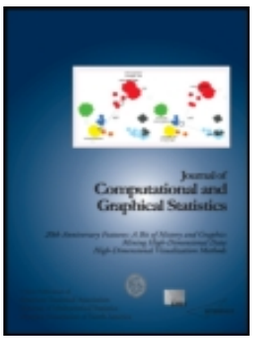

Taylor & Francis
Taylor & Francis Group

Journal of Computational and Graphical Statistics

ISSN: 1061-8600 (Print) 1537-2715 (Online) Journal homepage: http://amstat.tandfonline.com/loi/ucgs20

Fast and Stable Multiple Smoothing Parameter Selection in Smoothing Spline Analysis of Variance Models With Large Samples

Nathaniel E. Helwig & Ping Ma

To cite this article: Nathaniel E. Helwig & Ping Ma (2015) Fast and Stable Multiple Smoothing Parameter Selection in Smoothing Spline Analysis of Variance Models With Large Samples, Journal of Computational and Graphical Statistics, 24:3, 715-732, DOI: 10.1080/10618600.2014.926819

To link to this article: http://dx.doi.org/10.1080/10618600.2014.926819

Supplementary materials for this article are available online.
Please go to www.tandfonline.com/r/JCGS

Fast and Stable Multiple Smoothing Parameter Selection in Smoothing Spline Analysis of Variance Models With Large Samples

Nathaniel E. HELWIG and Ping MA

The current parameterization and algorithm used to fit a smoothing spline analysis of variance (SSANOVA) model are computationally expensive, making a generalized additive model (GAM) the preferred method for multivariate smoothing. In this article, we propose an efficient reparameterization of the smoothing parameters in SSANOVA models, and a scalable algorithm for estimating multiple smoothing parameters in SSANOVAs. To validate our approach, we present two simulation studies comparing our reparameterization and algorithm to implementations of SSANOVAs and GAMs that are currently available in R. Our simulation results demonstrate that (a) our scalable SSANOVA algorithm outperforms the currently used SSANOVA algorithm, and (b) SSANOVAs can be a fast and reliable alternative to GAMs. We also provide an example with oceanographic data that demonstrates the practical advantage of our SSANOVA framework. Supplementary materials that are available online can be used to replicate the analyses in this article.

Key Words: Algorithms; Multivariate analysis; Nonparametric methods; Smoothing and nonparametric regression.

1. INTRODUCTION

A typical nonparametric regression model can be written as $y_i = \eta(\mathbf{x}_i) + e_i$ for $i \in \{1, \ldots, n\}$ where $y_i \in \mathbb{R}$ is the response variable for the ith observation, $\mathbf{x}_i \equiv (x_{i1}, \ldots, x_{ip})'$ is the $p \times 1$ predictor vector for the ith observation, η is an unknown smooth function relating the response and predictor variables, and $e_i \sim \mathrm{N}(0, \sigma^2)$ is independent, normally distributed measurement error. Given a set of values $(\mathbf{x}_i, y_i)$ for $i \in \{1, \ldots, n\}$, a popular approach for estimating η is the minimization of a penalized least-squares

A shortened version of this article was selected as a winner of the 2013 Student Article Competition sponsored by the Statistical Computing and Statistical Graphics Sections of the American Statistical Association.

Nathaniel E. Helwig, School of Statistics, University of Minnesota, Minneapolis, MN 55455 (E-mail: *helwig@umn.edu*). Ping Ma, Department of Statistics, University of Georgia, Athens, GA 30602 (E-mail: *pingma@uga.edu*).

[1]This work was funded by NSF grants DMS-1055815, DMS-0800631, and DMS-1228288.

Journal of Computational and Graphical Statistics, Volume 24, Number 3, Pages 715–732
DOI: 10.1080/10618600.2014.926819
Color versions of one or more of the figures in the article can be found online at *www.tandfonline.com/r/jcgs*.

functional of the form

$$(1/n)\sum_{i=1}^{n}(y_i - \eta(\mathbf{x}_i))^2 + \lambda J(\eta), \tag{1}$$

where J is a quadratic penalty functional that quantifies the roughness of η, and $\lambda \in (0, \infty)$ is a global smoothing parameter that controls the trade-off between the goodness-of-fit of the data and the smoothness of η.

Two possible approaches for estimating the η_λ that minimizes Equation (1) are smoothing spline analysis of variance models (SSANOVAs; see Wahba 1990; Gu 2013b) and generalized additive models (GAMs; see Wood 2004, 2006); note that η_λ is subscripted because the optimal function depends on the chosen smoothing parameters. In general, SSANOVAs benefit from a solid theoretical foundation, but are computationally expensive to fit. In contrast, GAMs are fast to compute, but are not built upon the same solid theoretical framework as SSANOVAs. Due to their fast performance, GAMs are typically preferred over SSANOVAs in practice, despite the theoretical advantages of the SSANOVA framework.

Kim and Gu (2004) gave an SSANOVA approximation using $q \ll n$ selected knots, and their algorithm requires $O(nq^2)$ flops to estimate η_λ for each choice of smoothing parameters. In contrast, when using Wood's (2004) approach to fit a GAM with q selected knots, the algorithm requires $O(nq^2)$ flops for the initialization, followed by $O(q^3)$ flops for each choice of smoothing parameters. Furthermore, using the conventional SSANOVA parameterization, there are often more smoothing parameters than predictors, whereas a GAM uses one smoothing parameter for each predictor. As a result, fitting an SSANOVA (using Kim and Gu's 2004, algorithm) typically takes substantially longer than fitting the corresponding GAM (using Wood's 2004, algorithm), and the timing difference increases with n.

In this article, we propose an efficient reparameterization of the smoothing parameters in SSANOVA models, such that there is one smoothing parameter for each predictor. We also propose a new algorithm for fitting an SSANOVA using q selected knots. This new algorithm only requires $O(nq^2)$ flops for the initialization; then, after the initialization, the estimation of η_λ only requires $O(q^3)$ flops for each choice of smoothing parameters. In the following sections, we present our proposed SSANOVA reparameterization (Section 2), describe a new algorithm for estimating multiple smoothing parameters in SSANVOA models (Section 3), present two simulation studies comparing our approach to other approaches (Section 4), and demonstrate the benefits of our approach using an example (Section 5).

2. AN EFFICIENT REPARAMETERIZATION

2.1 CONVENTIONAL SSANOVA PARAMETERIZATION

The η_λ minimizing Equation (1) can be estimated via a tensor product reproducing kernel Hilbert space (RKHS) of functions $\mathcal{H} \equiv \otimes_{j=1}^{p}\mathcal{H}_{\mathcal{X}_j}$ on the domain $\mathcal{X} \equiv \prod_{j=1}^{p}\mathcal{X}_j$, where $\mathcal{H}_{\mathcal{X}_j}$ is a RKHS of functions on $\mathcal{X}_j$, which denotes the domain of the jth predictor. In most cases, $\mathcal{H}_{\mathcal{X}_j}$ can be decomposed into three orthogonal subspaces, such as $\mathcal{H}_{\mathcal{X}_j} = \mathcal{H}_{\bullet_j} \oplus \mathcal{H}_{\tilde{\mathrm{n}}_j} \oplus \mathcal{H}_{\mathrm{c}_j}$, where $\mathcal{H}_{\mathrm{n}_j} \equiv \mathcal{H}_{\bullet_j} \oplus \mathcal{H}_{\tilde{\mathrm{n}}_j}$ is the *null space* of the jth predictor (i.e., $\mathcal{H}_{\mathrm{n}_j} \equiv \{\phi : J(\phi) = 0\}$) with $\mathcal{H}_{\bullet_j}$ denoting a space of constant functions (i.e., $\mathcal{H}_{\bullet_j} \equiv \{\phi : \phi \propto 1\}$),

and $\mathcal{H}_{c_j}$ is the *contrast space* of the jth predictor (i.e., $\mathcal{H}_{c_j} \equiv \{\phi : 0 < J(\phi) < \infty\}$). The reproducing kernel (RK) of $\mathcal{H}$ has the form $\rho_{\mathcal{X}} = \prod_{j=1}^{p} \rho_{\mathcal{X}_j}$, where $\rho_{\mathcal{X}_j}$ is the RK of $\mathcal{H}_{\mathcal{X}_j}$, which has the form $\rho_{\mathcal{X}_j} \equiv \rho_{\bullet_j} + \rho_{\tilde{n}_j} + \rho_{c_j}$ with $\rho_{\bullet_j}$, $\rho_{\tilde{n}_j}$, and ρ_{c_j} denoting the RKs of $\mathcal{H}_{\bullet_j}$, $\mathcal{H}_{\tilde{n}_j}$, and $\mathcal{H}_{c_j}$, respectively (see Wahba 1990; Gu 2013b).

The marginal space decompositions $\mathcal{H}_{\mathcal{X}_j} = \mathcal{H}_{\bullet_j} \oplus \mathcal{H}_{\tilde{n}_j} \oplus \mathcal{H}_{c_j}$ imply that the tensor product space $\mathcal{H}$ can be decomposed into the summation of 3^p orthogonal subspaces. Typically, the tensor product space is written as $\mathcal{H} = \mathcal{H}_n \oplus \mathcal{H}_c$, where $\mathcal{H}_n$ and $\mathcal{H}_c$ are the tensor product null and contrast spaces, respectively. Using the conventional SSANOVA parameterization, the contrast space of $\mathcal{H}$ is written as $\mathcal{H}_c = \oplus_{k=1}^{s} \mathcal{H}_k^*$, where the $\mathcal{H}_k^*$ are orthogonal subspaces with corresponding inner products $\langle \cdot, \cdot \rangle_k^*$. Then, the inner product of $\mathcal{H}_c$ is defined as $\sum_{k=1}^{s} \theta_k^{-1} \langle \cdot, \cdot \rangle_k^*$, where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_s)'$ are local smoothing parameters with $\theta_k \in (0, \infty)$. Using this definition of the inner product for $\mathcal{H}_c$, it can be shown that the RK of $\mathcal{H}_c$ is given by $\rho_c = \sum_{k=1}^{s} \theta_k \rho_k^*$, where ρ_k^* denotes the RK of $\mathcal{H}_k^*$.

