Communications for Statistical Applications and Methods
2021, Vol. 28, No. 3, 281–294

https://doi.org/10.29220/CSAM.2021.28.3.281
Print ISSN 2287-7843 / Online ISSN 2383-4757

Stable activation-based regression with localizing property

Jae-Kyung Shin[a], Jae-Hwan Jhong[b], Ja-Yong Koo[1,a]

[a]Department of Statistics, Korea University, Korea;
[b]Department of Information Statistics, ChungBuk National University, Korea

Abstract

In this paper, we propose an adaptive regression method based on the single-layer neural network structure. We adopt a symmetric activation function as units of the structure. The activation function has a flexibility of its form with a parametrization and has a localizing property that is useful to improve the quality of estimation. In order to provide a spatially adaptive estimator, we regularize coefficients of the activation functions via ℓ_1-penalization, through which the activation functions to be regarded as unnecessary are removed. In implementation, an efficient coordinate descent algorithm is applied for the proposed estimator. To obtain the stable results of estimation, we present an initialization scheme suited for our structure. Model selection procedure based on the Akaike information criterion is described. The simulation results show that the proposed estimator performs favorably in relation to existing methods and recovers the local structure of the underlying function based on the sample.

Keywords: nonparametric regression, penalized least squares, coordinate descent algorithm, adaptive estimation, symmetric activation function

1. Introduction

A common problem in nonparametric regression is to estimate the unknown regression function based on the sample. Various methods have been suggested in this area including smoothing spline, regression splines, local polynomial estimators, projection estimators, and so on. One may refer to, for example, Tsybakov (2008), and Wasserman (2006) for an overview of nonparametric regression.

Traditional non-adaptive estimators often suffer from the lack of flexibility when the underlying regression function poses complicated nonlinear local trend. A variety of spatially adaptive estimators has been proposed to resolve this issue. Donoho and Johnstone (1995) considered a wavelet shrinkage method thresholding the empirical wavelet coefficients. Luo and Wahba (1997) introduced a hybrid smoothing procedure combining adaptive regression spline and traditional smoothing spline. The variable bandwidth selector for kernel estimation was proposed for adapting to inhomogeneous smoothness by Lepski *et al.* (1997). A free-knot spline method using two stocastic search algorithm are considered by Spiriti *et al.* (2013). More recently, Jhong *et al.* (2017) proposed penalized B-spline estimator using total variation penalty.

The research of Ja-Yong Koo was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2018R1D1A1B07049972).
The research of Jae-Hwan Jhong was supported by the NRF (NRF-2020R1G1A1A01100869).

[1] Corresponding author: Department of Statistics, Korea University,145 Anam-ro, Seongbuk-gu, Seoul 02841, Korea.
E-mail: jykoo@korea.ac.kr

Published 31 May 2021 / journal homepage: http://csam.or.kr

For the last decade, the artificial neural network has flourished in the machine learning area. The flexibility provided in the layer architectures has proven to be successful in estimating highly nonlinear structure from data, see Bani-Hani and Ghaboussi (1998) and Abdollahi *et al.* (2006). Most existing studies on the neural network model have been directed towards classification problems. We expect that the flexibility of the neural network offers a promising possibility in adaptive estimation of regression functions.

To obtain the adaptive property, a choice of an activation function in the neural network structure may be an useful tool. Popular activation functions include the sigmoidal function and the rectified linear unit (Nair and Hinton, 2010). Variants of activation functions are proposed for an improvement of learning abilities in neural network. Agostinelli *et al.* (2014) introduced the adaptive piecewise linear activation unit for improving upon deep neural network architectures. Zhao and Griffin (2016) considered the mirrored rectified linear unit for building robust convolutional neural network for classification problem, combining the standard convolutional neural network and symmetric activation functions. Although these research on incorporating an activation function into a neural network structure have been growing, most of them have been limited in solving classification problems.

In this paper, we develop an adaptive regression estimator based on the single-layer neural network structure. We adopt a symmetric activation function as units of our model. It has a localizing property and a small support, leading to improvement of estimation of its coefficients. The ℓ_1-penalty is imposed to induce sparsity in the nodes of the network, which results in spatial adaptation of the estimator. An initialization suitable for a type of activation functions is necessary to enhance the learning ability of the structure, refer to Glorot and Bengio (2010). An initialization scheme is proposed to improve the stability of the layer based learning. An efficient coordinate descent algorithm is devised to implement the proposed method. Through the pruning procedure of the algorithm, the number of activation functions is automatically determined. The results show the proposed estimator outperforms existing spline-based adaptive methods.

The rest of this paper is organized as follows. In Section 2, we define the estimator using a symmetric activation and ℓ_1-penalization and explore the characteristics of the activation function. The implementation of our estimator is described in Section 3, followed by numerical study via simulations and motorcycle data analysis in Section 4. The conclusion of the parer is summarized in Section 5.

2. Model and estimator

2.1. Activation-based penalized regression estimator

Consider the given data $\{(x_i, y_i)\}_{i=1}^N$ from the regression model

$$y_i = f(x_i) + \varepsilon_i, \quad \text{for } i = 1, \ldots, N, \tag{2.1}$$

where $x_i \in \mathbb{R}$ are predictors, $y_i \in \mathbb{R}$ are responses, and ε_i are independent errors with mean $\mathbb{E}(\varepsilon_i) = 0$. Our goal is to estimate the regression function f with the model given as

$$\mathsf{f}(x;\theta) = \beta_0(\alpha_{00} + \alpha_{01}x) + \sum_{m=1}^{M} \beta_m \sigma(\alpha_{m0} + \alpha_{m1}x),$$

where $\theta = (\beta, \alpha_0, \alpha_1) = (\beta_0, \ldots, \beta_M, \alpha_{00}, \ldots, \alpha_{M0}, \alpha_{01}, \ldots, \alpha_{M1}) \in \mathbb{R}^J$ with $J = 3(M+1)$ being dimensions of the model and σ is an activation function given by

$$\sigma(z) = \begin{cases} 1+z, & \text{for } -1 \le z < 0, \\ 1-z, & \text{for } 0 \le z \le 1, \\ 0, & \text{for otherwise,} \end{cases}$$

which is also mentioned by Zhao and Griffin (2016). We call it B-spline type activation function through this paper. The linear term and the activation terms, called nodes, are included to capture the global and the local trend of data, respectively.

