} \subseteq \bigcup_{i\in\Delta} U_i$. On the arrival of the first data point, x^1, a cover element or cloud U_1 is formed with the center on this point and with a distance threshold of D_T to form a ball around x^l. When the system receives a new data point x^2 with associated y^2 value, if the data point is within distance D_T of x^1, x^2 is assigned to the same cloud; otherwise a new cloud is created. The same procedure is repeated upon arrival of each data point x^k. A decision is made for the assignment of the new data to a new or existing cloud. If an existing cloud is assigned, the mean and the variance of this cloud is updated. Otherwise, a new cloud is created with the new point as its centroid. In the case that the arrival data is assigned to an existing cloud, the response of the cloud is updated using recursive least squares.

To check the assignment of a new arrival data point x^k, the distance between x^k and the centroids of the cloud, c_i with $i \in \{1, ..., N_c\}$ (N_c is the number of clouds or elements in the cover), is calculated using

$$D_i = \|x^k - c_i\|. \tag{4}$$

Let

$$I = \arg\min_i D_i.$$

Compare D_I, with distance threshold D_T; if $D_I < D_T$, update the ball indexed with U_I. Otherwise spawn a new cloud U_{N_c+1}, according to the previous description. When a new element is added to the cloud the number of the local balls, N_c is also updated to $N_c = N_c + 1$.

The algorithm retains a statistical representation of the data over a cover that is specified by the data stream and a distance threshold. The distance threshold is the only parameter in this algorithm that requires tuning. The value of this threshold depends on the variations of the response values over the domain. The data that is stored in the model are the number of points in each cloud, K_i, the centroid of each cloud, c_i, and the variance in each dimension of the data in each cloud, W_i. In addition the response coefficients for each cloud are also stored. Note that the model can be evaluated at any point of its domain after adapting the model using the new information. As the geometry of the data becomes more complex, there will be a need for more clouds in the cover to capture this behavior.

3.2 Recursive Update of the Model

There are two parts in the model that are updated during the training: the response to each local model associated to a local ball indexed I and an associated weight.

Recursive update for responses. We solve a local least squares problem of the form $\min_{\theta_I} \| X_I^{k_I}\theta_I - Y_I \|_2^2$ where $X_I^{k_I}$ is a matrix of the form

$$X_I^{k_I} = \begin{bmatrix} 1 & .. & x^1 \\ : & & : \\ 1 & .. & x^{k_I} \end{bmatrix},$$

where the vectors $x^1, ..., x^{k_I}$ are domain values. The vector $Y_I^{k_I}$ contains all the associated range values $Y_I^{k_I} = [y^1, ..., y^{k_I}]$. The vector $\theta_I = [\beta^I, \alpha_1^I, ..., \alpha_n^I]$ contains all the parameters for the linear response. The direct solution to this system is $\theta_I = ({X_I^{k_I}}^T X_I^{k_I})^{-1} {X_I^{k_I}}^T Y_I^{k_I}$. After receiving $n+1$ data points in a local neighborhood (we refer to this state of the model as the no-knowledge state), the matrix $X_I^{k_I}$ is formed with $k_I = n+1$. The recursive least squares update, [10], is used to compute the linear response model parameters. For $k_I = n+1$, let $P_I^{k_I} = ({X_I^{k_I}}^T X_I^{k_I})^{-1}$. When a new data point (x^k, y^k) is assigned to this local neighborhood, let $a^T = [x^k, 1]$.

The recursion formula is then:

$$P_I^{k_I+1} = P_I^{k_I} - \frac{P_I^{k_I} a a^T P_I^{k_I}}{1 + a^T P_I^{k_I} a}, \tag{5}$$

$$\theta_I^{k_I+1} = \theta_I^{k_I} + P_I^{k_I+1} a \left(y^k - a^T \theta_I^{k_I} \right), \tag{6}$$

where k_I is the recursion index for cloud indexed I. These equations are easily modified to include a discount factor which puts more weight on recent observations.

Recursive update for the weights. When a new data point (x^k, y^k) is assigned to the local neighborhood indexed I, the weights W_I associated with an element of cover C, cloud U_I, is also updated. For this purpose the number of data points that have been assigned to this local ball is updated as $k_I = k_I + 1$. The center of the kernels are updated via the recursion

$$c_I^{k_I+1} = c_I^{k_I} + \frac{x^k - c_I^{k_I}}{k_I}. \tag{7}$$

The width or scale of the model in each dimension of the cloud I is updated using the Welford formula [33]. Initialize $Q_1 = x^1$ and $S_1 = 0$. For each additional data point x^k assigned to this cloud, use the recurrence formulas

$$Q^{k_I} = Q^{k_I-1} + (x^k - Q^{k_I-1})/k_I, \tag{8}$$

$$S^{k_I} = S^{k_I-1} + (x^k - Q^{k_I-1})(x^k - Q^{k_I}). \tag{9}$$

The k_Ith estimate of the variance is $W_I^2 = \frac{1}{k_I-1} S^{k_I}$.

3.3 Model Evaluation

This section describes how to compute the approximation $\overline{F}^k(x)$ at a query point x. As shown in equation (3), the output of the proposed model is the weighted average of the linear responses over the local balls U_i associated to the cover C. In this study a Gaussian kernel, $\phi(r) = \exp(-r^2)$, determines the weight of each response function. The widths, W_i, and the centroids, c_i, of the kernels, $i \in \{1, ..., N_c\}$, were computed recursively as mentioned in Section 3.2. The model output is formulated as

$$f(x) = \frac{\sum_{i=1}^{N_c} \phi(\|x - c_i\|_{W_i})(\alpha_i x + \beta_i)}{\sum_{i=1}^{N_c} \phi(\|x - c_i\|_{W_i})}. \tag{10}$$

N_c is the total number of clouds. Note that $\alpha_i x + \beta_i$ is the least squares solution of the response over the ith local ball.

The weighted inner product is calculated from the recursive update of the standard deviation of data in each dimension. For the cases where the standard deviation of the data along a specific dimension is zero we introduce a penalty term to avoid singularity by writing $W_i = W_i + P$, where P is a specified constant. Note that in this study the weight matrices are diagonal. The summary of the above procedure is presented in DC-RBF Algorithm.

Algorithm DC-RBF algorithm

```
Input $D_T$
Initialize $N_c = 0$, $k = 1$, $\Delta = \emptyset$
while $x^k$ do
  if $k = 1$ then
    $c_1 = x^1$, $N_c = 1$, $k_1 = 1$, $\Delta = \{1\}$, form cloud $U_1$
  else
    compute $D_i = \|x^k - c_i\|$ according to Equation (4) for all $i \in \Delta$
    compute $I = \arg\min_i D_i$
    if $D_I < D_T$ then
      update cloud $U_I$, $k_I = k_I + 1$
      if $k_I \geq n + 1$ then
        update the response parameter, $\theta_I$, using Equations (5) and (6)
        update the cloud centroid, $c_I$, using Equation (7)
        update the scale of cloud, $W_I$, using Equations (8) and (9)
      else
        store $x^k$ assigned to cloud $I$
      end if
    else
      spawn new cloud $U_{N_c+1}$, $N_c = N_c + 1$, $k_{Nc} = 1$, $\Delta = \{1, ..., N_c\}$
    end if
  end if
  $k = k + 1$
end while
evaluate the model as needed using Equation (10)
```

3.4 Goodness of Fit and Stopping Criteria

In data modeling one of the major goals is to model data in such a way that the model does not over or under fit the data that is not used for training but is generated from the same process as the training data. One general approach to finding such a smooth model to observed data is known as regularization. Regularization describes the process of fitting a smooth function through the data set using a modified optimization problem that penalizes variation [54]. A standard technique to achieve regularization is via cross-validation [21, 55]. Such methods involve partitioning the data into subsets of training, validation, and testing data; for details see, e.g., [26]. To determine the tunable parameter of the model proposed in this work, i.e., the distance threshold, D_T, we use t-fold cross validation techniques [18]. In this scheme, the original data set is randomly partitioned into t subsample sets. One of the t subsamples is chosen as the validation data for testing the model, and the remaining subsamples are used as training data. This process is repeated t times with each of the t subsamples used only once as the validation data. All the t results are averaged to provide an overall estimate of the error. The advantage of this method over repeated random sub-sampling is that all observations are used for both training and validation, and each observation is used for validation exactly once. We have chosen to use five-fold cross validation. In the first iteration, we mark the first fold as the test set, and use the other four folds (the training set) to build a model [25]. We record the accuracy of the model on the testing data set. This procedure is repeated for all five folds, and the mean squared error (MSE) on the test data sets is computed at each time using,

$$MSE = \frac{1}{L}\sum_{l=1}^{L}\left(f(x^l) - y^l\right)^2, \tag{11}$$

where L is the size of the testing data set. Over-fitting is avoided by tuning the parameters to minimize MSE calculated through cross validation, as opposed to minimizing the MSE of points within the training data set [26].