For example, with $p = 2$ predictors using cubic marginals, $\mathcal{H} = \mathcal{H}_n \oplus \mathcal{H}_c$ where

$$\begin{aligned} \mathcal{H}_n &\equiv \mathcal{H}_\bullet \oplus \mathcal{H}_{\tilde{n}_1} \oplus \mathcal{H}_{\tilde{n}_2} \oplus (\mathcal{H}_{\tilde{n}_1} \otimes \mathcal{H}_{\tilde{n}_2}) \\ \mathcal{H}_c &\equiv \mathcal{H}_{c_1} \oplus \mathcal{H}_{c_2} \oplus (\mathcal{H}_{c_1} \otimes \mathcal{H}_{\tilde{n}_2}) \oplus (\mathcal{H}_{\tilde{n}_1} \otimes \mathcal{H}_{c_2}) \oplus (\mathcal{H}_{c_1} \otimes \mathcal{H}_{c_2}) \end{aligned} \tag{2}$$

are the null and contrast spaces of $\mathcal{H}$. The RK has the form $\rho_{\mathcal{X}} = \rho_n + \rho_c$, where

$$\begin{aligned} \rho_n &\equiv 1 + \rho_{\tilde{n}_1} + \rho_{\tilde{n}_2} + \rho_{\tilde{n}_1}\rho_{\tilde{n}_2} \\ \rho_c &\equiv \theta_1\rho_{c_1} + \theta_2\rho_{c_2} + \theta_3\rho_{c_1}\rho_{\tilde{n}_2} + \theta_4\rho_{\tilde{n}_1}\rho_{c_2} + \theta_5\rho_{c_1}\rho_{c_2} \end{aligned} \tag{3}$$

are the RKs of $\mathcal{H}_n$ and $\mathcal{H}_c$, respectively, and $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_5)'$ are the additional smoothing parameters. Note that there are $s = 5$ unique θ_k parameters for only $p = 2$ predictors; this allows for flexible smoothing, but creates a difficult multivariate optimization problem.

2.2 Proposed SSANOVA Reparameterization

To improve the efficiency of the SSANOVA framework, we define the marginal RKs as

$$\rho_{\mathcal{X}_j} = \rho_{n_j} + \gamma_j \rho_{c_j}, \tag{4}$$

where $\rho_{n_j} = \rho_{\bullet_j} + \rho_{\tilde{n}_j}$ is the RK of the jth predictor's null space, ρ_{c_j} is the RK of the jth predictor's contrast space, and $\gamma_j \in (0, \infty)$ is the jth predictor's smoothing parameter. Note that this definition of the marginal RKs corresponds to the marginal inner products $\langle \cdot, \cdot \rangle_{\mathcal{X}_j} = \langle \cdot, \cdot \rangle_{n_j} + \gamma_j^{-1} \langle \cdot, \cdot \rangle_{c_j}$, which implies that γ_j rescales the marginal penalty of the jth predictor. Also, note that this reparameterization imposes a structure on the smoothing parameters in the tensor product RKHS, which can result in substantial computational savings.

For example, with $p = 2$ predictors, the efficient SSANOVA reparameterization uses

$$\rho_c = \gamma_1\rho_{c_1}\rho_{n_2} + \gamma_2\rho_{n_1}\rho_{c_2} + \gamma_1\gamma_2\rho_{c_1}\rho_{c_2} \tag{5}$$

as the contrast space RK. In contrast to the conventional parameterization (see Equation (3)), the reparameterized RK in Equation (5) only uses two unique smoothing parameters. Thus, the reparameterization requires a bivariate optimization, whereas the traditional parameterization involves a five-variable optimization problem. The efficiency of the reparameterization is even more pronounced in higher dimensions: with $p = 3$ predictors, we

have a trivariate optimization problem using the reparameterization, whereas the traditional parameterization would require the optimization of a function with respect to 19 θ_k parameters.

2.3 PROPERTIES OF PROPOSED REPARAMETERIZATION

First, note that for $p = 1$ predictor, the efficient reparameterization is identical to the conventional parameterization because there is only one γ (or θ) parameter, which can be absorbed into the $\mathbf{c}$ coefficients. Second, note that for $p > 1$ predictors, the efficient reparameterization is identical to the conventional parameterization whenever the SSANOVA model contains strictly additive effects of the predictors; note that this differs from other SSANOVA reparameterizations (e.g., COSSO; Lin and Zhang 2006). So, the efficient and conventional parameterizations only differ with respect to their treatment of two-way (and higher-way) interactions between predictors. When one or more interaction effects are present, the efficient reparameterization will require fewer unique smoothing parameters than the conventional parameterization.

To better understand the constraints of the efficient reparameterization, it is helpful to consider the case of $p = 2$ predictors. Comparing the efficient reparameterization in Equation (5) to the conventional parameterization in Equation (3), note that the reparameterization assumes that $\theta_1 = \theta_3, \theta_2 = \theta_4$, and $\theta_5 = \theta_1\theta_2$ when defining the contrast space RK. Although seemingly limited, this reparameterization is actually rather flexible. For example, setting $\gamma_1, \gamma_2 \gg 1$ allows the nonparametric interaction space ($\mathcal{H}_{c_1} \oplus \mathcal{H}_{c_2}$) to dominate the contrast RK; in contrast, setting $\gamma_1, \gamma_2 \ll 1$ allows the nonparametric main effect spaces ($\mathcal{H}_{c_1}$ and $\mathcal{H}_{c_2}$) and parametric-nonparametric interaction effect spaces ($\mathcal{H}_{c_1} \oplus \mathcal{H}_{\tilde{n}_2}$ and $\mathcal{H}_{\tilde{n}_1} \oplus \mathcal{H}_{c_2}$) to dominate the contrast RK. The relative influence of each predictor on the contrast RK can be controlled by adjusting the ratio γ_1/γ_2; for example, setting $\gamma_1 > \gamma_2$ allows the first predictor more influence on the contrast RK (assuming the same spline type for each predictor).

The main limitation of the efficient reparameterization is its lack of flexibility (compared to the conventional parameterization) when the SSANOVA model is misspecified. Continuing with the case of $p = 2$ predictors, suppose that the true data-generating model is of the form $\hat{y} = \eta_1(x_1) + \eta_2(x_2)$, that is, there is no interaction effect. In theory, using the conventional SSANOVA parameterization it would be possible to obtain the correct (additive) model by estimating $\hat{\theta}_k = 0$ for $k \in \{3, 4, 5\}$. In contrast, using the efficient reparameterization the solution would contain a small bias, given that the parametric-nonparametric interaction effect spaces cannot be fully separated from the nonparametric main effect spaces. However, with large n and γ_j parameters selected using the GCV score (see Section 3), we have found that the bias introduced by the efficient reparameterization is negligible in most cases (see Section 4.2 and Helwig 2013).

3. FAST AND STABLE SSANOVA ALGORITHM

Overview. The scalable algorithm proposed in this section is unrelated to the efficient reparameterization given in the previous section. Throughout this section, we write the algorithm in terms of the conventional parameterization with θ_k parameters. However,

remembering that the efficient reparameterization imposes a certain structure on the θ_k parameters, this scalable algorithm could be easily applied to the efficient reparameterization using the γ_j parameters.

3.1 SSANOVA COMPUTATION

Given a set of smoothing parameters $\boldsymbol{\lambda} = (\lambda/\theta_1, \ldots, \lambda/\theta_k)$ and a set of selected knots $\{\check{\mathbf{x}}_t\}_{t=1}^q$, it is well-known that the η_λ minimizing Equation (1) can be approximated as $\eta_\lambda(\mathbf{x}) = \sum_{v=1}^u d_v\phi_v(\mathbf{x}) + \sum_{t=1}^q c_t\rho_c(\mathbf{x}, \check{\mathbf{x}}_t)$, where $\{\phi_v\}_{v=1}^u$ are basis functions spanning $\mathcal{H}_n$, ρ_c denotes the RK of $\mathcal{H}_c$, and $\mathbf{d} = \{d_v\}_{u\times 1}$ and $\mathbf{c} = \{c_t\}_{q\times 1}$ are the unknown function coefficient vectors (see Gu and Wahba 1991; Kim and Gu 2004). Using this representation, Equation (1) can be approximated as

$$(1/n)\|\mathbf{y} - \mathbf{K}\mathbf{d} - \mathbf{J}_{\boldsymbol{\theta}}\mathbf{c}\|^2 + \lambda\mathbf{c}'\mathbf{Q}_{\boldsymbol{\theta}}\mathbf{c}, \tag{6}$$

where $\|\cdot\|^2$ denotes the squared Frobenius norm, $\mathbf{y} \equiv \{y_i\}_{n\times 1}$, $\mathbf{K} \equiv \{\phi_v(\mathbf{x}_i)\}_{n\times u}$ for $i \in \{1, \ldots, n\}$ and $v \in \{1, \ldots, u\}$, $\mathbf{J}_{\boldsymbol{\theta}} = \sum_{k=1}^s \theta_k\mathbf{J}_k$ where $\mathbf{J}_k \equiv \{\rho_k^*(\mathbf{x}_i, \check{\mathbf{x}}_t)\}_{n\times q}$ for $i \in \{1, \ldots, n\}$ and $t \in \{1, \ldots, q\}$, and $\mathbf{Q}_{\boldsymbol{\theta}} = \sum_{k=1}^s \theta_k\mathbf{Q}_k$ where $\mathbf{Q}_k \equiv \{\rho_k^*(\check{\mathbf{x}}_t, \check{\mathbf{x}}_w)\}_{q\times q}$ for $t, w \in \{1, \ldots, q\}$.