We use the empirical risk in terms of $\theta \in \mathbb{R}^J$ defined as

$$\ell(\theta) = \frac{1}{2N} \sum_{i=1}^{N} (y_i - \mathsf{f}(x_i; \theta))^2 .$$

For our estimator to attain the adaptive property, we penalize the coefficients of activation functions via ℓ_1-norm. It regularizes the smoothness of the estimator, controlling the number of activation functions. The penalized objective function to be minimized is

$$\ell^\lambda(\theta) = \ell(\theta) + \lambda \mathsf{p}(\theta),$$

where $\lambda > 0$ is the complexity parameter and $\mathsf{p}(\theta) = |\beta|_1$ where $|\cdot|_1$ denotes the ℓ_1-norm. The role of the complexity parameter λ is to shrink the coefficients β_m's toward zero when λ becomes large. If the coefficient β_m is equal to zero, the corresponding activation function is ruled out since it is regarded to be no longer active. In addition, our estimator is zero function when all of the coefficients are shrunk toward zero with sufficiently large λ. In Subsection 3.3, we present the approximate maximum value of λ, with which all of terms are fully removed except for the linear term.

Let

$$\hat{\theta}^\lambda = \underset{\theta \in \mathbb{R}^J}{\operatorname{argmin}}\, \ell^\lambda(\theta).$$

The activation-based penalized regression estimator (APR) is given as

$$\hat{f}^\lambda = \mathsf{f}\left(\cdot\,; \hat{\theta}^\lambda\right).$$

2.2. Motivation for the activation function

The activation function used as components of our model is parametrized as $\sigma(a + bz)$ for $z \in \mathbb{R}$ with two parameters a and b. With this parametrization, it is observed that the center of the activation function is located at $-a/b$ and the length of interval is $|2/b|$ for a and non-zero b. Simple examples of the function with some a, b are displayed in Figure 1. This activation function has the flexibility of its form depending on the values of a and b. It suggests a possibility that the proposed estimator adapts to the local structure of the unknown function by appropriately positioning the activation functions into the region where there exists the local structure.

We also note that the activation function of the form $\sigma(a + bz)$ on $z \in \mathbb{R}$ is active only on support $[(-1-a)/b, (1-a)/b]$. It makes the proposed activation function have the localizing property in the sense that its coefficient is affected only by the sample in its support, refer to Fornberg *et al.* (2006). It

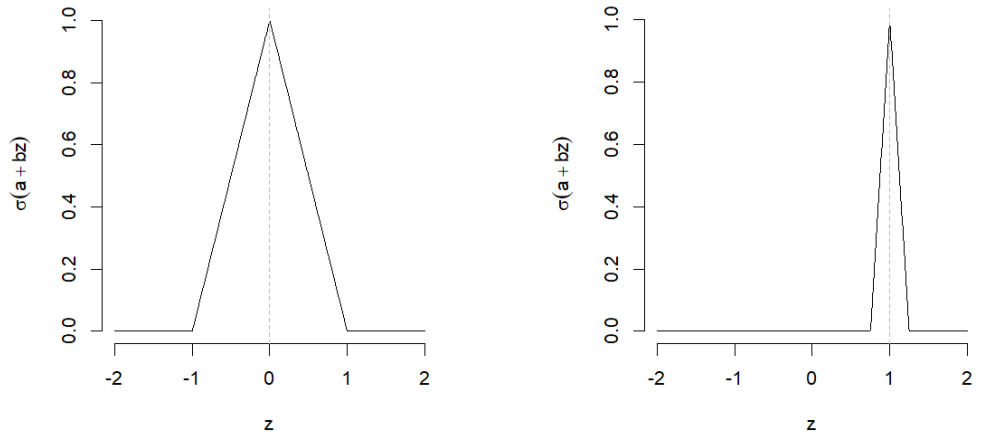

Figure 1: *Various activation functions for different values of a and b. $a = 0$, $b = 1$ (left) and $a = 4$, $b = -4$ (right). For each plot, the gray vertical dashed line represents the center of the activation function which are 0 and 1, respectively.*

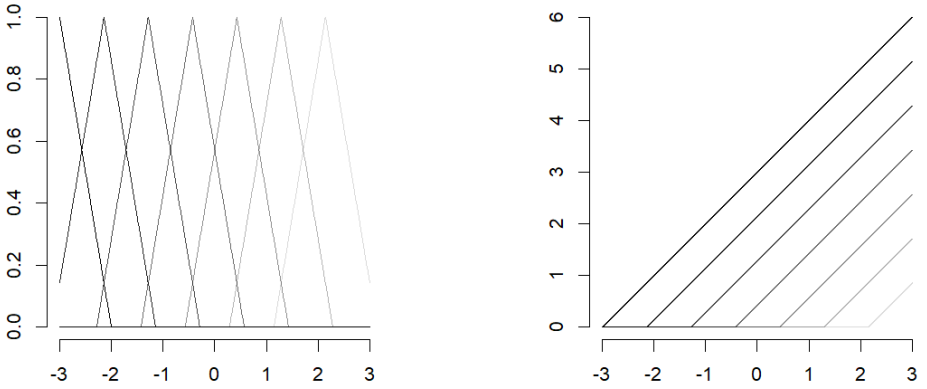

Figure 2: *B-spline type activation functions and Rectified linear unit activation functions are displayed on the left plot and right plot, respectively.*

gives the stability of estimation. Moreover, the activation function can have small support depending on the values of a and then interactive effect between the activation functions is rather small. For understanding, we present, in Figure 2, the proposed activation functions and the usual rectified linear units, which is one of the popular activation functions, to compare the interactive regions. A rectified linear unit in the right plot interacts with all of the others, while B-spline type activation function dose with some in its immediate vicinity due to its small support. This property results in numerical stability and improvement of quality of the fit. It works for spline scheme in a similar way; B-spline basis function is preferred to the truncated power basis function since B-spline basis function has smaller support compared to the truncated power basis function.

