IEICE TRANS. INF. & SYST., VOL.E94–D, NO.8 AUGUST 2011

PAPER

Nonparametric Regression Method Based on Orthogonalization and Thresholding

Katsuyuki HAGIWARA[†a)], ***Member***

SUMMARY In this paper, we consider a nonparametric regression problem using a learning machine defined by a weighted sum of fixed basis functions, where the number of basis functions, or equivalently, the number of weights, is equal to the number of training data. For the learning machine, we propose a training scheme that is based on orthogonalization and thresholding. On the basis of the scheme, vectors of basis function outputs are orthogonalized and coefficients of the orthogonalized vectors are estimated instead of weights. The coefficient is set to zero if it is less than a predetermined threshold level assigned component-wise to each coefficient. We then obtain the resulting weight vector by transforming the thresholded coefficients. In this training scheme, we propose asymptotically reasonable threshold levels to distinguish contributed components from unnecessary ones. To see how this works in a simple case, we derive an upper bound for the generalization error of the training scheme with the given threshold levels. It tells us that an increase in the generalization error is of $O(\log n/n)$ when there is a sparse representation of a target function in an orthogonal domain. In implementing the training scheme, eigen-decomposition or the Gram–Schmidt procedure is employed for orthogonalization, and the corresponding training methods are referred to as OHTED and OHTGS. Furthermore, modified versions of OHTED and OHTGS, called OHTED2 and OHTGS2 respectively, are proposed for reduced estimation bias. On real benchmark datasets, OHTED2 and OHTGS2 are found to exhibit relatively good generalization performance. In addition, OHTGS2 is found to be obtain a sparse representation of a target function in terms of the basis functions.

key words: *nonparametric regression, orthogonalization, hard thresholding, model selection*

1. Introduction

This paper considers a regression method using a learning machine that is defined by a linear combination of fixed basis functions. In particular, we focus on a machine in which the number of basis functions, or equivalently, the number of adjustable parameters or weights, is identical to the number of training data. This is viewed as a nonparametric regression problem in statistics. In machine learning, there are several approaches for this purpose, such as the support vector machine (SVM) [3]; least-squared SVMs (LS-SVMs) [10], or equivalently, regularized least squares (RLS) [9]; relevance vector machine (RVM) [11] and its variant called locally regularized orthogonal least squares (LROLS) [2]. In SVM, a kernel trick with the representer theorem yields a linear combination of kernel functions. LS-SVM/RLS is a quadratic type regularization method under squared error loss. In RVM, regularization parameters are assigned individually to weights, which are updated on the basis of Bayesian log evidence. LROLS is a variant of RVM in which basis function outputs are orthogonalized by a modified Gram–Schmidt procedure. In the procedure, some contributed orthogonal vectors are selected in a greedy manner in terms of residual reduction. Regularization parameters are individually assigned to coefficients of the orthogonal vectors and are updated as in RVM. The resulting weight vector is obtained by linear transformation of the coefficient vector.

In these training methods, adjustable parameters are needed to avoid over-fitting to training data and/or to obtain a sparse representation, although there are no systematic choices for these parameters. This problem is one of model selection. SVM parameters, C and ε, determine both the generalization capability and sparseness of the trained machine. In LS-SVM/RLS, a regularization parameter affects only the generalization capability. The training parameters are usually determined by using a resampling method such as cross-validation. In RVM and LROLS, there is no systematic method to determine the number of updates of regularization parameters and threshold levels necessary to remove fruitless weights.

In this paper, we propose a training scheme that is based on orthogonalization and hard thresholding. In this scheme, vectors of the outputs of basis functions are orthogonalized and some of the orthogonalized vectors are removed according to threshold levels. The advantage of the naive orthogonalization is that we can set theoretically reasonable component-wise threshold levels under some assumptions while it is difficult in a modified Gram–Schmidt procedure in LROLS. Thus, this scheme solves the above model selection problems automatically. For a simple situation, we also derive a generalization error bound when applying this training scheme with given threshold levels. In implementing this training scheme, we employ eigen-decomposition or the Gram–Schmidt procedure for orthogonalization, in which the corresponding methods are referred to as the orthogonalization and hard-thresholding method with eigen-decomposition (OHTED) and orthogonalization and hard-thresholding method with Gram–Schmidt procedure (OHTGS). We further modify OHTED and OHTGS in order to reduce estimation bias, which are referred to as OHTED2 and OHTGS2, respectively. The proposed methods are numerically tested on some real benchmark datasets and compared with a regularization method and LROLS in terms of generalization capability and sparseness.

Manuscript received January 7, 2011.
Manuscript revised April 12, 2011.

[†]The author is with the Faculty of Education, Mie University, Tsu-shi, 514–8507 Japan.
a) E-mail: hagi@edu.mie-u.ac.jp

DOI: 10.1587/transinf.E94.D.1610

In Sect. 2, we define the learning machine and describe its training scheme. In Sect. 3, we present the theoretical details of the training scheme. In Sect. 4, we discuss implementations of the training scheme detailed in Sect. 2. In Sect. 5, we present the results of numerical experiments on real benchmark datasets. Section 6 presents the main conclusions and the scope for future work.

2. Formulation of Training Procedure

In this section, we explain our training scheme after defining the learning machine.

2.1 Learning Machine

Let $\{(\boldsymbol{x}_i, y_i) : \boldsymbol{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}, i = 1, \ldots, n\}$ be a set of input–output training data. We assume that output data are generated by a rule:

$$y_i = h(\boldsymbol{x}_i) + e_i,\ i = 1, \ldots, n, \tag{1}$$

where h is a fixed function on $\mathbb{R}^d$ and $e_1, \ldots, e_n$ are i.i.d. Gaussian noise with mean zero and variance σ^2; that is, $e_i \sim N(0, \sigma^2)$. We refer to h by a target function or true function.

We consider a fitting problem by using a machine whose output for an input vector $\boldsymbol{x} \in \mathbb{R}^d$ is given by

$$f_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{j=1}^{n} w_j g_j(\boldsymbol{x}), \tag{2}$$

where $\boldsymbol{w} = (w_1, \ldots, w_n) \in \mathbb{R}^n$ is a weight vector. (2) is a linear combination of fixed basis functions. Note that the number of weights is identical to that of training data here. $g_j(\boldsymbol{x})$ can be written as $g(\boldsymbol{x}, \boldsymbol{x}_j)$ if we employ kernel functions as basis functions.

Let G be an $n \times n$ matrix whose (i, j)-th element is $g_j(\boldsymbol{x}_i)$. We define $\boldsymbol{g}_j = (g_j(\boldsymbol{x}_1), \ldots, g_j(\boldsymbol{x}_n))'$, which is the j-th column vector of G, where $'$ denotes the transpose. We assume that $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n$ are linearly independent. G has thus full rank. We redefine $\boldsymbol{w}$ as a vertical vector; that is, $\boldsymbol{w} = (w_1, \ldots, w_n)'$. We define $\boldsymbol{f}_{\boldsymbol{w}} = (f_{\boldsymbol{w}}(\boldsymbol{x}_1), \ldots, f_{\boldsymbol{w}}(\boldsymbol{x}_n))$, which is written as $\boldsymbol{f}_{\boldsymbol{w}} = G\boldsymbol{w}$. We also define $\boldsymbol{y} = (y_1, \ldots, y_n)'$, $\boldsymbol{e} = (e_1, \ldots, e_n)'$, and $\boldsymbol{h} = (h(\boldsymbol{x}_1), \ldots, h(\boldsymbol{x}_n))'$.

2.2 Transformation of G

Our training scheme is described step by step as follows: The flow of the training scheme is summarized in Fig. 1. In our training scheme, we first consider the transformation of G into a matrix with orthogonal column vectors.