Given a choice of $\boldsymbol{\lambda}$, the optimal function coefficients are given by

$$\begin{pmatrix}\hat{\mathbf{d}}\\ \hat{\mathbf{c}}\end{pmatrix} = \begin{pmatrix}\mathbf{K}'\mathbf{K} & \mathbf{K}'\mathbf{J}_\theta\\ \mathbf{J}_{\boldsymbol{\theta}}'\mathbf{K} & \mathbf{J}_{\boldsymbol{\theta}}'\mathbf{J}_{\boldsymbol{\theta}} + \lambda n\mathbf{Q}_{\boldsymbol{\theta}}\end{pmatrix}^\dagger \begin{pmatrix}\mathbf{K}'\\ \mathbf{J}_{\boldsymbol{\theta}}'\end{pmatrix}\mathbf{y} \tag{7}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudoinverse of the input matrix. The fitted values are given by $\hat{\mathbf{y}} = \mathbf{K}\hat{\mathbf{d}} + \mathbf{J}_{\boldsymbol{\theta}}\hat{\mathbf{c}} = \mathbf{S}_{\boldsymbol{\lambda}}\mathbf{y}$, where

$$\mathbf{S}_{\boldsymbol{\lambda}} = (\mathbf{K}\ \ \mathbf{J}_{\boldsymbol{\theta}})\begin{pmatrix}\mathbf{K}'\mathbf{K} & \mathbf{K}'\mathbf{J}_\theta\\ \mathbf{J}_{\boldsymbol{\theta}}'\mathbf{K} & \mathbf{J}_{\boldsymbol{\theta}}'\mathbf{J}_{\boldsymbol{\theta}} + \lambda n\mathbf{Q}_{\boldsymbol{\theta}}\end{pmatrix}^\dagger \begin{pmatrix}\mathbf{K}'\\ \mathbf{J}_{\boldsymbol{\theta}}'\end{pmatrix} \tag{8}$$

is the smoothing matrix, which depends on the chosen smoothing parameters in $\boldsymbol{\lambda}$. A popular criterion for estimating smoothing parameters in penalized regression models is the generalized cross-validation (GCV) score of Craven and Wahba (1979), which is given by

$$\mathrm{GCV}(\boldsymbol{\lambda}) = \left\{n\|(\mathbf{I}_n - \mathbf{S}_{\boldsymbol{\lambda}})\mathbf{y}\|^2\right\}\Big/\left\{[n - \mathrm{tr}(\mathbf{S}_{\boldsymbol{\lambda}})]^2\right\}. \tag{9}$$

The estimates $\hat{\lambda}$ and $\hat{\boldsymbol{\theta}}$ that minimize the GCV score have desirable properties (see Craven and Wahba 1979; Li 1986; Gu and Wahba 1991; Gu 2013b), so our algorithm focuses on a fast GCV score evaluation for given smoothing parameters.

Kim and Gu's (2004) algorithm seeks to find the smoothing parameters that minimize the GCV score in Equation (9). For each choice of smoothing parameters, Kim and Gu's algorithm forms the inner (cross-product) portion of $\mathbf{S}_{\boldsymbol{\lambda}}$, and uses a pivoted Cholesky decomposition to find the inverse $(\cdot)^\dagger$; given the needed inverse calculation, the fitted values can be easily obtained, so the GCV score can be easily evaluated. However, obtaining the inner (cross-product) portion of the smoothing matrix requires $O(nq^2)$ flops, which can be quite costly for large n. So, because Kim and Gu's algorithm requires iterative work that depends on the (possibly quite large) sample size n, the algorithm is not scalable for large samples.

Our key insight is the fact that the coefficient estimation (see Equation (7)) and the GCV score evaluations (see Equation (9)) only depend on various crossproduct vectors and matrices. The crossproduct vectors and matrices needed for Equation (7) are straightforward. Expanding the term in the numerator of Equation (9), we have $\|(\mathbf{I}_n - \mathbf{S}_\lambda)\mathbf{y}\|^2 = \|\mathbf{y}\|^2 - 2\mathbf{y}'\mathbf{S}_\lambda\mathbf{y} + \mathbf{y}'\mathbf{S}_\lambda^2\mathbf{y}$, which only depends on crossproduct vectors and matrices. Next, note that the trace calculation needed in the denominator of the GCV score can be written as

$$\text{tr}(\mathbf{S}_\lambda) = \text{tr}\left\{ \begin{pmatrix} \mathbf{K}'\mathbf{K} & \mathbf{K}'\mathbf{J}_\theta \\ \mathbf{J}_\theta'\mathbf{K} & \mathbf{J}_\theta'\mathbf{J}_\theta + \lambda n \mathbf{Q}_\theta \end{pmatrix}^\dagger \begin{pmatrix} \mathbf{K}'\mathbf{K} & \mathbf{K}'\mathbf{J}_\theta \\ \mathbf{J}_\theta'\mathbf{K} & \mathbf{J}_\theta'\mathbf{J}_\theta \end{pmatrix} \right\}. \tag{10}$$

So, after initializing the necessary crossproduct vectors and matrices, the SSANOVA can be fit using only $O(q^3)$ flops for each choice of smoothing parameters.

3.2 A Scalable Algorithm

First, form the $n \times sq$ matrix $\mathbf{J} \equiv (\mathbf{J}_1, \mathbf{J}_2, \ldots, \mathbf{J}_s)$ and the $q \times sq$ matrix $\mathbf{Q} \equiv (\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_s)$. Next, calculate the $u \times 1$ vector $\mathbf{y}_\text{K} \equiv \mathbf{K}'\mathbf{y}$ and the $sq \times 1$ vector $\mathbf{y}_\text{J} \equiv \mathbf{J}'\mathbf{y}$. Finally, initialize the $u \times u$ matrix $\mathbf{C}_\text{K} \equiv \mathbf{K}'\mathbf{K}$, the $u \times sq$ matrix $\mathbf{C}_\text{KJ} \equiv \mathbf{K}'\mathbf{J}$, and the $sq \times sq$ matrix $\mathbf{C}_\text{J} \equiv \mathbf{J}'\mathbf{J}$. Then, given a $\boldsymbol{\theta}$ vector, define $\tilde{\boldsymbol{\theta}} \equiv (\boldsymbol{\theta} \otimes_\text{K} \mathbf{I}_q)$ where the symbol $\otimes_\text{K}$ denotes the Kronecker product, and calculate $\mathbf{Q}_\theta = \mathbf{Q}\tilde{\boldsymbol{\theta}}$. Next, calculate the $q \times 1$ vector $\mathbf{y}_\text{J}^\theta = \tilde{\boldsymbol{\theta}}'\mathbf{y}_\text{J}$ and note that $\mathbf{y}_\text{J}^\theta \equiv \mathbf{J}_\theta'\mathbf{y}$. Then, calculate the $u \times q$ matrix $\mathbf{C}_\text{KJ}^\theta = \mathbf{C}_\text{KJ}\tilde{\boldsymbol{\theta}}$, and note that $\mathbf{C}_\text{KJ}^\theta \equiv \mathbf{K}'\mathbf{J}_\theta$. Likewise, calculate the $q \times q$ matrix $\mathbf{C}_\text{J}^\theta = \tilde{\boldsymbol{\theta}}'\mathbf{C}_\text{J}\tilde{\boldsymbol{\theta}}$, and note that $\mathbf{C}_\text{J}^\theta \equiv \mathbf{J}_\theta'\mathbf{J}_\theta$.

Next, note that $\mathbf{S}_\lambda = \mathbf{X}_\theta[\mathbf{C}_\theta + \lambda n \tilde{\mathbf{Q}}_\theta]^\dagger\mathbf{X}_\theta'$, where $\mathbf{X}_\theta \equiv (\mathbf{K}, \mathbf{J}_\theta)$ is the $n \times (u+q)$ design matrix,

$$\mathbf{C}_\theta \equiv \begin{pmatrix} \mathbf{C}_\text{K} & \mathbf{C}_\text{KJ}^\theta \\ (\mathbf{C}_\text{KJ}^\theta)' & \mathbf{C}_\text{J}^\theta \end{pmatrix} \quad \text{and} \quad \tilde{\mathbf{Q}}_\theta \equiv \begin{pmatrix} \mathbf{0}_{u\times u} & \mathbf{0}_{u\times q} \\ \mathbf{0}_{q\times u} & \mathbf{Q}_\theta \end{pmatrix}. \tag{11}$$

Now, if we let $\mathbf{ABA}'$ denote the full-rank spectral decomposition of $[\mathbf{C}_\theta + \lambda n \tilde{\mathbf{Q}}_\theta]$, we know that $\text{tr}(\mathbf{S}_\lambda) = \text{tr}(\mathbf{X}_\theta\mathbf{AB}^{-1}\mathbf{A}'\mathbf{X}_\theta') = \text{tr}(\mathbf{AB}^{-1}\mathbf{A}'\mathbf{C}_\theta)$; note that we define the full-rank spectral decomposition by calculating the full spectral decomposition, and then setting to zero the eigenvalues that are smaller than the first eigenvalue multiplied by machine epsilon. Furthermore, we have $\|(\mathbf{I}_n - \mathbf{S}_\lambda)\mathbf{y}\|^2 = \|\mathbf{y}\|^2 - 2\mathbf{y}_\theta'\mathbf{z} + \mathbf{z}'\mathbf{C}_\theta\mathbf{z}$, where $\mathbf{y}_\theta \equiv (\mathbf{y}_\text{K}; \mathbf{y}_\text{J}^\theta)$ is a $(u+q) \times 1$ vector and $\mathbf{z} \equiv \mathbf{AB}^{-1}\mathbf{A}'\mathbf{y}_\theta$. This implies that

$$\text{GCV}(\boldsymbol{\lambda}) = n\left\{\|\mathbf{y}\|^2 - 2\mathbf{y}_\theta'\mathbf{z} + \mathbf{z}'\mathbf{C}_\theta\mathbf{z}\right\} \Big/ \left\{[n - \text{tr}(\mathbf{AB}^{-1}\mathbf{A}'\mathbf{C}_\theta)]^2\right\} \tag{12}$$

can be used to evaluate the GCV score.

