Orthogonal Least Square with Boosting for Regression Modeling

S. Chen [†], X.X. Wang [‡] and D.J. Brown [‡]

[†] School of Electronics and Computer Science
University of Southampton, Southampton SO17 1BJ, U.K.

[‡] Department of Electronic and Computer Engineering
University of Portsmouth, Portsmouth PO1 3DJ, U.K.

Abstract

A novel technique is presented to construct sparse regression models based on the orthogonal least square method with boosting. This technique tunes the mean vector and diagonal covariance matrix of individual regressor by incrementally minimizing the training mean square error. An efficient weighted optimization method is developed based on boosting to append regressors one by one in an orthogonal forward selection procedure. Experimental results obtained using this construction technique demonstrate that it offers a viable alternative to the existing state-of-art kernel modeling methods for constructing parsimonious regression models.

I. Introduction

The orthogonal least square (OLS) algorithm [1]–[4] is popular for nonlinear data modeling practicians, 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 modeling techniques, such as the support vector machine and relevant vector machine [5]–[8], have widely been adopted in data modeling applications. In most of these sparse regression modeling techniques, a fixed common variance is used for all the regressor kernels and the kernel centers are placed at the training input data.

We present a flexible construction method that can tune the mean vector and diagonal covariance matrix of individual regressor by incrementally minimizing the training mean square error in an orthogonal forward selection procedure. To incrementally append regressor one by one, a weighted optimization search algorithm is developed, which is based on the idea from boosting [9]–[11]. 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 that generalize well.

II. Orthogonal least square regression modeling

Consider the modeling problem of approximating the N pairs of training data $\{\mathbf{x}_l, y_l\}_{l=1}^{N}$ with the regression model

$$y(\mathbf{x}) = \hat{y}(\mathbf{x}) + e(\mathbf{x}) = \sum_{i=1}^{M} w_i g_i(\mathbf{x}) + e(\mathbf{x}) \tag{1}$$

where $\mathbf{x}$ is the m-dimensional input variable, $y(\mathbf{x})$ is the target or desired output, $\hat{y}(\mathbf{x})$ is the model output, and $e(\mathbf{x})$ denotes the modeling error at $\mathbf{x}$; w_i, $1 \leq i \leq M$, denote the model weights, M is the number of regressors, and $g_i(\bullet)$, $1 \leq i \leq M$, denote the regressors. We allow the regressor to be chosen as the general Gaussian function

$g_i(\mathbf{x}) = G(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ with

$$G(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)\right) \tag{2}$$

where the diagonal covariance matrix has the form of $\boldsymbol{\Sigma}_i = \text{diag}\{\sigma_{i,1}^2, \cdots, \sigma_{i,m}^2\}$. We will adopt an orthogonal forward selection to build up the regression model (1) by appending regressors one by one. By defining $\mathbf{y} = [y_1 \; y_2 \cdots y_N]^T$,

$$\mathbf{G} = [\mathbf{g}_1 \; \mathbf{g}_2 \cdots \mathbf{g}_M] \quad \text{with} \quad \mathbf{g}_k = [g_k(\mathbf{x}_1) \; g_k(\mathbf{x}_2) \cdots g_k(\mathbf{x}_N)]^T \tag{3}$$

$\mathbf{w} = [w_1 \; w_2 \cdots w_M]^T$ and $\mathbf{e} = [e(\mathbf{x}_1) \; e(\mathbf{x}_2) \cdots e(\mathbf{x}_N)]^T$, the regression model (1) over the training data set can be written in the matrix form

$$\mathbf{y} = \mathbf{G}\mathbf{w} + \mathbf{e} \tag{4}$$

Let an orthogonal decomposition of the regression matrix $\mathbf{G}$ be

$$\mathbf{G} = \mathbf{P}\mathbf{A} \tag{5}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & a_{1,2} & \cdots & a_{1,M} \\ 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{M-1,M} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \tag{6}$$