$\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n\}$ can be regarded as a coordinate system since $\boldsymbol{f}_{\boldsymbol{w}}$ is represented by a weighted sum of those vectors. In this paper, we introduce another coordinate system obtained by transformation of the original coordinate system and is also an orthogonal system. Let Q be an invertible $n \times n$ matrix. We define $A = GQ$ and denote the jth column vector of A as $\boldsymbol{a}_j$. We assume that A has full rank and $A'A$ is a diagonal matrix by an appropriate choice of Q. Such Q exists since $\boldsymbol{g}_k$'s are linearly independent. We define $\Gamma = A'A$, where $\Gamma = \mathrm{diag}(\gamma_1, \ldots, \gamma_n)$. $\mathrm{diag}(\gamma_1, \ldots, \gamma_n)$ denotes a diagonal matrix whose (j, j)-th element is γ_j. $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\}$ is thus a set of orthogonal vectors. Since $A'A = Q'(G'G)Q$, a transformation by Q implements a diagonalization of $G'G$, which is the Gram matrix of $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n\}$. For given input data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ and fixed basis functions $g_1, \ldots, g_n$, Q can be viewed as a matrix that transforms $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n\}$ into $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\}$. We call $\boldsymbol{a}_j$ an orthogonal component or orthogonal basis vector. We define

$$\boldsymbol{v} = (v_1, \ldots, v_n)' = Q^{-1}\boldsymbol{w} \tag{3}$$

by which we can write $\boldsymbol{f}_{\boldsymbol{w}} = G\boldsymbol{w} = A\boldsymbol{v}$. We call $\boldsymbol{v}$ an orthogonal coefficient vector.

$$\begin{array}{ccc} (G, \boldsymbol{w}) & & (G, \overline{\boldsymbol{w}}) \\ \Big\downarrow Q^{-1}: \text{orthogonalization} & & Q \Big\uparrow \\ (A, \boldsymbol{v}) & \xrightarrow[\text{estimation}]{} (A, \widehat{\boldsymbol{v}}) \xrightarrow[\text{thresholding}]{} & (A, \overline{\boldsymbol{v}}) \end{array}$$

Fig. 1 Flow of training scheme using orthogonalization and thresholding.

2.3 Estimation of Orthogonal Coefficient Vector

In training of a machine with many degrees of freedom, there are two problems encountered: over-fitting and instability of the training process. If such a machine is trained under a squared error loss function, then the machine highly over-fits training data, resulting in poor prediction. In addition, for the typical choices for basis functions such as Gaussian and sigmoidal functions, vectors of the outputs of the basis functions can be nearly linearly dependent when we use many basis functions. This leads to instability of a trained weight vector.

The regularization method is one of the strategies for solving these two problems simultaneously. In the regularization method, we obtain an estimator of $\boldsymbol{v}$ that minimizes a regularized cost function defined by

$$C(\boldsymbol{v}) = \|\boldsymbol{y} - A\boldsymbol{v}\|^2 + \boldsymbol{v}'\Lambda\boldsymbol{v}, \tag{4}$$

where the first term is a squared error function and the second term is a regularization term or regularizer. Λ is an n-dimensional diagonal matrix defined by $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, where $\lambda_j \geq 0$ for all j. λ_j's are called regularization parameters. In a typical application of a regularization method, $\lambda = \lambda_j$ for all j. Note that we may introduce a regularizer only for stabilizing the training process since the generalization capability is improved by another source. Further note that in implementing our scheme, we will not introduce a regularizer and consider stability to be guaranteed by another technique. In this case, we just set $\lambda_j = 0$ for all j. In theoretical analyses, however, we take account of

a regularizer as a general case. A regularized estimator that minimizes $C(\boldsymbol{v})$ is defined by $\widehat{\boldsymbol{v}}$. By simple calculations, we have

$$\widehat{\boldsymbol{v}} = (\Gamma + \Lambda)^{-1} A' \boldsymbol{y} \tag{5}$$

since $A'A = \Gamma$. By the definition of Γ and Λ, we can write

$$\widehat{v}_j = \boldsymbol{a}_j' \boldsymbol{y} / (\gamma_j + \lambda_j) \tag{6}$$

for $j = 1, \ldots, n$.

2.4 Thresholding of Orthogonal Coefficient Vector

By introducing a set of threshold levels defined by $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)'$, we consider the application of a hard thresholding operator $T_{\boldsymbol{\theta}}$ to an orthogonal coefficient vector. The thresholded orthogonal coefficient vector is then given by $T_{\boldsymbol{\theta}}(\boldsymbol{v}) = (T_{\theta_1}(v_1), \ldots, T_{\theta_n}(v_n))'$, where

$$T_{\theta_j}(v_j) = \begin{cases} v_j & v_j^2 > \theta_j \\ 0 & v_j^2 \leq \theta_j \end{cases}, \ j = 1, \ldots, n. \tag{7}$$

For a regularized estimator given by (5), we define $\overline{\boldsymbol{v}} = T_{\boldsymbol{\theta}}(\widehat{\boldsymbol{v}})$. By (3), we then obtain $\overline{\boldsymbol{w}} = Q\overline{\boldsymbol{v}}$ as the resulting weight vector in our scheme.

3. Theoretical Details on Threshold Levels

In this section, we consider threshold levels in the above-mentioned training scheme.

3.1 Statistical Properties of $\widehat{\boldsymbol{v}}$

We assume that there exists a $\boldsymbol{w}^*$ such that $\boldsymbol{h} = G\boldsymbol{w}^*$ holds. We call $\boldsymbol{w}^*$ a true weight vector and define $\boldsymbol{v}^* = Q^{-1}\boldsymbol{w}^*$ on the basis of (3). We call $\boldsymbol{v}^* = (v_1^*, \ldots, v_n^*)'$ a true orthogonal coefficient vector. We define $\boldsymbol{\xi}_n = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ as the set of training inputs. We simply measure the generalization error on $\boldsymbol{\xi}_n$, which is written as

$$R(\boldsymbol{w}|\boldsymbol{\xi}_n) = \frac{1}{n}\sum_{i=1}^{n}(h(\boldsymbol{x}_i) - f_w(\boldsymbol{x}_i))^2 = \frac{1}{n}\|\boldsymbol{h} - G\boldsymbol{w}\|^2. \tag{8}$$

(8) is minimized if $\boldsymbol{w}^* = (G'G)^{-1}G'\boldsymbol{h}$ holds. We omit conditioning by $\boldsymbol{\xi}_n$ below.

We now have $\boldsymbol{e} \sim N(\boldsymbol{0}_n, \sigma^2 I_n)$, where $\boldsymbol{0}_n$ is the n-dimensional zero vector and I_n is the $n \times n$ identity matrix. We define $\boldsymbol{\mu} = (\mu_1 v_1^*, \ldots, \mu_n v_n^*)'$ in which

$$\mu_j = \frac{\gamma_j}{\gamma_j + \lambda_j}. \tag{9}$$

By (5), we have

$$\mathbb{E}[\widehat{\boldsymbol{v}}] = (\Gamma + \Lambda)^{-1} A' \boldsymbol{h} = (\Gamma + \Lambda)^{-1} A' G \boldsymbol{w}^* = \boldsymbol{\mu} \tag{10}$$

where we used $\boldsymbol{y} = \boldsymbol{h} + e$, $\boldsymbol{h} = G\boldsymbol{w}^*$, $\boldsymbol{v}^* = Q^{-1}\boldsymbol{w}^*$, and $A'A = \Gamma$. We define $S = \sigma^2(\Gamma + \Lambda)^{-2}\Gamma = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$ in which

$$\sigma_j^2 = \frac{\sigma^2 \gamma_j}{(\gamma_j + \lambda_j)^2}. \tag{11}$$

By (5) and (10), we have

$$\mathbb{E}[(\widehat{\boldsymbol{v}} - \mathbb{E}[\widehat{\boldsymbol{v}}])(\widehat{\boldsymbol{v}} - \mathbb{E}[\widehat{\boldsymbol{v}}])'] = S, \tag{12}$$

where we used $\mathbb{E}\boldsymbol{e}\boldsymbol{e}' = \sigma^2 I_n$ and $A'A = \Gamma$. Since $\boldsymbol{e} \sim N(\boldsymbol{0}_n, \sigma^2 I_n)$ and $\widehat{\boldsymbol{v}}$ is a linear transformation of $\boldsymbol{e}$, we have

$$\widehat{\boldsymbol{v}} \sim N(\boldsymbol{\mu}, S). \tag{13}$$

Since S is diagonal, $\widehat{v}_j \sim N(\mu_j v_j^*, \sigma_j^2)$, $j = 1, \ldots, n$ hold and are independent.