Below we outline the full algorithm. First, note smoothing is fully parameterized by $\lambda_k = \lambda\theta_k^{-1}$ for $k \in \{1, \ldots, s\}$. However, when estimating multiple smoothing parameters in SSANOVA models, it has proven useful to separate the overall level of smoothing (captured by λ) from the relative smoothing of each subspace of $\mathcal{H}_\text{c}$ (captured by the θ_k's), see Gu and Wahba (1991). As a result, the below algorithm iterates between estimating λ for a fixed $\boldsymbol{\theta}$, and then estimating $\boldsymbol{\theta}$ for a fixed λ. In the below algorithm, ω denotes the maximum number of iterations, which is a user-provided positive integer. In general, we have found that setting $\omega = 5$ works well (see Helwig 2013).

Fast SSANOVA Algorithm for Large Samples

I. Initializations:
1. Define $\mathbf{J} \equiv (\mathbf{J}_1, \mathbf{J}_2, \ldots, \mathbf{J}_s)$ and $\mathbf{Q} \equiv (\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_s)$
2. Define $\mathbf{y}_\mathrm{K} \equiv \mathbf{K}'\mathbf{y}$, $\mathbf{y}_\mathrm{J} \equiv \mathbf{J}'\mathbf{y}$, $\mathbf{C}_\mathrm{K} \equiv \mathbf{K}'\mathbf{K}$, $\mathbf{C}_\mathrm{KJ} \equiv \mathbf{K}'\mathbf{J}$, and $\mathbf{C}_\mathrm{J} \equiv \mathbf{J}'\mathbf{J}$
3. Initialize $\boldsymbol{\theta} = \mathbf{1}_s$, $\epsilon = 10^{-5}$, $\tau = 0$, $\omega = 5$, and $\mathrm{GCV}(\boldsymbol{\lambda}_0) = \|\mathbf{y}\|^2$

II. Iterative Procedure:
1. Update λ for fixed $\boldsymbol{\theta}$
   a. $\tilde{\boldsymbol{\theta}} = (\hat{\boldsymbol{\theta}} \otimes_\mathrm{K} \mathbf{I}_q)$, $\mathbf{Q}_{\boldsymbol{\theta}} = \mathbf{Q}\tilde{\boldsymbol{\theta}}$, $\mathbf{y}_\mathrm{J}^{\boldsymbol{\theta}} = \tilde{\boldsymbol{\theta}}'\mathbf{y}_\mathrm{J}$, $\mathbf{C}_\mathrm{KJ}^{\boldsymbol{\theta}} = \mathbf{C}_\mathrm{KJ}\tilde{\boldsymbol{\theta}}$, $\mathbf{C}_\mathrm{J}^{\boldsymbol{\theta}} = \tilde{\boldsymbol{\theta}}'\mathbf{C}_\mathrm{J}\tilde{\boldsymbol{\theta}}$
   b. Form the $\mathbf{C}_{\boldsymbol{\theta}}$ and $\tilde{\mathbf{Q}}_{\boldsymbol{\theta}}$ matrices from Equation (11)
   c. Minimize the GCV score w.r.t. λ using Equation (12)
2. Update $\boldsymbol{\theta}$ for fixed λ
   a. Given current $\hat{\lambda}$, calculate $\hat{\lambda}n$
   b. Minimize the GCV score w.r.t. $\boldsymbol{\xi} = \ln(\boldsymbol{\theta})$ using Equation (12)
3. Check for Convergence
   a. If $[\mathrm{GCV}(\boldsymbol{\lambda}_0) - \mathrm{GCV}(\hat{\boldsymbol{\lambda}})]/\mathrm{GCV}(\boldsymbol{\lambda}_0) < \epsilon$, stop (algorithm converged)
   b. Else if $\tau = \omega - 1$, stop (iteration limit reached)
   c. Else set $\mathrm{GCV}(\boldsymbol{\lambda}_0) = \mathrm{GCV}(\hat{\boldsymbol{\lambda}})$ and $\tau = \tau + 1$ and return to step 1

III. Estimate Parameters:
1. Let $\mathbf{ABA}'$ denote the full-rank spectral decomposition of $[\mathbf{C}_{\boldsymbol{\theta}} + \lambda n \tilde{\mathbf{Q}}_{\boldsymbol{\theta}}]$
   a. $(\hat{\mathbf{d}}', \hat{\mathbf{c}}') = \mathbf{y}_{\boldsymbol{\theta}}'\mathbf{AB}^{-1}\mathbf{A}'$ and $\hat{\mathbf{y}} = \mathbf{K}\hat{\mathbf{d}} + \mathbf{J}_{\boldsymbol{\theta}}\hat{\mathbf{c}}$
   b. $\hat{\sigma}^2 = \{\|\mathbf{y}\|^2 - 2(\hat{\mathbf{d}}', \hat{\mathbf{c}}')\mathbf{y}_{\boldsymbol{\theta}} + (\hat{\mathbf{d}}', \hat{\mathbf{c}}')\mathbf{C}_{\boldsymbol{\theta}}(\hat{\mathbf{d}}', \hat{\mathbf{c}}')'\}/\{n - \mathrm{tr}(\mathbf{AB}^{-1}\mathbf{A}'\mathbf{C}_{\boldsymbol{\theta}})\}$

For the minimization of the GCV score with respect to λ (in Step 1(c) of the iterative procedure), it is possible to use some Newton-type (or other optimization) method. However, given that each GCV score evaluation is rather cheap using this algorithm, we prefer a brute-force search because this provides a better chance of avoiding local optima. Consequently, throughout this article, we evaluate the GCV score for $\lambda \in \{10^{-k}\}$ for $k \in \{0, 1, \ldots, 9\}$. For the minimization of the GCV score with respect to $\boldsymbol{\xi}$ (in Step 2(b) of the iterative procedure), we follow the suggestion of Kim and Gu (2004) and use the quasi-Newton methods of Dennis and Schnabel (1996) through the `nlm.R` function. Finally, note that we parameterize the problem in terms of $\boldsymbol{\xi}$ (instead of $\boldsymbol{\theta}$) because $\boldsymbol{\xi}$ is unconstrained (see Gu and Wahba 1991).

3.3 SMART STARTING VALUES

In the algorithm, the θ_k values are initialized to one, but this is arbitrary. When fitting an SSANOVA using the standard parameterization, much better starting values for the θ_k parameters can be obtained by using Algorithm 3.2 from Gu and Wahba (1991), which will be briefly described. Suppose that the true function can be written as $\eta = \sum_{k=1}^{s} P_k\eta$, where P_k is the projection operator corresponding to kth orthogonal subspace of $\mathcal{H}_\mathrm{c}$. In this case, it would be sensible to weight each subspace according to $\|P_k\eta\|^2$, so that the roughness penalties of the different subspaces are balanced. So, if η were known, one could define $\theta_k = \|P_k\eta\|^2$ for $k \in \{1, \ldots, s\}$, and then minimize the GCV score with respect to λ.

Clearly, η will be unknown in practice, so Gu and Wahba (1991) proposed the following procedure for initializing θ_k. First, set $\theta_k = \mathrm{tr}(\mathbf{Q}_k)^{-1}$ for $k \in \{1, \ldots, s\}$. Then, given the

θ_k values, select the λ that minimizes the GCV score (or some similar criterion), and let $\hat{\mathbf{c}}$ denote the contrast space function coefficient vector that corresponds to the optimal λ. Next, define the starting smoothing parameter values as $\hat{\theta}_{k0} = \theta_k^2 \hat{\mathbf{c}}' \mathbf{Q}_k \hat{\mathbf{c}}$ for $k \in \{1, \ldots, s\}$. Note that the $\hat{\theta}_{k0}$ values are obtained using the relation $\theta_k = \| P_k \eta \|^2$ with the true projection $P_k \eta$ replaced by the estimated projection $P_k \eta_{\boldsymbol{\lambda}} \equiv \theta_k \sum_{t=1}^{q} \hat{c}_t [P_k \rho(\check{\mathbf{x}}_t, \cdot)]$.

When fitting an SSANOVA using the efficient reparameterization, Algorithm 3.2 from Gu and Wahba (1991) cannot be used to initialize the smoothing parameters (because of the assumed interdependencies between the smoothing parameters of the different subspaces). However, it is possible to use a similar logic to initialize the smoothing parameters in this reparameterized model. Assuming that $\mathbf{x}_i = (x_{i1}, x_{i2})$ with $x_{ij} \in [0, 1]$ for $i \in \{1, \ldots, n\}$ and $j \in \{1, 2\}$, the contrast space can be decomposed into $s = 3$ orthogonal subspaces, such as $\mathcal{H}_{\mathrm{c}} = \mathcal{H}_1 \oplus \mathcal{H}_2 \oplus \mathcal{H}_3$, where $\mathcal{H}_1 \equiv \mathcal{H}_{\mathrm{c}_1} \otimes \mathcal{H}_{\mathrm{n}_2}$, $\mathcal{H}_2 \equiv \mathcal{H}_{\mathrm{n}_1} \otimes \mathcal{H}_{\mathrm{c}_2}$, and $\mathcal{H}_3 \equiv \mathcal{H}_{\mathrm{c}_1} \otimes \mathcal{H}_{\mathrm{c}_2}$. In this case, the γ_1 and γ_2 parameters can be initialized using the following procedure.

First set $\check{\theta}_k = \mathrm{tr}(\mathbf{Q}_k)$, where $\mathbf{Q}_k$ is the penalty matrix corresponding to the subspace $\mathcal{H}_k$ (for $k \in \{1, 2, 3\}$), and define $\gamma_1 = \check{\theta}_2 / \check{\theta}_3$ and $\gamma_2 = \check{\theta}_1 / \check{\theta}_3$. Then, given $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)'$ with $\theta_j \equiv \gamma_j$ for $j \in \{1, 2\}$ and $\theta_3 \equiv \gamma_1 \gamma_2$, select the λ that minimizes the GCV score (or some similar criterion), and let $\hat{\mathbf{c}}$ denote the contrast space function coefficient vector that corresponds to the optimal λ. Next, define $\tilde{\theta}_k = \theta_k^2 \hat{\mathbf{c}}' \mathbf{Q}_k \hat{\mathbf{c}}$ for $k \in \{1, 2, 3\}$. Finally, define the starting smoothing parameter values as $\hat{\gamma}_1 = \tilde{\theta}_3 / \tilde{\theta}_2$ and $\hat{\gamma}_2 = \tilde{\theta}_3 / \tilde{\theta}_1$. Note that the $\hat{\gamma}_k$ values are obtained using a similar logic to that used by Algorithm 3.2 from Gu and Wahba (1991), with the additional assumption that $\theta_3 \equiv \gamma_1 \gamma_2$.