To increase the number of estimates, one could run the above t-fold cross validation multiple times. For this purpose the data needs to be reshuffled each round.

To determine the predictive ability of the models we use several measures to show the accuracy of the predictions. The R^2 value is calculated through the following formula:

$$R^2 = 1 - \frac{\sum_{l=1}^{L}(y^l - f(x^l))^2}{\sum_{l=1}^{L}(y^l - \bar{y})^2},$$

where $\bar{y} = \frac{1}{L}\sum_{l}^{T_L} y^l$ is the mean of y^l. The variance of the data with regards to the true values is calculated in the numerator. The denominator presents the variance of the model output with respect to the mean of the outcomes. This measure tells how good the model is performing in comparison to the case where the mean value is used as the predictor. An R^2 value of 1 indicates an exactly fitted model, while a value of 0 indicates a model that adds no predictive power. We have also used the notion of Mean Absolute Error to record the performance of the model. The MAE is defined as following:

$$MAE = \frac{1}{L}\sum_{l=1}^{L}|f(x^l) - y^l|. \tag{12}$$

In the worst case, if the distance threshold is set too high, the proposed method will fit a single hyperplane to the entire data set.

4 The Convergence Properties of the DC-RBF Algorithm

This section describes facts about the DC-RBF algorithm. Theorems describe the finite time and asymptotic behavior of this algorithm. Note that the proofs are carried out for the algorithm in steady state where there are enough clouds to cover the desired domain of the function and the clouds have been stabilized. Theorems are provided for both homoscedastic and heteroscedastic types of noise.

Assume we have

$$y^k = \theta x^k + \epsilon^k, \tag{13}$$

for $k \in \{1, ..., K\}$. Vector θ is the parameter that needs to be estimated. The set $\mathcal{D}^{\mathcal{K}} = \{(x^k, y^k)\}$, with $k \in \{1, ..., K\}$ denotes the collection of sequentially observed data points. The error variables, ϵ^k, are random.

Remark 1. Gauss-Markov theorem [17]: The Gauss-Markov theorem states that in a linear regression model in which the errors have expectation zero and are uncorrelated and have equal variances, the lowest possible mean squared error (variance) of the linear unbiased estimator (BLUE) of the coefficients is given by the ordinary least squares estimator. Actual errors need not be normal, nor independent and identically distributed (only uncorrelated and homoscedastic). That is, $\mathbb{E}(\epsilon^k) = 0$, $\text{var}(\epsilon^k) = \sigma^2 < \infty$ and $\text{cov}(\epsilon^{k_1}, \epsilon^{k_2}) = 0$ for $k_1 \neq k_2$ and $k_1, k_2 \in \{1, ..., K\}$. The proof of this follows from a contradiction on the mean and variance of another linear estimator for coefficient θ. An example could be adding another term to the least squares solution given by $(X^T X)^{-1} X^T$. It can be shown that the mean of this new estimator is not unbiased and it produces a variance that is greater than what is produced by the mean square error solution, i.e., $\sigma^2 (X^T X)^{-1}$.

Note that in our work we use recursive least squares solution on the local clouds as new data points arrive to that local region.

Lemma 1. For a given D_T, if the arrival data points sample an n dimensional ambient space, the DC-RBF Algorithm will produce null output for a cloud cover that does not accumulate $M_o = n + 1$ data points (n is the dimension of the ambient space). Therefore $D_T > 0.5||N_{M_o}(x^{k_1}) - x^{k_2}||$ for $x^{k_1}, x^{k_2} \in \mathcal{X}$. $N_{M_o}(x)$ denotes the M_oth nearest neighbor of point x.

Proof. According to the construction of the model in the DC-RBF algorithm, there is a need for $n + 1$ data points to construct a hyperplane from $\mathbb{R}^n$ to $\mathbb{R}$. Therefore if D_T is chosen to be a sufficiently small number then there is a chance that the model produces no output. □

Proposition 1. Given $D_T = \text{diam}(\mathcal{X})$, assuming the underlying data structure is a hyperplane (line in dimension 1) with homoscedastic additive noise the DC-RBF Algorithm is unbiased asymptotically and in finite time larger than $n + 1$. The diameter of domain $\mathcal{X}$ is defined as

$$\text{diam}(\mathcal{X}) = \max_{1 \leq k_1, k_2 \leq K} ||x^{k_1} - x^{k_2}||.$$

Proof. Under the hypothesis H_I^k that cloud I has a linear structure, the rank one update of the recursive least squares solution at each iteration given by equations (5) and (6) provides the best linear unbiased estimator for the coefficient θ. Note that in this argument the local iteration counter k_I (the kth data point that arrives to cloud I) is the same as the global counter, $k_I = k$. We offer a proof based on induction. At $k = n - 1$ the argument is true since the algorithm saves $n - 1$ data points to initialize matrix P and the LSE is computed directly. Let the argument be true for $k = m \in \mathbb{N}$, we show that at iteration $k = m + 1$, the rank one update solution via the recursive least squares solution is also unbiased. This follows from the derivation of the recursive least squares via the Sherman-Morrison formula,

$$(A + uv^T)^{-1} = A^{-1} - \frac{(A^{-1}u)(v^T A^{-1})}{1 + v^T A^{-1} u}.$$

Let $G^m = X^{m^T} X^m$ and $a^{m+1^T} = [x^{m+1}, 1]$. Then $\theta^{m+1} = G^{m+1^{-1}} X^{m+1^T} Y^{m+1} = G^{m+1^{-1}}(X^{m^T} Y^m + a^{m+1} y^m)$. Notice that $X^{m^T} Y^m = G^m G^{m^{-1}} X^{m^T} Y^m = G^m \theta^m = G^{m+1}\theta^m - a^{m+1} a^{m+1^T} \theta^m$. Therefore, $\theta^{m+1} = \theta^m - G^{m+1^{-1}} a^{m+1}(a^{m+1^T}\theta^m - y^m)$. Let $P^{m+1} = G^{m+1^{-1}} = (G^m + a^{m+1} a^{m+1^T})^{-1}$. Therefore,

$$\theta^{m+1} = \theta^m + P^{m+1} a^{m+1} \left(y^m - a^{m+1^T} \right).$$

Hence, at each iteration the estimate is unbiased.

Asymptotically, when the number of data points becomes very large the same induction is true. Therefore when the number of local iterations $k \to \infty$ we deduce that $\theta^{k+1} = \theta^k + P^{k+1} a \left(y^k - a^T \theta^k \right)$ leads to an unbiased estimator for θ, according to Remark 1. $\square$

Lemma 2. Given $D_T = \mathrm{diam}_1(X)$, assuming the underlying data structure is a hyperplane with heteroscedastic or correlated additive noise, the DC-RBF Algorithm is unbiased asymptotically and in finite time larger than $n+1$.

Proof. The proof follows the proof of Proposition 1 and incorporating the notion of generalized least squares. Generalized least squares assumes the conditional variance of Y given X is a known or estimated matrix Ω. This matrix is used as the weight in computing the minimum squared Mahalanobis length, hence, $\hat{\beta} = (X'\Omega^{-1}X)^{-1} X'\Omega^{-1} Y$. Ω rescales the input data to make it uncorrelated, then the Gauss-Markov theorem is applied. For the case where the variance of the noise changes, the weight matrix Ω is diagonal and this is a weighted least squares problem. In this case the weight associated to each point $x^k \in \mathcal{X}$ is $\frac{1}{\sigma^k}$.