3.2 Thresholding Operation with Threshold Levels

By (3), we have $\boldsymbol{h} = G\boldsymbol{w}^* = A\boldsymbol{v}^*$. We define $V^* = \{j | v_j^* \neq 0,\ 1 \leq j \leq n\}$ and $V = \{j | v_j^* = 0, 1 \leq j \leq n\}$. V^* is a subset of indexes for which true orthogonal coefficients are nonzero. We refer to $\{\boldsymbol{a}_j | j \in V^*\}$ as true components or contributed components. K_n^* denotes the cardinality of V^*, and we define $K_n = n - K_n^*$ as the cardinality of V. Orthogonal coefficients that have indexes in V do not relate to a target function and are affected only by noise. We thus refer to $\{\boldsymbol{a}_j | j \in V\}$ as noise components. The purpose of thresholding is to remove noise components and keep contributed components. We should determine appropriate threshold levels to achive this purpose in some meanings. We here define

$$C_{n,\epsilon} = (2 + \epsilon)\log n, \tag{14}$$

where ϵ is a constant. Let δ be a positive constant below. We consider a thresholding operation $T_{\boldsymbol{\theta}_{n,\epsilon}}$ in which $\boldsymbol{\theta}_{n,\epsilon} = (\theta_{1,\epsilon}, \ldots, \theta_{n,\epsilon})$,

$$\theta_{j,\epsilon} = \sigma_j^2 C_{n,\epsilon}, \ j = 1, \ldots, n. \tag{15}$$

We present a theoretical implication of the threshold levels defined in (15). We define $Z_j = (\widehat{v}_j - \mu_j v_j^*)/\sigma_j$. By (13) and the definition of V, $Z_j = \widehat{v}_j/\sigma_j$ for $j \in V$ and $\{Z_j : j \in V\}$ are i.i.d. samples from $N(0, 1)$. By the definition of $\theta_{j,\epsilon}$, we have

$$\begin{aligned}\mathbb{P}\left[\bigcup_{j\in V}\left\{\widehat{v}_j^2 > \theta_{j,\delta}\right\}\right] &= \mathbb{P}\left[\bigcup_{j\in V}\left\{\widehat{v}_j^2/\sigma_j^2 > C_{n,\delta}\right\}\right] \\ &= \mathbb{P}\left[\max_{j\in V} Z_j^2 > C_{n,\delta}\right].\end{aligned}$$

Since $K_n \leq n$ and $\delta > 0$,

$$\lim_{n\to\infty}\mathbb{P}\left[\bigcup_{j\in V}\left\{\widehat{v}_j^2 > \theta_{j,\delta}\right\}\right] = 0 \tag{16}$$

by (A·4) in Appendix A. On the other hand, we consider the case where $K_n \geq \rho n$ for $\rho \in (0, 1]$. We then have

$$\mathbb{P}\left[\bigcap_{j\in V}\left\{\widehat{v}_j^2 \le \theta_{j,-\delta}\right\}\right] = \mathbb{P}\left[\bigcap_{j\in V}\left\{\widehat{v}_j^2/\sigma_j^2 \le C_{n,-\delta}\right\}\right] = \mathbb{P}\left[\max_{j\in V} Z_j^2 \le C_{n,-\delta}\right].$$

Since $\delta > 0$,

$$\lim_{n\to\infty}\mathbb{P}\left[\bigcap_{j\in V}\left\{\widehat{v}_j^2 \le \theta_{j,-\delta}\right\}\right] = 0 \tag{17}$$

by (A· 5) in Appendix A.

(16) tells us that for any $j \in V$, $\widehat{v}_j^2$ cannot exceed $\theta_{j,\delta}$ with high probability when n is large. If we employ $\theta_{j,\delta}$'s as component-wise threshold levels, then (16) implies that they have a high probability of removing all noise components if n is large. On the other hand, (17) tells us that there is at least one noise component for which $\widehat{v}_j^2 > \theta_{j,-\delta}$, $j \in V$ with high probability when n is large. In other words, there exists at least one noise component for which $\widehat{v}_j^2$ is close to $\theta_{j,0}$. Thus, we cannot distinguish a contributed component with $\widehat{v}_j^2 \le \theta_{j,-\delta}$ from a noise component. Alternatively, the j-th component should be recognized as a noise component if $\widehat{v}_j^2 \le \theta_{j,-\delta}$, even when it is not. Since δ can be set to an arbitrarily small value here, $\theta_j = \theta_{j,0}$, $j = 1,\ldots,n$ are critical levels and reasonable for deciding whether to remove or keep orthogonal components if $K_n \ge \rho n$. As a summary, (16) tells us that $\theta_j = \theta_{j,0}$, $j = 1,\ldots,n$ can be used for the worst case evaluations for noise levels and are appropriate threshold levels for removing noise components. Additionally, (16) and (17) says that those are tight if $K_n \ge \rho n$.

It may be important to consider a generalization property of the proposed training scheme. For a simple situation, we will give an upper bound on the generalization error defined by (8) when applying threshold levels defined by (15). We make the following two assumptions here:

(A1) $\gamma_j \ge 4\sigma^2 C_{n,\epsilon}/|v_j^*|^2$ for $j \in V^*$.
(A2) $\lambda_j \le \sqrt{\gamma_j}$ for $j \in V^*$.

(A1) implies that γ_j, $j \in V^*$ should be larger than $c_j \log n$ for some $c_j > 0$. Recall that γ_j is the squared norm of jth orthogonal basis vector with length n. If (A1) does not hold for a contributed component then almost all of output values of the component may be nearly zeros or exactly zeros since the right hand side of (A1) is of $O(\log n)$. This implies that the component may not contribute a target output; that is, for examples, it gives spikey outputs. We exclude this type of contributed outputs by (A1), which may be naturally accepted. On the other hand, if λ_j is large for $j \in V^*$ then the corresponding $\widehat{v}_j$ is biased from v_j^*. (A2) avoids this. Roughly speaking, regularization parameters may be required to go to zero as $n \to \infty$ for a consistency of an estimated model outputs. If this is true, (A2) is satisfied for a sufficiently large n under (A1).

Under (A1) and (A2), we have

$$\mathbb{P}\left[R(\overline{\boldsymbol{v}}|\boldsymbol{\xi}_n) > \frac{1}{n}\sum_{j=1}^{n}\alpha_j\right] \to 0 \tag{18}$$

as $n \to \infty$, where

$$\alpha_j = \begin{cases}\gamma_j\theta_{j,\epsilon} & j \in V^* \\ 0 & j \in V\end{cases} \tag{19}$$

for any $\epsilon > 0$. The proof is given in Appendix B. Note that (18) holds for any h while it gives an important implications under a sparseness condition as mentioned later. We conclude this section with some remarks including this point.

- The aim of applying a thresholding scheme is to remove noise components and keep contributed components. As seen in evaluation of P_2 in Appendix B and also evaluation of (16), component-wise threshold levels defined by (15) remove all noise components with high probability when n is large. Moreover, as seen in the evaluation of $P_{1,1}$ in Appendix B, component-wise threshold levels retain all of the contributed components with high probability when n is large.
- If $\gamma_j = 1$ and $\lambda_j = 0$ for all j, then our thresholding method on the orthogonal domain is consistent with a hard thresholding method with a universal threshold level in wavelet denoising, which has asymptotic optimality [4].
- If we define a generalization error by $R(\boldsymbol{w}) = E_{\boldsymbol{u}}[(h(\boldsymbol{u}) - f_{\boldsymbol{w}}(\boldsymbol{u}))^2]$ and assume that there exists $\boldsymbol{w}^* = (w_1^*,\ldots,w_n^*)'$ such that $h(\boldsymbol{u}) = \sum_{j=1}^n w_j^* g_j(\boldsymbol{u})$ for any $\boldsymbol{u}$, then we have $R(\boldsymbol{w}) = (\boldsymbol{w}-\boldsymbol{w}^*)'H(\boldsymbol{w}-\boldsymbol{w}^*)$, where H is an $n \times n$ matrix whose (k,l)-th element is $E_{\boldsymbol{u}}[g_k(\boldsymbol{u})g_l(\boldsymbol{u})]$. Here, $E_{\boldsymbol{u}}$ denotes an expectation with respect to a probability distribution P on $\mathbb{R}^d$. On the other hand, $R(\widehat{\boldsymbol{w}}|\boldsymbol{\xi}_n) = (\boldsymbol{w}^* - \widehat{\boldsymbol{w}})'(G'G/n)(\boldsymbol{w}^* - \widehat{\boldsymbol{w}})$ holds by (8), since $\boldsymbol{h} = G\boldsymbol{w}^*$ holds. If training inputs are randomly drawn from P, then by the strong law of large numbers, we have $\frac{1}{n}\sum_{i=1}^n g_k(\boldsymbol{x}_i)g_l(\boldsymbol{x}_i) \to \mathbb{E}_{\boldsymbol{u}}[g_k(\boldsymbol{u})g_l(\boldsymbol{u})]$ as $n \to \infty$ almost everywhere if basis functions are sufficiently smooth. In this situation, we can therefore expect that $R(\widehat{\boldsymbol{w}}|\boldsymbol{\xi}_n)$ approximates $R(\widehat{\boldsymbol{w}})$ for a sufficiently large n. However, we need a proof to rigorously support the validity of the proposed threshold levels in this case.
- If $\lambda_j = 0$ for all j then $\alpha_j = \sigma^2 C_{n,\epsilon}$. In this case, (18) can be written as