This modified starting algorithm can be extended to various other situations, for example, different types of predictors or $p > 2$ predictors. The general structure of the modified starting algorithm is (1) initialize the γ_j parameters so that the different subspaces have (approximately) equal influence, (2) select the λ that minimizes the GCV score and let $\hat{\mathbf{c}}$ denote the optimal contrast space function coefficient vector, (3) define $\tilde{\theta}_k = \theta_k^2 \hat{\mathbf{c}}' \mathbf{Q}_k \hat{\mathbf{c}}$ for $k \in \{1, \ldots, s\}$, and (4) initialize the $\hat{\gamma}_j$ values by taking the appropriate ratio of the $\tilde{\theta}_k$ values; we recommend using the $\tilde{\theta}_k$ values corresponding to the highest-order interaction in the model.

4. SIMULATION STUDIES

Overview. We conducted two simulation studies. Simulation A compares our scalable algorithm to Kim and Gu's (2004) algorithm using the conventional SSANOVA parameterization. Simulation B compares our efficient SSANOVA parameterization (in combination with the scalable algorithm) to Wood's (2004) GAM algorithm. See the supplementary online materials for R code that can be used to replicate these simulations. Note: simulations were conducted in R (ver 3.0.3) on an iMac (3.1 GHz Intel Core i5) using Apple's `vecLib` BLAS (see R documentation).

4.1 SIMULATION A

4.1.1 Simulation A: Design. Simulation A compares the scalable algorithm proposed in Section 3 to Kim and Gu's (2004) algorithm when using the conventional SSANOVA

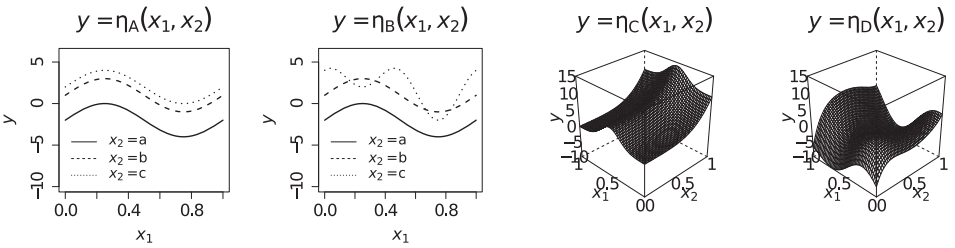

Figure 1. Four simulation functions. See supplementary R code for function specifics.

parameterization (see Section 2.1). The scalable algorithm is implemented in Helwig's (2014) `bigssp.R` function, and Kim and Gu's algorithm is implemented in Gu's (2013a) `ssanova.R` function. As a part of Simulation A, we manipulated two conditions relevant to the problem at hand: (a) the function type (4 levels: see Figure 1), and (b) the number of observations (5 levels: $n = 1000k$ for $k \in \{1, 5, 10, 25, 50\}$).

Using cubic and nominal marginal splines for x_1 and x_2 (respectively), there are $s = 4$ smoothing parameters for η_A and η_B when using the conventional SSANOVA parameterization with the two-way interaction included. In contrast, there are $s = 5$ smoothing parameters for η_C and η_D when using the conventional SSANOVA parameterization with cubic marginals and the interaction included.

4.1.2 Simulation A: Analyses. For each combination of η and n, we generated y_i by (a) independently sampling x_{i1} and x_{i2} from a uniform distribution on the appropriate range for the given η, see Figure 1, (b) independently sampling e_i from a N(0, 9) distribution, and (c) defining the observed response as $y_i = \eta(x_{i1}, x_{i2}) + e_i$ for $i \in \{1, \ldots, n\}$. Then, we fit a nonparametric regression model using four different methods: Method 1 is our scalable SSANOVA algorithm with fully optimized θ_k parameters, Method 2 is our scalable algorithm with partially optimized (i.e., fixed after smart start) θ_k parameters, Method 3 is Kim and Gu's (2004) SSANOVA algorithm with fully optimized θ_k parameters, and Method 4 is Kim and Gu's algorithm with partially optimized θ_k parameters. In both the `bigssp.R` and `ssanova.R` functions, the fully optimized smoothing parameters are obtained using the option `skip.iter=FALSE`, whereas the partially optimized smoothing parameters are obtained using the option `skip.iter=TRUE`.

For each function and method, we fit a two-way SSANOVA model with an interaction term included (syntax y ∼ x1 ∗ x2). For η_A and η_B, we (a) used cubic and nominal marginal splines for x_1 and x_2, respectively, and (b) examined the solution using $q = 30$ basis function knots. In contrast, for η_C and η_D, we (a) used cubic marginal splines for both x_1 and x_2, and (b) examined the solution using $q = 100$ basis function knots. We used a bin-sampling approach to select knots spread throughout the covariate domain, and we used the same knots for each of the four methods. For each method, we selected the smoothing parameters that minimized the GCV score. Given the optimal smoothing parameters, we calculated the fitted values, and then defined the true mean-squared-error (TMSE) as

$$\text{TMSE} = (1/n) \sum_{i=1}^{n} (\eta(\mathbf{x}_i) - \hat{y}_i)^2, \tag{13}$$

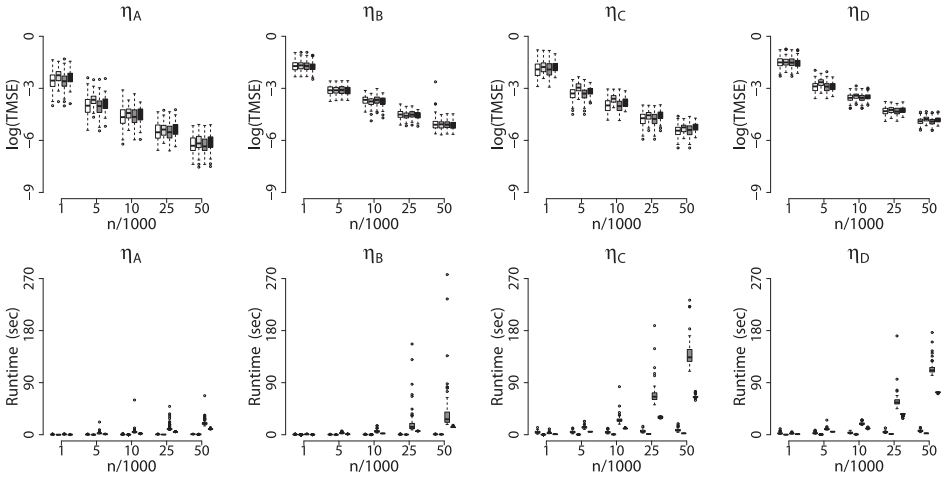

Figure 2. Simulation A log-TMSE (top) and runtime (bottom) boxplots. Within each sample size, left (white) box is Method 1 (scalable algorithm, fully optimized θ_k's), left-middle (light gray) box is Method 2 (scalable algorithm, partially optimized θ_k's), right-middle (dark gray) box is Method 3 (classic algorithm, fully optimized θ_k's), and right (black) box is Method 4 (classic algorithm, partially optimized θ_k's).

where η denotes the true function from Figure 1, $\mathbf{x}_i$ denotes the predictor variable scores for the ith observation, and $\hat{y}_i$ is the fitted value for the ith observation. Finally, we used 100 replications of the above procedure within each cell of the simulation design.

4.1.3 Simulation A: Results. The log-TMSE for each combination of Simulation A conditions is plotted in Figure 2. First, note that all of the methods performed reasonably well, and note that (as expected) the TMSE approached zero as n increased for all examined conditions. Furthermore, as expected, the TMSE values associated with the partially optimized θ_k parameters (i.e., Methods 2 and 4) tended to be larger than the TMSE values associated with the fully optimized θ_k parameters (i.e., Methods 1 and 3); however, the TMSE difference between the fully and partially optimized solutions was negligible. Comparing the TMSE values of Methods 1 and 2 to those of Methods 3 and 4, the effectiveness of our approach is readily apparent: our scalable algorithm performed nearly identically to Kim and Gu's (2004) algorithm in all of the Simulation A conditions.

The only mentionable difference between the two algorithms occurs when analyzing η_B with $n = 5 \times 10^4$ observations; in this case, Method 1 produced three poorer solutions (i.e., solutions with larger TMSE values) that did not occur when using the other methods. For these three outlying cases, Method 2 produced better solutions, suggesting that the three poorer results from Method 1 are due to a poor iterative update of the smoothing parameters. So, in some cases, it appears that our scalable algorithm may be slightly more stable when using the partially optimized θ_k parameters; however, for a vast majority of the simulation conditions, our scalable algorithm produced TMSEs comparable to those of Kim and Gu's (2004) classic algorithm.

The runtime for each combination of Simulation A conditions is also plotted in Figure 2. First, note that (as expected) the runtimes increased as n increased for all methods. Furthermore, as expected, the runtimes associated with the partially optimized θ_k parameters

(i.e., Methods 2 and 4) tended to be smaller than the runtimes associated with the fully optimized θ_k parameters (i.e., Methods 1 and 3); the runtime difference between the fully and partially optimized solutions differed depending on the algorithm and n. Comparing the runtimes of Methods 1 and 2 to those of Methods 3 and 4, it is evident that our scalable algorithm saves a substantial amount of computation time when fitting SSANOVAs to large samples; the scalable algorithm was anywhere from 10 to 200 times more efficient than Kim and Gu's (2004) algorithm, and the runtime difference between the algorithms increased substantially as n increased.