3. Implementation

In implementation, we use a coordinate descent algorithm which is comparatively easy to deal with in the sense that a univariate function problem is given. The first-order Taylor approximation is applied for changing a nonlinear problem into a linear problem. In the structure based on neural network, initial values has a critical effect on the stability and performance of estimator. We suggest a rule for generating initial values suited for B-spline type activation function. For model selection, a strategy for generating an increasing sequence of the complexity parameter λ is presented. An approximate upper bound of λ is induced, for which our estimator is a linear function or zero function. We select the optimal complexity parameter in the increasing sequence based on the Akaike information criterion.

3.1. Reformulation of the minimization problem

We apply coordinate descent algorithm to acquire our estimator, which is useful to effectively optimize a real-valued continuous function. Define univariate functions

$$\ell_j^\lambda(\theta_j) = \ell^\lambda\left(\tilde{\theta}_1, \ldots, \tilde{\theta}_{j-1}, \theta_j, \tilde{\theta}_{j+1}, \ldots, \tilde{\theta}_J\right) \quad \text{for } j = 1, \ldots, J, \tag{3.1}$$

where $\tilde{\theta}_1, \ldots, \tilde{\theta}_J$ are current values. The proposed algorithm is motivated by gradient boosting of Friedman (2001), but it is characterized for APR.

The minimization problem of (3.1) is a nonlinear problem with respect to α_{m0} and α_{m1}. Let $\tilde{\theta} = (\tilde{\theta}_1, \ldots, \tilde{\theta}_J)$ denote the current vector and $\tilde{\theta}^{-j} = \left(\tilde{\theta}_1, \ldots, \tilde{\theta}_{j-1}, \theta_j, \tilde{\theta}_{j+1}, \ldots, \tilde{\theta}_J\right)$ is a current vector replacing the jth entry by θ_j. To solve the minimization problem of (3.1), we consider the first-order Taylor approximation of a univariate function $\mathsf{f}(\cdot; \tilde{\theta}^{-j})$ at $\theta_j = \tilde{\theta}_j$ in the following way:

$$\begin{aligned}
\mathsf{f}\left(x; \tilde{\theta}^{-j}\right) &= \mathsf{f}\left(x; \tilde{\theta}\right) + \left.\frac{\partial \mathsf{f}(x;\theta)}{\partial \theta_j}\right|_{\theta=\tilde{\theta}} \left(\theta_j - \tilde{\theta}_j\right) \\
&= \left(\mathsf{f}\left(x; \tilde{\theta}\right) - \left.\frac{\partial \mathsf{f}(x;\theta)}{\partial \theta_j}\right|_{\theta=\tilde{\theta}} \tilde{\theta}_j\right) + \left.\frac{\partial \mathsf{f}(x;\theta)}{\partial \theta_j}\right|_{\theta=\tilde{\theta}} \theta_j \quad \text{for } j = 1, \ldots, J.
\end{aligned}$$

Note that there are some non-differentiable points in the B-splines type activation function. For numerical stability, we treat the differential values of these points as 1 or -1. Introducing the concept of the pseudoresponses considered by Friedman (2001), we define the empirical loss

$$\begin{aligned}
R_j(\theta_j) &= \frac{1}{2N} \sum_{i=1}^N \left(y_i - \left(\mathsf{f}(x_i; \tilde{\theta}) - \left.\frac{\partial \mathsf{f}(x_i;\theta)}{\partial \theta_j}\right|_{\theta=\tilde{\theta}} \tilde{\theta}_j\right) - \left.\frac{\partial \mathsf{f}(x_i;\theta)}{\partial \theta_j}\right|_{\theta=\tilde{\theta}} \theta_j\right)^2 \\
&= \frac{1}{2N} \sum_{i=1}^N \left(y_{ij} - z_{ij}\theta_j\right)^2,
\end{aligned}$$

where

$$z_{ij} = \left.\frac{\partial \mathsf{f}(x_i;\theta)}{\partial \theta_j}\right|_{\theta=\tilde{\theta}} \quad \text{and} \quad y_{ij} = y_i - \mathsf{f}\left(x_i; \tilde{\theta}\right) + z_{ij}\tilde{\theta}_j.$$

Then, we observe that minimizing (3.1) with respect to θ_j is *approximately* equivalant to minimizing

$$R_j^\lambda\left(\theta_j\right) = R_j\left(\theta_j\right) + \lambda \mathsf{p}\left(\tilde{\theta}_1, \ldots, \tilde{\theta}_{j-1}, \theta_j, \tilde{\theta}_{j+1}, \ldots, \tilde{\theta}_J\right). \tag{3.2}$$