$$\mathbb{P}\left[R(\overline{\boldsymbol{v}}|\boldsymbol{\xi}_n) > \sigma^2(2+\epsilon)\frac{K_n^*}{n}\log n\right] \to 0 \tag{20}$$

  as $n \to \infty$.
- If $K_n^* = n$ then (20) is not informative bound on the generalization error unfortunately. However, if K_n^* is small relative to n then (20) gives us some non-trivial implications. This is the case that there is a sparse representation of a target function in an orthogonal domain. The important point is that, in this case, we can expect $K_n \ge \rho n$ and the proposed threshold levels in (15) can be critical levels. (20) gives us non-trivial results in this case as follows. If $K_n^* = o(n/\log n)$, then by (20), the generalization error goes to zero as $n \to \infty$. If K_n^* is a constant, then the generalization error is less than $O(\log n/n)$. As seen in the evaluation of $P_{1,2}$, we

need $O(\log n/n)$ for uniformly bounding the fluctuations of orthogonal coefficients for contributed components. (20) is tight when $K_n^* = O(n)$ while the generalization error cannot be shown to go to zero in this case. On the other hand, (20) is not tight under a sparseness condition while it guarantees the convergence of generalization error. Roughly speaking, if K_n^* is constant then we may just need to uniformly bounding a sequence of random variables with a constant length. From the viewpoint of the aysmptotic theory, $\log n$ appeared in (20) may be replaced with $\log\log n$ for this case while the rigorous proof is somewhat complicated and is included in our future works.

- In applications, it is natural to consider the case where $K_n^* = n$ but many v_j^*'s are small. This depends on the shape of a true function. As well known, under a limited number of training data, small values of v_j^* should be ignored because the corresponding components are used to fit noise, which causes over-fitting. Unfortunately, for this case, (20) cannot give us valid implications when using the proposed threshold levels. In other words, the magnitude of v_j^* is not reflected in (20). However, let us recall (16) and (17). Those tell us that we cannot distinguish a contributed component from noise components if the magnitude of coefficients of the contributed components is less than the proposed threshold levels. This is true regardless of the shape of a true function. We should note that a certain theoretical plausibility of our threshold levels in practical applications comes from an evaluation of noise levels.

4. Implementation

In this section, we describe an implementation of the above training scheme.

4.1 Estimation of Noise Variance

We employ $\boldsymbol{\theta}_{n,0}$ as component-wise threshold levels since δ can be set to an arbitrarily small value in (16) and (17). In practical applications of threshold levels, we need an estimate of noise variance σ^2. Fortunately, in nonparametric regression methods, [1] suggested

$$\widehat{\sigma}^2 = \frac{\boldsymbol{y}'(I-H)^2\boldsymbol{y}}{\mathrm{trace}[(I-H)^2]}, \tag{21}$$

where

$$H = A(\Gamma+\Lambda)^{-1}A', \tag{22}$$

by which an estimated machine output vector is given by $H\boldsymbol{y}$; that is, it is written as a linear estimator based on $\boldsymbol{y}$.

4.2 Orthogonalization Procedure

In this paper, we consider two different procedures for determining Q by which $A'A$ becomes a diagonal matrix, or equivalently, $\{\boldsymbol{a}_1,\ldots,\boldsymbol{a}_n\}$ becomes a set of orthogonal vectors.

The first procedure is based on eigen-decomposition of $G'G$. We choose Q in which the k-th column vector is the k-th eigenvector of $G'G$. Q is then an orthonormal matrix; that is, $Q^{-1} = Q'$. By this choice of Q, the column vectors of $A = GQ$, in which $\boldsymbol{a}_1,\ldots,\boldsymbol{a}_n$, are orthogonal . Actually, $A'A = (GQ)'GQ = Q'(G'G)Q = \Gamma$, where γ_k is the eigenvalue corresponding to $\boldsymbol{a}_k$. Since G is nondegenerate, we have $\gamma_k > 0$ for any k. Without loss of generality, we assume that $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_n > 0$.

The second procedure is based on Gram–Schmidt orthogonalization of $G = (\boldsymbol{g}_1,\ldots,\boldsymbol{g}_n)$. In the Gram–Schmidt procedure, $\boldsymbol{g}_k$ is orthogonalized on the basis of $\boldsymbol{g}_1,\ldots,\boldsymbol{g}_{k-1}$. We assume that orthogonal vectors $\boldsymbol{q}_1,\ldots,\boldsymbol{q}_{k-1}$ are already obtained by a Gram–Schmidt procedure for $\boldsymbol{g}_1,\ldots,\boldsymbol{g}_{k-1}$. At the k-th step, we define $p_{j,k} = \langle \boldsymbol{q}_j,\boldsymbol{g}_k\rangle/\|\boldsymbol{q}_j\|^2$, $j = 1,\ldots,k-1$, where $\langle\cdot,\cdot\rangle$ denotes the Euclidean inner product. We then obtain $\boldsymbol{q}_k = \boldsymbol{g}_k - \sum_{j=1}^{k-1} p_{j,k}\boldsymbol{q}_j$, which is orthogonal to $\boldsymbol{q}_1,\ldots,\boldsymbol{q}_{k-1}$. This procedure is successively applied by starting from $\boldsymbol{q}_1 = \boldsymbol{g}_1$. We define $P_1 = I_n$. Let P_k, $k = 2,\ldots,n$ be a $n\times n$ matrix whose (j,k)-th element is $p_{j,k}$ if $j = 1,\ldots,k-1$, is 1 if $j = k$, and is 0 otherwise. We define $Q_k = \prod_{j=1}^{k} P_j$. We then have $GQ_k = (\boldsymbol{q}_1,\ldots,\boldsymbol{q}_k,\boldsymbol{g}_{k+1},\ldots,\boldsymbol{g}_n)$; that is, multiplication of Q_k brings us to the k-th step of the Gram–Schmidt procedure. We choose Q_n as Q. Then $A = GQ = (\boldsymbol{q}_1,\ldots,\boldsymbol{q}_n)$, by which $\boldsymbol{a}_k = \boldsymbol{q}_k$ and $\gamma_k = \|\boldsymbol{q}_k\|^2$. In implementing the Gram–Schmidt procedure, we consider the choice of a vector of basis function outputs to be orthogonalized at each step. At the k-th step, we calculate $p_{j,i}$, $\boldsymbol{q}_i$, and γ_i for all i in $\{k,\ldots,n\}$. We then find $c_k = \arg\max_{k\leq i\leq n}\gamma_i$. We set $p_{j,k} = p_{j,c_k}$ and $\boldsymbol{q}_k = \boldsymbol{q}_{c_k}$. We then swap $\boldsymbol{g}_{c_k}$ with $\boldsymbol{g}_k$ and repeat this procedure. This process chooses the k-th orthogonal component that maximizes its norm, or equivalently, is possibly unrelated to previously chosen components. This process may stabilize an orthogonalization procedure. In doing this, we need to store an index set $\{c_1,\ldots,c_n\}$ for calculating $\overline{\boldsymbol{w}}$.

