Journal of Machine Learning Research 24 (2023) 1-49 Submitted 12/20; Revised 9/23; Published 10/23

Scalable high-dimensional Bayesian varying coefficient models with unknown within-subject covariance

Ray Bai RBAI@MAILBOX.SC.EDU
Department of Statistics
University of South Carolina
Columbia, SC 29201, USA

Mary R. Boland MARY.BOLAND@STVINCENT.EDU
Department of Mathematics
Saint Vincent College
Latrobe, PA 15650, USA

Yong Chen YCHEN123@UPENN.EDU
Department of Biostatistics, Epidemiology, and Informatics
University of Pennsylvania
Philadelphia, PA 19104, USA

Editor: Sayan Mukherjee

Abstract

Nonparametric varying coefficient (NVC) models are useful for modeling time-varying effects on responses that are measured repeatedly for the same subjects. When the number of covariates is moderate or large, it is desirable to perform variable selection from the varying coefficient functions. However, existing methods for variable selection in NVC models either fail to account for within-subject correlations or require the practitioner to specify a parametric form for the correlation structure. In this paper, we introduce the nonparametric varying coefficient spike-and-slab lasso (NVC-SSL) for Bayesian high-dimensional NVC models. Through the introduction of functional random effects, our method allows for flexible modeling of within-subject correlations without needing to specify a parametric covariance function. We further propose several scalable optimization and Markov chain Monte Carlo (MCMC) algorithms. For variable selection, we propose an Expectation Conditional Maximization (ECM) algorithm to rapidly obtain maximum a posteriori (MAP) estimates. Our ECM algorithm scales linearly in the total number of observations N and the number of covariates p. For uncertainty quantification, we introduce an approximate MCMC algorithm that also scales linearly in both N and p. We demonstrate the scalability, variable selection performance, and inferential capabilities of our method through simulations and a real data application. These algorithms are implemented in the publicly available R package `NVCSSL` on the Comprehensive R Archive Network.

Keywords: functional random effects, spike-and-slab group lasso, variable selection, varying coefficient model, within-subject covariance

1. Introduction

1.1 Model set-up

Consider the nonparametric varying coefficient (NVC) model with p covariates,

$$y_i(t_{ij}) = \sum_{k=1}^{p} x_{ik}(t_{ij})\beta_k(t_{ij}) + \varepsilon_{ij}(t_{ij}), \quad i = 1, \ldots, n, \ j = 1, \ldots, m_i, \tag{1}$$

where $y_i(t)$ is the response for the ith subject at time point $t \in \mathcal{T}$, $\mathcal{T}$ is the time interval on which the m_i different measurements are taken, $x_{ik}(t)$ is a possibly time-dependent covariate with corresponding smooth coefficient function $\beta_k(t)$, and $\varepsilon_{ij} := \varepsilon_{ij}(t_{ij})$ is random error. Throughout this paper, we denote $N = \sum_{i=1}^{n} m_i$ as the total number of observations. We also assume that the error terms $\boldsymbol{\varepsilon}_i = (\varepsilon_{i1}, \ldots, \varepsilon_{im_i})^\top$, $i = 1, \ldots, n$, are independent, zero-mean Gaussian processes. That is, $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_i), i = 1, \ldots, n$, where $\boldsymbol{\Sigma}_i$ is the $m_i \times m_i$ variance-covariance matrix that captures the temporal correlation between the n_i responses, $y_i(t_{i1}), \ldots, y_i(t_{im_i})$, for the ith subject.

NVC models (1) arise in many real applications. A prominent example is in longitudinal data analysis where we aim to model the response for the ith experimental subject at m_i different time points (Hoover et al., 1998). NVC models can also be used for functional data analysis where the objective is to model functional responses $y_i(t), i = 1, \ldots, n$, varying over a continuum $t \in \mathcal{T}$ (Rice, 2004). Hastie and Tibshirani (1993) and Fan and Zhang (2008) provide some further examples of applications of these models. Under (1), the primary aim is to estimate and conduct inference for the varying coefficients $\beta_k(t), k = 1, \ldots, p$.

