Incremental Error-Adaptive Modeling of Time-Series Data Using Radial Basis Functions

Xiaofeng Ma, Manuchehr Aminian, Michael Kirby

Department of Mathematics, Colorado State University, Fort Collins, Colorado, 80523

Abstract

We propose an error-adaptive method for constructing time-series models using radial basis functions. The quality of the data fit is optimized as part of the objective function in a linear program. The resulting algorithm is applied to streaming time-series data. Novel points are identified using a infeasibility criterion, while feasible points are characterized as nominal and not used for training. A sparsity promoting term in the objective function serves to determine the location and number of RBFs required to fit the data. We illustrate the approach with several examples and show that the fitting procedure is robust to additive noise.

Keywords: Radial Basis Functions, Adaptive Error Modeling, Dual Simplex Algorithm, Anomaly Detection.

1. Introduction to Radial Basis Functions

The Radial Basis Function (RBF) expansion is a widely used tool for data driven modeling of large data sets. It takes the form

$$f(x) = w_0 + \sum_{k=1}^{N_c} w_k \phi(\|x - c_k\|), \tag{1}$$

where the RBF functions can be selected from several options [1, 2]. For simplicity, in this paper the function f and weights w_k are assumed to be scalars.

*Corresponding author:
Email address: `Michael.Kirby@Colostate.Edu` (Michael Kirby)

One of the main challenges to using RBFs to discover nonlinear relationships in data is the need to determine the model parameters. How many RBFs N_c should be used? Further, where should these RBFs be centered on the data, i.e., how should these locations $\{c_k\}$ be determined? Much of the algorithmically oriented RBF literature centers around providing efficient solutions to these problems, see, e.g., [3, 4] and references therein.

The theoretical foundation of RBFs is supported by a *Universal Approximation Theorem* [5] which states that continuous functions on a compact domain are dense with respect to certain families of RBFs. RBFs have also been explored in the context of approximation theory [6, 7] and more recently families of compact RBFs have been proposed [8].

Radial basis function approximations provide a flexible optimization framework for solving for the model parameters. For example, if the centers and shape parameters of the RBFs are preselected, then the RBF problem for the unknown weights is simply a linear system. This fact suggests hybrid optimization routines, where the centers are selected, then refined, using nonlinear optimization after the weights have been found using a linear systems approach. Updates may also be made either on-line or in batch modes depending on the application.

In this paper we propose a method for learning radial basis functions where the quality of the model to be fit is a decision variable to be determined by the optimization problem. This program is set in the context of streaming data and is solved using an incremental solution to the simplex algorithm. This approach has the advantage that sparsity promoting terms may be added to the objective function. This results in a solution to both the order determination problem as well as the center location problem.

2. The RBF Learning Problem

We assume that the data is observed sequentially with time being indexed by i. So, at time t_i the point x_i is observed. The RBF equation for this point is then

$$b_i = w_0 + \sum_{k=1}^{m} w_k \phi(\|x_i - c_k\|).$$

where b_i represents the target value of the function f, i.e.,

$$f(x_i) = b_i$$

If we define the interpolation matrix as

$$\Phi = \begin{bmatrix} 1 & \phi(\|x_1 - c_1\|) & \dots & \phi(\|x_1 - c_{N_c}\|) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \phi(\|x_P - c_1\|) & \dots & \phi(\|x_P - c_{N_c}\|) \end{bmatrix}$$

where we assume a total of P observations and a selection of N_c centers. The weights are then found by solving the system

$$\Phi w = b. \tag{2}$$

This approach is widely used for computing radial basis function solutions. Typically there are far more data points than centers, i.e., $P >> N_c$.

However, since the problem is over-determined, it has a unique approximate optimal solution. The number of centers must be selected in advance, so there is no opportunity for sparsity promotion in the weight vector w.

We propose an alternative approach to computing the RBF model that revolves around the idea of solving Equation (2) as an *underdetermined* system with a sparsity promoting constraint. Specifically, we randomly select far more centers than data points, i.e., $N_c >> P$. As we shall see, the regularizing sparsity term in combination with the adaptive error constraint serve to prevent over-fitting of the data in a manner that can be controlled by an *ad hoc* parameter.

As we are concerned with streaming data, we need notation for the additional information associated with observing the ith point x_i. Thus, we use Φ_i to denote the ith row of this matrix (as a column vector).