LROLS [2] employs a modified Gram–Schmidt procedure in which the vector of basis function outputs to be orthogonalized is selected according to residual error reduction in a greedy manner. In this case, our threshold levels cannot be applied because the procedure corresponds to select basis functions as to well fit training data. It is a setup similar to that employed in optimizing a multilayer perceptron or choosing hyperparameters in kernel methods in a data-dependent manner (see, e.g., [7]). In this meaning, it may also be difficult for LROLS to confirm the effectiveness of an update procedure for the regularization parameters on the basis of Bayesian log evidence, although [2] reports good performance by LROLS in numerical experiments. In LROLS, after the update, sparseness is maintained by removing coefficients of small magnitude, although there is no systematic method available to set appropriate cut-off values. This problem also emerges in RVM.

4.3 Numerical Stability

Although the previous theoretical analysis covers a regularization method, we consider the following alternative training procedure. Since in our method, over-fitting can be avoided by thresholding in an orthogonal domain, we are left with only the stability problem in training. For this, we choose orthogonal components for which γ_k's are larger than a positive constant η. This is because estimation of the coefficients of orthogonal components with small values for γ_k's are no longer accurate because of numerical errors. Their contributions are suppressed when we introduce a regularization method. So they are not useful in explaining training data. A set of orthogonal components that satisfy this condition is referred to as a stable set. In this procedure, regularization parameters are set to zeros; that is, $\lambda_j = 0$ for all j. If we set a large value for η then the number of components in the stable set are not sufficient to represent a target function. On the other hand, over-fitting may occur if we set a small value for η while a stability of training procedure is kept in this case. Our thresholding method avoid this over-fitting by removing noise components in the stable set. In other words, η is set to a small value for just avoiding instability in the training process, and the orthogonal components in a stable set are targets of thresholding. Note that η is not a hyperparameter which affects the generalization performance and, thus, there is no need to optimize η. However, we note also that the number of candidates in the stable set, that is the size of the stable set can vary with the hyperparameter values.

In addition, restriction of the set of orthogonal components by η reduces the number of candidates that must be compared with the threshold levels. When applying the Gram–Schmidt procedure, especially, the number of steps is equal to the size of a stable set. This may considerably reduce computational cost. Note that eigen-decomposition is efficiently implemented in many software packages, including Matlab and R, and requires only a few CPU cycles.

4.4 Summary of Procedure for Hard Thresholding

By incorporating the above implementation issues, we propose the procedure summarized below.

1. We set η to a small positive value and set $\lambda_j = 0$ for all j, as discussed in 4.3. We apply eigen-decomposition or the Gram–Schmidt procedure in 4.2 for orthogonalization of $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n\}$ and obtain $\{(\boldsymbol{a}_j, \gamma_j, \widehat{v}_j) : j = 1, \ldots, l \le n\}$, where $\widehat{v}_j$ is obtained by (6) with $\lambda_j = 0$ and $l = l(\eta)$ is the number of orthogonal components in a stable set in 4.3. We also obtain R_l, which is the first $n \times l$ submatrix of Q when using eigen-decomposition and the first $l \times l$ submatrix of Q_l when using the Gram–Schmidt procedure.
2. We define $\widehat{\theta}_j = \widehat{\sigma}_j^2 C_{n,0}$, where $C_{n,0} = 2 \log n$ and $\widehat{\sigma}_j^2 = \widehat{\sigma}^2/\gamma_j$. $\widehat{\sigma}^2$ is obtained by (21), where $A = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_l)$, $\Gamma = \mathrm{diag}(\gamma_1, \ldots, \gamma_l)$, and Λ is an $l \times l$ zero matrix. We then obtain $\overline{\boldsymbol{v}}_l = (T_{\widehat{\theta}_1}(\widehat{v}_1), \ldots, T_{\widehat{\theta}_l}(\widehat{v}_l))'$, where $T_{\widehat{\theta}_j}$ is defined by (7). If $m < l$ and $T_{\widehat{\theta}_j}(\widehat{v}_j) = 0$ for any $m < j \le l$, then we set $\overline{\boldsymbol{v}}_m = (T_{\widehat{\theta}_1}(\widehat{v}_1), \ldots, T_{\widehat{\theta}_m}(\widehat{v}_m))'$. In this case, we replace l with m in constructing R_l in 1.
3. The final weight vector is obtained by $\overline{\boldsymbol{w}} = R_m \overline{\boldsymbol{v}}_m$. $\overline{\boldsymbol{w}}$ is an n-dimensional vector when using eigen-decomposition and is an m-dimensional vector when using the Gram–Schmidt procedure. As a result, the output of $f_{\overline{\boldsymbol{w}}}$ for $\boldsymbol{x} \in \mathbb{R}^d$ is $f_{\overline{\boldsymbol{w}}}(\boldsymbol{x}) = \sum_{j=1}^{n} \overline{w}_j g_j(\boldsymbol{x})$ when using eigen-decomposition and $f_{\overline{\boldsymbol{w}}}(\boldsymbol{x}) = \sum_{j=1}^{m} \overline{w}_j g_{c_j}(\boldsymbol{x})$ when using the Gram–Schmidt procedure.

We refer to the methods with eigen-decomposition and the Gram–Schmidt procedure as OHTED and OHTGS respectively. The two methods yield different types of estimates for a weight vector. It is easy to see that the least squares estimate of a weight vector is $\widehat{\boldsymbol{w}} = (G'G)^{-1}G'\boldsymbol{y}$. The least squares estimate of an orthogonal coefficient vector is $\widehat{\boldsymbol{v}} = \Gamma^{-1}A'\boldsymbol{y}$ by (5). We obtain $\widehat{\boldsymbol{w}} = Q\widehat{\boldsymbol{v}}$ due to (3). We then have $\|\widehat{\boldsymbol{w}}\|^2 = \widehat{\boldsymbol{v}}'Q'Q\widehat{\boldsymbol{v}} = \|\widehat{\boldsymbol{v}}\|^2$ since Q is orthonormal. It is also easy to see that $R_m'R_m = I_m$ in OHTED. We thus have $\|\overline{\boldsymbol{w}}\|^2 = \overline{\boldsymbol{v}}_m'R_m'R_m\overline{\boldsymbol{v}}_l = \|\overline{\boldsymbol{v}}_m\|^2$. By the definition of $T_{\widehat{\theta}_m}$, we have $\|\overline{\boldsymbol{v}}_m\|^2 \le \|\widehat{\boldsymbol{v}}\|^2$. This implies that $\|\overline{\boldsymbol{w}}\|^2 \le \|\widehat{\boldsymbol{w}}\|^2$ while $\overline{\boldsymbol{w}}$ has the same dimensions as $\widehat{\boldsymbol{w}}$. Thus, $\overline{\boldsymbol{w}}$ is a shrinkage estimator of a weight vector. On the other hand, in OHTGS, we have $\{g_{c_1}, \ldots, g_{c_m}\}$ as a set of unremoved basis functions. Since $m \le n$, OHTGS works as a thresholding method also on the set of original basis functions. In practical situations, $l \ll n$ holds (see, e.g., [2]). In addition, as can be seen in the numerical experiments described below, $m < l$ holds. Since some of $\overline{v}_k$, $k \in \{c_1, \ldots, c_m\}$ are possibly set to zero by thresholding, $\overline{\boldsymbol{w}}$ can also be a shrinkage estimate of a weight vector associated with $\{g_{c_1}, \ldots, g_{c_m}\}$. Note that the performances of the proposed methods including the generalization capabilities and/or the numbers of unremoved orthogonal basis vectors are affected by the choice of basis functions. If an assumed basis function has hyperparameters then its performance depends on the choice of hyperparameters. Therefore, in practical applications, we need to choose appropriate hyperparameters by using such as cross-validation method.