$\square$

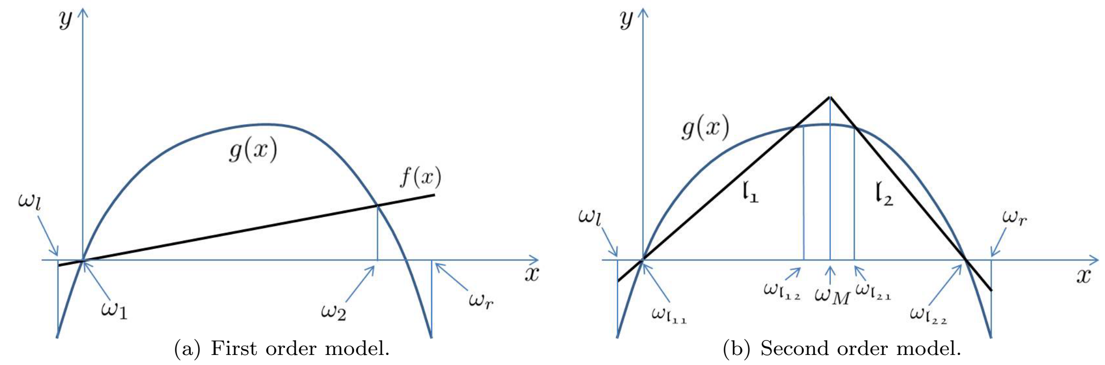
We denote the bias generated by the DC-RBF algorithm at each point of the domain of function $f(x)$ as $B(x) = (f(x) - \mathbb{E}[y|x])^2$. The variance at each point of the domain is $V(x) = \frac{1}{K}(f(x) - \bar{f}(x))^2$, where $\bar{f}(x)$ is the average of $f(x)$.

Proposition 2. If the underlying nonlinear functional structure $g(x)$ is Lipschitz continuous with Lipschitz constant L_c, and the observed data points are drawn from $y^k = g(x^k) + \epsilon^k$, then $\sum_{x \in \mathcal{X}} B(x) < Q(L_c, \omega_1, \omega_2) = L_c \frac{(\omega_1 - \omega_2)^2}{2} + O$. Here $D_T = \mathrm{diam}(\mathcal{X})$ is the chosen distance threshold, and ω_1 and ω_2 determine the intersecting points of the linear estimation and the underlying true structure $g(x)$ and O denotes a residual term.

Proof. We assume that the underlying function $g(x)$ is Lipschitz continuous, i.e., there are restrictions on the variations in the underlying structure. This implies that for every pair of points on the graph of this function, the absolute value of the slope of the line connecting them is no greater than a definite real number (Lipschitz constant, $L_c(g)$). The slope of the secant line passing through $x^{k_1}, x^{k_2} \in \mathcal{X}$ is equal to the difference quotient $\frac{g(x^{k_1}) - g(x^{k_2})}{||x^{k_1} - x^{k_2}||} < L_c$ when $x^{k_1} \neq x^{k_2}$. Without loss of generality we only consider the portion of the data that has an ascent followed by a descent slope in its structure. This could be achieved by the appropriate choice of D_T. To show this result we consider $\phi(.)$ to be uniform weights. First we show that the least squares solution to this structure intersects at two points, ω_1 and ω_2 (see Figure 1(a) for a visual presentation). The proof follows from a contradiction argument.

We denote the residuals of the least squares fit line at iteration k, $f^k(x) = a^k x$ with the truth function $g(x)$, as $e^k(x)$. Note that the residuals at each point are discrete samplings of the mapping

$$e : x \in [\omega_l, \omega_r] \to e^k(x) \in \mathbb{R}.$$



(a) First order model. (b) Second order model.

Figure 1: The visual demonstration of the regression lines intersection the underlying non-linear function $g(x)$. In this figure we provide the geometry for the case where the model consists of a single line, followed by the case where the model consists of two lines.

We assume that the data used for training is given by $\mathcal{X}_I \in U_I$ (index I denotes a specific cloud, in this case the whole data diameter with $I = 1$.) Note that the global iteration k is the same as the local index k_I. The following analysis is carried out for a given iteration k. This data is contained in the closed interval $x \in B_I = [\omega_l, \omega_r] \in U_I$. In addition, we assume that both g and f intersect at the origin of the coordinate system, i.e., $e(\omega_1) = 0$ and that $e(x) = 0$ at $x \notin B_I$.

Let a_+ denote the optimized parameters for the linear estimator f, i.e.,

$$a_+ = \operatorname*{argmin}_a \sum_{x^l \in B_I} (ax^l - e(x^l))^2.$$

We would like to show that $\exists\, x \in [\omega_l, \omega_r]$, such that

$$a_+x - e(x) = 0.$$

This indicates that the line a_+x intersects $g(x)$ at another point. If $a_+x - e(x) \neq 0$, $\forall\, x \in [\omega_r, \omega_l]$, then, without loss of generality, one could assume $a_+x - e(x) > 0$, $\forall\, x \in [\omega_l, \omega_r]$.

So, there must be a smallest distance between the optimal hyperplane a_+x and the residual curve $e(x)$. This distance is defined as

$$\alpha_+ = \min_{x \in [\omega_l, \omega_r]} a_+x - e(x).$$

Say this minimum occurs at the point x_+. Note that this point x_+ may not actually correspond to a sampled data point x^k.

Now consider another hyperplane a_*x which is obtained by scaling the optimal hyperplane so that it produces another point of intersection with $e(x)$, i.e.,

$$a_*x - e(x) = 0$$

for $x \neq 0$ and $x \in B_I$. By assumption we have $a_*x = \kappa a_+x$ where $0 < \kappa < 1$ so

$$0 < e(x) \leq a_*x < a_+x$$

for all $x \in (\omega_l, \omega_r)$. Hence $a_* x - e(x) < a_+ x - e(x)$ and

$$\sum_{x^l \in [\omega_l, \omega_r]} (a_* x - e(x^l))^2 < \sum_{x^l \in [\omega_l, \omega_r]} (a_+ x - e(x^l))^2,$$

which is a contradiction as we assumed a_+ was the optimal solution. Thus, the parameters a_+ could not have been obtained using least squares minimization.

Using the geometry that is produced by the intersection line and the nonlinear geometry, we could contain the residuals within three triangles that is generated by the lines passing through the intersecting points ω_1 and ω_2 with slop L_c (see Figure 1(a) for a visual presentation). We conclude that

$$\sum_{x^l \in B_I} |f(x^l) - g(x^l)| < L_c[\frac{(\omega_1 - \omega_2)^2}{2} + (\omega_l - \omega_1)^2 + (\omega_r - \omega_2)^2].$$

Note that B_I is an interval that contains the region of interest, $\mathrm{diam}(B_I) > |\omega_1 - \omega_2|$, where ω_r and ω_l are the right and left ends of the interval B_I, respectively. f^k is the line structure that is the least squares solution. We present this proof for a single dimension; however, the higher dimensional argument follows directly by considering a ball B_I with a diameter that covers the intersecting hyperplane with co-dimension 2 at $\{\Omega_1, ..., \Omega_n\}$ with $g(x)$. The argument follows the volume that is contained between the hyperplane of co-dimension 1 and the polyhedron (Tetrahedra in dimension 2) that contains the nonlinear structure and has slope equal to L_c for all its faces. □

Note that in one dimension we have the notion of a line and a tangent line and the model output is the weighted sum of piecewise linear approximation to the data. In higher dimensions we have the notion of a hyperplane with the tangent plane to make a weighted piecewise linear approximation to the underlying manifold.

Definition 1. Partition of Unity [37]: A partition of unity of a topological space $\mathcal{X}$ is a set of continuous functions, $\{\rho_i\}_{i \in \Delta}$, from $\mathcal{X}$ to the unit interval $[0, 1]$, such that for every point, $x \in \mathcal{X}$, there is a neighborhood of x where all but a finite number of the functions are 0, and the sum of all the function values at x is 1, i.e., $\sum_{i \in \Delta} \rho_i(x) = 1$.

The notion of partition of unity allows the extension of a local construction to the whole space. We use the following way to identify our partition of unity. Given any open cover U_i, $i \in \Delta$ (Δ is an index set) of a space, there exists a partition ρ_i, $i \in \Delta$, such that $\mathrm{supp}\,\rho_i \in U_i$ ($\mathrm{supp}\,\rho_i$ indicates the support of function ρ_j). Such a partition is said to be subordinate to the open cover U_i. Thus we choose to have the supports indexed by the open cover.

If functions ρ_i are compactly supported, given any open cover U_i, $i \in \Delta$, of a space, there exists a partition ρ_j, $j \in \Lambda$ indexed over a possibly distinct index set Λ such that each ρ_j has compact support and for each $j \in \Lambda$, $\mathrm{supp}\,\rho_j \in U_i$ for some $i \in \Delta$.

Theorem 1. Assuming the underlying data structure, $g(x)$, is nonlinear, and the distance threshold parameter, $D_{T_1} = \mathrm{diam}\mathcal{X}$, the DC-RBF Algorithm produces an upper bound on the bias denoted by $Q_{D_{T_1}}$ (Q is defined in Proposition 2); however if $D_{T_2} = \frac{\mathrm{diam}\mathcal{X}}{N}$ ($N \in \mathbb{N}$, is a natural number) with $N > 1$, and upper bound on the bias $Q_{D_{T_2}}$ then $Q_{D_{T_2}} < Q_{D_{T_1}}$.