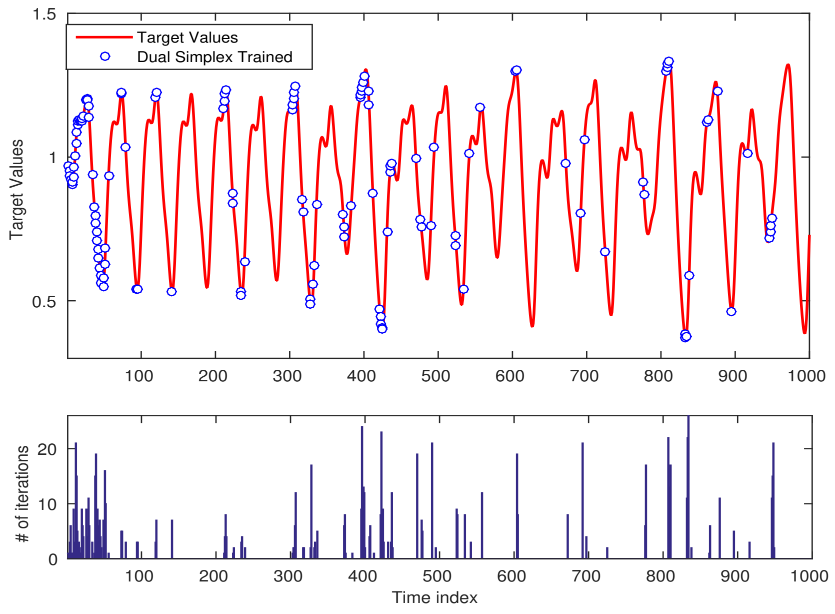

Figure 1: The result of applying the simplex algorithm with fixed ϵ to the Mackey-Glass data set using sequential training over 1000 data points. The lower bars denote that an infeasible LP resulted from adding the point. The location of the *novel* point is shown in blue. A total of $N_c = 14$ RBFs were automatically selected from an initial set of 500 during the streaming algorithm.

3. Linear Programming Formulation

We propose to determine the parameters in the RBF approximations by solving the optimization problem

$$\begin{aligned} \underset{w,\epsilon}{\text{minimize}} \quad & \|w\|_1 + C\epsilon \\ \text{subject to} \quad & \|\Phi w - b\|_\infty \leq \epsilon \end{aligned} \tag{3}$$

where the fitting parameter ϵ and the weight vector w are the decision variables in the optimization problem. As we will see below, this optimization problem is actually a linear programming problem. This LP formulation builds on our earlier work where we viewed ϵ as a fixed constant [9]. Although the modification may appear minor, as we shall illustrate, it fundamentally improves the resulting models and utility of the algorithm. First, observe in Figure 1 that the learning using a fixed ϵ is spread out over a longer time period than we will observe below. We will see that taking ϵ as a decision variable leads to more rapid learning of nominal data and the ability to adapt variations in the level of noise. In contrast, using a fixed quality parameter ϵ requires a search for this parameter, but more importantly, does not permit the flexibility of being adapted as part of the learning problem.

Appeal of the formulation. This formulation has several attractive features over standard techniques for learning RBF models:

Automated Goodness of Fit. The parameter ϵ serves as the upper bound on the error of the data fit. This value is minimized via the solution of the linear program. We see that it adapts over the course of learning. When noise is artificially added to data this is seen to increase the optimal value of ϵ.

Fast Updates. If the new point is not feasible, one can implement a *warm start* of the dual simplex algorithm. In practice, we have observe that very few iterations of the dual simplex are required to recover optimality.

Updates not always required. In general, incremental RBF algorithms implement rank-one updates for every new data point [10]. The algorithm proposed here allows for incremental learning without the need for rank-one updates to the solution of a linear system. When a new point arrives, one determines if the solution to the LP is still feasible. If so, it is still optimal, and *NO UPDATE* is required.

Low Memory Requirements. This algorithm is designed to operate on streaming data. Because only infeasible points are required to update the model, only a small subset of the observed data must be stored.

Data Selection. This algorithm identifies novel points in the data stream. In practice, we observe that approximately 10% or less of the data is novel.

Order Determination. The number of RBFs is automatically determined using this algorithm.

Center Locations. The location of the centers $\{c_k\}$ is determined automatically by the algorithm. If one desires, a further location refinement may be implemented using, e.g., BFGS for nonlinear optimization of the objective function. We note that this is now much faster using the smaller novel data set and an appropriate initial number of centers.

A Paradigm for Novelty Detection. We observe that at the beginning of training that many updates are required but that after many points have been observed the updates become much less frequent. This is due to the fact that the data space has been modeled by the RBF and no additional complexity is required when a new point arrives. However, if the system changes, e.g., the temperature in a weather system that is changing, then the new data triggers a model update.

The ℓ_1-norm component of the objective function can be converted into the more tractable form of a hyperplane by introducing t_i such that

$$-t_i \leq w_i \leq t_i$$

where $t_i \geq 0, i = 0, \dots, N_c$. Substituting this into the optimization problem

$$\begin{aligned} \underset{w,\epsilon}{\text{minimize}} \quad & \sum_{i=0}^{N_c} t_i + C\epsilon \\ \text{subject to} \quad & \|\Phi w - b\|_\infty \leq \epsilon \\ & -t_i \leq w_i \leq t_i, \quad i = 0, \dots, N_c, \\ & \epsilon, t_i \geq 0. \end{aligned} \tag{4}$$

Similarly, the constraint $\|\Phi w - b\|_\infty \leq \epsilon$ can be rewritten equivalently as

$$|(\Phi w - b)_i| \leq \epsilon$$

for each of the components $i = 1, \dots P$. Denoting the i'th row of Φ as Φ_i this can be further rewritten to

$$-\epsilon \leq \Phi_i^T w - b_i \leq \epsilon$$

again for each $i = 1, \dots, P$.

