Sparse Regression Modelling Using an Incremental Weighted Optimization Method Based on Boosting with Correlation Criterion

S. Chen†, X.X. Wang† and D.J. Brown†
† School of Electronics and Computer Science
University of Southampton, Southampton SO17 1BJ, U.K.
E-mail: sqc@ecs.soton.ac.uk
† Department of Creative Technologies
University of Portsmouth, Portsmouth PO1 3HE, U.K.

Abstract

A novel technique is presented to construct sparse Gaussian regression models. Unlike most kernel regression modelling methods, which restrict kernel means to the training input data and use a fixed common variance for all the regressors, the proposed technique can tune the mean vector and diagonal covariance matrix of individual Gaussian regressor to best fit the training data based on the correlation between the regressor and the training data. An efficient repeated weighted optimization method is developed based on boosting with the correlation criterion to append regressors one by one in incremental regression modelling. Experimental results obtained using this construction technique demonstrate that it offers a viable alternative to the existing state-of-art kernel modelling methods for constructing parsimonious regression models.

Index Terms — Regression, construction algorithm, correlation, mean square error, boosting

1 Introduction

A basic principle in practical nonlinear data modelling is the parsimonious principle of ensuring the smallest possible model that explains the training data. Forward selection using the orthogonal least square (OLS) algorithm [1]–[4] is popular for nonlinear data modelling practitioners, for the reason that the algorithm is simple and efficient, and is capable of producing parsimonious linear-in-the-weights nonlinear models. Recently, the state-of-art sparse kernel modelling techniques, such as the support vector machine and relevant vector machine [5]–[7], have widely been adopted in data modelling applications. In most of these sparse regression modelling techniques, a fixed common variance is used for all the regressor kernels and the kernel centers or means are placed at the training input data.

We present a flexible construction method for Gaussian kernel models. The correlation between a Gaussian regressor and the training data is used as the criterion in positioning (mean adjustment) and shaping (diagonal covariance adjustment) the regressor. To incrementally append regressor one by one, a repeated weighted optimization search algorithm is developed, which is based on the idea from boosting [8]–[10]. Because kernel means are not restricted to the training input data and each regressor has an individually tuned diagonal covariance matrix, our method can produce very sparse models. The proposed repeated weighted optimization algorithm is simple, robust and easy to implement. Experimental results are used to demonstrate the effectiveness of this incremental construction algorithm.

2 The Proposed Construction Method

We consider the modelling problem of approximating the \( N \) pairs of training data \( \{x_i, y_i\}_{i=1}^N \) with the regression model

\[
\hat{y}(x) = \sum_{i=1}^{M} w_i g_i(x)
\]

where \( x \) is the \( m \)-dimensional input variable; \( w_i, 1 \leq i \leq M \), denote the model weights; \( M \) is the number of regressors; and \( g_i(\cdot), 1 \leq i \leq M \), denote the regressors. We allow the regressor to be chosen as the general Gaussian function \( g_i(x) = G(x; \mu_i, \Sigma_i) \) with

\[
G(x; \mu_i, \Sigma_i) = e^{-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i)}
\]

where \( \Sigma_i = \text{diag}\{\sigma_{i,1}^2, \ldots, \sigma_{i,m}^2\} \). We will adopt an incremental approach to build up the regression model (1) by appending regressors one by one.

2.1 Correlation criterion for selecting regressor

Let us first introduce the following notation

\[
\begin{align*}
\begin{cases}
y_i^{(0)} = y_i \\
y_i^{(k)} = y_i^{(k-1)} - w_k g_k(x_i)
\end{cases}
\end{align*}
\]

\[1 \leq i \leq N \quad (3)\]
Obviously, $y_i^{(k)}$ is the modelling error at $x_i$ after the $k$th regressor has been fitted. At the $k$th stage of incremental modelling, the regressor $g_k(x)$ is fitted to the training data set $\{y_i^{(k-1)}, x_i\}_{i=1}^N$. The correlation function between the regressor and the training data set given by

$$C_k = \frac{\sum_{i=1}^N g_k(x_i)y_i^{(k-1)}}{\sqrt{\sum_{i=1}^N g_k^2(x_i)}\sqrt{\sum_{i=1}^N (y_i^{(k-1)})^2}}$$

(4)

defines the “similarity” between the regressor $g_k(x)$ and the training data $\{y_i^{(k-1)}, x_i\}_{i=1}^N$. The larger value of $|C_k|$ is, the more similar they are.