Proof. As new data points arrive, more of the true underlying geometry reveals itself. We begin by posing as a null hypothesis that the underlying structure is described by a line. As the number of observations increases, we test this hypothesis against alternative hypotheses that the underlying structure is described by a series of lines over regions. At this stage the

DC-RBF algorithm given $D_T = \text{diam}\mathcal{X}$ attempts to model the secant line passing through the curvature (see Figure 1(a) for a visual presentation). What we mean by the secant line is the regression line passing through the data points associated to cloud indexed I, where at this stage $I = 1$. Note that the linear solution produces bias, with an upper bound provided in Proposition 2.

Similar to Proposition 2, we assume the regression line intersects the underlying nonlinear structure at points ω_1 and ω_2 with upper bound on the bias Q. The domain of interest is $B_I = [\omega_l, \omega_r]$. As D_T shrinks, (without loss of generality we assume, $D_T = \frac{\text{diam}\mathcal{X}}{2}$), the space is broken down into smaller sets (see Figure 1(b) for a visual presentation). For a one dimensional space, this choice of D_T breaks down the space into two parts, hence $\Delta = \{1, 2\}$. The RBFs, $\phi(r)$ with $\sum_i \frac{\phi(\|x-c_i\|_W)}{\sum_i \phi(\|x-c_i\|_W)} = 1$, form a partition of unity as provided in Definition 1. This would allow the expansion of each solution to a larger domain to produce a smooth transition from one local model to the neighboring models. With this set up, we assume that the two regression lines $\mathfrak{l}_1$ and $\mathfrak{l}_2$ intersect at point ω_M. According to Proposition 2, and a uniform choice for kernel $\phi(.)$, we get the following bounds for the bias on $\mathfrak{l}_1$ and $\mathfrak{l}_2$ denoted by Q_{l_1} and Q_{l_2}, respectively,

$$Q_{l_1} = L_c[\frac{(\omega_{\mathfrak{l}_{11}} - \omega_{\mathfrak{l}_{12}})^2}{2} + (\omega_l - \omega_{\mathfrak{l}_{11}})^2 + (\omega_M - \omega_{\mathfrak{l}_{12}})^2]$$

and

$$Q_{l_2} = L_c[\frac{(\omega_{\mathfrak{l}_{21}} - \omega_{\mathfrak{l}_{22}})^2}{2} + (\omega_M - \omega_{\mathfrak{l}_{21}})^2 + (\omega_r - \omega_{\mathfrak{l}_{12}})^2].$$

One could simply verify that $Q_{l_1} + Q_{l_2} < Q$. This would result in reduction of the upper bound on the bias, hence $Q_{D_{T_2}} < Q_{D_{T_1}}$ with the cost of having more clouds in the cover (more parameters in the model) and a corresponding increase in the variance of the model, $V_{D_{T_1}} < V_{D_{T_2}}$.

The argument provided here is for one dimension, but with similar arguments made in Proposition 2, could be generalized to higher dimensions. □

Variable subset selection and shrinkage are methods that introduce bias and try to reduce the variance of the estimate. These methods trade a little bias for a larger reduction in variance. In practice we choose D_T via cross validation among other available methods that are developed for this purpose.

Theorem 2. Asymptotically, the DC-RBF Algorithm is unbiased for heteroscedastic noise.

Proof. This theorem follows from Theorem 1 and taking into account that in the limit $D_T \to 0$ and $K \to \infty$. Let

$$f_i(x) = \frac{\sum_i p_i(x)\phi(\|x - c_i\|_{W_i})}{\sum_i \phi(\|x - c_i\|_{W_i})}.$$

As $D_T \to 0$, $i \to \infty$, therefore

$$\lim_{i\to\infty} f_i(x) = \frac{\int p_i(x)\phi(\|x - c_i\|_{W_i})dx}{\int \phi(\|x - c_i\|_{W_i})dx}.$$

Since $D_T \to 0$ leads to $W_i \to 0$, $\lim\limits_{W\to 0} \phi(\|x - c_i\|_W) = \delta(x - c_i)$ where $\delta(.)$ is the Dirac delta function. We then have $\lim\limits_{i\to\infty} \int \phi(\|x - c_i\|)dx = 1$. Note that $K \to \infty$ guarantees that in every local ball around a given point in domain x with radius $\xi > 0$, $B(x, \xi)$ contains at least $n+1$ data points as suggested by Lemma 1, thus $\lim_{i\to\infty} \int p_i(x)\phi(\|x-c_i\|_{W_i})dx = p_i(c_i)$ where $x = c_i$.

Let the true underlying nonlinear function $g \in \mathcal{C}^\infty$ be given by the Taylor expansion,

$$g(x,a) = \sum_{n=0}^{\infty} \frac{g^{(n)}(x)}{n!}\,(x-a)^n,$$

where $g(x,a)$ denotes the approximation of function $g(x)$ around point a and $g^{(n)}(x)$ denotes the nth derivative of the function g at point x. If the underlying function $g(x) \in \mathcal{C}^1$, we consider a first order linear approximation to function $g(x)$ at each point, i.e., $g(x) \approx f(x) = g^1\,(x-a)^1$. In other words at each point x, the first order approximation is a hyperplane. In the DC-RBF algorithm if $p_i(x)$ is chosen to be a linear function, $p_i(x)$ is the tangent line to each point x. This is true since the limit of the secant line passing through the points x^{k_1} and x^{k_2} when $x^{k_1} \to x^{k_2}$ is a tangent line at point x^{k_1}. When $x^{k_1} \to x^{k_2}$, $||x^{k_1} - x^{k_2}|| \to 0$ and in the limit this secant line forms the tangent at the meeting point x_j. As a result the difference quotient $\frac{f(x^{k_1})-f(x^{k_2})}{||x^{k_1}-x^{k_2}||}$ approaches the slope of the tangent line of $g(x)$ at that x^{k_2}, i.e.,

$$\lim_{||x^{k_1}-x^{k_2}||\to 0} \frac{f(x^{k_1}) - f(x^{k_2})}{||x^{k_1} - x^{k_2}||} = g^1(x^{k_2}).$$

This is true as $K \to \infty$. Up to the first order approximation within a ball $B(x,\xi)$ with $\rho > 0$, the function is assumed to be linear. According to Remark 1, and considering the fact that the variation of the noise within $B(x,\xi)$ is negligible, the solution is BLUE. Since this is true for every point in domain X, the DC-RBF algorithm is therefore asymptotically unbiased. $\square$

5 The Empirical Results

Here we demonstrate the performance of the algorithm on a variety of synthesized and real data sets. The data sets have distinct features in terms of input dimension, complexity of the response function as well as the noise content. Note that the algorithm provides a functional representation of the underlying data. The first three synthetic examples concentrate on the performance and discussions on DC-RBF algorithm followed by two real examples with comparisons to other statistical techniques.

5.1 Synthesized Data Sets

In this section, the performance of the DC-RBF algorithm on modeling the newsvender problem is shown. In the newsvendor problem, a newsvendor is trying to determine how many units of an item x to stock. The stocking cost and selling price for the item is c and p, respectively. One could assume an arbitrary distribution on demand D. The expected profit is given by

$$F(x) = \mathbb{E}\left[p \min(x, D)\right] - cx.$$

This problem poses a challenge to on-line data analysis due to the special behavior in the function around the optimal solution which is highly dependent on the characteristics of a dataset. A data set composed of an oscillatory behavior with varying noise of the form $y = \sin(\frac{\pi x}{4}) + 0.1xN(0,1)$ is also used in this section. In this problem the challenge is to discover the underlying nonlinear function while the observations are highly corrupted with various levels of noise. The last synthesized data set provides an opportunity to approximate a two dimensional surface from a data set composed of noisy observation of a saddle shape surface given by $z = x^2 - y^2 + N(0, \sqrt{5})$.