$$\begin{aligned} \underset{w,\epsilon}{\text{minimize}} \quad & \sum_{k=0}^{N_c} t_k + C\epsilon \\ \text{subject to} \quad & \|\Phi w - b\|_\infty \leq \epsilon \\ & -t_k \leq w_k \leq t_k, \quad k = 0, \dots, N_c. \end{aligned} \tag{5}$$

Collecting the constraints and the (now linear) objective function, we have

$$\begin{aligned} \text{minimize} \quad & \sum_{k=0}^{N_c} t_k + C\epsilon \\ \text{subject to} \quad & \Phi_i^T w - \epsilon \leq b_i \qquad i = 1, \dots, P. \\ & -\Phi_i^T w - \epsilon \leq -b_i \qquad i = 1, \dots, P. \\ & w_k - t_k \leq 0, \qquad k = 0, \dots, N_c \\ & -w_k - t_k \leq 0, \qquad k = 0, \dots, N_c \\ & \epsilon, t_k \geq 0, \qquad k = 0, \dots, N_c. \end{aligned} \tag{6}$$

where the decision variable is $x = (\epsilon, t, w)$ with $t, w \in \mathbb{R}^{N_c+1}, \epsilon \in \mathbb{R}$. Augmenting the decision variable to include the slack variables, we have

$$\begin{array}{lrl}
\text{minimize} & \displaystyle\sum_{k=1}^{N_c} t_k + C\epsilon & \\
\text{subject to} & \Phi_i^T w - \epsilon + u_i = b_i & i = 1, \ldots, P. \\
& -\Phi_i^T w - \epsilon + v_i = -b_i & i = 1, \ldots, P. \\
& w_k - t_k + s_k = 0, & k = 0, \ldots, N_c \\
& -w_k - t_k + z_k = 0, & k = 0, \ldots, N_c \\
& \epsilon, u_i, v_i, s_k, z_k, t_k \geq 0, & k = 0, \ldots, N_c.
\end{array} \tag{7}$$

where now $x = (\epsilon, t, w, s, z, u, v)$ with $t, w, s, z \in \mathbb{R}^{N_c+1}, u, v \in \mathbb{R}^P, \epsilon \in \mathbb{R}$. If we let $Q = N_c + 1$ and define $c = (C\ e_Q\ 0_Q\ 0_Q\ 0_Q\ 0_P\ 0_P)^T$. In other words,

$$\text{minimize} \sum_{k=1}^{N_c} t_k + C\epsilon$$

subject to

$$\begin{pmatrix} -e_P & 0_{P\times Q} & \Phi & 0_{P\times Q} & 0_{P\times Q} & I_{P\times P} & 0_{P\times P} \\ -e_P & 0_{P\times Q} & -\Phi & 0_{P\times Q} & 0_{P\times Q} & 0_{P\times P} & I_{P\times P} \\ 0_Q & -I_{Q\times Q} & I_{Q\times Q} & I_{Q\times Q} & 0_{Q\times Q} & 0_{Q\times P} & 0_{Q\times P} \\ 0_Q & -I_{Q\times Q} & -I_{Q\times Q} & 0_{Q\times Q} & I_{Q\times Q} & 0_{Q\times P} & 0_{Q\times P} \end{pmatrix} \begin{pmatrix} \epsilon \\ t \\ w \\ s \\ z \\ u \\ v \end{pmatrix} = \begin{pmatrix} b \\ b \\ 0_Q \\ 0_Q \end{pmatrix}$$

To complete the conversion to standard form we need to remove the only free variable w. The standard mechanism for doing this is to let $w = w^+ - w^-$ where $w^+, w^- \geq 0$. Now we have

$$A = \begin{pmatrix} -e_P & 0_{P\times Q} & \Phi & -\Phi & 0_{P\times Q} & 0_{P\times Q} & I_{P\times P} & 0_{P\times P} \\ -e_P & 0_{P\times Q} & -\Phi & \Phi & 0_{P\times Q} & 0_{P\times Q} & 0_{P\times P} & I_{P\times P} \\ 0_Q & -I_{Q\times Q} & I_{Q\times Q} & -I_{Q\times Q} & I_{Q\times Q} & 0_{Q\times Q} & 0_{Q\times P} & 0_{Q\times P} \\ 0_Q & -I_{Q\times Q} & -I_{Q\times Q} & I_{Q\times Q} & 0_{Q\times Q} & I_{Q\times Q} & 0_{Q\times P} & 0_{Q\times P} \end{pmatrix} \tag{8}$$

$$x = \begin{pmatrix} \epsilon \\ t \\ w^+ \\ w^- \\ s \\ z \\ u \\ v \end{pmatrix}$$

and

$$\hat{b} = \begin{pmatrix} b \\ b \\ 0_Q \\ 0_Q \end{pmatrix}$$

With the above definitions for $A, x, \hat{b}, c$ the standard form is

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^T x \\ \text{subject to} \quad & Ax = b, \\ & x \geq 0. \end{aligned} \tag{9}$$

4. Incremental Learning

Rather than solve the above optimization problem for a set of points, we proceed with an incremental learning algorithm in the sense that the points arrive one at a time and the model is updated. The initialization of the LP is immediate with no points.