4.5 Modifications to OHTED and OHTGS

As previously mentioned, orthogonal components with large γ_j capture the smooth part of training data and dominate machine outputs. Accidental removal of such orthogonal components by, for example, hard thresholding, will yield a large bias. Actually, in wavelet denoising, only fast/detail components are the targets of the thresholding method and slow/approximation components are considered harmless [4]. In our methods, such accidental removal occurs when there are some zeros in $\overline{\boldsymbol{v}}_m = (\overline{v}_1, \ldots, \overline{v}_m)'$. If $\widehat{v}_j \le \widehat{\theta}_j$ and it is a coefficient of a contributed component, then it yields a bias that causes a large increase in the gen-

eralization error. It is usually of $O(1)$. If we set $\overline{v}_j = \widehat{v}_j$, $j \in \{1, \ldots, m\}$ even when it is a coefficient of a noise component, then the generalization error of the j-th component is $\gamma_i \widehat{v}_j^2$ in (A·6). By (16), this is bounded above by $2\sigma^2 \log n/\gamma_j$ with high probability, which is very small for a component with large γ_j. If there is a sparse representation and K^* is constant, then it is of $O(\log n/n)$ by (20). It is thus safer to avoid accidental removal and retain such components. We thus consider the use of $\overline{\boldsymbol{v}}_m = (\widehat{v}_1, \ldots, \widehat{v}_m)'$; that is, we retain all m orthogonal components. In other words, this procedure is a stopping point search, which is m here, in increasing order of magnitudes of γ_j. OHTED and OHTGS with this modification are referred to as OHTED2 and OHTGS2 respectively. We note that OHTGS2 works as a thresholding method also on the original basis function but does not work as a shrinkage method since we do not remove any of the $\boldsymbol{a}_k$, $k \in \{c_1, \ldots, c_m\}$.

5. Numerical Experiments

In this section, we compare the generalization performance of the proposed methods to the alternatives through numerical experiments on real benchmark datasets from [12]. As an alternative, we consider the regularization method with the squared norm of weights as a regularizer, where the regularization parameter is selected by leave-one-out cross-validation (LOOCV). We refer this method just by regularization method (RM) below. We also test LROLS here while we do not apply a sparsity control with heuristics. In addition, we discuss the performance of OHTGS/OHTGS2 in terms of sparseness.

We employ a Gaussian basis function for all methods. A hyperparameter is a width parameter that is common across all basis functions. Each dataset is divided into a training set and test set. A machine is trained for a training set by using each method and tested on a test set. The test error is defined as the mean squared error on the test set. We repeat this procedure for five different randomly chosen pairs of sets and calculate the average test error. The names of the datasets are shown in Table 1, together with the number of training data, test data, and inputs. For each dataset, values for all variables are normalized to zero mean and unit variance. The value of a hyperparameter is determined by two-fold cross-validation. The candidate set of values for the hyperparameter is $T = \{1, 2, 5, 8\}$ and is common for all datasets.

Table 1 Names of datasets, number of training data (n_{train}), test data (n_{test}), and inputs (d) of each dataset.

names of datasets	n_{train}	n_{test}	d
Ailerons	4000	2000	39
Delta_ailerons	4000	2000	5
Elevators	4000	2000	17
Delta_elevators	4000	2000	6
Kin8nm	4000	2000	8
Puma8NH	3000	1499	8
Puma32H	3000	1499	32
Auto	300	92	7

In RM, candidate values for a regularization parameter are $\{k \times 10^j;\ k = 1, 5,\ j = -4, -3, \ldots, 2\}$. In LROLS, we set $\lambda_{\text{init}} = 10^{-3}$ as the common initial value of regularization parameters, $\eta_L = 10^{-6}$ as the lower bound for the squared norms of orthogonalized vectors in a modified Gram–Schmidt procedure, $K = 20$ as the number of updates of the regularization parameters on the basis of Bayesian log evidence (see, e.g., [2], [11] for the details of the parameters). Here, η_L plays the same role as η in our methods. In LROLS, we do not apply heuristics when choosing the number of basis functions. For our methods, we set $\eta = 10^{-10}$. In all datasets, the number of unremoved components, m, is smaller than that of candidates in a stable set determined by this value of η.

Averaged test errors for 50 repetitions are summarized in Table 2, in which test error is divided by the variance of the test data for normalization. In this table, we also show twice the standard deviation for 50 repetitions. In Table 3, we show the numbers of unremoved orthogonal basis vectors in OHTGS and OHTGS2 together with twice the standard deviations. Both of OHTGS and OHTGS2 yield the same number of unremoved original basis functions, which is consistent with the value for OHTGS2 in Table 3; see Sect. 4.4. As explained in 4.5, the number of unremoved orthogonal components in OHTGS is smaller than that in OHTGS2 if the selected hyperparameter value is identical. We can see this in Table 3 while we should note that the selected hyperparameter value for OHTGS can be different from that for OHTGS2 in each trial. In Table 3, by comparing the number of data in Table 1, we can see that, for all data sets, OHTGS and OHTGS2 obtained sparse representations in the orthogonal domain and also in the original domain.

In Table 2, we can see that OHTED2 outperforms the other methods in the "ailerons" datasets, and is comparable to RM in the other datasets; that is, the average per-

Table 2 Averaged test errors (mean ± twice the standard deviation).

	RM	OHTED	OHTED2	OHTGS	OHTGS2	LROLS
ailerons	0.422±0.031	0.178±0.011	0.178±0.012	0.526±0.053	0.525±0.054	0.500±0.039
delta_ailerons	0.305±0.023	0.315±0.030	0.307±0.023	0.312±0.022	0.306±0.024	0.306±0.023
elevators	0.116±0.016	0.106±0.015	0.100±0.014	0.115±0.016	0.109±0.015	0.106±0.015
delta_elevators	0.376±0.019	0.378±0.018	0.379±0.018	0.379±0.019	0.381±0.022	0.376±0.019
kin8nm	0.117±0.011	0.158±0.025	0.123±0.023	0.171±0.020	0.135±0.020	0.124±0.017
puma8NH	0.342±0.026	0.373±0.029	0.350±0.056	0.414±0.030	0.360±0.027	0.352±0.029
puma32H	0.798±0.037	0.819±0.038	0.801±0.039	0.820±0.037	0.801±0.039	0.800±0.039
auto	0.129±0.046	0.144±0.047	0.141±0.059	0.145±0.057	0.136±0.049	0.146±0.058

Table 3 Averaged number of unremoved basis functions (mean ± twice the standard deviation).

	OHTGS	OHTGS2
ailerons	6.70 ± 2.92	10.54 ± 2.59
delta_ailerons	17.64 ± 7.04	53.20 ± 11.76
elevators	40.88 ± 7.22	101.42 ± 5.38
delta_elevators	7.36 ± 3.79	42.00 ± 160.84
kin8nm	208.84 ± 17.90	612.28 ± 17.23
puma8NH	64.26 ± 7.93	254.82 ± 128.47
puma32H	16.92 ± 4.06	31.96 ± 3.59
auto	10.30 ± 4.38	24.52 ± 34.58

formance depends on the dataset, although the differences are almost within twice the standard deviations. The performance of OHTED and OHTGS tends to be worse than those of the other methods. This may be due to an estimation bias caused by accidental removing contributing components. We can therefore say that the modification in OHTED2 and OHTGS2 is effective in improving the generalization performance in practical situations.

The performance of RM and OHTED2 seems to be superior to that of OHTGS2 in terms of the average test error. This fact may tell us that sparseness on basis functions may not necessarily imply better generalization capability. The other reason for this fact may be due to the orthogonalization procedure in OHTGS/OHTGS2. In OHTED/OHTED2, the components in a stable set represent the smooth part of data. This is intuitively understood since the eigendecomposition procedure is essentially the same as the principal component analysis of vectors of basis function outputs and the stable set is constructed by the components with relatively large eigen values. Therefore, OHTED/OHTED2 may be possible to reduce a bias effectively. On the other hand, if we pick up the components so as to reduce residuals in the Gram-Schmidt procedure as in LROLS then it could effectively represent a target function by a few components. However, it would be difficult to derive theoretical threshold levels for this procedure. In OHTGS/OHTGS2, we pick up components successively so as to cover the input space by using information only about input data. Therefore, OHTGS/OHTGS2 do not see output data in the orthogonalization procedure. This may cause a bias in training since some contributed basis functions may not be in a stable set. However, Table 2 shows that such a impact is small in OHTGS2. Although constructing the stable set without using output data may be a disadvantage of our methods, we could construct theoretically reasonable threshold levels because of this restriction. However, there may be effective methods for constructing a stable set in OHTGS/OHTGS2 under this restriction.

The relatively good generalization performance of LROLS may stem from the update of regularization parameters since the number of basis functions may be larger than the number of effective basis functions; that is, the weights are small for ineffective basis functions. Note again that in LROLS, and also in RVM, there is no systematic choice for the number of updates and threshold levels for removing weights. On the other hand, although RM exhibits good generalization performance in this set of experiments, it requires a proper choice of candidates and number of folds in cross-validation. The important point is that OHTED2 and OHTGS2 do not present such a model selection problem.