and $\mathbf{P} = [\mathbf{p}_1 \; \mathbf{p}_2 \cdots \mathbf{p}_M]$ with orthogonal columns that satisfy $\mathbf{p}_i^T \mathbf{p}_j = 0$, if $i \neq j$. The regression model (4) can alternatively be expressed as

$$\mathbf{y} = \mathbf{P}\boldsymbol{\theta} + \mathbf{e} \tag{7}$$

where the orthogonal weight vector $\boldsymbol{\theta} = [\theta_1 \; \theta_2 \cdots \theta_M]^T$ satisfies the triangular system $\mathbf{A}\mathbf{w} = \boldsymbol{\theta}$. For the orthogonal regression model (7), the least square cost $J = \mathbf{e}^T\mathbf{e}/N$ can be expressed as

$$J = \frac{1}{N}\mathbf{e}^T\mathbf{e} = \frac{1}{N}\mathbf{y}^T\mathbf{y} - \frac{1}{N}\sum_{i=1}^{M} \mathbf{p}_i^T \mathbf{p}_i \theta_i^2 \tag{8}$$

Thus the least square cost for the k-term subset model can be expressed recursively as

$$J_k = J_{k-1} - \frac{1}{N}\mathbf{p}_k^T \mathbf{p}_k \theta_k^2 \tag{9}$$

where $J_0 = \mathbf{y}^T\mathbf{y}/N$. At the kth stage of regression, the kth term is selected to maximize the error reduction criterion $\text{ER}_k = \mathbf{p}_k^T \mathbf{p}_k \theta_k^2 / N$. However, unlike the original OLS algorithm [1]–[4], the maximization is with respect to the weight θ_k, the mean vector $\boldsymbol{\mu}_k$ and the diagonal covariance matrix $\boldsymbol{\Sigma}_k$ of the kth regressor. The forward selection procedure is terminated at the kth stage if

$$J_k < \xi \tag{10}$$

is satisfied, where the small positive scalar ξ is a chosen tolerance. This produces a parsimonious model containing k significant regressors. The termination of the model construction process can alternatively be decided using cross validation.

III. Orthogonal least square with boosting

At the kth stage of regression, the task is to maximize $f(\mathbf{u}) = \mathrm{ER}_k(\mathbf{u})$ over $\mathbf{u} \in U$, where the vector $\mathbf{u}$ contains the regressor mean vector $\boldsymbol{\mu}$ and diagonal covariance matrix $\boldsymbol{\Sigma}$. We use the following weighted search method to perform this optimization. Given s points of $\mathbf{u}$, $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_s$, let $\mathbf{u}_{best} = \arg\max\{f(\mathbf{u}_i), 1 \leq i \leq s\}$ and $\mathbf{u}_{worst} = \arg\min\{f(\mathbf{u}_i), 1 \leq i \leq s\}$. A $(s+1)$th value is generated by a weighted combination of $\mathbf{u}_i$, $1 \leq i \leq s$. A $(s+2)$th value is then generated as the mirror image of $\mathbf{u}_{s+1}$, with respect to $\mathbf{u}_{best}$, along the direction defined by $\mathbf{u}_{best} - \mathbf{u}_{s+1}$. The best of $\mathbf{u}_{s+1}$ and $\mathbf{u}_{s+2}$ then replaces $\mathbf{u}_{worst}$. The process is repeated until it converges. With the weightings updated by boosting [9]–[11], this leads to the following **OLSwB** algorithm.

Initialization: Give the training data $\{\mathbf{x}_i, y_i\}_{i=1}^N$ and J_{k-1}, and the s randomly chosen initial values for $\mathbf{u}$, $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_s$. Set iteration index $t = 0$ and $\delta_i^{(t)} = \frac{1}{s}$ for $1 \leq i \leq s$.