There has been extensive frequentist work on fitting NVC models. Typical approaches to fitting (1) use local polynomial kernel smoothing (Fan and Zhang, 2000; Wu and Chiang, 2000) or basis expansions (Huang et al., 2004; Qu and Li, 2006; Xue and Qu, 2012) to estimate the varying coefficients. Bayesian approaches to NVC models have also been developed. Liu et al. (2018) and Guhaniyogi et al. (2022) endow the varying coefficients with a Gaussian process (GP) prior. Biller and Fahrmeir (2001) and Huang et al. (2015) use splines to model the $\beta_k(t)$'s in (1) and place multivariate normal priors on the groups of basis coefficients. Li et al. (2015) use a scale-mixture of a multivariate normal distribution as a prior to shrink groups of basis coefficients towards zero. Deshpande et al. (2020) use Bayesian additive regression trees (BART) to model the varying coefficients.

1.2 Related work

When the number of covariates p is large, it is often desirable to perform variable selection from the varying coefficient functions. In the frequentist literature, many authors have applied penalty functions such as the group lasso (Yuan and Lin, 2006) in order to threshold many of the $\beta_k(t)$'s to zero. See, e.g. Wang and Xia (2009), Wang et al. (2008), and Wei et al. (2011). These frequentist penalized NVC models do not account for the within-subject temporal correlations, essentially solving penalized likelihood objective functions with $\boldsymbol{\varepsilon} = (\boldsymbol{\varepsilon}_1^\top, \ldots, \boldsymbol{\varepsilon}_n^\top)^\top \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_N)$ in (1).

In low-dimensional settings and without regularizing the parameter space, Krafty et al. (2008) and Chen and Wang (2011) incorporated estimation of within-subject correlations into NVC models. However, to the best of our knowledge, no similar extension has been made for high-dimensional, penalized NVC models. While many researchers, e.g. Wang and Xia (2009), Wang et al. (2008), Wei et al. (2011), and Xue and Qu (2012), have shown that consistent estimation of the β_k's and model selection consistency can still be achieved for penalized NVC models, failing to account for the error variances can nevertheless lead to invalid *inferences* in finite samples (Liang and Zeger, 1993). Thus, it seems prudent to explicitly model temporal dependence in NVC models. Furthermore, while point estimates are easily attained, another major limitation of current penalized NVC models is their lack of inferential capabilities.

Unlike frequentist penalized approaches, the Bayesian approaches of Liu et al. (2018), Li et al. (2015), Deshpande et al. (2020), and Guhaniyogi et al. (2022) explicitly model dependencies by specifying a parametric correlation structure for the residual errors or the cross-covariance correlation function. Liu et al. (2018) employ subject-specific random effects with a random intercept and a random slope, Deshpande et al. (2020) use a compound symmetry covariance structure, Li et al. (2015) use a first-order autoregressive process, and Guhaniyogi et al. (2022) use the exponential or the Gneiting's correlation functions. The choices of covariance structure in Deshpande et al. (2020), Li et al. (2015), and Guhaniyogi et al. (2022) are parameterized by one to three hyperparameters (e.g. an autocorrelation parameter ρ). Suitable priors are then placed on these hyperparameters. Apart from being able to properly handle correlations, Bayesian NVC models also allow for natural uncertainty quantification of the varying coefficients through their posterior distributions.

While the aforementioned Bayesian approaches enable modeling of within-subject correlation, one of their limitations is the need to prespecify a parametric correlation structure. In practice, the final estimates can be sensitive to the choice of kernel function (Stephenson et al., 2022), and misspecifying the correlation structure may lead to incorrect inferences about the model parameters. For example, if there are long-range dependencies, then correlation functions that decay exponentially with distance (such as the ones used in Guhaniyogi et al. (2022)) will not be able to adequately capture the dependence between responses observed at far apart time points.