4.2 SIMULATION B

4.2.1 Simulation B: Design. Simulation B compares our efficient SSANOVA reparameterization (see Section 2.2) in combination with our scalable SSANOVA algorithm (see Section 3) to Wood's (2004) GAM algorithm. Our efficient reparameterization and algorithm are implemented in Helwig's (2014) `bigssa.R` function, and Wood's algorithm is implemented in Wood's (2014) `gam.R` function. For comparability, we use the same simulation conditions (i.e., η's and n's) that were examined in Simulation A. Note that the efficient reparameterization uses $s = 2$ unique γ_j parameters for each of the four simulation functions. For η_A and η_B, the GAM essentially estimates a different function for each level of the nominal predictor (assuming that the two-way interaction term is included), so there are $s = 3$ unique smoothing parameters; in contrast, for η_C and η_D the GAM separately penalizes the partial derivative of the estimated function with respect to each predictor, so there are $s = 2$ unique smoothing parameters regardless of whether or not the two-way interaction term is included (see Wood 2006, 2014).

4.2.2 Simulation B: Analyses. We used the same data simulation procedure that was used in Simulation A. Then, we fit a nonparametric regression model using four different methods: Method 1 is our scalable SSANOVA algorithm with fully optimized γ_j parameters, Method 2 is our scalable algorithm with partially optimized (i.e., fixed after smart start) γ_j parameters, Method 3 is Wood's (2014) `gam.R` function, and Method 4 is Gu's (2013a) `ssanova.R` function with partially optimized θ_k parameters; note that Method 4 in Simulation A is identical to Method 4 in Simulation B (to facilitate comparisons between the two simulations). For each function and method, we fit a two-way model with an interaction. For the `bigssa.R` function, the syntax is straightforward (`y~x1 * x2`). For the `gam.R` function, the smooths function (`s.R`) is used (with the `by=x2` option) for fitting η_A and η_B, whereas the tensor product function (`te.R`) is needed for fitting η_C and η_D; see the online supplementary R code for specific details and function syntax.

For all four methods, we used cubic marginal splines for the continuous variable(s). For Methods 1, 2, and 4 we used the same knots that were used in Simulation A; that is, $q = 30$ bin-sampled knots for η_A and η_B, and $q = 100$ bin-sampled knots for η_C and η_D. For Method 3, we used the default `gam.R` knot-selection algorithm, which places an equidistant grid of points throughout the covariate domain (see Wood 2014). The "cardinal" spline parameterization used by Wood's `gam.R` function is not directly comparable to the SSANOVA framework (see Wood 2006, 2014), so it is not possible to fit a GAM with an

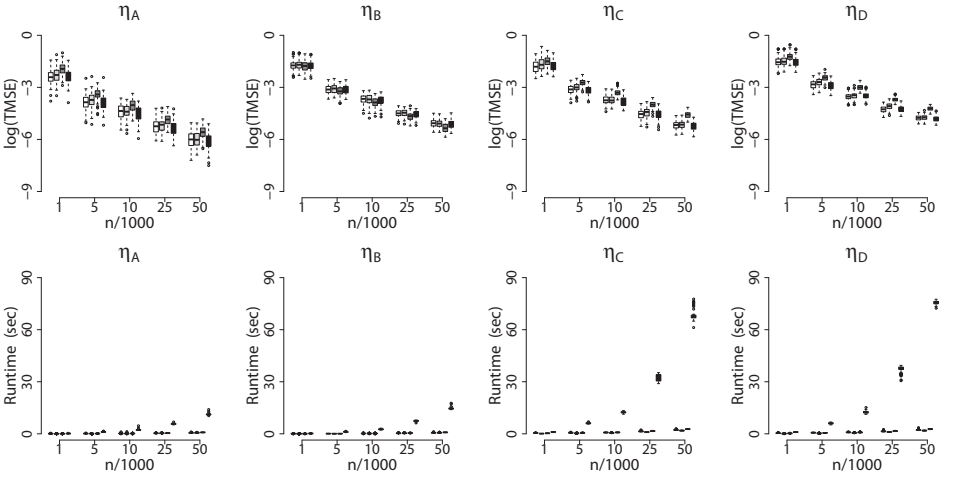

Figure 3. Simulation B log-TMSE (top) and runtime (bottom) boxplots. Within each sample size, left (white) box is Method 1 (scalable algorithm, fully optimized γ_j's), left-middle (light gray) box is Method 2 (scalable algorithm, partially optimized γ_j's), right-middle (dark gray) box is Method 3 (Wood's GAM algorithm), and right (black) box is Method 4 (classic algorithm, partially optimized θ_k's).

identical number of parameters. To allow for a fair comparison, we used (a) $k = 11$ knots for each level of x_2 when fitting η_A and η_B, for a total of 33 basis function coefficients, and (b) $k = 11$ knots for each marginal when fitting η_C and η_D, for a total of 121 basis function coefficients. Finally, as in Simulation A, for each method we (a) selected the smoothing parameters that minimized the GCV score, (b) used the TMSE to evaluate the quality of the solutions, and (c) used 100 replications of the above procedure within each cell of the simulation design.

4.2.3 Simulation B: Results. The log-TMSE for each combination of Simulation B conditions is plotted in Figure 3. Similar to Simulation A, all of the methods in Simulation B performed reasonably well, and (as expected) the TMSE approached zero as n increased for all examined conditions. Furthermore, as expected, the TMSE values associated with the partially optimized γ_j parameters (i.e., Method 2) tended to be larger than the TMSE values associated with the fully optimized γ_j parameters (i.e., Method 1); however, the TMSE difference between the fully and partially optimized solutions was negligible.

Comparing the TMSE values of Methods 1 and 2 to those of Methods 3 and 4, the effectiveness of our efficient reparameterization is obvious. In all of the examined conditions, our efficient reparameterization (i.e., Methods 1 and 2) performed comparable to Wood's GAM (Method 3) and the conventional SSANOVA parameterization (Method 4) with respect to the observed TMSE values. Furthermore, for a majority of the examined functions (i.e., η_A, η_C, and η_D), the GAM introduced a noticeable bias (compared to the other methods) that increased with n; at the largest sample size ($n = 5 \times 10^4$), the TMSEs associated with Method 3 tended to be about 50% larger than the TMSEs associated with the other three methods (for η_A, η_C, and η_D). So, for a majority of the function shapes, an

SSANOVA with bin-sampled knots outperformed the corresponding GAM with respect to recovering the unknown true function.

The analysis runtime for each combination of simulation conditions is also plotted in Figure 3. As expected, the runtimes (a) increased as n increased for all methods, (b) were slightly larger for the fully optimized γ_j's (i.e., Method 1) compared to the partially optimized γ_j's (i.e., Method 2). Furthermore, as expected, the runtimes for the efficient parameterization (with scalable algorithm) were noticeably smaller than the corresponding runtimes for the conventional parameterization (with scalable algorithm); for example, for η_D the median runtimes for Methods 1 and 2 with $n = 5 \times 10^4$ were (a) 6.6 and 2.4 sec (respectively) in Simulation A, and (b) 2.2 and 1.8 seconds (respectively) in Simulation B. In comparison, for η_D with $n = 5 \times 10^4$, the median runtime was 2.7 sec for the GAM, 75.7 sec for the partially optimized SSANOVA (with classic algorithm), and 111.3 sec for the fully optimized SSANOVA (with classic algorithm). Thus, when smoothing large samples, a two-way SSANOVA model (using our efficient reparameterization and scalable algorithm) can be fit more quickly than the corresponding GAM.

5. EL NIÑO EXAMPLE

5.1 DATA

The data used in the example were obtained from Bache and Lichman (2013), but are originally from the Tropical Atmosphere Ocean project (TAO; see NOAA 2014). The TAO project collects oceanographic data from approximately 70 buoys positioned throughout the equatorial Pacific Ocean, see Figure 4(a). The TAO project monitors the El Niño and La Niña phenomena, which involve unusually warm and cold equatorial ocean temperatures, respectively. For this example, we analyzed ocean surface temperature data collected from

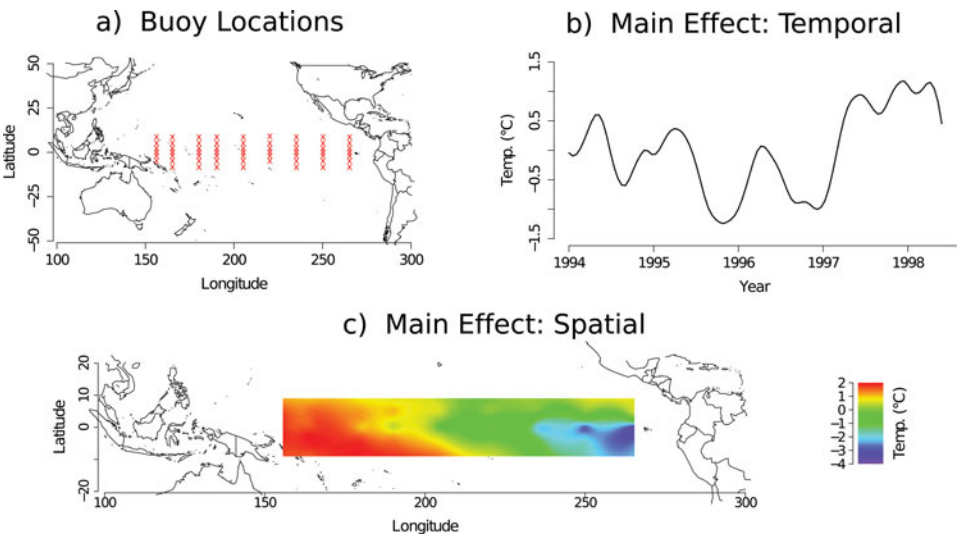

Figure 4. (a) Approximate locations of the buoys in the example, (b) main effect of time variable, and (c) main effect of space variable.

January 1994 to May 1998, and we only analyzed data from buoys east of 156° longitude. There were $n = 86{,}501$ data points included in the example; see the online supplementary R code for more specifics.