4.1. Initialization

The feasibility condition is

$$A_0 x_0 = b_0$$

where

$$A_0 = \begin{pmatrix} 0_Q & -I_{Q\times Q} & I_{Q\times Q} & -I_{Q\times Q} & I_{Q\times Q} & 0_{Q\times Q} \\ 0_Q & -I_{Q\times Q} & -I_{Q\times Q} & I_{Q\times Q} & 0_{Q\times Q} & I_{Q\times Q} \end{pmatrix} \tag{10}$$

$$x_0 = \begin{pmatrix} \epsilon \\ t \\ w^+ \\ w^- \\ s \\ z \end{pmatrix}$$

where u, v are not part of the decision vector since there are no points. and

$$\hat{b}_0 = \begin{pmatrix} 0_Q \\ 0_Q \end{pmatrix}$$

A basic feasible solution is simply given by $x_0 = 0$.

4.2. Rank 2 update

Now that the LP is initialized for the case of no data we outline the procedure for updating the RBF solution as each data point is added. We focus on the case of the first point given the rest of the updates follow the same pattern. Given a new data point we test the feasibility of the LP, i.e., if either

$$\Phi_1^T w - \epsilon \leq b_1$$

or

$$-\Phi_1^T w - \epsilon \leq -b_1$$

fails to hold the LP may become non-optimal and a new solution must be found. Here we take advantage of the structure of the updated constraints and the fact that since the new LP has an infeasible primal solution, it is dual feasible.

This is accomplished, as the notation above suggests, by introducing the new basic variables $u_1, v_1 \geq 0$ that appear as slack variables

$$\Phi_1^T w - \epsilon + u_1 = b_1$$

or

$$-\Phi_1^T w - \epsilon + v_1 = -b_1$$

The new constraint matrix now becomes

$$A_1 = \begin{pmatrix} 0_Q & -I_{Q\times Q} & I_{Q\times Q} & -I_{Q\times Q} & I_{Q\times Q} & 0_{Q\times Q} & 0_Q & 0_Q \\ 0_Q & -I_{Q\times Q} & -I_{Q\times Q} & I_{Q\times Q} & 0_{Q\times Q} & I_{Q\times Q} & 0_Q & 0_Q \\ -1 & 0_Q^T & \Phi_1^T & -\Phi_1^T & 0_Q^T & 0_Q^T & 1 & 0 \\ -1 & 0_Q^T & -\Phi_1^T & \Phi_1^T & 0_Q^T & 0_Q^T & 0 & 1 \end{pmatrix} \tag{11}$$

This has the form

$$A_1 = \begin{pmatrix} A_0 & \mathbf{0} & \mathbf{0} \\ f_1^T & 1 & 0 \\ g_1^T & 0 & 1 \end{pmatrix}$$

where the row vector

$$f_1^T = \begin{pmatrix} -1 & 0_Q^T & \Phi_1^T & -\Phi_1^T & 0_Q^T & 0_Q^T \end{pmatrix},$$

and

$$g_1^T = \begin{pmatrix} -1 & 0_Q^T & -\Phi_1^T & \Phi_1^T & 0_Q^T & 0_Q^T \end{pmatrix}.$$

$$x_1 = \begin{pmatrix} x_0 \\ u_1 \\ v_1 \end{pmatrix}$$

where u_1 and v_1 are scalars. Also,

$$\hat{b}_1 = \begin{pmatrix} \hat{b}_0 \\ b_1 \\ -b_1 \end{pmatrix}$$

i.e., the resource vector has been augmented by the scalar RBF target of the first data point.

Thus, if the old basis for the LP was B_0, then the new basis, augmented by u, v, becomes

$$B_1 = \begin{pmatrix} B_0 & \mathbf{0} & \mathbf{0} \\ \tilde{f}_1^T & 1 & 0 \\ \tilde{g}_1^T & 0 & 1 \end{pmatrix}$$

where the tilde represents extracting the components of f, g associated with the basic variables. The inverse of B_1 may be efficiently computed from B_0 as described in [11].

5. Results

In this section we present apply the algorithm to several illustrative data sets. In each case we explore the behavior of the fit over the incremental learning process, as exhibited the evolution of ϵ. Also, since we have introduced a new *ad hoc* parameter C we explore its impact on the modeling and the extent to which tuning this parameter can be used to an advantage.

In each example we use the idea of time delayed embeddings [12]. We fit a model to predict based on previous delays, i.e.,

$$x_{n+T} = f(x_n, x_{n-T}, x_{n-2T})$$

where f is mapped by RBFs using the proposed algorithm. In each case we have selected an embedding dimension of 3 and a time delay T based on the autocorrelation.

5.1. Synthetic Time Series

To investigate the role of the parameter C we study the performance of the algorithm on two synthetic time series:

$$y_1(t;\delta) = \sin(t) + \sin(2t) + \delta\, d\mathrm{W}(t) \tag{12}$$

$$y_2(t;\delta) = \sin(t) + \delta\, \exp\left[\frac{-(t-t_0)^2}{2\sigma^2}\right] d\mathrm{W}(t) \tag{13}$$

where $d\mathrm{W}(t)$ are increments of white noise, with realizations of $\mathrm{W}(t) \sim \mathcal{N}(0,t)$ (distributed normally with mean zero and variance t), and $\delta \in [0,1]$, t_0, and σ parameters determining the shape of the data. For simplicity in this study, we fix the domain $t \in [0, 20\pi]$, $t_0 = 10\pi$, and $\sigma = 2.5\pi$.