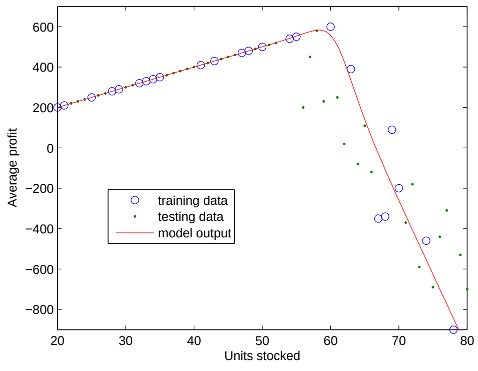
One dimensional newsvender data set. Figure 2 describes the experimental setup and the model output for two different distributions on D. Figure 2 shows the training, testing and model output for each case. The specification of the algorithm for both problems is identical. The number and position of training and testing data is the same in both experiments. We employ our function approximation technique to find the maximum number of units to stock. The MSE value for the first experiment is 107.39 with a model that has three clouds. The second experiment has resulted in a model with three clouds with MSE value of 426.63. We observe that the maximum stocking point is well identified in both experiments. Note that the algorithm learns the underlying functional behavior of the given data set within the scope that is determined by the radius of the local balls. The model generalizes well within local regions.

The underlying function forms a shape that is not in the span of a polynomial of degree p, in particular for the problems where D has low variance (making this a more challenging problem). Therefore parametric methods do not perform well, especially given the fact that the basis functions of a parametric model must be known a priori. For the noisy data set the interpolation methods such as cubic splines (despite the regression schemes) do not perform well and often result in over-fitting the data. Interpolation techniques also require more data points to produce an accurate response, see e.g., [55]. We observe that, in this experiment, our method outperforms kernel smoothing regression both in noisy and no noise data sets. The poor performance of kernel smoothing in this experiment is due to its local averaging property that tends to under-fit the function around the local extreme points. Therefore, the predicted value of the maximum is worse than the true value. In addition, this technique requires storage of the history of data points and does not provide an analytical form for the underlying function.

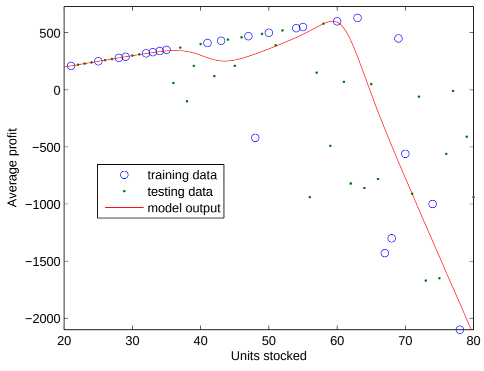
We have tested our algorithm on various other single dimensional noisy data sets and we have observed a similar conclusion. Therefore we only report on a newsvender data set which plays a key role in resource management and produces an interesting case for data analysis due to the shape around the optimum result. Our method has the lowest or a very comparable mean squared error compared to the techniques described in this section and captures well the local structure of the signal.

An oscillatory data set with varying additive noise. To demonstrate the performance of the algorithm on signals that have varying additive noise, we have tested the method on a synthesized data set that is produced by $y = \sin(\frac{\pi x}{4}) + 0.1xN(0, 1)$. Figure 3(a) shows the noisy and noise free data sets. The challenge is to find the underlying function sin(.) from the noisy input signal with large variation in the variance of noise. Figure 3(b) shows the testing data set and the model output. One could observe that the proposed algorithm has performed well in recovering the underlying function with only five clouds in the function expansion. The final MSE is 0.8 and MAE 0.62. The distance threshold is 3.

Two dimensional saddle data set. A data set generated from $z = x^2 - y^2 + N(0, \sqrt{5})$ which produces a noisy saddle shape is used in this study. Figure 4 shows the data points that are used for training and testing. Figure 5 demonstrates four major snapshots of the model making process. Each panel in this figure shows the starting point of adding a new cloud when each cloud receives three data points. The first data point that activates the cloud is also plotted in each panel. At first there is not enough points to form a model. Then, as new data points arrive the first cloud is formed, as shown in Figure 5(a). Depending on the spatial location of the arrival points the first cloud might get updated or the second



(a) $D \sim (50, 60)$.



(b) $D \sim (30, 70)$.

Figure 2: Newsvendor data set is shown that is generated using $[p \min(x, D)] - cx$. The training and testing data sets consist of 24 and 36 data points, respectively. To generate this data set, $c = 50$, $p = 60$, and D is a random uniform integer between 50 and 60 for panel (a) and between 30 and 70 for panel (b). Inventory stock levels x were sampled from a random uniform integer distribution from 20 to 80. The MSE and MAE of the model output on test data set in the original scale of the data are 33847 and 107.39 for panel (a) and 453470 and 426.63, for panel (b), respectively. The model distance threshold is 15. There are three clouds in each model.

cloud might get formed. The same process repeats till the fourth cloud of the model is shaped as shown in panel (d) of Figure 5. Finally, Figure 6 shows the final model after presenting 125 data points in the presence of the 100 testing points.

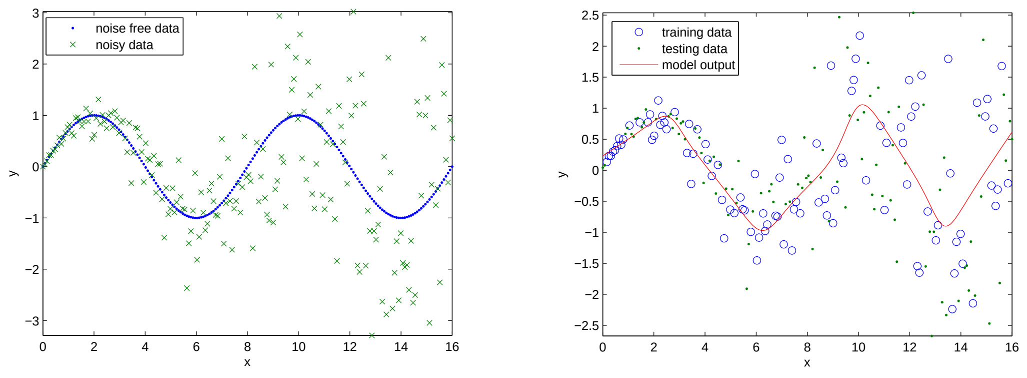
We observe that the accuracy of the model is similar to LOESS. However, unlike LOESS our method does not require the storage of the data points and has superior speed. In addition, DC-RBF provides an analytical formulation for the underlying structure. Our test results on other synthesized surfaces provide the same conclusion. For further elaboration on the LOESS method its comparison to DC-RBF, see Section 6.

5.2 Real Benchmark Data Sets and Comparison Results

In this section we show the performance of the proposed method on two real data sets and compare results to the related methods in the literature in terms of accuracy, speed, batch v.s. on-line and the requirement of storing the previous data points. We select two data sets with continuous response values. The selected data sets represent regression challenges on noise heteroscedasticity and moderate dimensionality. We compare our results to techniques that we find are the closest to the spirit of our work.

The list of the benchmark algorithms is as following:

- **Dirichlet Process Mixtures of Generalized Linear Models (DP-GLM):** A Bayesian nonparametric method that finds a global model of the joint input-response pair distribution through a mixture of local generalized linear models, [24].
- **Ordinary Least Squares (OLS):** A parametric method that is widely used for data fitting problems and often provides a reasonable fit to data. In this study



(a) Actual data. (b) Model output.

Figure 3: Panel (a) shows a data set of 200 data points in the interval $[0, 16]$ is generated with $y = \sin(\frac{\pi x}{4}) + 0.1xN(0, 1)$. Both noise free and noisy data are shown in this figure. Panel (b) shows the training, testing data set each with 100 points and the model output. In this graph the data is re-scaled using the STD of the whole data.

$[1, X_1, ..., X_n]^T$ has been chosen as basis functions ($X_i, i \in \{1, ..., n\}$ denotes the i th coordinate).

- **CART:** This is a nonparametric tree regression method [4]. This method is available in the MATLAB function classregtree.
- **Bayesian CART:** A tree regression model with a prior over tree size [8], implemented in R with the tgp package.
- **Bayesian Treed Linear Model:** A tree regression model with a prior over tree size and a linear model in each of the leaves [9], implemented in R with the tgp package.
- **Gaussian Processes (GP):** A nonparametric method for continuous inputs and responses [47]. This algorithm is available in MATLAB with program name gpml.
- **Treed Gaussian Processes:** A tree regression model with a prior over tree size and a GP on each leaf node [22]; This is implemented in R with the tgp package.

DP-GLM, [24], is closest to the spirit of our work, although algorithmicly they are quite distinct. So we have most emphasized on this comparison in this work and used results reported in [24] on the performance of other competing methods in the literature. DP-GLM was designed for batch applications and is much slower and data intensive than DC-RBF method proposed in this paper.