Another challenge with Bayesian NVC models is computational. In practice, Markov chain Monte Carlo (MCMC) is typically used to fit these Bayesian NVC models. However, when the number of observations N and/or the number of predictors p is large, MCMC can be computationally prohibitive. Recently, there have been efforts to scale up Bayesian NVC models when N is large. Guhaniyogi et al. (2022) employ divide-and-conquer MCMC which divides the N data points into subsets of much smaller size, runs MCMC in parallel on each subset, and then combines the posterior samples in a principled manner to approximate the full data posterior. In a separate line of work, Guhaniyogi et al. (2023) employed data sketching, which first compresses the dataset to a much smaller size through a random linear transformation and then fits an NVC model to the compressed data using MCMC.

The approaches in Guhaniyogi et al. (2022) and Guhaniyogi et al. (2023) do not perform variable selection and are specifically designed to handle the "large N, small p" situation. In contrast, the methodology and algorithms that we introduce in this paper are meant to be applied in the "small N, large p" scenario. This scenario arises often in practice, for example, in genome-wide association studies (GWAS) and other analyses of high-throughput biological data (Li et al., 2015). For example, Li et al. (2015) used NVC modeling to model the changes in body mass index (BMI) for $n = 865$ subjects using $p = 33{,}239$ single nucleotide polymorphisms (SNPs) from the Framingham Heart Study. In this study, BMI was measured at irregular time points from age 29 to age 61 for each of the n subjects, and a multivariate Laplace prior was used to select the SNP varying coefficients that are significantly associated with BMI.

1.3 Our contributions

The goal of this paper is to introduce a comprehensive methodological and computational framework for high-dimensional Bayesian varying coefficient models. Our framework addresses all of the issues of variable selection, estimation, and uncertainty quantification. Methodologically, we introduce a new approach to Bayesian function selection in NVC models that flexibly accounts for unknown within-subject covariances. Despite the utility of Bayesian methods for inference, there is currently a lack of Bayesian variable selection methods for NVC models that are scalable in the number of *covariates* p. This work addresses this gap by proposing scalable optimization and MCMC algorithms when p is large and variable selection is a primary objective for the data analyst.

In fitting a Bayesian NVC model, we have the following desiderata: 1) our method should perform variable selection, 2) our method should be able to flexibly accommodate a wide variety of unknown within-subject correlation structures, and 3) our method should be scalable for large p. To the best of our knowledge, there are no existing Bayesian methods for NVC models that accomplish all three goals. In this paper, we address all of these issues. We focus mainly on methodology and computation. However, theoretical considerations for "small N, large p" Bayesian varying coefficient models are briefly discussed and are reported in much greater detail in a follow-up work by Bai (2023b).

Recently, there has been a rapid development in spike-and-slab lasso (SSL) methods to solve various high-dimensional problems, including (generalized) linear models (Ročková and George, 2018; Tang et al., 2017; Deshpande et al., 2019; Bai et al., 2022; Bai, 2023a), factor analysis (Ročková and George, 2016; Moran et al., 2021), graphical models (Gan et al., 2019a; Li et al., 2019; Gan et al., 2019b), and nonparametric additive regression (Bai et al., 2022). SSL methods endow regression coefficients with spike-and-slab priors such that the posterior mode gives exact sparsity. In this work, we extend the SSL methodology to functional and longitudinal data analysis. Our contributions can be summarized as follows:

- We introduce the *nonparametric varying coefficient spike-and-slab lasso* (NVC-SSL) for Bayesian estimation and variable selection in NVC models. Our method provides several advantages over previously proposed methodology for high-dimensional varying coefficient models. First, unlike existing frequentist penalized NVC models, NVC-SSL incorporates estimation of the within-subject covariance structure and borrows information across functional components through a *non*-separable beta-Bernoulli prior. Second, unlike existing Bayesian approaches, the NVC-SSL model does not assume known within-subject covariance functions.

- For scalable variable selection, we propose an ECM algorithm for MAP estimation that scales linearly in both p and N. Our approach gives exact sparsity, thereby allowing the MAP estimator to automatically perform selection from the varying coefficient functions.