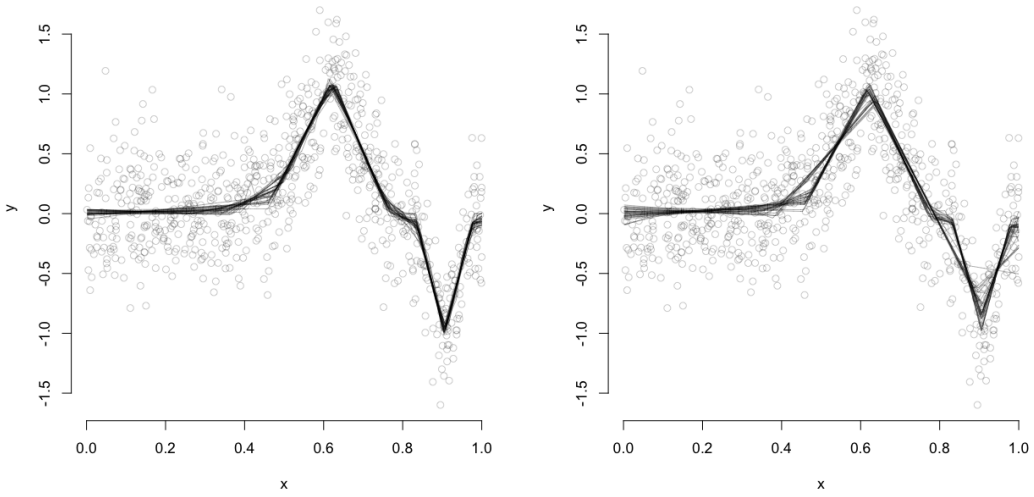

Figure 3: *The each* 30 *estimated functions with initial values using the proposed rule, and with random initialization are displayed on the left and right plot, respectively.*

We note that the solution of minimizing (3.2) is the ordinary least squares estimator with respect to α_{m0} and α_{m1} since the penalty function p only depends on β. On the other hand, the solution of β_m is the one to the lasso problem. For the lasso problem, see Tibshirani (1996). We coordinate-wisely update θ_j by

$$\tilde{\theta}_j \leftarrow \underset{\theta_j \in \mathbb{R}}{\operatorname{argmin}} R_j^1\left(\theta_j\right) \quad \text{for } j = 1, \ldots, J.$$

The algorithm iterates in the order of β_m, α_{m1}, and α_{m0} for $m = 0, \ldots, M$, namely in order of each node. The iteration terminates when the difference between the current and updated objective function values of the empirical risk is sufficiently small, that is 10^{-6}.

3.2. Initialization

Initialization procedure becomes a key ingredient for gaining the desirable results in neural network structure. Figure 3 indicates each 30 estimates obtained with initialization using a specific rule to be mentioned below on the left plot, and with random initialization on the right plot. The 30 repetitions for each case are executed from the same dataset to identify the effect of initial values on estimation. We see that the estimated functions with our strategy for generating initial values gives more stable results. To underline the effect of initial values on the stability, we report the MSE for each case, mentioned in Subsection 4.1, which also evaluate the accuracy of estimator. The values of MSE and standard error in parenthesis are given as 0.0026(0.0007) for our scheme and 0.0080(0.0056)

for random initialization, respectively. Our procedure for finding an appropriate set of initial values follows as

1. Set $\beta_m = 0$ for $m = 0, \ldots, M$ where M denotes the number of nodes.

2. α_{00} and α_{01} are the ordinary least squared estimators obtained by regressing y on x. And we set the center of initial activation functions $c_m = -\alpha_{m0}/\alpha_{m1}$ for $m = 1, \ldots, M$ as the quantiles of the given design points x_i.

3. Calculate $z_i = x_{i+1} - x_i$ for $i = 1, \ldots, N-1$ and generate random numbers w_m from normal distribution whose parameters are sample average and variance of $\{z_i\}_{i=1}^{N-1}$.

4. Calculate $\alpha_{m1} = 1/(S w_m)$ for $m = 1, \ldots, M$ where S is a scale parameter controlling the interval of the initial activation functions, and then $\alpha_{m0} = -c_m \alpha_{m1}$.

Such as scheme of spline methodologies where the interior knots are located on interval of design points, the center of activation functions is placed on the interval. As results, there is no activation functions which are inactive. Similarly, Wu (2012) adopt the way to place the centers of the radial basis functions at the design points in the beginning. we also need to determine length of the support of activation functions, adjusting the value of the scale parameter S,or the value of α_{m1}. Depending on the value of S, the amount of local information to be used for estimation is determined. If we choose the large value of S, a lot of information is used, but localizing property becomes meaningless and the interactive effect between activation functions gets large as discussed in Section 2. That is, there is a trade-off between the locality and an amount of information used. In this regard, a compromise throughout this paper is the choice of $S = 8N/M$, which means that the eight observations is averagely used for each activation function at the beginning when the sample size N is equal to the number of nodes M.

3.3. Model selection and pruning procedure

We consider an increasing sequence $\lambda_1 < \cdots < \lambda_K$ of the complexity parameter λ. The sequence $\{\lambda_1, \ldots, \lambda_K\}$ is generated as follows. We first compute an approximate upper bound λ_K, for which the proposed estimator is a linear function or zero function. Then, we set $\lambda_1 = \epsilon \lambda_K$; here, the typical choice of $\epsilon = 10^{-8}$ and $K = 100$. The increasing sequence of complexity parameter is generated in between λ_1 and λ_K on the log-scale. For the calculation of the λ_K, we refer to an upper bound for the complexity parameter to the lasso problem, shown by Osborne *et al.* (2000). If we use the way of lasso problem in calculating an upper bound of the complexity parameter for our estimator, it is given by $\max_{1 \le m \le M} |\sum_{i=1}^{N} y_i \sigma(\alpha_{m0} + \alpha_{m1} x_i)|$. Using the fact that $0 \le \sigma(z) \le 1$ for $z \in \mathbb{R}$, we obtain an upper bound of the complexity parameter as $\lambda_K = \max\{\sum_{\{i:y_i>0\}} y_i, \left|\sum_{\{i:y_i<0\}} y_i\right|\}$. Figure 4 shows an example of a solution path for a sequence $\{\lambda_1, \ldots, \lambda_{500}\}$ generated from the proposed strategy. The rainbow-colored lines are used to display the solution path from the red line corresponding to λ_1 to the purple line corresponding to λ_{500}. If λ is small, then so is the amount of penalization, resulting in a wiggly fit. As λ becomes larger, the proposed estimator gets smoother. For λ_{500}, the estimator is a simple linear function displayed by the purple line.