6. Conclusions and Future Work

For a nonparametric regression problem, we proposed a training scheme based on orthogonalization and thresholding with theoretically reasonable threshold levels. For the training scheme, we also obtained an upper bound for the generalization error in a simple case. As an implication of the bound, we found that the increase in the generalization error is of $O(\log n/n)$ if there is a sparse representation of the target function in an orthogonal domain. Analyses for more general cases are left to future work. We also described a practical implementation of the training scheme and proposed the training methods OHTED/OHTED2 and OHTGS/OHTGS2, which are based on eigen-decomposition and the Gram–Schmidt orthogonalization procedure, respectively. OHTED2/OHTGS2 are modifications of OHTED/OHTGS with reduced incidence of accidental estimation bias. In numerical experiments on real benchmark datasets, we compared the performance of our methods to those of RM and LROLS. We found that the generalization capabilities of OHTED2 and OHTGS2 are comparable to those of RM and LROLS. In addition, OHTGS2 could obtain a sparse representation. The salient point is that our methods are automatic, including model selection, which is an advantage of using orthogonalization. Moreover, our methods are easily implemented, except eigen-decomposition, which is nevertheless included in most software.

In applications, to reduce a bias effectively for OHTGS and OHTGS2, we need to improve a choice of basis functions in each step of the orthogonalization procedure. On the other hand, from a theoretical viewpoint, it is important to consider a target function for which $K_n = n$ but many v_j^* is small, which is mentioned in the remark of Sect. 3.2. Unfortunately, the generalization error bound given in this paper does not reflect the magnitude of v_j^*. Furthermore, the derived bound is not tight in the sparse case as noted in Sect. 3.2. To derive more precise and tight bound of the generalization error is a part of our future works. Also, the rigorous analysis when the inputs are stochastic is also left as a future work.

Acknowledgements

The author would like to thank anonymous reviewers for their helpful comments. This work was supported by Grant-in-Aid for Scientific Research (C), No.21500215 from Japan Society for the Promotion of Science.

References

[1] C.K. Carter and G.K. Eagleson, "A comparison of variance esti-

mators in nonparametric regression," J.R. Statist. Soc., B, vol.54, pp.773–780, 1992.
[2] S. Chen, "Local regularization assisted orthogonal least squares regression," Neurocomputing, vol.69, pp.559–585, 2006.
[3] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, 2000.
[4] D.L. Donoho and I.M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," Biometrika, vol.81, pp.425–455, 1994.
[5] K. Hagiwara, "Model selection with componentwise shrinkage in orthogonal regression," IEICE Trans. Fundamentals, vol.E86-A, no.7, pp.1749–1758, July 2003.
[6] K. Hagiwara, "A consistent model selection for orthogonal regression under component-wise shrinkage," J. Statistical Planning and Inference, vol.136, pp.1181–1195, 2006.
[7] K. Hagiwara and K. Fukumizu, "Relation between weight size and degree of over-fitting in neural network regression," Neural Netw., vol.21, pp.48–58, 2008.
[8] M.R. Leadbetter, G. Lindgren, and H. Rootźen, Extremes, and Related Properties of Random Sequences and Processes, Springer-Verlag, 1983.
[9] R. Rifkin, Everything Old is New Again: A Fresh Look at Historical Approaches in Machine Learning, Ph.D. thesis, MIT, 2002.
[10] J.A.K. Suykens, J.D. Brabanter, L. Lukas, and J. Vandewalle, "Weighted least squares support vector machines: Robustness and sparse approximation," Neurocomputing, vol.48, pp.85–105, 2002.
[11] M.E. Tipping, "Sparse Bayesian learning and the Relevance vector machine," J. Machine Learning Research, vol.1, pp.211–244, 2001.
[12] L. Torgo, www.liaad.up.pt/¯ltorgo/Regression/DataSets.html, Regression datasets, 2005.

Appendix A: Properties of Gaussian and χ^2 Random Variables

Let $Y_1, \ldots, Y_m$ be i.i.d. random variables from $N(0,1)$, that is, a Gaussian distribution whose mean is 0 and variance 1. $Y_1^2, \ldots, Y_m^2$ are then i.i.d. random variables from χ_1^2, where χ_1^2 denotes a χ^2 distribution with one degree of freedom. We define $\phi(x) = e^{-x^2/2}/\sqrt{2\pi}$ and $\psi_1(t) = (2^{1/2}\Gamma(1/2))^{-1}t^{-1/2}\exp(-t/2)$, which are probability density functions of $N(0,1)$ and χ_1^2 respectively. We define $C_{n,\delta} = (2+\delta)\log n$, where δ is a constant and $n \geq m$. We have

$$\mathbb{P}[Y_1 > x] \sim \phi(x)/x \tag{A·1}$$

for large x([5], [6], [8]), where the notation of $p(x) \sim q(x)$ implies $p(x)/q(x) \to 1$ as $x \to \infty$. By replacing x with $\sqrt{C_{n,\delta}}$ in (A·1), we have

$$\begin{aligned}\mathbb{P}[\max_{1\leq i\leq m} Y_i > \sqrt{C_{n,\delta}}] &\leq \sum_{i=1}^{m}\mathbb{P}[Y_i > \sqrt{C_{n,\delta}}] \\ &\sim \frac{mc_1}{n^{1+\delta/2}\sqrt{\log n}} \leq \frac{c_1}{n^{\delta/2}\sqrt{\log n}}\end{aligned} \tag{A·2}$$

for sufficiently large n, where c_1 is a constant. We thus have $\mathbb{P}[\max_{1\leq i\leq m} Y_i > \sqrt{C_{n,\delta}}] \to 0$ as $n \to \infty$ if $\delta \geq 0$.

We have

$$\mathbb{P}[Y_1^2 > x] \sim 2\psi_1(x) \tag{A·3}$$

for large x([5], [6], [8]). By replacing x with $C_{n,\delta}$ in (A·3), we have

$$\begin{aligned}\mathbb{P}[\max_{1\leq i\leq m} Y_i^2 > C_{n,\delta}] &\leq \sum_{i=1}^{m}\mathbb{P}[Y_i^2 > C_{n,\delta}] \\ &\sim \frac{mc_2}{n^{1+\delta/2}\sqrt{\log n}} \\ &\leq \frac{c_2}{n^{\delta/2}\sqrt{\log n}}\end{aligned} \tag{A·4}$$

for large n, where c_2 is a constant. We thus have $\mathbb{P}[\max_{1\leq i\leq m} Y_i^2 > C_{n,\delta}] \to 0$ as $n \to \infty$ if $\delta \geq 0$.

On the other hand, we assume that $m \geq \rho n$ where $\rho \in (0,1]$. Then, for a sufficiently large n, we have

$$\begin{aligned}\mathbb{P}[\max_{1\leq i\leq m} Y_i^2 \leq C_{n,\delta}] &= \left(\mathbb{P}[Y_1^2 \leq C_{n,\delta}]\right)^m \\ &= \left(1-\mathbb{P}[Y_1^2 > C_{n,\delta}]\right)^m \\ &\sim \left(1-\frac{c_3}{n^{1+\delta/2}\sqrt{\log n}}\right)^m \\ &\leq \left(1-\frac{\rho c_3}{n^{\delta/2}\sqrt{\log n}}\frac{1}{m}\right)^m \\ &\sim \exp\left(-\rho c_3/(n^{\delta/2}\sqrt{\log n})\right),\end{aligned} \tag{A·5}$$

where c_3 is a constant. In this case, thus, $\mathbb{P}[\max_{1\leq i\leq m} Y_i^2 \leq C_{n,\delta}] \to 0$ as $n \to \infty$ if $\delta < 0$.