- For scalable uncertainty quantification, we provide both an exact MCMC algorithm and an approximate MCMC algorithm. The exact algorithm scales linearly in p and quadratically in N, while the approximate MCMC algorithm scales linearly in *both* p and N. The approximate MCMC algorithm is shown to provide massive speed-ups over the exact algorithm as p increases. We quantify the tradeoffs of using the approximate MCMC algorithm in place of the exact algorithm.

The rest of this paper is structured as follows. In Section 2, we introduce the NVC-SSL model. In Section 3, we propose a fast ECM algorithm for rapidly obtaining MAP estimates of the varying coefficients under NVC-SSL. In Section 4, we provide exact and approximate MCMC algorithms for scalable uncertainty quantification. Section 5 presents simulation studies validating the variable selection performance, scalability, and inferential capability of NVC-SSL. Section 6 applies NVC-SSL to a real data application of identifying important transcription factors in the yeast cell cycle. Finally, Section 7 concludes the paper with a brief discussion.

We use the following notation in this paper. A Gaussian process with mean function $m := m(x)$ and covariance function $k := k(x, x')$ is denoted as $\mathcal{GP}(m, k)$. For a vector $\boldsymbol{v}$, $\|\boldsymbol{v}\|_2$ denotes its ℓ_2-norm. For two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, the Kronecker product is denoted by $\boldsymbol{A} \otimes \boldsymbol{B}$, and the direct sum of $\boldsymbol{A}$ and $\boldsymbol{B}$ is denoted by $\boldsymbol{A} \oplus \boldsymbol{B}$. For a square matrix $\boldsymbol{C}$, $\det(\boldsymbol{C})$ denotes its determinant and $\text{tr}(\boldsymbol{C})$ denotes its trace. For two square matrices $\boldsymbol{C}$ and $\boldsymbol{D}$ of the same dimension, $\boldsymbol{C} \geq \boldsymbol{D}$ means that $\boldsymbol{C} - \boldsymbol{D}$ is non-negative definite.

2. The Nonparametric varying coefficient spike-and-slab lasso

2.1 Modeling of unknown within-subject correlations

In order to accommodate unknown within-subject correlations, we suppose that we can decompose the error $\varepsilon_i(t_{ij})$ in (1) into two terms: a functional random effect (Guo, 2002) and a measurement error term. Thus, our model is

$$y_i(t_{ij}) = \sum_{k=1}^{p} x_{ik}(t_{ij})\beta_k(t_{ij}) + \alpha_i(t_{ij}) + r_{ij}, \quad \alpha_i(t) \sim \mathcal{GP}(0, k_i), \quad r_{ij} \sim \mathcal{N}(0, \sigma^2), \tag{2}$$

where the r_{ij}'s are independent measurement errors at each time point t_{ij}, and the $\alpha_i(t)$'s are subject-specific functional random effects that independently follow zero-mean Gaussian processes $\mathcal{GP}(0, k_i)$. In (2), the covariance function k_i models the ith subject's within-subject temporal correlations. In particular, the m_i-dimensional random effects vector $\boldsymbol{\alpha}_i(\boldsymbol{t}_i) = (\alpha_i(t_{i1}), \ldots, \alpha_i(t_{im_i}))^\top$ follows a multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{K}(\boldsymbol{t}_i))$, where the (j, j')th entry of $\boldsymbol{K}(\boldsymbol{t}_i)$ is $k_i(t_{ij}, t_{ij'})$, $1 \leq j, j' \leq m_i$.