Cosmic Microwave Background (CMB) [2]. This data set maps positive integers $x = 1, 2, ..., 899$, called "multipole moments," to power spectrum C_l. It consists of 899 observations, shown in Figure 7. This data set is highly nonlinear and heteroscedastic. These features make this data set an interesting benchmark choice. The underlying function relates continuous domains values to their corresponding continuous responses.

Figure 8 shows the test data set and the model output. The distance threshold parameter value, 95, is determined using 5-fold cross validation. The output is a continuous plot of a

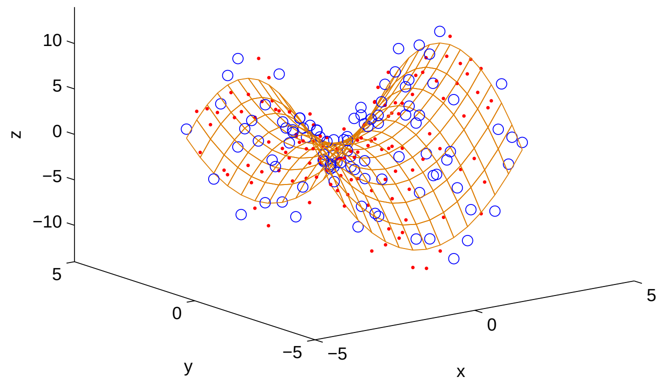


Figure 4: Saddle data set generated by $z = x^2 - y^2 + N(0, \sqrt{5})$. There are 125 training and 100 testing data points over the domain $x \in [-3, 3] \times [-3, 3]$. This nonlinear and noisy example provides an interesting problems to show the performance of our proposed algorithm.

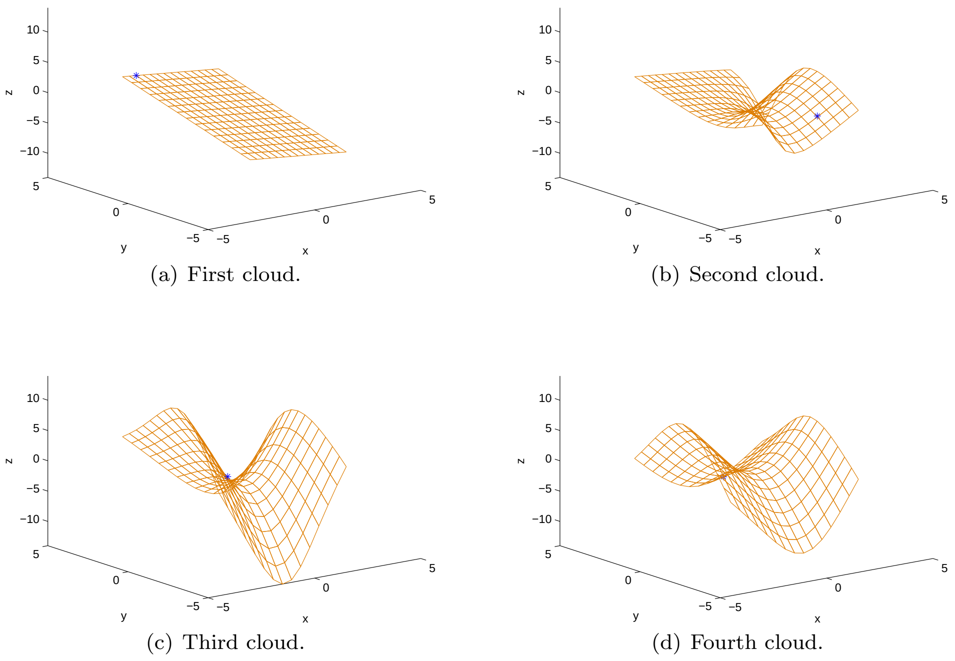


Figure 5: The model behavior at the starting point of adding a new cloud to the model. The final model consists of four clouds.

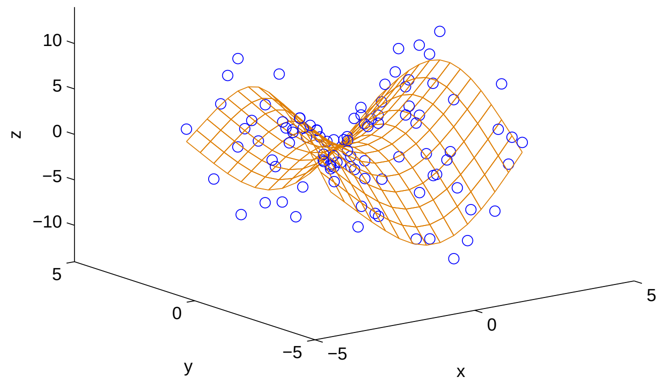


Figure 6: The final model output after receiving 125 training points is shown in the presence of 100 testing points. The final model with four clouds has MSE of 4.49 with MAE of 1.7.

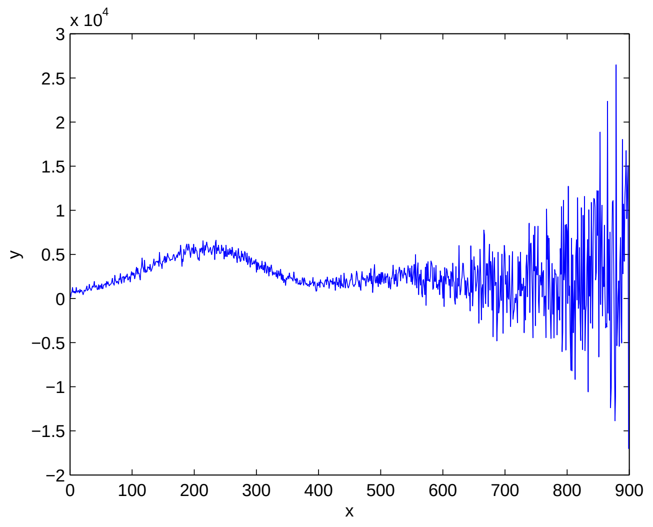


Figure 7: This plot shows the CMB data set that consists of 899 data points.

parsimonious model with only seven clouds. The MSE and MAE for the testing data are 0.7810 and 0.4546 respectively. The result shows that the method is capable of handling the heteroscedasticity in this data set quite well. The comparison between DC-RBF method and the six methods described above are summarized in table 1. In terms of error rates our proposed algorithm slightly under-performs on the smaller data sets compared to the DP-GLM, although with a dramatic reduction in CPU time. However it is quite competitive on larger data sets. Note that even though our proposed algorithm is in the same algorithmic class as DP-GLM, it only uses a given data point once and does not require storage of the data points. In contrast, DP-GLM is a batch algorithm and relies on exhaustive computation to compute a model and requires all the history to rebuild the model once a new data point arrives. The new method has one adjustable hyper-parameter, while the DP-GLM has multiple hyper-parameters to tune. The DC-RBF method is orders of magnitude faster, since DP-GLM requires the use of Markov Chain Monte Carlo methods to reoptimize the clustering as each data point is added, where as DC-RBF updates instantly.

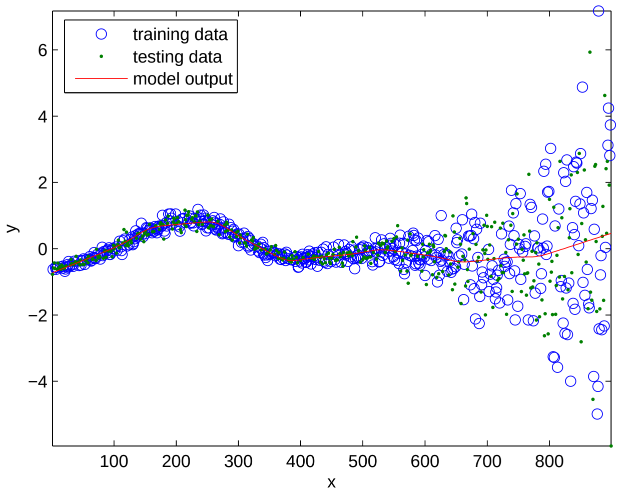


Figure 8: The training and testing data set and the model output for the CMB data set is shown in this plot. The training data set consists of 500 points and the remaining 499 are used for testing. Note that the range of the signal is re-scaled by the standard deviation of the total input data.

Concrete Compressive Strength (CCS) [58] The CCS data set provides a mapping from an eight dimensional input space to a continuous output. This data set has low noise and variance. The eight continuous input dimensions are: the components cement, blast furnace slag, fly ash, water, super-plasticizer, coarse aggregate and fine aggregate, all measured in kg per m^3, and the age of the mixture in days. The response is the compressive strength of the resulting concrete. There are $1,030$ observations in this data set. The challenge working with this data set is to find a continuous mapping that represents the input-output relationship well in this multi-dimensional data.