The purpose of studying these equations is twofold. The first equation (12) has a stationary additive noise. Since the function is periodic in expected value,

ϵ and the number of RBFs will asymptotically approach a level characteristic for the time series and the choice of C. The second equation (13) uses a noise with time-dependent amplitude. This allows us to study how the algorithm adapts to novel information, and to what degree a choice of C allows us to choose a balance between a large number of RBFs when $C \ll 1$, or fewer RBFs and a large ϵ when $C \gg 1$.

Figure 2 illustrates the results of one such experiment. Here we set $\delta = 0.4$ and a lag time of $T = \pi/2$ is chosen for the time delayed embedding. We observe the dynamic behavior of the number of RBFs and ϵ for ten repetitions of random initial centers. The average of the timeseries is plotted, with the pointwise bound across all repetitions shown as transparent. In the left panels we illustrate $y_1(t, \delta)$ (top) and the evolution of the two model parameters. There is a learning phase for the algorithm over the course of approximately two periods of the function before the values of ε and the number of RBFs level off. Note that there are two vertical axes in the bottom row; the grid uses the left axes in the two panels, and reflects $\varepsilon(t)$, while the right axes are used for the number of RBFs. In the right panels the results for $y_2(t, \delta)$ are shown. Similar to $y_1(t; \delta)$, there is an initial period where new RBFs are introduced. As the noise in the signal increases drastically, it is important to note that the algorithm incorporates the new data into the model by increasing ϵ rather than continuing to add further RBFs, which we interpret as a resilience in the algorithm towards overfitting the data. Depending on the application, a balance between these extremes can be struck based on C alone. We propose that this feature of the algorithm makes it useful for a broad class of time series.

5.2. Mackey-Glass

In this experiment we apply the algorithm to Mackey-Glass data set in Figure 3. Details on this well-studied benchmarking problem may be found in [3] and references therein. The blue circles identify the locations on the graph where the new data point is determined to be infeasible and training is required via updates to the dual simplex algorithm. The black line shows the solution

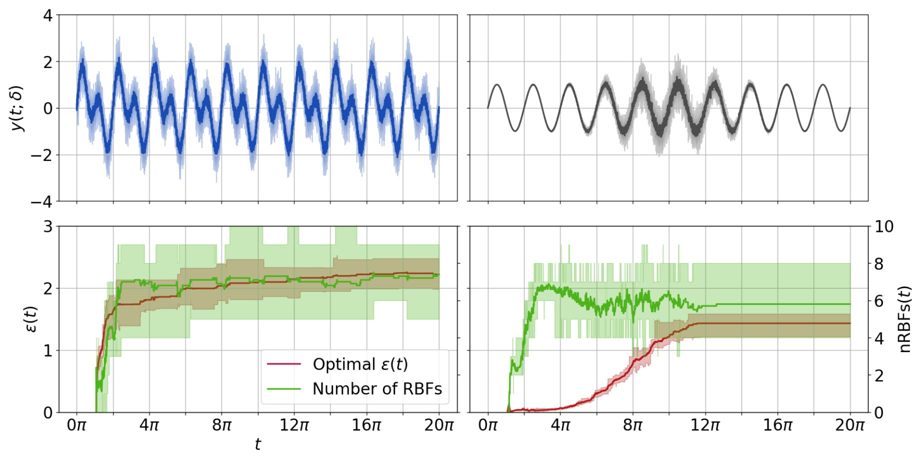

Figure 2: Ten realizations of time series (12) and (13) are plotted (top row) with corresponding $\epsilon(t)$ and $nRBFs(t)$ (bottom row), with the pointwise mean (solid) and interval between the pointwise minimum/maximum (shaded) shown in all panels. The typical behavior is an initial growth of RBFs and ϵ which then stabilizes. The right column demonstrates that the algorithm incorporates data with non-stationary noise by increasing ϵ rather than the number of RBFs.

for ϵ from the optimization problem as a function of time. We observe in Figure 3 a cluster of blue circles showing up in the first 100 time points. Meanwhile, ϵ is adjusted to an "optimal" value region such that the model starts learning the newly observed pattern with a reasonably small error threshold ϵ. From time point 200 to 1000, we see that blue circle occurs at peaks and valleys of the function, as well as at what appear to be inflection points.

As a comparison, the result of sequential simplex algorithm with fixed $\epsilon = 0.006$ on the same data is shown in Figure 1. In the first 100 time points, less blue circles are observed in Figure 1 when comparing to the new result in Figure 3. Also we see the "optimal" ϵ value automatically determined by the optimization problem is less than the pre-determined fixed ϵ in the first 100 time points. This suggests that, in contrast to the the model with a fixed $\epsilon = 0.006$, the automated ϵ selection model is increasing the learning rates at the beginning of the modeling process.