The functional random effect $\alpha_i(t)$ in (2) deserves some explanation. Functional random effect models *generalize* mixed effects models with a random intercept $\alpha_i \sim \mathcal{N}(0, \sigma_\alpha^2)$ (Guo, 2002). The random intercept model is equivalent to the compound symmetry (CS) covariance function, $k(t, t') = \sigma^2 \mathbb{I}(t = t') + \sigma_\alpha^2$. Models that specify a random intercept (and a random slope), e.g. the approach in Liu et al. (2018), are thus imposing a *specific* parametric covariance function on the model. The more general formulation in (2) where $\boldsymbol{\alpha}_i(\boldsymbol{t}_i) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}(\boldsymbol{t}_i))$ allows for many other covariance functions to characterize the within-subject correlations. For example, if the squared exponential (SE) kernel function is used, then $k_i(t,t') = s^2 \exp\{-(t-t')^2/2\ell^2\}$, where s is the scale factor and ℓ is the length-scale. The SE covariance function assumes that the correlation between time points t and t' decreases exponentially as the distance $|t-t'|$ grows.

In this work, we treat the within-subject covariance functions k_i's as *completely unknown* and model these k_i's nonparametrically. This makes our model more flexible than previous works which ignore within-subject correlations (Wang and Xia, 2009; Wang et al., 2008; Wei et al., 2011) or which require specific parametric forms for the covariance functions (Liu et al., 2018; Li et al., 2015; Deshpande et al., 2020; Guhaniyogi et al., 2022).

2.2 Basis expansion representation of the NVC model

Following the development in Wang and Xia (2009), Wang et al. (2008), and Wei et al. (2011), we approximate each coefficient function β_k in (1) by a linear combination of d basis functions, i.e. at a particular time t,

$$\beta_k(t) \approx \sum_{l=1}^{d} \gamma_{kl} B_{kl}(t), \tag{3}$$

where $B_{kl}(t), l = 1, \ldots, d$, are the basis functions with corresponding basis coefficients γ_{kl}. In addition, we approximate the unknown functional random effect $\alpha_i(t)$ in (2) as a linear combination of q basis functions with *random* coefficients, i.e. for a particular time t,

$$\alpha_i(t) \approx \sum_{l=1}^{q} \widetilde{B}_{il}(t)\eta_{il}, \quad \boldsymbol{\eta}_i = (\eta_{i1}, \ldots, \eta_{iq})^\top \sim \mathcal{N}_q(\mathbf{0}, \boldsymbol{\Omega}), \tag{4}$$

where $\boldsymbol{\Omega}$ is a $q \times q$ positive-definite matrix and $\widetilde{B}_{il}(t), l = 1, \ldots, q$, are the basis functions. Combining (3)-(4), the model (2) can be approximated as

$$y_i(t_{ij}) \approx \sum_{k=1}^{p} \sum_{l=1}^{d} x_{ik}(t_{ij}) \gamma_{kl} B_{kl}(t_{ij}) + \sum_{l=1}^{q} \widetilde{B}_{il}(t_{ij}) \eta_{ik} + r_{ij}. \tag{5}$$

From (4), it is clear that the within-subject covariance function $k_i(t,t')$ is approximated by

$$k_i(t,t') \approx \widetilde{\boldsymbol{B}}_i^\top(t) \boldsymbol{\Omega} \widetilde{\boldsymbol{B}}_i(t'), \tag{6}$$

where $\widetilde{\boldsymbol{B}}_i(t) = (\widetilde{B}_{i1}(t), \ldots, \widetilde{B}_{iq}(t))^\top \in \mathbb{R}^q$. We see from (6) that our formulation affords a great deal of flexibility in modeling the unknown within-subject covariances. In particular, the covariance function $k_i(t_{ij}, t_{ij'})$ for the ith subject is modeled by a quadratic form of the subject-specific basis function vectors $\widetilde{\boldsymbol{B}}_i(t_{ij})$ and $\widetilde{\boldsymbol{B}}_i(t_{ij'})$. Therefore, if we choose a flexible family of basis functions for the $\widetilde{\boldsymbol{B}}_i$'s, then we can capture a wide variety of within-subject covariance functions. In practice, we use B-splines with equispaced knots as the basis functions for both the $B_{kl}(t)$'s and $\widetilde{B}_{il}(t)$'s, due to their computational simplicity, numerical stability, and excellent local approximation properties (Wei et al., 2011; Yoo and Ghosal, 2016). However, other basis functions such as natural splines, trigonometric functions, and wavelets could also be used to model the $B_{kl}(t)$ and $\widetilde{B}_{il}(t)$'s.