For selecting the optimal tuning parameter λ_{opt} in the set $\{\lambda_1, \ldots, \lambda_K\}$, we adopt the Akaike information criterion (AIC) given by

$$\text{AIC}_k = 2 \log \ell\left(\hat{\theta}^{\lambda_k}\right) + 2J_k \qquad \text{for } k = 1, \ldots, K,$$

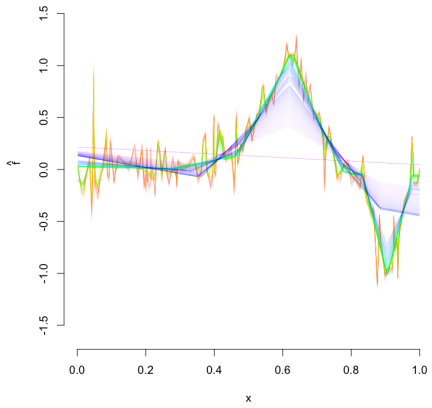

Figure 4: *A solution path of 500 estimates for an example function as the corresponding λ changes from the smallest (wiggly fit in red) to the largest (smooth fit in purple).*

where J_k and $\hat{\theta}^{\lambda_k}$ denotes the dimensions of the estimator and the estimated coefficients with λ_k, respectively. We choose the λ_l with $l = \text{argmin}_{k=1,\ldots,K} \text{AIC}_k$ as the optimal complexity parameter λ_{opt}. The optimal model can also be selected based on the Bayesian information criterion, which leads to a sparser fit than AIC.

In algorithm, we execute a pruning step through which the m^{-th}activation function to be regarded as unnecessary is pruned when the coefficient of activation function β_m becomes zero. As the complexity parameter λ in the set $\{\lambda_1, \ldots, \lambda_K\}$ gets large, the number of activation functions decreases. For $\lambda = \lambda_K$, we just find a linear function or zero function; refer to Figure 4.

4. Numerical study

4.1. Simulation

We consider three example functions to identify the performance of the proposed estimator via simulations with 100 repetitions. Response variables are generated through the model (2.1) where design points $\{x_i\}_{i=1}^N$ are generated from uniform distribution $[0, 1]$ and $\{\varepsilon_i\}_{i=1}^N$ are i.i.d. $N(0, \sigma^2)$. The example functions and variances of error used for simulation are given as

- $f(x) = \sin^3\left(2\pi x^3\right)$, with $\sigma^2 = 0.1$.
- $f(x) = (4x - 2) + 2\exp\{-256(x - 0.5)^2\}$, with $\sigma^2 = 0.1$.
- $f(x) = x(1 - x)\sin\left(\frac{2\pi(1+2^{-3/5})}{x+2^{-7/5}}\right)$, with $\sigma^2 = 0.15^2$.

Example functions and dataset are displayed in Figure 5. In the first example function, it is noticeable that there is a subtle change at around 0.8 and it seems to not easy to detect the inhomogeneous pattern around the region using a given sample. The second example function has a bump on $[0.4, 0.6]$. An accuracy of detecting the starting, the peak ,and the end point of the bump may affect the performance of estimators. The third example function has the inhomogeneous wiggly form. For each example function, we consider the cases of sample size $N = 200, 400$, and 800.

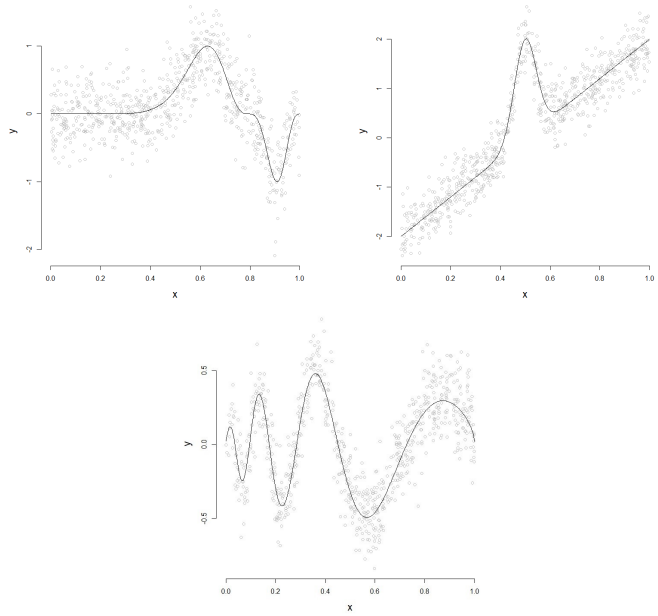

Figure 5: *The example functions and the corresponding data points with $N = 800$ are displayed.*

We compare the performance of the proposed estimator with those of penalized regression B-spline estimator (PBSE) by Jhong *et al.* (2017), and free knot least-squares splines using the genetic algorithm (freelsgen) and free knot penealized spline using the genetic algorithm (freepsgen) by Spiriti *et al.* (2013). Both competitors are based on B-spline and are spatially adaptive estimators. PBSE use all of design points as initial knots and delete unnecessary knots via pruning procedure. freelsgen and freepsgen select the location of knots, with the number of knots fixed, via a continuous generic algorithm and determine the number of knots based on the adjusted generalized cross-validation. In all simulations of this paper, we commonly use spline order $r = 2$ with the competitors, that is, linear splines. To reduce the computational burden in implementing free-knot estimators, we explore the number of knots between 2 and 12 for the first two example functions and 5 and 20 for the third example function. Free-knot splines are available in the `R` package `freeknotsplines`. For the proposed estimator, we set the number of initial nodes $M = N/4$ for all cases.