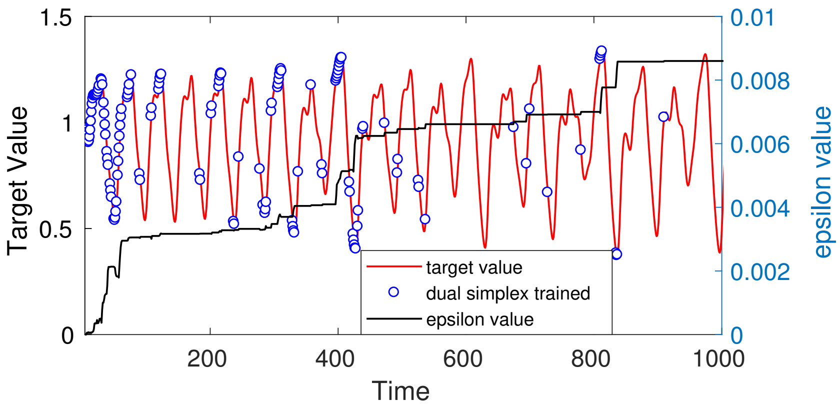

Figure 3: Mackey-Glass data training results. A 3-dimensional embedding with time delay $T = 1$ is used and we are predicting 1 step ahead. A total of 1000 input-output pairs are generated for training. A total of 500 initial centers are sampled from the domain of training data. The RBFs are selected to be Gaussians all with width 10. The sequential simplex algorithm identifies 164 out of 1000 data points as infeasible requiring dual simplex updates to solve for primal feasibility.

5.3. Fort Collins Weather Data

The Fort Collins weather data was collected at Christman Field Weather Station at Colorado State University [13]. The data set contains the temperature measurements sampled every 5 minutes over September, 2016. The data is smoothed via simple moving average. The temperature at k-th time point is the mean of the previous n temperatures. In particular, n is set to be 30 for this experiment.

In this example the propsed incremental algorithm is used on 8640 data points corrsponding to data over collected over September, 2016. It is observed in Figure 4 that when $C = 100$, 769 data points are identified as novel during the streaming process. The red dots identify the location of novel data points which we see occur a lot in the first 1000 time points as a learning process. After the first period of learning, we can observe novel data points shows up at extrema and inflection points. The number of iterations is capped at 15 through the whole streaming process.