**Regressor positioning and shaping.** With the Gaussian regressor $g_k(\bullet) = G(\bullet; \mu_k, \Sigma_k)$, the correlation $C_k$ is a function of the kernel mean $\mu_k$ and covariance matrix $\Sigma_k$. Thus, the correlation criterion (4) can be used for positioning and shaping $g_k(\bullet)$ by maximizing $|C_k|$ with respect to $\mu_k$ and $\Sigma_k$.

**Calculation of the model weight.** After the determination of the $k$th regression $g_k(x)$, the corresponding model weight $w_k$ is calculated by minimizing the mean square error (MSE) for the $k$-term regression model

$$MSE_k = \frac{1}{N} \sum_{i=1}^N \left( y_i^{(k-1)} - w_k g_k(x_i) \right)^2$$

(5)

This leads to the usual least square solution

$$w_k = \frac{\sum_{i=1}^N y_i^{(k-1)} g_k(x_i)}{\sum_{i=1}^N g_k^2(x_i)}$$

(6)

We now prove that selecting regressors by incrementally maximizing $|C_k|$ is equivalent to that by incrementally minimizing the modeling MSE (5). In fact, substituting $w_k$ in (5) by the least square solution (6), it can be shown that

$$MSE_k = \left( \frac{1}{N} \sum_{i=1}^N \left( y_i^{(k-1)} \right)^2 \right) \left( 1 - C_k^2 \right)$$

(7)

Obviously, maximizing $|C_k|$ with respect to $\mu_k$ and $\Sigma_k$ is identical to minimizing $MSE_k$ with respect to $\mu_k$ and $\Sigma_k$.

### 2.2 Repeated weighted search method

It is seen that at each increment regression stage, the basic task is to maximize some function $f(\mathbf{u})$ over $\mathbf{u} \in U$, where $f(\mathbf{u}) = |C_k(\mathbf{u})|$, and $\mathbf{u}$ contains $\mu$ and $\Sigma$. We use the following simple search method to perform this optimization. Given $s$ points of $\mathbf{u}$, $\mathbf{u}_1, \cdots, \mathbf{u}_s$, let $\mathbf{u}_{best} = \arg \min \{ f(\mathbf{u}_i), 1 \leq i \leq s \}$ and $\mathbf{u}_{word} = \arg \max \{ f(\mathbf{u}_i), 1 \leq i \leq s \}$. A $(s+1)$th value is generated by a weighted combination of $\mathbf{u}_i$, $1 \leq i \leq s$. Because this weighted combination is a convex combination, the point $\mathbf{u}_{s+1}$ is always within the convex hull defined by the $s$ values. A $(s+2)$th value is then generated as the mirror image of $\mathbf{u}_{s+1}$, with respect to $\mathbf{u}_{best}$. The best of $\mathbf{u}_{s+1}$ and $\mathbf{u}_{s+2}$ then replaces $\mathbf{u}_{word}$. The process is repeated until it converges. Obviously, the weighting values used to perform this combination are critical, and we adopt the idea of boosting [8]-[10] to adapt these weightings. This leads to the following basic weighted optimization algorithm.

**WeightedOp**

*Initialization:* Given $\{x_i, y_i^{(k-1)}\}_{i=1}^N$ and the $s$ initially chosen values for $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_s$, set iteration index $t = 0$ and $\delta_{i}^{(t)} = \frac{1}{s}$ for $1 \leq i \leq s$.