To identify the discrepancy between a function g and the underlying function f, we adopt the mean squared error (MSE), the mean absolute error (MAE), and the maximum deviation (MXDV) as the performance measures which are given by

$$\mathrm{MSE}(g) = \frac{1}{1000}\sum_{i=1}^{1000}(g(z_i) - f(z_i))^2\,,$$

$$\mathrm{MAE}(g) = \frac{1}{1000}\sum_{i=1}^{1000}|g(z_i) - f(z_i)|,$$

and

$$\mathrm{MXDV}(g) = \max_{1\le i\le 1000}|g(z_i) - f(z_i)|,$$

Table 1: The simulation results of the first example function

		MSE	MAE	MXDV
$N = 200$	APR	0.0088(0.0041)	0.0673(0.0175)	0.2990(0.1196)
	PBSE	0.0113(0.0041)	0.0784(0.0141)	0.3143(0.0806)
	freelsgen	0.0106(0.0040)	0.0746(0.0140)	0.3677(0.1668)
	freepsgen	0.0102(0.0037)	0.0734(0.0140)	0.3526(0.1526)
$N = 400$	APR	0.0047(0.0023)	0.0498(0.0123)	0.2193(0.0825)
	PBSE	0.0068(0.0027)	0.0605(0.0121)	0.2471(0.0646)
	freelsgen	0.0063(0.0022)	0.0568(0.0108)	0.3123(0.1233)
	freepsgen	0.0060(0.0020)	0.0560(0.0110)	0.2824(0.1082)
$N = 800$	APR	0.0026(0.0009)	0.0372(0.0068)	0.1678(0.0654)
	PBSE	0.0040(0.0018)	0.0456(0.0099)	0.2021(0.0431)
	freelsgen	0.0038(0.0016)	0.0431(0.0083)	0.2780(0.1399)
	freepsgen	0.0033(0.0014)	0.0416(0.0080)	0.2374(0.1065)

The standard error multiplied by 10 is reported in parenthesis.

Table 2: The simulation results of the second example function

		MSE	MAE	MXDV
$N = 200$	APR	0.0058(0.0037)	0.0505(0.0167)	0.2704(0.1108)
	PBSE	0.0103(0.0045)	0.0695(0.0133)	0.3301(0.0926)
	freelsgen	0.0074(0.0040)	0.0579(0.0154)	0.3321(0.1274)
	freepsgen	0.0072(0.0039)	0.0575(0.0151)	0.3233(0.1215)
$N = 400$	APR	0.0033(0.0016)	0.0395(0.0110)	0.2067(0.0571)
	PBSE	0.0058(0.0030)	0.0519(0.0119)	0.2441(0.0759)
	freelsgen	0.0043(0.0020)	0.0448(0.0110)	0.2881(0.1714)
	freepsgen	0.0041(0.0018)	0.0441(0.0106)	0.2562(0.0782)
$N = 800$	APR	0.0018(0.0009)	0.0298(0.0086)	0.1591(0.0591)
	PBSE	0.0029(0.0013)	0.0376(0.0080)	0.1858(0.0486)
	freelsgen	0.0027(0.0011)	0.0352(0.0077)	0.2573(0.1605)
	freepsgen	0.0026(0.0010)	0.0349(0.0075)	0.2314(0.1101)

The standard error multiplied by 10 is reported in parenthesis.

Table 3: The simulation results of the third example function

		MSE	MAE	MXDV
$N = 200$	APR	0.0032(0.0011)	0.0426(0.0073)	0.1830(0.0502)
	PBSE	0.0034(0.0015)	0.0431(0.0081)	0.1860(0.0492)
	freelsgen	0.0039(0.0014)	0.0475(0.0065)	0.2286(0.0840)
	freepsgen	0.0037(0.0013)	0.0468(0.0065)	0.2113(0.0579)
$N = 400$	APR	0.0017(0.0006)	0.0317(0.0050)	0.1369(0.0385)
	PBSE	0.0018(0.0006)	0.0321(0.0053)	0.1456(0.0331)
	freelsgen	0.0024(0.0006)	0.0378(0.0043)	0.1862(0.0803)
	freepsgen	0.0022(0.0005)	0.0371(0.0042)	0.1750(0.0613)
$N = 800$	APR	0.0011(0.0003)	0.0256(0.0037)	0.1117(0.0259)
	PBSE	0.0012(0.0004)	0.0257(0.0038)	0.1354(0.0280)
	freelsgen	0.0015(0.0003)	0.0294(0.0031)	0.1592(0.0383)
	freepsgen	0.0014(0.0003)	0.0289(0.0033)	0.1491(0.0302)

The standard error multiplied by 10 is reported in parenthesis.

where the measures are calculated at 1,000 equidistant points $z_1, \ldots, z_{1000}$ over the range of the design points. The MSE and MAE is generally used to evaluate the global performance of an estimator. The local performance over regions where underlying regression function has inhomogeneous changes is assessed by MXDV.

Tables 1–3 summarize the results of the simulations for each example function, where the standard errors multiplied by 10 are provided in parenthesis. We observe that the performance of APR is