Similar to the experiment using the CMB data set, we have summarized the comparison results for this data set in Table 2. We observe that as the number of observations increase the performs of the new method enhances due to the nature of our design that requires certain number of data points to activate a cloud. From the Table 2, we see that in terms of error our method remains competitive with DP-GLM, which is computationally much more demanding.

6 Concluding Remarks

We propose a fast, recursive, function approximation method which avoids the need to store the complete history of data (typical of nonparametric methods) or the specification of

Training Set Size	30	50	100	250	500	30	50	100	250	500
Method	Mean Absolute Error					Mean Square Error				
DP-GLM	0.58	0.51	0.49	0.48	0.45	1.00	0.94	0.91	0.94	0.83
Bayesian CART	0.66	0.64	0.54	0.50	0.47	1.04	1.01	0.93	0.94	0.84
Bayesian TLM	0.64	0.52	0.49	0.48	0.46	1.10	0.95	0.93	0.95	0.85
Gaussian Process	0.55	0.53	0.50	0.51	0.47	1.06	0.97	0.93	0.96	0.85
Treed GP	0.52	0.49	0.48	0.48	0.46	1.03	0.95	0.95	0.96	0.89
Linear Regression	0.66	0.65	0.63	0.65	0.63	1.08	1.04	1.01	1.04	0.96
CART	0.62	0.60	0.60	0.56	0.56	1.45	1.34	1.43	1.29	1.41
DC-RBF	0.61	0.53	0.49	0.47	0.46	1.40	1.13	0.98	0.91	0.85

Table 1: This table summarizes the performance of various regression techniques on CMB data set. The MSEs and MAEs on different sizes of testing data sets are reported in this table. The presented results are the mean of the performances for various methods for different permutations of data points selected for training and testing data sets. For DC-RBF, the hyper-parameter D_T is kept fixed for a given data set size. The hyper-parameter could be identified using for example a 5-fold cross validation or another suitable technique. The hyper parameters of the other techniques are calculated by sweeping over a set of candidate parameters that varies by 3 to 5 orders of magnitude, then the parameter that does the best on the training set is chosen, these number are reported from [24].

Training Set Size	30	50	100	250	500	30	50	100	250	500
Method	Mean Absolute Error					Mean Square Error				
DP-GLM	0.54	0.50	0.45	0.42	0.40	0.47	0.41	0.33	0.28	0.27
Bayesian CART	0.78	0.72	0.63	0.55	0.54	0.95	0.80	0.61	0.49	0.46
Bayesian TLM	1.08	0.95	0.60	0.35	1.10	7.85	9.56	4.28	0.26	1,232
Gaussian Process	0.53	0.52	0.38	0.31	0.26	0.49	0.45	0.26	0.18	0.14
Treed GP	0.73	0.40	0.47	0.28	0.22	1.40	0.30	3.40	0.20	0.11
Linear Regression	0.61	0.56	0.51	0.50	0.50	0.66	0.50	0.43	0.41	0.40
CART	0.72	0.62	0.52	0.43	0.34	0.87	0.65	0.46	0.33	0.23
DC-RBF	0.57	0.54	0.51	0.47	0.39	0.54	0.50	0.44	0.40	0.29

Table 2: The performance table for CCS data. In this table the results are reported in terms of Mean Square Error and Mean Absolute Error for various competing methods, including DP-GLM. The experimental set up here is the same as we described for CMB data set.

hyperparameters for Bayesian priors. The method assigns locally parametric models (such as linear polynomials) for regions of the covariate space that are created dynamically from the data without the need for prespecified classification schemes. A weighting scheme is associated to each locally linear approximation using normalized RBF. Each local model is updated recursively with the arrival of each new observation, which is then used to update the cloud representation before the data is discarded.

The new method is robust in the presence of homoscedastic and heteroscedastic noise. Our method automatically determines the model order for a given dataset (one of the most challenging tasks in nonlinear function approximation). Unlike similar algorithms in the literature, our method has only one tunable parameter and is asymptotically unbiased.

Table 3 provides a comparison of different statistical methods that we have considered in this paper. This table brings together features such as speed, complexity of implementation, storage requirement, recursivity, the type of noise that can be handled and the number of tunable parameters.

More specifically, in contrast to DP-GLM, DC-RBF offers orders of magnitude faster updates, does not require as many tunable parameters, avoids the complexity of its implementation, and can be updated recursively without need for storing the data history.

The power of DC-RBF lies in its adaptibility to approximate complex surfaces, compactness in model representation, parsimony in model specification, recursivity and speed. Parametric methods will always suffer from the need to tune the choice of basis functions. Nonparametric methods require the storage of the entire history which complicates function computations in stochastic optimization settings (our motivating application). DP-GLM, which fits local polynomial approximations to clusters, is extremely slow and cannot be adapted to a recursive setting. Gaussian process models can work very well, but only in particular problem classes with continuous covariates and they also require more tunable parameters. Gaussian process models, as well as techniques such as splines, also introduce the risk of overfitting.

Our experience with Bayesian methods (in particular, in the development of DP-GLM, in which the second author was a developer) is that they are quite sensitive to the tuning of the hyperparameters that make up the priors. This introduces an undesirable degree of human intervention that is likely to limit their usefulness in black box stochastic optimization algorithms.

DC-RBF appears to satisfy most of our needs for approximating continuous functions in the recursive setting of stochastic optimization. It starts with a linear, parametric model, and adds clouds only as the range of covariates expand. It can adapt to quite general surfaces, and requires only that we retain history in the form of a relatively compact set of clouds. It does not require the specification of basis functions (we believe the local linear models should be kept very simple), and offers fast, recursive updating algorithms.

Performance in terms of solution quality is always going to be an evaluation limited to a specific set of experimental tests. We used the newsvendor setting to demonstrate the ability of DC-RBF to adapt to both a sharp, nondifferentiable function (which is difficult to approximate using low-order polynomials) and a smooth function. The method works well, theoretically and experimentally, in the presence of heteroscedastic noise. Finally, it appears to be competitive against much more complex algorithms such as DP-GLM.

DC-RBF does introduce the need to tune a distance threshold parameter D_T. This is not a minor requirement, as the choice of D_T captures the overall behavior of the surface. We suspect the choice of D_T will generally become apparent within any specific problem class. We also doubt that this method would be effective for high dimensional applications (say, more than 10 or 20 covariates).

We believe that there are opportunities to build on this basic idea to overcome the need to specify D_T. A natural extension is to specify a dictionary of possible values of D_T (possibly on a log scale), which can be viewed as models at different levels of granularity. We might then estimate a family of approximations, one for each D_T, which can then be combined using a hierarchical weighting scheme such as that proposed in [20].

We have shown numerical results on synthetic and real data which demonstrate the success of the algorithm for multiple dimensional function approximation. Despite a model generated by DP-GLM which remains biased toward the early observations, (this form of modeling may not serve well in applications that provide update for the existing information such as ADP), DC-RBF could adapt very quickly to new information. This type of approximation tool is specially useful for value function approximation in the context of Approximate Dynamic Programming, where there is a stream of exogenous information contributing to the system dynamics. We intend to use the approximate technique for value function approximation in the context of Approximate dynamic programming, [44] and optimal learning [45] for energy resource allocation under uncertainty.

Algorithm	Spd	Comp	Stor	Rec	Noise	TP
DP-GLM	V	C	A	N	HE	F
Bayesian CART	V	C	A	N	HO	F
Bayesian TLM	F	I	A	N	HO	F
Gaussian Process	V	C	A	N	HO	F
Treed GP	V	C	A	N	HO	F
Linear Regression	U	S	N	Y	HO	N
CART	F	I	A	N	HO	F
Kernel Regression	F	I	A	N	HO	F
Locally Linear	F	I	A	Y	HO	F
LOWESS	F	I	A	N	HO	F
DC-RBF	U	S	SR	Y	HE	O

Table 3: The benchmark table to compare various statistical function approximation techniques in terms of overall Speed in terms of model Updating and model evaluation (Ultra fast, Fast, Very slow), Complexity of implementation (Complex, Intermediate, Simple), Storage requirement (All the history, Statistical Representation, None), Recursivity (Yes, No), Noise type they could handle (HEteroscedasticity, HOmoscedasticity), the number of Tunable Parameters, (None, One, Few).

References

[1] C. Andrieu, N. Freitas, and A. Doucet. Robust Full Bayesian Learning for Radial Basis Networks. *Neural Computation*, 13:2359–2407, 2001.