**Step 1: Boosting**

1. Calculate the loss of each point, namely

$$\text{cost}(j) = 1 - |C_k(\mathbf{u}_j)|, \quad 1 \leq j \leq s$$

2. Find $\mathbf{u}_{best} = \arg \min \{ \text{cost}(j), 1 \leq j \leq s \}$

3. Find $\mathbf{u}_{word} = \arg \max \{ \text{cost}(j), 1 \leq j \leq s \}$

4. Normalize the loss

$$\text{loss}(j) = \frac{\text{cost}(j)}{\sum_{l=1}^{s} \text{cost}(l)}, \quad 1 \leq j \leq s$$

5. Compute a weighting factor $\beta_t$ according to

$$\epsilon_t = \sum_{j=1}^{s} \delta_{j}^{(t)} \text{loss}(j), \quad \beta_t = \frac{\epsilon_t}{1 - \epsilon_t}$$

6. Update the weighting vector

$$\delta_{j}^{(t+1)} = \left\{ \begin{array}{ll} \delta_{j}^{(t)} \beta_{t}^{\text{loss}(j)} & \text{for } \beta_{t} \leq 1, \\ \delta_{j}^{(t)} \beta_{t}^{1 - \text{loss}(j)} & \text{for } \beta_{t} > 1, \end{array} \right.$$  

where $j = 1, \cdots, s$

6. Normalize the weighting vector

$$\delta_{j}^{(t+1)} = \frac{\delta_{j}^{(t+1)}}{\sum_{l=1}^{s} \delta_{l}^{(t+1)}}, \quad 1 \leq j \leq s$$

**Step 2: Parameter updating**

1. Construct the $(s+1)$th point using the formula

$$\mathbf{u}_{s+1} = \sum_{i=1}^{s} \delta_{i}^{(t+1)} \mathbf{u}_i$$

2. Construct the $(s+2)$th point using the formula

$$\mathbf{u}_{s+2} = \mathbf{u}_{best} + (\mathbf{u}_{best} - \mathbf{u}_{s+1})$$

3. Choose a better point from $\mathbf{u}_{s+1}$ and $\mathbf{u}_{s+2}$ to replace $\mathbf{u}_{word}$. Repeat from Step 1 until the $(s+1)$th value changes very little compared with the last round, or a preset maximum number of iterations has been reached.
Fig. 1. Incremental modelling results for Example 1: in (a)–(f), the light curves are the modelling errors of the previous stage, \( y^{(k-1)} \), and the dark curves are the fitted current regressors, \( w_k \phi_k(\mathbf{x}_i) \), for \( 1 \leq k \leq 6 \), respectively; in (g), the light curve is the noisy training data \( y_i \) and the dark curve is the final 6-term model \( \hat{g} \); and (h) shows the final modelling errors.

The above \textit{WeightedOp} algorithm performs a guided random search and solution obtained may depend on the initial choice of the population. To derive a robust algorithm that ensures a stable “global” solution, we simply augment the algorithm into the following \textit{ReWeightedOp} algorithm.

\textbf{ReWeightedOp}

\textit{Repeat loop: For } \( l = 1 : M_R \)

Set \( \mathbf{u}_l^{(l)} = \mathbf{u}_{l-1}^{(l-1)} \), and randomly generate the other \( s - 1 \) points \( \mathbf{u}_i^{(l)} \) for \( 2 \leq i \leq s \)

Call the \textit{WeightedOp} algorithm to obtain a solution \( \mathbf{u}_{best}^{(l)} \)

\textbf{If } \| \mathbf{u}_{best}^{(l-1)} - \mathbf{u}_{best}^{(l)} \| < \xi, \text{ Exit loop}

\textbf{End for}

The solution is \( \mathbf{u}_{best}^{(l)} \)

In the \textit{ReWeightedOp}, \( M_R \) is the number of maximum repeating runs and \( \xi \) the termination threshold value.

2.3 Incremental regression modelling

The completed algorithm for incremental regression modelling can now be summarized. Choose a preset modelling accuracy \( \xi \), and set \( k = 0 \).

\textit{Do: } \( k = k + 1 \)

1. Call \textit{ReWeightedOp} to determine \( \mu_k \) and \( \Sigma_k \), the position and diagonal covariance matrix of the \( k \)th regressor

2. Calculate the weight \( w_k \) for the \( k \)th regressor according to (6) and compute the modelling errors \( y_i^{(k)} = y^{(k-1)} - w_k \phi_k(\mathbf{x}_i) \), \( 1 \leq i \leq N \)

\textbf{While } \text{MSE}_k > \xi

The termination of the model construction process can also be decided using cross validation [11],[12]. A simple method is to have a separate validation data set. The model construction is based on the training data set, while the performance of the selected model, the MSE, is monitored over the validation data set. The construction process is terminated when the MSE over the
validation data set stops improving. Alternatively, the Akaike information criterion [13], the optimal experimental design criteria [4] and the leave-one-out generalization criterion [14] may be adopted to automatically terminate the model construction process without the need for a separate validation data set.