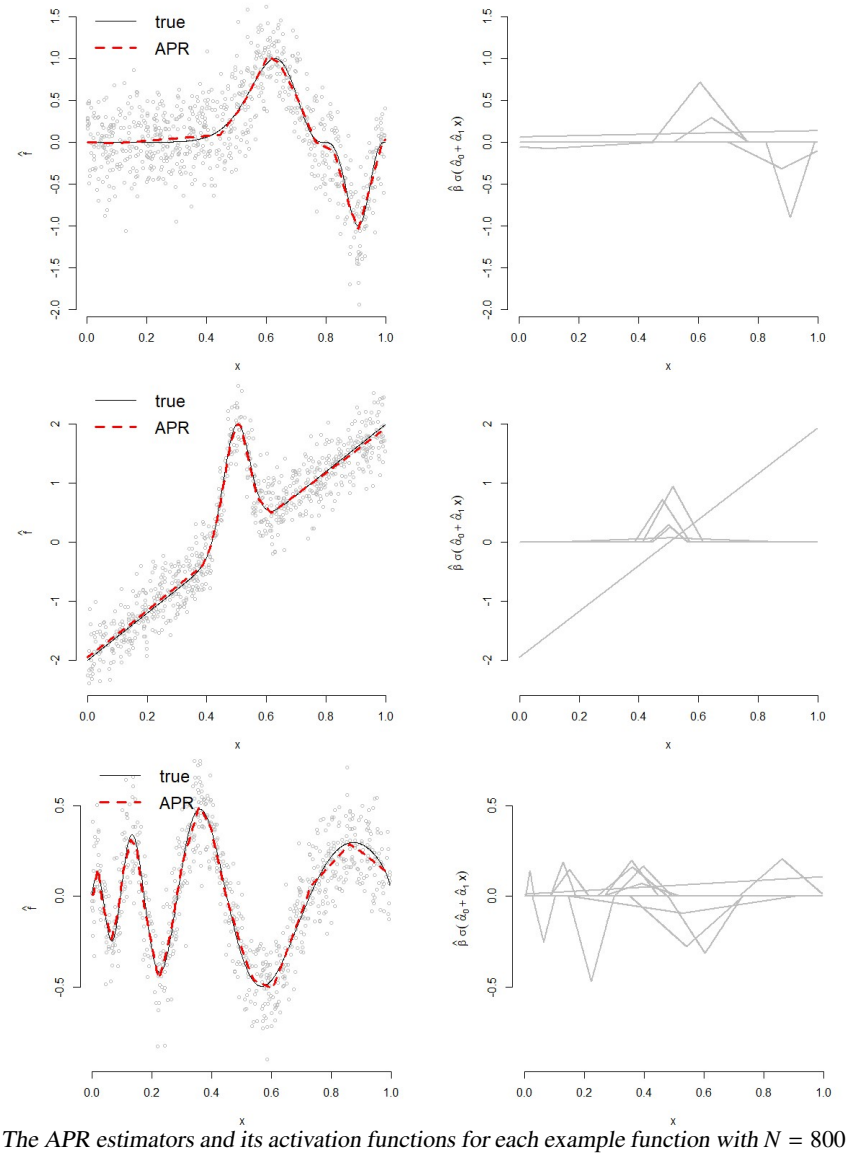

Figure 6: *The APR estimators and its activation functions for each example function with* $N = 800$ *are displayed from the top to bottom.*

competitive to those of PBSE and the free-knot estimators for the first two example functions in terms of the proposed measures. For the third example function, our estimator shows comparable results to the competitors. In Figure 6, the plot results for the each example functions are displayed from the top to the bottom, with $N = 800$ in a repetition. The left panels and the right panels represent the APR estimator and its activation functions, respectively. Figure 6 demonstrates that APR spatially adapts to the local trend, appropriately locating the activation functions. Especially, APR detects the local change at around 0.8 in the first example function. Also, the estimator recovers the bump for the

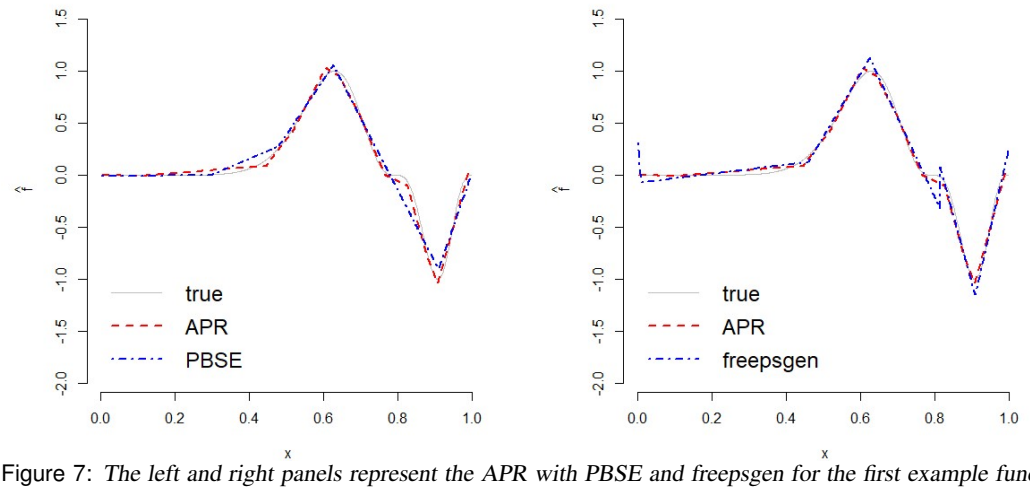

Figure 7: *The left and right panels represent the APR with PBSE and freepsgen for the first example function with $N = 800$. The APR and the competitor are drawn by the red dashed line and the blue dotdashed line on each plot. Refer to the legend on each panel.*

second example function with an approach to perfection. For the third example function, the estimated B-spline type activation functions are located, detecting the inhomogeneous trend. To compare our estimator with the competitors, we also present, in Figure 7, PBSE and freepsgen with APR for the first example function on the left panel and right panel, respectively. APR and freepsgen detect the local trend at around 0.8. But, PBSE doesn't capture it since knots in the region were removed via pruning process and freepsgen has a slight oscillation in the region. The results suggest that APR is outstanding to both estimators in that APR recovers the local structure of the underlying function based on a given sample.

4.2. Motorcycle data analysis

We apply the proposed method to motorcycle dataset (Silverman, 1985). This dataset had been recorded during an experiment to determine the efficacy of crash-helmets and consists of two variables; time measured in mileseconds after a simulated impact of a motorcycle and the acceleration readings of the driver's head taken through time. Motorcycle dataset is also used in Jhong *et al.* (2017).