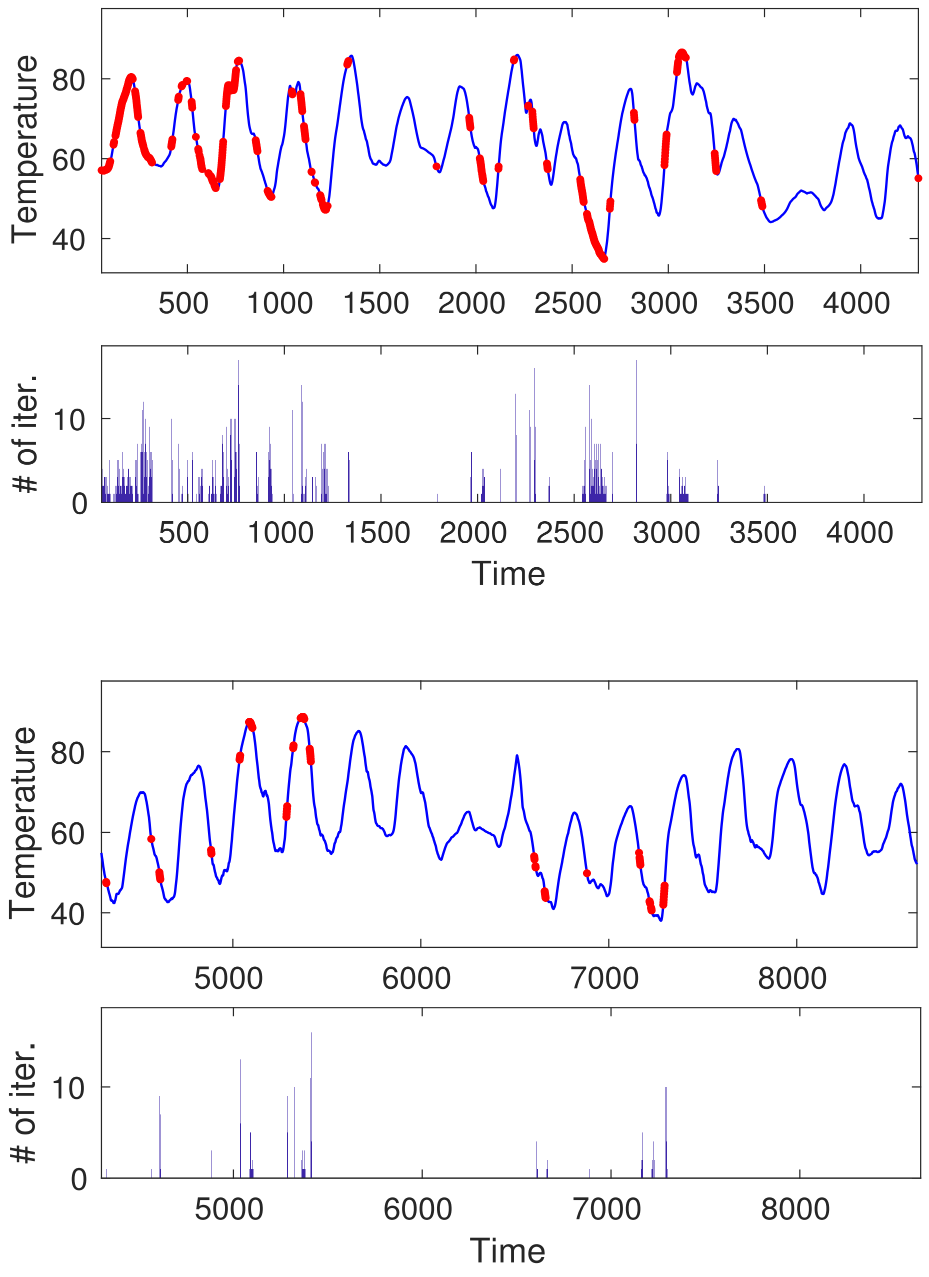

Figure 4: Fort Collins weather data training results (split into two panels). A 3-dimensional embedding with time delay $T = 6$ (30 minutes) is used and we are predicting 12 steps (1 hour) ahead. A total of 8511 input-output pairs are generated for training. A total of 500 initial centers are sampled from the domain of training data. The RBFs are selected to be Gaussians all with width 10. The sequential simplex algorithm identifies 769 out of 8511 data points as infeasible requiring dual simplex updates to solve for primal feasibility. This plot shows the location of the novel points with number of iterations used in dual simplex algorithm directly in the panel below. A total of $N_c = 16$ centers were selected for this model.

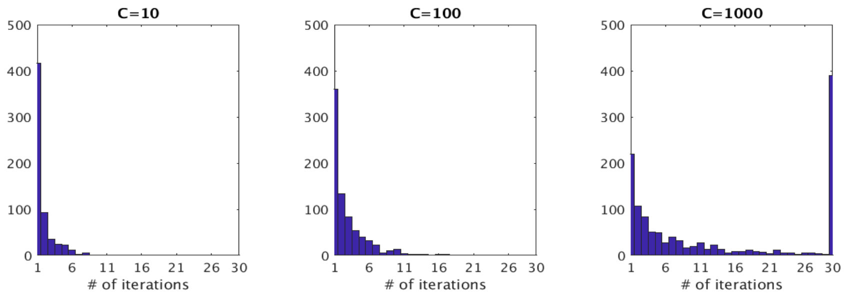

Figure 5: Histogram of number of iterations needed to obtain new optimal solution in dual simplex algorithm when novel data points are observed in the process of streaming Fort Collins weather data set.

Analysis of the ad hoc parameter C. The impact of the ad hoc parameter C is explored using the Fort Collins weather data set. In Table 1, we see as C varies from 10 to 1000, the average number of iterations in dual simplex algorithm is varying from 0.1239 to 1.958, while the median number of iterations is constantly 0. The number of novel data points is increasing from 613 when $C = 10$ to 1208 when $C = 1000$. Also we see in Figure 5 the number of iterations required to obtain a new optimal solution is increasing as we vary C from 10 to 1000. When $C = 10000$, we see 7216 out of 8511 data points are identified as novel and median number of iterations required to obtain optimal solution becomes 30 at which the maximum number of iteration is capped,i.e. optimal solution is not obtained at these novel points. The expense of model training is increasing as we turn up this ad hoc parameter C.

One fact we also observe from Table 1 is when $C = 10$, only 3 RBFs are needed to fit weather data while as we bring C up to 1000, 31 RBFs are selected. The optimal error threshold ϵ given by the model at the end of streaming process is decreasing from 17.15 to 3.39. We see a better goodness of fit with larger value of C.

C can be viewed as a knob to adjust the desired complexity of the model. More complex model result in significant increment in the training expense while simple model is lacking in goodness of fit. The ad hoc parameter C in this model serves as a balancing tool between sparsity and accuracy.

Table 1: Impact of the ad hoc parameter C on the learning of the weather data.

	$C = 10$	$C = 100$	$C = 1000$	$C = 10000$
Iterations (mean)	0.1239	0.2463	1.958	24.0629
Iterations (median)	0	0	0	30
Iterations (STD)	0.5862	1.1115	6.6931	11.6483
Novel data points	613	769	1208	7216
Number of RBFs	3	16	31	50
Error threshold(ϵ)	17.15	5.64	3.39	0.64

6. Prior Art

Radial Basis Functions have been established as a theoretically and algorithmically attractive tool for approximating functions see, e.g., [5, 6, 7, 8]. They have been applied a broad variety of problems including the solution of PDEs [14, 15] and modeling data [1, 2].

Our interest in RBFs, in addition to basic applications to data modeling, concern addressing fundamental algorithm aspects that relieve the user from making parameter decisions. Determining the number of basis functions and there locations is a fundamental challenge experience by anyone attempting nonlinear modeling. We made a basic first step in this direction using an autocorrelation test [16]; see also [17]. Additional extensions have been proposed, including the use of skew RBFs which greatly increase the representation power of the approximate functions [18, 19, 20]. We have provided convergence results for this method in [3].

The work here builds on [21] where a two stage optimization approach was proposed for learning RBFs where one component of the learning was sparsity driven and employed the simplex algorithm. This paper fundamentally extends our previous work [9] to the case where the error fitting parameter selection is automated.