3 EXPERIMENTAL RESULTS

Example 1. The 500 points of training data were generated from

\[ y(x) = 0.1x + \frac{\sin x}{x} + \sin 0.5x + \epsilon \]

with \( x \in [-10, 10] \), where \( \epsilon \) was a Gaussian white noise with zero mean and variance 0.01. The population size and the number of maximum repeating times used in \textit{ReWeightedOp} were \( s = 9 \) and \( M_R = 10 \). With the modelling accuracy set to \( \xi = 0.012 \), the incremental regression modelling produced 6 Gaussian regressors, as summarized in Table I, and the construction process is also illustrated graphically in Fig. 1 (a)–(f). In Fig. 1 (g), the model output from the constructed 6-term model is superimposed on the noisy training data, and the final modelling errors are shown in Fig. 1 (h).

Example 2. This example constructed a model for the gas furnace data set (Series J in [15]). The data set contained 296 pairs of input-output points, where the input \( u(t) \) was the coded input gas feed rate and the output \( y(t) \) represented \( \text{CO}_2 \) concentration from the gas furnace. The input-output data are depicted in Fig. 2 (a) and (b), respectively. The training data set was constructed with \( y_i = y(i) \) and \( x_i = [y(i-1) y(i-2) y(i-3) u(i-1) u(i-2) u(i-3)]^T \) for \( i = 4, 5, \ldots, 296 \). In the previous work [14], it was found out that various existing state-of-art kernel modeling techniques required at least 28 model terms to achieve a modelling accuracy of \( \xi = 0.054 \) for this data set.

With a population size \( s = 147 \) and the number of maximum repeating times \( M_R = 20 \) in \textit{ReWeightedOp}, and a preset modelling accuracy of \( \xi = 0.054 \), the proposed incremental modelling procedure produced 18 Gaussian regressors, and the resulting model is listed in Table II. Fig. 2 (b) depicts the model prediction \( \hat{y}(t) \) for this 18-term model, in comparison with the system output \( y(t) \). The corresponding model prediction error \( \epsilon(t) = y(t) - \hat{y}(t) \) is shown in Fig. 2 (c). It is seen that the proposed approach is able to produce a sparser regression model with an equally good modeling performance over the various existing state-of-art kernel modeling methods.

4 CONCLUSIONS

A construction algorithm has been proposed to incrementally fit sparse Gaussian regression models. The algorithm has the ability to tune the mean vector and diagonal covariance matrix of individual Gaussian regressors to best fit the training data based on the correlation between the regressor and the training data. A repeated weighted optimization search method has been developed based on boosting with the correlation criterion to append regressors one by one in incremental regression modelling. Experimental results presented have demonstrated the effectiveness of the proposed technique.