[2] L. Bennett, M. Halpern, G. Hinshaw, N. Jarosik, A. Kogut, M. Limon, S. S. Meyer, L. Page, D. N. Spergel, G. S. Tucker, E. Wollack, E. L. Wright, C. Barnes, M. R. Greason, R. S. Hill, E. Komatsu, M. R. Nolta, N. Odegard, H. V. Peiris, L. Verde, and J. L. Weiland. First-year Wilkinson Microwave Anisotropy Probe (WMAP) 1 Observations: Preliminary Maps and Basic Results. *The Astrophysical Journal Supplement Series*, 148(1):1–27, 2003.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, U.K., 1995.

[4] L. Brieman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* hapman & Hall/CRC, New York, NY, 1984.

[5] D. S. Broomhead and D. Lowe. Multivariable Functional Interpolation and Adaptive Networks. *Complex Systems*, 2:321–355, 1988.

[6] G. Bugmann. Normalized Gaussian Radial Basis Function Networks. *Neurocomputing*, 20(1-3):97–110, 1998.

[7] M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, 2003.

[8] H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian CART Model Search. *Journal of the American Statistical Association*, 93(443):935–948, 1998.

[9] H. A. Chipman, E. I. George, and R. E. McCulloch. Bayesian Treed Models. *Machine Learning*, 48(1):299–320, 2002.

[10] E. K. P. Chong. *An Introduction to Optimization*. Wiley, 2008.

[11] W. S. Cleveland. Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, 74(368):829–836, 1979.

[12] W. S. Cleveland. LOWESS: A program for Smoothing Scatterplots by Robust Locally Weighted Regression. *The American Statistician*, 35(1):54, 1981.

[13] W. S. Cleveland and S. J. Devlin. LOWESS: A program for Smoothing Scatterplots by Robust Locally Weighted Regression. *Journal of the American Statistical Association*, 83(403):596–610, 1988.

[14] R. L. Eubank. *Spline Smoothing and Nonparametric Regression*. Marcel Dekker, New York, 1988.

[15] J. Fan. Design-adaptive Nonparametric Regression. *Journal of the American Statistical Association*, 87(420):998–1004, Dec 1992.

[16] J. Fan and I. Gijbels. *Local Polynomial Modelling and Its Applications*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability, London, 1996.

[17] C. F. Gauss. *Theoria Combinationis Observationum Erroribus Minimis Obnoxiae*. Gottingae, 1825.

[18] S. Geisser. *Predictive Inference: An Introduction*. Chapman and Hall/CRC, 1 edition, June 1993.

[19] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, second edition, July 2003.

[20] A. George, W. B. Powell, and S. Kulkarni. Value Function Approximation using Multiple Aggregation for Multiattribute Resource Management. *J. Machine Learning Research*, 9:20792111, 2008.

[21] F. Girosi, M. Jones, and T. Poggio. Regularization Theory and Neural Network Architectures. *Neural Computation*, 7:219–269, 1995.

[22] R. B. Gramacy and H. K. H. Lee. Bayesian Treed Gaussian Process Models with an Application to Computer Modeling. *Journal of the American Statistical Association*, 103(483):1119–1130, 2008.

[23] H.-M. Gutmann. A Radial Basis Function Method for Global Optimization. *Journal of Global Optimization*, 19:201–227, 2001.

[24] L. A. Hannah, D. M. Blei, and W. B. Powell. Dirichlet Process Mixtures of Generalized Linear Models. *Journal of Machine Learning Research*, 12:1923–1953, 2011.

[25] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer, 2009.

[26] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall, 2nd edition, 1999.

[27] C. C. Holmes and B. K. Mallick. Bayesian Radial Basis Functions of Variable Dimension. *Neural Computation*, 10(5):1217–1233, 1998.

[28] T. Ishikawa and M. Matsunami. An Optimization Method Based on Radial Basis Function. *IEEE Transactions on Magnetics*, 33(2):1868–1871, March 1997.

[29] A. A. Jamshidi and M. J. Kirby. Towards a Black Box Algorithm for Nonlinear Function Approximation over High-Dimensional Domains. *SIAM Journal of Scientific Computation*, 29(3):941–963, May 2007.

[30] A. A. Jamshidi and M. J. Kirby. Skew-Radial Basis Function Expansions for Empirical Modeling. *SIAM Journal on Scientific Computation*, 31(6):4715–4743, 2010.

[31] A. A. Jamshidi and M. J. Kirby. Modeling Multivariate Time Series on Manifolds with Skew Radial Basis Functions. *Neural Computation*, 23(1):97–123, 2011.

[32] R. D. Jones, Y.C. Lee, C. W. Barnes, G. W. Flake, K. Lee, P. S. Lewis, and S. Qian. Function Approximation and Time Series Prediction with Neural Networks. *IJCNN International Joint Conference on Neural Networks*, 1:649–665, 1990.

[33] D. E. Knuth. *Art of Computer Programming*, volume 2. Addison-Wesley Professional, 3rd edition, Nov. November 14, 1997.

[34] D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to Linear Regression Analysis.* Wiley-Interscience, 3 edition edition, April 2001.

[35] J. Moody and C. Darken. Fast Learning in Networks of Locally-tuned Processing Units. *Neural Computation*, 1:281–294, 1989.

[36] H. G. Müller. Weighted Local Regression and Kernel Methods for Nonparametric Curve Fitting. *Journal of the American Statistical Association*, 82:231–238, 1988.

[37] J. Munkres. *Topology.* Prentice Hall, 2 edition, January 2000.

[38] E. A. Nadaraya. On Estimating Regression. *Theory of Probability and Its Applications*, 9:141–142, 1946.

[39] J. Park and I. W. Sandberg. Universal Approximation Using Radial-Basis-Function Networks. *Neural Computation*, 3:246–257, 1991.

[40] J. Park and I. W. Sandberg. Approximation and Radial-Basis-Function Networks. *Neural Computation*, 5:305–316, 1993.

[41] T. Poggio and F. Girosi. Regularization Algorithm for Learning that Are Equivalent to Multilayer Networks. *Science*, 247:978–982, 1990.

[42] M. J. D. Powell. Radial Basis Functions for Multivariable Interpolation: A Review. In J. C. Mason and M. G. Cox, editors, *Algorithms for approximation*, pages 143–167. Clarendon Press, Oxford, 1987.

[43] M. J. D. Powell. The Theory of Radial Basis Functions in 1990. In W. Light, editor, *Advances in Numerical Analysis*, volume II of *Wavelets, Subdivision Algorithms, and Radial Basis Functions*, pages 105–210. Oxford University Press, 1992.

[44] W. B. Powell. *Approximate Dynamic Programming.* Wiley, 2011.

[45] W. B. Powell and Ilya O. Ryzhov. *Optimal Learning.* Wiley, 2012.

[46] C. E. Rasmussen and Z. Ghahramani. Infinite Mixtures of Gaussian Process Experts. In *In Advances in Neural Information Processing Systems 14*, pages 881–888. MIT Press, 2001.

[47] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* MIT Press, Cambridge, MA, 2006.

[48] R. G. Regis and C. A. Shoemaker. A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions. *Informs Journal on Computing*, 19(4):497–509, 2007.

[49] R. G. Regis and C. A. Shoemaker. Improved Strategies for Radial Basis Function Methods for Global Optimization. *Journal of Global Optimization*, 37(1):113–135, 2007.

[50] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Internal Representations by Error Propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, pages 318–362. 1986.

[51] D. Ruppert, M. P. Wand, and R. J. Carroll. *Semiparametric Regression.* Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1 edition edition, July 2003.

[52] B. Shahbaba and R. M. Neal. Nonlinear Models Using Dirichlet Process Mixtures. *Journal of Machine Learning Research*, 10:1829–1850, 2009.

[53] J. S. Simonoff. *Smoothing Methods in Statistics.* Springer, June 1996.

[54] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems.* John Wiley & Sons, New York, 1977.

[55] G. Wahba. Spline Bases, Regularization, and Generalized Cross Validation for Solving Approximation Problems with Large Quantities of Data. In W. Cheney, editor, *Approximation Theory III*, pages 905–912. Academic Press, 1980.

[56] G. S. Watson. Smooth Regression Analysis. *Sankhya, Ser. A*, 26:359–372, 1946.

[57] P. J. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* Ph.D. dissertation, Division of Applied Mathematics, Harvard University, Boston, MA, Aug. 1974.

[58] I. C. Yeh. Modeling of Strength of High-performance Concrete Using Artificial Neural Networks. *Cement and Concrete Research*, 28(12):1797–1808, 1998.