5.2 ANALYSES

We analyzed the ocean temperature data using a two-way SSANOVA where the first predictor was the bidimensional space effect (i.e., longitude and latitude) and the second predictor was time effect (in years). We used (a) a cubic thin-plate spline for the bidimensional space effect (see Appendix), (b) a cubic smoothing spline for the time effect, and (c) a total of $q = 676$ bin-sampled knots. Finally, we fit the model using the efficient reparameterization (see Section 2) with partially optimized smoothing parameters (`skip.iter=TRUE` in Helwig's `bigssa.R` function) and the scalable algorithm (see Section 3). We tried fitting the model both without and with the two-way interaction between the space and time effects.

5.3 RESULTS

The relevant fit statistics for the additive and interaction models are given in Table 1. From the fit statistics in Table 1, there is clear evidence that the interaction model should be preferred; however, the additive model can explain about 75% of the data variation, suggesting that the main effects (from the interaction model) are worth examining. The temporal main effect plot in Figure 4(b) reveals the intense 1995 La Niña event (i.e., drop in ocean temperatures) and the 1997 El Niño event (i.e., increase in ocean temperatures). The spatial main effect plot in Figure 4(c) reveals that the equatorial ocean surface temperatures are generally warmer in the western Pacific and colder in the southeastern Pacific.

The general trends in Figure 4 account for about 75% of the variation in the Pacific Ocean surface temperatures. However, the space-time interaction effect accounts for approximately 20% of the variation that is unexplained by the main effect trends. To visualize the significant interaction effect, it is helpful to examine the predicted ocean surface temperatures at different time points. Selected ocean surface temperatures from January 1994 to May 1998 are plotted in Figure 5, and an animation of the predicted temperatures can be found with the online supplementary material accompanying this article. Note that the 1995 La Niña event is characterized by an unusually large plume of cold water in the southeast Pacific in September, whereas the 1997 El Niño event has no plume of cold water in the southeast Pacific in September. Furthermore, from the animation in the online supplementary materials, note that the warmer waters of the western Pacific tend to move eastward during the 1997 El Niño.

Table 1. Fit statistics for the two-way SSANOVA models fit to the El Niño data

Model	GCV	R^2	AIC	BIC
Additive	0.93	0.75	239422.7	242027.9
Interaction	0.20	0.95	106029.6	112383.3

Note. R^2 is explained variation and AIC/BIC is Akaike's/Bayesian information criterion.

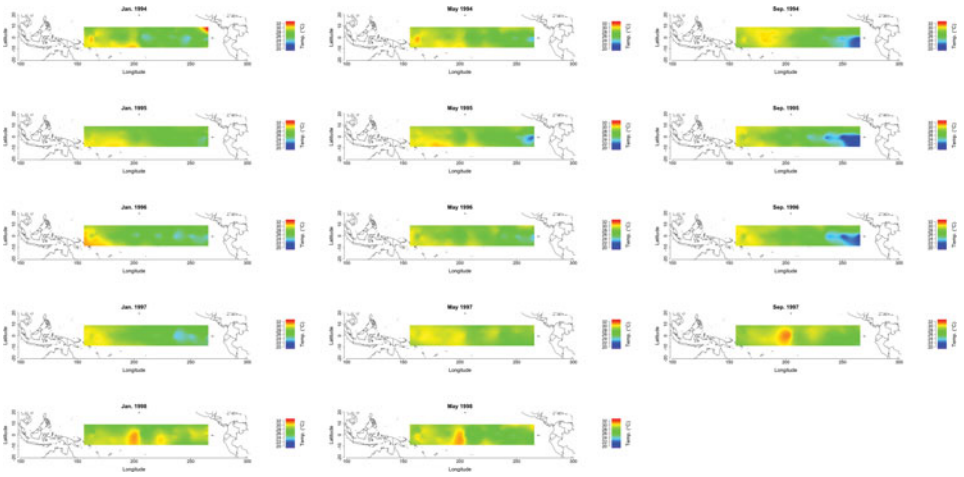

Figure 5. Some predicted equatorial ocean surface temperatures from Jan. 1994 to May 1998. See the online supplementary material for an animation of the predicted temperatures.

6. DISCUSSION

Our results clearly reveal the benefits of our SSANOVA reparameterization and algorithm. Specifically, Simulation A demonstrates that our scalable SSANOVA algorithm (a) produces TMSE values comparable to the classic SSANOVA algorithm, and (b) is much more computationally efficient than the classic SSANOVA algorithm when analyzing large samples. Furthermore, Simulation B reveals that SSANOVAs can (a) outperform GAMs when analyzing many different function shapes, and (b) be fit more quickly than a GAM when n is large. Also, comparing the results of the two simulations, it is evident that our efficient reparameterization is effective under a wide variety of different data situations; for example, the efficient reparameterization performed well when the SSANOVA model was misspecified (i.e., fitting an interaction model to η_A and η_C) or correctly specified (i.e., fitting an interaction model to η_B and η_D).

Our results also allow for an interesting comparison between Gu's (2013) SSANOVA approach and Wood's (2004, 2006) GAM approach. For a majority of the simulation functions, the SSANOVA methods produced smaller TMSE values than the comparable (interaction) GAM; this was particularly true for the functions with nonparametric effects of two continuous predictors (i.e., η_C and η_D). However, the difference between the SSANOVA and GAM solutions was minor (e.g., TMSE of GAM was 0.005 larger), and the GAM runtime was substantially smaller than the classic SSANOVA runtime. So, Wood's `gam.R` function seems to be a reasonable alternative to Gu's `ssanova.R` function when fitting two-way SSANOVA models to large samples. Furthermore, Helwig's `bigssp.R` and `bigssa.R` functions offer fast and accurate alternatives to Gu's `ssanova.R` function when fitting SSANOVAs to large samples.

Finally, our results also demonstrate the flexibility and potential of the SSANOVA approach for analyzing real data. The El Niño data example demonstrates that SSANOVAs can provide a powerful nonparametric framework for analyzing spatiotemporal data. And,

using the reparameterization and algorithm developed in this article, it is now possible to use the powerful SSANOVA framework to analyze large spatiotemporal datasets. As a result, we suspect that SSANOVAs will prove quite useful for discovering interesting functional relationships among large noisy datasets in the physical and social sciences.

APPENDIX: SPLINE TYPES AND REPRODUCING KERNELS

Table A.1. Marginal reproducing kernels for different spline types

Spline type	$\rho_{\mathrm{n}}(x_g, x_h)$	$\rho_{\mathrm{c}}(x_g, x_h)$
mth Order Smoothing	$\sum_{v=0}^{m-1} \kappa_v(x_g)\kappa_v(x_h)$	$\kappa_m(x_g)\kappa_m(x_h) + (-1)^{m-1}\kappa_{2m}(\lvert x_g - x_h\rvert)$
mth Order Periodic	$\kappa_0(x_g)\kappa_0(x_h)$	$(-1)^{m-1}\kappa_{2m}(\lvert x_g - x_h\rvert)$
Nominal	$1/f$	$I_{x_g = x_h} - 1/f$

Note 1. The m-order splines assume $x \in [0, 1]$, and nominal spline assumes $x \in \{1, \ldots, f\}$.
Note 2. For linear and cubic splines, the needed functions are $\kappa_0(x) \equiv 1$, $\kappa_1(x) \equiv x - 0.5$, $\kappa_2(x) \equiv 0.5[\kappa_1^2(x) - 1/12]$, and $\kappa_4(x) \equiv \{\kappa_1^4(x) - [\kappa_1^2(x)/2] + 7/240\}/24$.

The null and contrast space RKs for various different types of splines are given in Table A.1. The thin-plate spline RK is more complicated (see Gu 2013b, for a thorough discussion). For practical computation with a set of selected knots $\{\check{\mathbf{x}}_t\}_{t=1}^q$, the following procedure can be used to fit a cubic thin-plate spline with $\mathbf{x}_i \in \mathbb{R}^d$ (assuming $d \leq 3$).

First, define $\check{\boldsymbol{\Psi}} \equiv \{\psi_v(\check{\mathbf{x}}_t)\}_{q\times M}$ for $t \in \{1, \ldots, q\}$ and $v \in \{0, \ldots, d\}$, where $\psi_0(\check{\mathbf{x}}_t) \equiv 1$, $\psi_j(\check{\mathbf{x}}_t) \equiv \check{x}_{tj}$ for $j \in \{1, \ldots, d\}$, and $M \equiv d + 1$. Next, let $\mathbf{F}_1\mathbf{R}$ denote the QR decomposition of $\check{\Psi}$ such that $\mathbf{F}_1$ is a $q \times M$ matrix of orthonormal columns with the first column proportional to a column of ones, and $\mathbf{R}$ is a $M \times M$ upper triangular matrix. Also, let $\mathbf{F}_2$ denote a $q \times (q - M)$ matrix of orthonormal columns that are orthogonal to the columns of $\mathbf{F}_1$, so that the columns of $\mathbf{F} \equiv (\mathbf{F}_1, \mathbf{F}_2)$ form an orthonormal basis for $\mathbb{R}^q$. Given the knots $\{\check{\mathbf{x}}_t\}_{t=1}^q$, the null space RK is $\rho_{\mathrm{n}}(\mathbf{x}_g, \mathbf{x}_h) = \sum_{v=1}^M \phi_v(\mathbf{x}_g)\phi_v(\mathbf{x}_h)$ where $\boldsymbol{\phi}_g \equiv \{\phi_1(\mathbf{x}_g), \ldots, \phi_M(\mathbf{x}_g)\}_{1\times M}$ is obtained using the relation $\boldsymbol{\phi}_g = \sqrt{q}\boldsymbol{\psi}_g\mathbf{R}^{-1}$ with $\boldsymbol{\psi}_g \equiv \{\psi_v(\mathbf{x}_g)\}_{1\times M}$.