1. For $1 \leq i \leq s$, generate $\mathbf{g}_k^{(i)}$ from $\mathbf{u}_i$, the s candidates for the kth model column, and orthogonalize them

$$\alpha_{j,k}^{(i)} = \frac{\mathbf{p}_j^T \mathbf{g}_k^{(i)}}{\mathbf{p}_j^T \mathbf{p}_j}, \; 1 \leq j < k, \quad \mathbf{p}_k^{(i)} = \mathbf{g}_k^{(i)} - \sum_{j=1}^{k-1} \alpha_{j,k}^{(i)} \mathbf{p}_j$$

2. For $1 \leq i \leq s$, calculate the loss of each point, namely

$$\theta_k^{(i)} = \frac{\left(\mathbf{p}_k^{(i)}\right)^T \mathbf{y}}{\left(\mathbf{p}_k^{(i)}\right)^T \mathbf{p}_k^{(i)}}, \quad J_k^{(i)} = J_{k-1} - \frac{1}{N}\left(\mathbf{p}_k^{(i)}\right)^T \mathbf{p}_k^{(i)} \left(\theta_k^{(i)}\right)^2$$

Step 1: Boosting

1. Find

$$\mathbf{u}_{best} = \arg\min\{J_k^{(i)}, \; 1 \leq i \leq s\}, \quad \mathbf{u}_{worst} = \arg\max\{J_k^{(i)}, \; 1 \leq i \leq s\}$$

2. Normalize the loss

$$\bar{J}_k^{(i)} = \frac{J_k^{(i)}}{\sum_{l=1}^s J_k^{(l)}}, \; 1 \leq i \leq s$$

3. Compute a weighting factor β_t according to

$$\epsilon_t = \sum_{i=1}^s \delta_i^{(t)} \bar{J}_k^{(i)}, \quad \beta_t = \frac{\epsilon_t}{1 - \epsilon_t}$$

4. Update the weighting vector

$$\delta_i^{(t+1)} = \begin{cases} \delta_i^{(t)} \beta_t^{\bar{J}_k^{(i)}} & \text{for } \beta_t \leq 1, \\ \delta_i^{(t)} \beta_t^{1-\bar{J}_k^{(i)}} & \text{for } \beta_t > 1, \end{cases} \quad 1 \leq i \leq s$$

5. Normalize the weighting vector

$$\delta_i^{(t+1)} = \frac{\delta_i^{(t+1)}}{\sum_{l=1}^s \delta_l^{(t+1)}}, \; 1 \leq i \leq s$$

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. Orthogonalize these two candidate model columns and compute their losses.

4. Choose a better point from $\mathbf{u}_{s+1}$ and $\mathbf{u}_{s+2}$ to replace $\mathbf{u}_{worst}$ (which inherits the weighting δ value from $\mathbf{u}_{worst}$).

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.

From the converged population of s points, find $i_k = \arg\min\{J_k^{(i)}, \ 1 \le i \le s\}$ and select $\alpha_{j,k} = \alpha_{j,k}^{(i_k)}$, $1 \le j < k$,

$$\mathbf{p}_k = \mathbf{p}_k^{(i_k)} = \mathbf{g}_k^{(i_k)} - \sum_{j=1}^{k-1} \alpha_{j,k} \mathbf{p}_j$$

with $J_k = J_k^{(i_k)}$, and $\theta_k = \theta_k^{(i_k)}$. This also determines the kth regressor's mean vector and diagonal covariance matrix.

IV. Experimental results

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

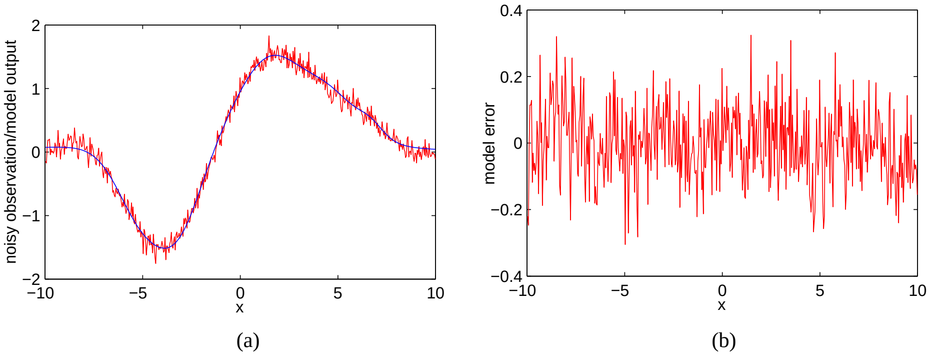
$$y(x) = 0.1x + \frac{\sin x}{x} + \sin 0.5x + \epsilon \tag{11}$$