Appendix B: Proof of (18)

By (8), we have

$$\begin{aligned}nR(\overline{\boldsymbol{v}}|\boldsymbol{\xi}_n) = \|A\boldsymbol{v}^* - A\overline{\boldsymbol{v}}\|^2 &= (\overline{\boldsymbol{v}}-\boldsymbol{v}^*)'A'A(\overline{\boldsymbol{v}}-\boldsymbol{v}^*) \\ &= \sum_{i=1}^{n}\gamma_i(\overline{v}_i - v_i^*)^2,\end{aligned} \tag{A·6}$$

since $\boldsymbol{h} = A\boldsymbol{v}^*$, $G\overline{\boldsymbol{w}} = A\overline{\boldsymbol{v}}$ and $A'A = \Gamma$. Then we have

$$\begin{aligned}&\mathbb{P}\left[nR(\overline{\boldsymbol{v}}|\boldsymbol{\xi}_n) > \sum_{i=1}^{n}\alpha_i\right] \\ \leq&\mathbb{P}\left[\bigcup_{i=1}^{n}\left\{\gamma_i(\overline{v}_i - v_i^*)^2 > \alpha_i\right\}\right] \\ \leq&\mathbb{P}\left[\bigcup_{i\in V^*}\left\{\gamma_i(\overline{v}_i - v_i^*)^2 > \alpha_i\right\}\right] + \mathbb{P}\left[\bigcup_{i\in V}\left\{\gamma_i\overline{v}_i^2 > 0\right\}\right].\end{aligned} \tag{A·7}$$

We denote the first and second terms in (A·7) as P_1 and P_2 respectively. We note that P_1 relates to contributed components and P_2 relates to noise components. We define $Z_i = (\widehat{v_i} - \mu_i v_i^*)/\sigma_i$. We first evaluate P_2. By (13) and the definition of V, $Z_i = \widehat{v_i}/\sigma_i$ for $i \in V$ and $\{Z_i : i \in V\}$ are i.i.d. samples from $N(0,1)$. By the definition of $\theta_{i,n,\epsilon}$, we have

$$P_2 = \mathbb{P}\left[\bigcup_{i\in V}\left\{\overline{v}_i^2 > 0\right\}\right]$$

$$\leq \mathbb{P}\left[\bigcup_{i\in V}\left\{\widehat{v}_i^2 > \theta_{i,n,\epsilon}\right\}\right]$$
$$= \mathbb{P}\left[\bigcup_{i\in V}\left\{\widehat{v}_i^2/\sigma_i^2 > C_{n,\epsilon}\right\}\right]$$
$$= \mathbb{P}\left[\max_{i\in V} Z_i^2 > C_{n,\epsilon}\right]. \quad (A\cdot 8)$$

Since $K_n \leq n$, it is easy to see (A· 8) is bounded above by $O((\log n)^{-1/2}n^{-\epsilon/2})$ by (A· 4) in Appendix A. This goes to zero as $n \to \infty$ since $\epsilon > 0$.

We evaluate P_1: We define events $E_{1,i} = \{\gamma_i(\overline{v}_i - v_i^*)^2 > \alpha_i\}$ and $E_{2,i} = \{\widehat{v}_i^2 \leq \theta_{i,\epsilon}\}$. P_1 is bounded by

$$P_1 = \mathbb{P}\left[\bigcup_{i\in V^*}\left(E_{1,i}\cap(E_{2,i}\cup E_{2,i}^C)\right)\right]$$
$$\leq \mathbb{P}\left[\bigcup_{i\in V^*}(E_{1,i}\cap E_{2,i}))\right] + \mathbb{P}\left[\bigcup_{i\in V^*}\left(E_{1,i}\cap E_{2,i}^C\right)\right], \quad (A\cdot 9)$$

where $E_{2,i}^C$ denotes the complement of $E_{2,i}$. We denote the first and second terms of (A· 9) as $P_{1,1}$ and $P_{1,2}$ respectively. By the definition of α_i, we obtain

$$P_{1,2} \leq \mathbb{P}\left[\bigcup_{i\in V^*}\left(E_{1,i}\bigcap\{\overline{v}_i^2 = \widehat{v}_i^2\}\right)\right]$$
$$\leq \mathbb{P}\left[\bigcup_{i\in V^*}\left\{\gamma_i(\widehat{v}_i - v_i^*)^2 > \alpha_i\right\}\right]$$
$$= \mathbb{P}\left[\bigcup_{i\in V^*}\left\{(\widehat{v}_i - v_i^*)^2/\sigma_i^2 > C_{n,\epsilon}\right\}\right]$$
$$= \mathbb{P}\left[\bigcup_{i\in V^*}\left\{|\widehat{v}_i - v_i^*|/\sigma_i > \sqrt{C_{n,\epsilon}}\right\}\right]$$
$$= \mathbb{P}\left[\bigcup_{i\in V^*}\left\{|\widehat{v}_i - \mu_i v_i^* + \mu_i v_i^* - v_i^*|/\sigma_i > \sqrt{C_{n,\epsilon}}\right\}\right]$$
$$\leq \mathbb{P}\left[\bigcup_{i\in V^*}\left\{|Z_i| > \sqrt{C_{n,\epsilon}} - D_n\right\}\right], \quad (A\cdot 10)$$

where $D_n = |v_i^*||\mu_i - 1|/\sigma_i$. By the definition of μ_i and σ_i^2, we have $D_n = |v_i^*|\lambda_i/(\sigma\sqrt{\gamma_i})$. This is bounded above by $O(1)$ if (A2) holds. By the definition of $C_{n,\epsilon}$, we thus have $\sqrt{C_{n,\epsilon/2}} < \sqrt{C_{n,\epsilon}} - D_n$ for large n. (A· 10) is bounded above by $\mathbb{P}\left[\bigcup_{i\in V^*}\left\{Z_i^2 > C_{n,\epsilon/2}\right\}\right]$. By (A· 4) in Appendix A, this is bounded above by $O((\log n)^{-1/2}n^{-\epsilon/4})$ since $Z_1,\ldots,Z_n$ are i.i.d. samples from $N(0,1)$. Therefore, $P_{1,2}$ goes to zero as $n \to \infty$ since $\epsilon \geq 0$.

If we assume that $v_i^* > 0$, we obtain, by (15),

$$E_{2,i} = \left\{|\widehat{v}_i| \leq \sigma_i\sqrt{C_{n,\epsilon}}\right\}$$
$$= \left\{|Z_i + \mu_i v_i^*/\sigma_i| \leq \sqrt{C_{n,\epsilon}}\right\}$$
$$= \left\{-\sqrt{C_{n,\epsilon}} \leq Z_i + \mu_i v_i^*/\sigma_i\right\}$$
$$\bigcap\left\{Z_i + \mu_i v_i^*/\sigma_i \leq \sqrt{C_{n,\epsilon}}\right\}$$
$$\subseteq \left\{Z_i + \mu_i v_i^*/\sigma_i \leq \sqrt{C_{n,\epsilon}}\right\}$$
$$= \left\{Z_i \leq \sqrt{C_{n,\epsilon}} - |v_i^*|\sqrt{\gamma_i}/\sigma\right\}$$
$$\subseteq \left\{Z_i \leq -\sqrt{C_{n,\epsilon}}\right\}, \quad (A\cdot 11)$$

where we use the definition of Z_i in the second line, $\mu_i/\sigma_i = \sqrt{\gamma_i}/\sigma$ in the fifth line, and (A1) in the last line. Similarly, we also have $E_{2,i} \subseteq \{Z_i > \sqrt{C_{n,\epsilon}}\}$ if $v_i^* < 0$. Since $Z_i \sim N(0,1)$, we have $\mathbb{P}[E_{2,i}] \leq \mathbb{P}\left[Z_i > \sqrt{C_{n,\epsilon}}\right]$ for any $v_i^* \neq 0$. We then have

$$P_{1,1} \leq \mathbb{P}\left[\bigcup_{i\in V^*} E_{2,i}\right]$$
$$\leq \mathbb{P}\left[\bigcup_{i\in V^*}\{Z_i > \sqrt{C_{n,\epsilon}}\}\right]$$
$$= \mathbb{P}\left[\max_{i\in V^*} Z_i > \sqrt{C_{n,\epsilon}}\right]. \quad (A\cdot 12)$$

Since $K_n^* \leq n$, this is bounded by $O((\log n)^{-1/2}n^{-\epsilon/2})$ by (A· 2) in Appendix A. This goes to zero as $n \to \infty$ since $\epsilon > 0$.

Katsuyuki Hagiwara received the Ph.D. degree from Toyohashi University of Technology in 1995. The same year, he joined Mie University as a Research Associate in the Faculty of Engineering and is now an Associate Professor in the Faculty of Education. His research interests include machine learning, statistical model selection, and statistical signal processing.