For a cubic thin-plate spline, the RK of the contrast space has the form

$$\rho_{\mathrm{c}}(\mathbf{x}_g, \mathbf{x}_h) = \xi(\mathbf{x}_g, \mathbf{x}_h) - q^{-1}\boldsymbol{\phi}_g\check{\boldsymbol{\Phi}}'\check{\boldsymbol{\xi}}_h - q^{-1}\check{\boldsymbol{\xi}}_g'\check{\boldsymbol{\Phi}}\boldsymbol{\phi}_h' + q^{-2}\boldsymbol{\phi}_g\check{\boldsymbol{\Phi}}'\check{\boldsymbol{\Xi}}\check{\boldsymbol{\Phi}}\boldsymbol{\phi}_h',$$

where $\check{\boldsymbol{\Phi}} \equiv \{\phi_v(\check{\mathbf{x}}_t)\}_{q\times M}$ for $t \in \{1, \ldots, q\}$ and $v \in \{1, \ldots, M\}$, $\check{\boldsymbol{\xi}}_g \equiv \{\xi(\check{\mathbf{x}}_t, \mathbf{x}_g)\}_{q\times 1}$ for $t \in \{1, \ldots, q\}$, $\check{\boldsymbol{\Xi}} \equiv \{\xi(\check{\mathbf{x}}_t, \check{\mathbf{x}}_w)\}_{q\times q}$ for $t, w \in \{1, \ldots, q\}$, and the semikernel is given by

$$\xi(\mathbf{x}_g, \mathbf{x}_h) = \begin{cases} \alpha\|\mathbf{x}_g - \mathbf{x}_h\|^2 \ln(\|\mathbf{x}_g - \mathbf{x}_h\|) & \text{if } d = 2 \\ \beta\|\mathbf{x}_g - \mathbf{x}_h\|^{4-d}\mathrm{sign}(2-d) & \text{if } d \in \{1, 3\} \end{cases}, \tag{A.1}$$

where α and β are positive scalars that can be absorbed into the smoothing parameters by setting $\alpha = \beta = 1$ when defining the RK.

Note that the penalty matrix $\mathbf{Q} \equiv \{\rho_{\mathrm{c}}(\check{\mathbf{x}}_t, \check{\mathbf{x}}_w)\}_{q\times q}$ for $t, w \in \{1, \ldots, q\}$ has the form

$$\begin{aligned} \mathbf{Q} &= \check{\boldsymbol{\Xi}} - q^{-1}\check{\boldsymbol{\Phi}}\check{\boldsymbol{\Phi}}'\check{\boldsymbol{\Xi}} - q^{-1}\check{\boldsymbol{\Xi}}\check{\boldsymbol{\Phi}}\check{\boldsymbol{\Phi}}' + q^{-2}\check{\boldsymbol{\Phi}}\check{\boldsymbol{\Phi}}'\check{\boldsymbol{\Xi}}\check{\boldsymbol{\Phi}}\check{\boldsymbol{\Phi}}' \\ &= (\mathbf{I} - \mathbf{F}_1\mathbf{F}_1')\check{\boldsymbol{\Xi}}(\mathbf{I} - \mathbf{F}_1\mathbf{F}_1') \\ &= \mathbf{F}_2\mathbf{F}_2'\check{\boldsymbol{\Xi}}\mathbf{F}_2\mathbf{F}_2', \end{aligned}$$

where the relation $\sqrt{q}\mathbf{F}_1 = \check{\mathbf{\Phi}}$ is used. Similarly, the basis function matrix $\mathbf{J} \equiv \{\rho_c(\mathbf{x}_i, \check{\mathbf{x}}_t)\}_{n\times q}$ for $i \in \{1, \ldots, n\}$ and $t \in \{1, \ldots, q\}$ has the form

$$\begin{aligned}\mathbf{J} &= \mathbf{\Xi} - q^{-1}\mathbf{\Phi}\check{\mathbf{\Phi}}'\check{\mathbf{\Xi}} - q^{-1}\mathbf{\Xi}\check{\mathbf{\Phi}}\check{\mathbf{\Phi}}' + q^{-2}\mathbf{\Phi}\check{\mathbf{\Phi}}'\check{\mathbf{\Xi}}\check{\mathbf{\Phi}}\check{\mathbf{\Phi}}' \\ &= \mathbf{\Xi}(\mathbf{I} - q^{-1}\check{\mathbf{\Phi}}\check{\mathbf{\Phi}}') - q^{-1}\mathbf{\Phi}\check{\mathbf{\Phi}}'\check{\mathbf{\Xi}}(\mathbf{I} - q^{-1}\check{\mathbf{\Phi}}\check{\mathbf{\Phi}}') \\ &= (\mathbf{\Xi} - \mathbf{\Psi}\mathbf{R}^{-1}\mathbf{F}_1'\check{\mathbf{\Xi}})\mathbf{F}_2\mathbf{F}_2',\end{aligned}$$

where $\mathbf{\Xi} \equiv \{\xi(\mathbf{x}_i, \check{\mathbf{x}}_t)\}_{n\times q}$ for $i \in \{1, \ldots, n\}$ and $t \in \{1, \ldots, q\}$, $\mathbf{\Phi} \equiv \{\phi_v(\mathbf{x}_i)\}_{n\times M}$ for $i \in \{1, \ldots, n\}$ and $v \in \{1, \ldots, M\}$, and $\mathbf{\Psi} \equiv \{\psi_v(\mathbf{x}_i)\}_{n\times M}$ for $i \in \{1, \ldots, n\}$ and $v \in \{0, \ldots, d\}$; the relation $\sqrt{q}\mathbf{\Psi}\mathbf{R}^{-1} = \Phi$ is used.

It should be noted that SSANOVAs formed with marginal RKs from Table A.1 will be scale invariant in the sense that scale transformations of the covariates would not change the result. This is because cubic and periodic smoothing splines assume that the observed data domain has been transformed to the interval [0,1] as a preprocessing step, so the covariate scales are implicitly removed before the formation of the tensor product space. In contrast, the thin-plate spline RK retains the data scale; however, note that the thin-plate null space RK $\boldsymbol{\phi}_g = \sqrt{q}\boldsymbol{\psi}_g\mathbf{R}^{-1}$ is scale free because the data in $\boldsymbol{\psi}_g$ are post-multiplied by $\mathbf{R}^{-1}$, where $\mathbf{R}$ contains the data scale. Furthermore, assuming that the jth predictor has a thin-plate spline marginal, note that γ_j is always attached to the jth predictor's contrast space RK using our reparameterization; as a result the scale of the thin-plate contrast space RK can be absorbed into γ_j, so the thin-plate RK is effectively scale free using our efficient reparameterization.

SUPPLEMENTARY MATERIALS

README (Overview): Summary of the other supplementary materials. (README.txt)

Simulation Functions Plot: R code for creating functions plot in Figure 1. (sim_funplot.R)

Simulation A Script: R code for replicating Simulation A analyses and Figure 2. (simA_script.R)

Simulation B Script: R code for replicating Simulation B analyses and Figure 3. (simB_script.R)

El Niño Script: R code for replicating El Niño analyses, Figures 4–5, and movie. (elnino_script.R)

El Niño Animation: An animation of the predicted equatorial Pacific Ocean surface temperatures from January 1994 to May 1998. (ElNinoMovie.mp4)

ACKNOWLEDGMENTS

This work was funded by NSF grants DMS-1055815, DMS-0800631, and DMS-1228288.

[Received July 2013. Revised April 2014.]

REFERENCES

Bache, K., and Lichman, M. (2013), *UCI Machine Learning Repository*. Available at *http://archive.ics.uci.edu/ml/citation_policy.html*. [727]

Craven, P., and Wahba, G. (1979), "Smoothing Noisy Data With Spline Functions: Estimating the Correct Degree of Smoothing by the Method of Generalized Cross-Validation," *Numerische Mathematik*, 31, 377–403. [719]

Dennis, J. E., and Schnabel, R. B. (1996), *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Philadelphia, PA: Society for Industrial and Applied Mathematics. [721]

Gu, C. (2013a), *gss: General Smoothing Splines*, R package version 2.1-0. [723,725]

——— (2013b), *Smoothing Spline ANOVA Models* (2nd ed.), New York: Springer-Verlag. [716,719,730]

Gu, C., and Wahba, G. (1991), "Minimizing GCV/GML Scores With Multiple Smoothing Parameters via the Newton Method," *SIAM Journal on Scientific and Statistical Computing*, 12, 383–398. [719,720,721,722]

Helwig, N. E. (2013, May), "Fast and Stable Smoothing Spline Analysis of Variance Models for Large Samples With Applications to Electroencephalography Data Analysis," Ph.D. thesis, University of Illinois at Urbana-Champaign. [718,720]

——— (2014), *bigsplines: Big Splines (Smoothing Splines for Large Samples)*, R package version 1.0-0. [723,725]

Kim, Y.-J., and Gu, C. (2004), "Smoothing Spline Gaussian Regression: More Scalable Computation via Efficient Approximation," *Journal of the Royal Statistical Society,* Series B, 66, 337–356. [716,719,721,722,723,724]

Li, K.-C. (1986), "Asymptotic Optimality for C_p, C_L, Cross-Validation and Generalized Cross-Validation: Discrete Index Set," *The Annals of Statistics*, 15, 958–975. [719]

Lin, Y., and Zhang, H. H. (2006), "Component Selection and Smoothing in Multivariate Nonparametric Regression," *The Annals of Statistics*, 34, 2272–2297. [718]

NOAA (2014), *Tropical Atmosphere Ocean Project*. Available at *http://www.pmel.noaa.gov/tao/*. [727]

Wahba, G. (1990), *Spline Models for Observational Data*, Philadelphia, PA: Society for Industrial and Applied Mathematics. [716]

Wood, S. N. (2004), "Stable and Efficient Multiple Smoothing Parameter Estimation for Generalized Additive Models," *Journal of the American Statistical Association*, 99, 673–686. [716,722,725]

——— (2006), *Generalized Additive Models: An Introduction With R*, Chapman & Hall: Boca Raton. [716,725]

——— (2014), *mgcv: Mixed GAM Computation Vehicle With GCV/AIC/REML Smoothness Estimation and GAMMs by REML/PQL*, R package version 1.7-28. [725]