REFERENCES


Fig. 2. Example 2, the gas furnace data set: (a) system input $u(t)$, (b) model output $\hat{y}(t)$ (dashed) superimposed on system output $y(t)$ (solid), and (c) model prediction error $e(t) = y(t) - \hat{y}(t)$. 
<table>
<thead>
<tr>
<th>( k )</th>
<th>( \mu_k )</th>
<th>( \Sigma_k )</th>
<th>( w_k )</th>
<th>( MSE_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \begin{pmatrix} 61.0000 \ 6.0783 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 50.9525 \ 16.6549 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 3.3340 \ 2.4340 \end{pmatrix} )</td>
<td>( 2.8443 \times 10^{-3} )</td>
</tr>
<tr>
<td>1</td>
<td>( \begin{pmatrix} 60.2279 \ 0.0217 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 45.1000 \ 5.9837 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 3.3340 \ 6.2630 \end{pmatrix} )</td>
<td>( 59.276 )</td>
</tr>
<tr>
<td>2</td>
<td>( \begin{pmatrix} 46.8987 \ 0.0936 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 58.9860 \ 0.0762 \end{pmatrix} )</td>
<td>( \begin{pmatrix} -1.2067 \ 6.0783 \end{pmatrix} )</td>
<td>( 4.0451 )</td>
</tr>
<tr>
<td>3</td>
<td>( \begin{pmatrix} 45.1000 \ 5.7572 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 45.1000 \ 0.0476 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 3.3340 \ 7.9013 \end{pmatrix} )</td>
<td>( 2.4931 )</td>
</tr>
<tr>
<td>4</td>
<td>( \begin{pmatrix} 57.9404 \ 0.293 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 51.5699 \ 22.1583 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 3.3340 \ 19.0917 \end{pmatrix} )</td>
<td>( 29.190 )</td>
</tr>
<tr>
<td>5</td>
<td>( \begin{pmatrix} 58.0935 \ 0.101 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 45.1000 \ 0.2753 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 3.3340 \ 24.6911 \end{pmatrix} )</td>
<td>( 58.659 )</td>
</tr>
<tr>
<td>6</td>
<td>( \begin{pmatrix} 56.3757 \ 0.010 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 49.5204 \ 25.0000 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 2.2450 \ 0.0547 \end{pmatrix} )</td>
<td>( 21.472 )</td>
</tr>
<tr>
<td>7</td>
<td>( \begin{pmatrix} 45.1000 \ 0.1052 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 51.6922 \ 1.5136 \end{pmatrix} )</td>
<td>( \begin{pmatrix} -0.0144 \ 0.0501 \end{pmatrix} )</td>
<td>( -5.2669 )</td>
</tr>
<tr>
<td>8</td>
<td>( \begin{pmatrix} 46.0303 \ 25.0000 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 54.9481 \ 0.0385 \end{pmatrix} )</td>
<td>( \begin{pmatrix} -2.6996 \ 25.0000 \end{pmatrix} )</td>
<td>( 1.4557 )</td>
</tr>
<tr>
<td>9</td>
<td>( \begin{pmatrix} 58.7624 \ 0.010 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 55.1555 \ 0.0100 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 2.5233 \ 0.1178 \end{pmatrix} )</td>
<td>( 46.918 )</td>
</tr>
<tr>
<td>10</td>
<td>( \begin{pmatrix} 53.9952 \ 25.0000 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 54.8236 \ 0.0100 \end{pmatrix} )</td>
<td>( \begin{pmatrix} -1.7173 \ 0.0113 \end{pmatrix} )</td>
<td>( -1.7379 )</td>
</tr>
<tr>
<td>11</td>
<td>( \begin{pmatrix} 45.1000 \ 0.1052 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 54.9041 \ 0.0620 \end{pmatrix} )</td>
<td>( \begin{pmatrix} -2.7555 \ 15.7145 \end{pmatrix} )</td>
<td>( 48.259 )</td>
</tr>
<tr>
<td>12</td>
<td>( \begin{pmatrix} 54.7481 \ 0.0100 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 3.3340 \ 0.1000 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 61.0000 \ 25.0000 \end{pmatrix} )</td>
<td>( 46.2857 )</td>
</tr>
<tr>
<td>13</td>
<td>( \begin{pmatrix} 53.5265 \ 25.0000 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 48.3381 \ 0.0992 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 1.9158 \ 25.0000 \end{pmatrix} )</td>
<td>( 53.5265 )</td>
</tr>
<tr>
<td>14</td>
<td>( \begin{pmatrix} 53.4362 \ 0.0100 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 48.3381 \ 0.0100 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 1.9158 \ 0.0100 \end{pmatrix} )</td>
<td>( 7.962 )</td>
</tr>
<tr>
<td>15</td>
<td>( \begin{pmatrix} 53.7756 \ 0.0100 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 51.6104 \ 0.0100 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 0.7165 \ 25.0000 \end{pmatrix} )</td>
<td>( -0.9918 )</td>
</tr>
<tr>
<td>16</td>
<td>( \begin{pmatrix} 45.1000 \ 1.0087 \end{pmatrix} )</td>
<td>( \begin{pmatrix} 45.1000 \ 17.3402 \end{pmatrix} )</td>
<td>( \begin{pmatrix} -3.2160 \ 10.6542 \end{pmatrix} )</td>
<td>( 0.2511 )</td>
</tr>
</tbody>
</table>