with $x \in [-10, \ 10]$, where ϵ was a Gaussian white noise with zero mean and variance 0.01. The population size used in **OLSwB** was $s = 7$. With the modeling accuracy set to $\xi = 0.012$, the model construction procedure produced 6 Gaussian regressors, as summarized in Table I. Fig. 1 (a) depicts the model output $\hat{y}(x)$ generated from the constructed 6-term model, in comparison with the noisy training data $y(x)$, and Fig. 1 (b) shows the corresponding modeling error $e(x) = y(x) - \hat{y}(x)$.

TABLE I

OLSwB MODELING PROCEDURE FOR THE SIMPLE FUNCTION EXAMPLE.

regression step k	mean μ_k	variance σ_k^2	weight w_k	MSE J_k
0	–	–	–	0.8431
1	2.6911	4.2480	2.3527	0.3703
2	-4.0652	2.1710	-2.5197	0.0339
3	3.0314	2.0059	-1.0609	0.0172
4	-4.1771	1.0909	0.8982	0.0151
5	-1.9783	64.0000	0.1190	0.0129
6	6.6853	0.3894	0.1548	0.0118



(a)

(b)

Fig. 1. The simple function approximation: (a) noisy training data $y(x)$ (rough light curve) and model output $\hat{y}(x)$ (smooth dark curve), and (b) modeling error $e(x) = y(x) - \hat{y}(x)$.

Example 2. This example constructed a model representing the relationship between the fuel rack position (input $u(t)$) and the engine speed (output $y(t)$) for a Leyland TL11 turbocharged, direct injection diesel engine operated at low engine speed. Detailed system description and experimental setup can be found in [12]. The input-output data set contained 410 samples. The first 210 data points were used in training and the last 200 points in model validation. The previous study [4] has shown that this data set can be modeled adequately as $y_i = f_s(\mathbf{x}_i) + \epsilon_i$, with $y_i = y(i)$, $\mathbf{x}_i = [y(i-1)\; u(i-1)\; u(i-2)]^T$, where $f_s(\bullet)$ describes the unknown underlying system to be identified and $\epsilon_i = \epsilon(i)$ denotes the system noise.

With a population size $s = 37$ and a preset modeling accuracy of $\xi = 0.00055$, the **OLSwB** modeling procedure produced 6 Gaussian regressors, as listed in Table II. The MSE value of the constructed 6-term model over the testing set was 0.000573. Fig. 2 (a) depicts the model prediction $\hat{y}(t)$ superimposed on the system output $y(t)$ and Fig. 2 (b) shows the model prediction error $e(t) = y(t) - \hat{y}(t)$ for this 6-term model. It is worth pointing out that to achieve a same modeling accuracy for this data set the existing state-of-art kernel regression techniques required at least 22 regressors [4],[13].

V. Conclusions

A novel construction algorithm has been proposed for parsimonious regression modeling based on the OLS algorithm with boosting. The proposed algorithm has the ability to tune the mean vector and diagonal covariance matrix of individual regressor to incremen-

TABLE II

OLSwB MODELING PROCEDURE FOR THE ENGINE DATA SET.

step k	mean vector $\boldsymbol{\mu}_k$			diagonal covariance $\boldsymbol{\Sigma}_k$			weight w_k	MSE $J_k \times 100$
0		–			–		–	1558.9
1	5.2219	5.5839	5.6416	7.3532	21.0894	22.4661	6.0396	0.3866
2	4.2542	5.2741	4.1028	1.8680	10.0863	49.8826	-1.2845	0.1311
3	3.8826	5.1707	6.3200	0.1600	0.1600	64.0000	-0.1539	0.0996
4	2.3154	3.2544	5.4897	0.9447	0.3329	11.7564	-0.1433	0.0913
5	4.0673	4.4276	3.5963	0.1608	18.3731	0.2207	0.1945	0.0740
6	2.3663	3.2377	5.1376	0.1754	0.9317	0.1600	0.9658	0.0547

tally minimize the training mean square error. A weighted optimization search method has been developed based on boosting to append regressors one by one in an orthogonal forward regression procedure. Experimental results presented have demonstrated the effectiveness of the proposed technique.