7. Conclusions

We have proposed and algorithm for time-series prediction that employs an adaptive error to determine the quality of the model fit. The quality of the fit ϵ is determined as part of the linear program formulation and serves to regularize the model. We observe that when noise is added to the data in our synthetic example, the solution value for ϵ compensates to avoid over-fitting. We have introduced an ad hoc parameter C that can be adjusted to tune the desired complexity of the representation.

We have illustrated that this approach can be viewed as a tool for anomaly detection using the LP feasibility test. This also serves the dual purpose of a data selection tool. Although we did not consider it here, this small data set can be used to refine the model via a second stage nonlinear optimization problem, see, [21].

We have focused on fitting the time-series using the time-delayed embedding framework to predict future values. In future work we will look at the batch training of spatial data using the adaptive fit idea.

Acknowledgements

This paper is based on research partially supported by the National Science Foundation under Grants No. DMS-1513633, and DMS-1322508 as well as DARPA awards N66001-17-2-4020 and D17AP00004.

References

[1] D. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, Complex Systems 2 (1988) 321–355.

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

[3] A. A. Jamshidi, M. J. Kirby, A radial basis function algorithm with automatic model order determination, SIAM Journal on Scientific Computing 37 (3) (2015) A1319–A1341.

[4] A. Jamshidi, Modeling spatio-temporal systems with skew radial basis functions: Theory, algorithms and applications, Ph.D. dissertation, Colorado State University, Department of Mathematics (2008).

[5] J. Park, I. W. Sandberg, Universal approximation using radial-basis-function networks, Neural Computation 3 (2) (1991) 246–257. arXiv:https://doi.org/10.1162/neco.1991.3.2.246, doi:10.1162/neco.1991.3.2.246.
URL https://doi.org/10.1162/neco.1991.3.2.246

[6] M. Powell, Radial basis functions for multivariable interpolation: A review, in: J. Mason, M. Cox (Eds.), Algorithms for Approximation, Oxford: Clarendon Press, 1987, pp. 143–167.

[7] M. Powell, The theory of radial basis functions in 1990, in: W. Light (Ed.), Advances in Numerical Analysis II: Wavelets Subdivision and Radial Basis Functions, Oxford Univeristy Press, 1992, pp. 105–210.

[8] M. D. Buhmann, Radial Basis Functions: Theory and Implementations, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2003. doi:10.1017/CBO9780511543241.

[9] X. Ma, T. Ghosh, M. Kirby, A sequential simplex algorithm for automatic data and center selecting radial basis functions, in: 2017 International Joint Conference on Neural Networks (IJCNN), 2017, pp. 549–556. doi:10.1109/IJCNN.2017.7965901.

[10] M. Orr, Introduction to radial basis function networks, Technical Report, Institute for Adaptive and Neural Computation, Edinburgh University, Edinburgh, U.K. (1996).

[11] D. Bertsimas, J. Tsitsiklis, Introduction to Linear Optimization, Athena Scientific, 1997.

[12] F. Takens, Detecting Strange Attractors in Turbulence, Dynamical Systems and Turbulence, Warwick 1980; Proceeding of a symposium held at University of Warwick 1979-1980 366–381.

[13] C. S. University, Christman field weather station, `www.atmos.colostate.edu/wx/fcc/` (2017).

[14] C. Franke, R. Schaback, Solving partial differential equations by collocation using radial basis functions, Applied Mathematics and Computation 93 (1) (1998) 73 – 82. `doi:https://doi.org/10.1016/S0096-3003(97)10104-7`.
URL `http://www.sciencedirect.com/science/article/pii/S0096300397101047`

[15] B. Fornberg, N. Flyer, Solving pdes with radial basis functions, Acta Numerica 24 (2015) 215258. `doi:10.1017/S0962492914000130`.

[16] A. Jamshidi, M. Kirby, Towards a black box algorithm for nonlinear function approximation over high-dimensional domains, SIAM J. Sci. Comput. 29 (2007) 941.

[17] M. Anderle, M. Kirby, Correlation feedback resource allocation rbf, in: Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on, Vol. 3, 2001, pp. 1949–1953 vol.3. `doi:10.1109/IJCNN.2001.938462`.

[18] A. Jamshidi, M. Kirby, Skew-radial basis functions for modeling edges and jumps, in: Mathematics in Signal Processing Conference Digest, The Insititute for Mathematics and its Applications, Royal Agricultural College, Cirencester, U.K., 2008.

[19] A. Jamshidi, M. Kirby, Skew-radial basis function expansions for empirical modeling, SIAM J. Sci. Comput. 31 (6) (2010) 4715–4743.

[20] A. Jamshidi, M. Kirby, Modeling multivariate time series on manifolds with skew radial basis functions, Neural Computation 23 (1) (2011) 97–123.

[21] T. Ghosh, M. Kirby, X. Ma, Dantzig-Selector Radial Basis Function Learning with Nonconvex Refinement, Springer International Publishing, Cham, 2017, pp. 313–327. `doi:10.1007/978-3-319-55789-2_22`.
URL `https://doi.org/10.1007/978-3-319-55789-2_22`