We use the same set-up with the simulation for all estimators. We just round off $N/4$ for getting an integer and for freepsgen, we explore the number of knots between 2 and 15. The motorcycle dataset and APR, PBSE, and freepsgen are displayed in Figure 8 where x-axis and y-axis represent time after impact and acceleration readings through time (accel). APR seems to fit data well over the entire range. Furthermore, APR adapts quite well to the local trend. The time has no effects on the the acceleration in the range $[0, 15]$ and $[45, 60]$ and local changes is detected in $[15, 40]$.

5. Conclusion

In this paper, we have developed the penalized regression estimator based on the single-layer neural network structure combined with a symmetric activation function.

The proposed estimator attains the adaptive property, with a symmetric activation function and ℓ_1-penalization which leads to a data-driven activation function selection. Use of the proposed strategy

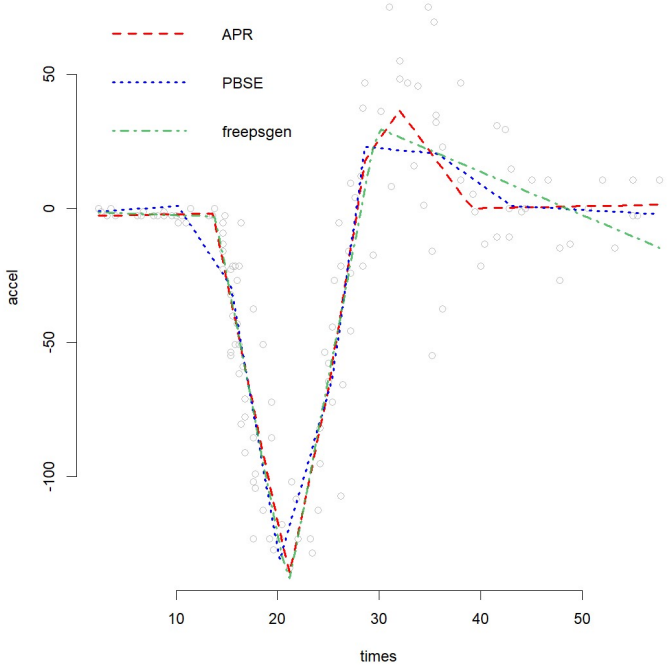

Figure 8: *The results of analysis on motorcycle data. The red dashed, blue dotted, and green dotdash lines represent APR, PBSE, and freepsgen, respectively.*

of initialization results in the stability of estimation. An approximation upper bound of complexity parameter is induced, for which the proposed estimator is a linear function or zero function. We also have devised an efficient coordinate descent algorithm for the proposed estimator. The results of based on simulations and a real data analysis demonstrate the satisfactory performance of the proposed estimator.

In the future research, a variety of extensions of the model is expected due to the flexibility of the structure. We can consider introducing various activation functions to our framework, for example, one with a higher degree. An extension to multivariate regression also suggests a promising possibility, especially with tensor product structure. A model with tensor product structure is currently under development, with taking an additional penalization for variable selection into consideration.

Acknowledgement

The research of Ja-Yong Koo was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2018R1D1A1B07049972). The research of Jae-Hwan Jhong was supported by the NRF (NRF-2020R1G1A1A01100869).

References

Abdollahi F, Talebi HA, and Patel RV (2006). Stable identification of nonlinear systems using neural networks: Theory and experiments, *IEEE/ASME Transactions On Mechatronics*, **11**, 488–495.

Agostinelli F, Hoffman M, Sadowski P, and Baldi P (2014). *Learning activation functions to improve*

deep neural networks.

Bani-Hani K and Ghaboussi J (1998). Nonlinear structural control using neural networks, *Journal of Engineering Mechanics*, **124**, 319–327.

Donoho DL and Johnstone IM (1995). Adapting to unknown smoothness via wavelet shrinkage, *Journal of the American Statistical Association*, **90**, 1200–1224.

Fornberg B, Flyer N, Hovde S, and Piret C (2006). Localization properties of rbf expansion coefficients for cardinal interpolation. i. equispaced nodes, *Advances in Computational Mathematics*, **47**, 5–20.

Friedman JH (2001). Greedy function approximation: a gradient boosting machine, *Annals of Statistics*, **29**, 1189-1232.

Glorot X and Bengio Y (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 249–256.

Jhong JH, Koo JY, and Lee SW (2017). Penalized b-spline estimator for regression functions using total variation penalty, *Journal of Statistical Planning and Inference*, **184**, 77–93.

Lepski OV, Mammen E, Spokoiny VG, *et al.* (1997). Optimal spatial adaptation to inhomogeneous smoothness: an approach based on kernel estimates with variable bandwidth selectors, *Annals of Statistics*, **25**, 929–947.

Luo Z and Wahba G (1997). Hybrid adaptive splines, *Journal of the American Statistical Association*, **92**, 107–116.

Nair V and Hinton GE (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 807–814.

Osborne MR, Presnell B, and Turlach BA (2000). On the lasso and its dual, *Journal of Computational and Graphical statistics*, **9**, 319–337.

Silverman BW (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting, *Journal of the Royal Statistical Society: Series B*, **47**, 1–21.

Spiriti S, Eubank R, Smith PW, and Young D (2013). Knot selection for least-squares and penalized splines, *Journal of Statistical Computation and Simulation*, **83**, 1020–1036.

Tibshirani R (1996). Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society: Series B*, **58**, 267–288.

Tsybakov AB (2008). *Introduction to Nonparametric Estimation* (1st ed), Springer Publishing Company, Incorporated, New York.

Wasserman L (2006). *All of Nonparametric Statistics (Springer Texts in Statistics)*, Springer-Verlag, Berlin.

Wu Y, Wang H, Zhang B, and Du KL (2012). Using radial basis function networks for function approximation and classification, *ISRN Applied Mathematics*.

Zhao Q and Griffin LD (2016). *Suppressing the unusual: towards robust cnns using symmetric activation functions*.

Received January 23, 2021; Revised February 17, 2021; Accepted February 17, 2021