REFERENCES

[1] S. Chen, S.A. Billings and W. Luo, "Orthogonal least squares methods and their application to non-linear system identification," *Int. J. Control*, Vol.50, No.5, pp.1873–1896, 1989.

[2] S. Chen, C.F.N. Cowan and P.M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Networks*, Vol.2, No.2, pp.302–309, 1991.

[3] S. Chen, Y. Wu and B.L. Luk, "Combined genetic algorithm optimisation and regularised orthogonal least squares learning for radial basis function networks," *IEEE Trans. Neural Networks*, Vol.10, No.5, pp.1239–1243, 1999.

[4] S. Chen, X. Hong and C.J. Harris, "Sparse kernel regression modelling using combined locally regularized orthogonal least squares and D-optimality experimental design," *IEEE Trans. Automatic Control*, Vol.48, No.6, pp.1029–1036, 2003.

[5] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

[6] V. Vapnik, S. Golowich and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," in: M.C. Mozer, M.I. Jordan and T. Petsche, Eds., *Advances in Neural Information Processing Systems 9*. Cambridge, MA: MIT Press, 1997, pp.281–287.

[7] B. Schlkopf and A.J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.

[8] M.E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Machine Learning Research*, Vol.1, pp.211–244, 2001.

[9] Y. Freund and R.E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Computer and System Sciences*, Vol.55, No.1, pp.119–139, 1997.

[10] R.E. Schapire, "The strength of weak learnability," *Machine Learning*, Vol.5, No.2, pp.197–227, 1990.

[11] R. Meir and G. Rätsch, "An introduction to boosting and leveraging," in: S. Mendelson and A. Smola, eds., *Advanced Lectures in Machine Learning*. Springer Verlag, 2003, pp.119–184.

[12] S.A. Billings, S. Chen and R.J. Backhouse, "The identification of linear and non-linear models of a turbocharged automotive diesel engine," *Mechanical Systems and Signal Processing*, Vol.3, No.2, pp.123–142, 1989.

[13] S. Chen, X. Hong, C.J. Harris and P.M. Sharkey, "Sparse modelling using orthogonal forward regression with PRESS statistic and regularization," *IEEE Trans. Systems, Man and Cybernetics, Part B*, to appear, 2004.

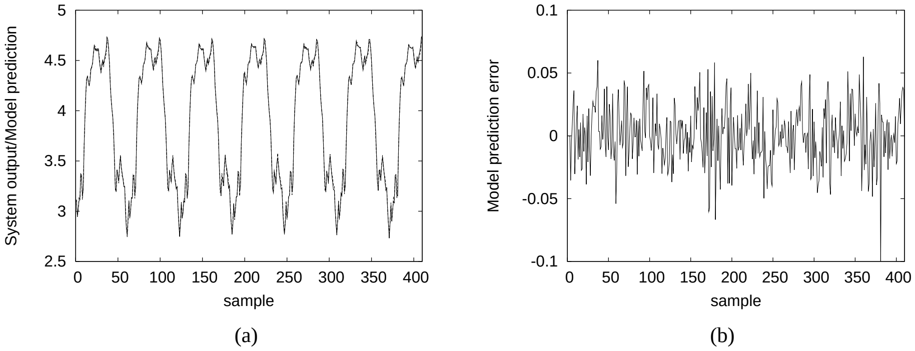


Fig. 2. The engine data set: (a) model prediction $\hat{y}(t)$ (dashed) superimposed on system output $y(t)$ (solid), and (b) model prediction error $e(t) = y(t) - \hat{y}(t)$.