Recall that $N = \sum_{i=1}^n m_i$ is the total number of observations. Let $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p] \in \mathbb{R}^{N \times p}$, with $\boldsymbol{x}_k = (x_{1k}(t_{11}), \ldots, x_{1k}(t_{1m_1}), \ldots, x_{nk}(t_{n1}), \ldots, x_{nk}(t_{nm_n}))^\top$. Further, we define $\boldsymbol{B}(t)$ as a $p \times dp$ basis expansion matrix containing the $B_{kl}(t)$'s from (3),

$$\boldsymbol{B}(t) = \begin{pmatrix} B_{11}(t) & B_{12}(t) & \ldots & B_{1d}(t) & 0 & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \ldots & 0 & 0 & \ldots & B_{p1}(t) & B_{p2}(t) & \ldots & B_{pd}(t) \end{pmatrix},$$

and we define

$$\boldsymbol{U}_i = (\boldsymbol{u}_{i1}, \ldots, \boldsymbol{u}_{im_i})^\top \in \mathbb{R}^{m_i \times dp}, \tag{7}$$

where

$$\boldsymbol{u}_{ij}^\top = \boldsymbol{x}^\top(t_{ij})\boldsymbol{B}(t_{ij})$$

for $i = 1, \ldots, n, j = 1, \ldots, m_i$, and $\boldsymbol{x}(t_{ij}) \in \mathbb{R}^p$ denotes the row of $\boldsymbol{X}$ corresponding to the jth observation for the ith subject. We also define $\boldsymbol{Z}_i$ as the matrix with (j, l)th entry $\widetilde{B}_{il}(t_{ij})$ from (4), i.e.

$$\boldsymbol{Z}_i = (\widetilde{\boldsymbol{B}}_i(t_{i1}), \ldots, \widetilde{\boldsymbol{B}}_i(t_{im_i}))^\top \in \mathbb{R}^{m_i \times q}. \tag{8}$$

Let $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1^\top, \ldots, \boldsymbol{\gamma}_p^\top)^\top \in \mathbb{R}^{dp}$, where the kth subvector $\boldsymbol{\gamma}_k = (\gamma_{k1}, \ldots, \gamma_{kd}) \in \mathbb{R}^d$ consists of the basis coefficients γ_{kl}'s in (3) corresponding to the kth varying coefficient $\beta_k(t)$. Let $\boldsymbol{Y}_i = (y_i(t_{i1}, \ldots, y_i(t_{im_i}))^\top$ and $\boldsymbol{r}_i = (r_{i1}, \ldots, r_{im_i})^\top$ denote the m_i-dimensional vectors of responses and measurement errors for the ith subject. Then (5) can be written in matrix form as

$$\boldsymbol{Y}_i = \boldsymbol{U}_i\boldsymbol{\gamma} + \boldsymbol{Z}_i\boldsymbol{\eta}_i + \boldsymbol{r}_i, \quad \boldsymbol{\eta}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Omega}), \quad \boldsymbol{r}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}_{m_i}), \quad i = 1, \ldots, n, \tag{9}$$

where $\boldsymbol{U}_i$ and $\boldsymbol{Z}_i$ are as in (7) and (8). Letting $\boldsymbol{Y} = (\boldsymbol{Y}_1^\top, \ldots, \boldsymbol{Y}_n^\top)^\top \in \mathbb{R}^N$, $\boldsymbol{U} = (\boldsymbol{U}_1^\top, \ldots, \boldsymbol{U}_n^\top)^\top \in \mathbb{R}^{N \times dp}$, $\boldsymbol{Z} = \boldsymbol{Z}_1 \oplus \cdots \oplus \boldsymbol{Z}_n \in \mathbb{R}^{N \times nq}$, $\boldsymbol{\eta} = (\boldsymbol{\eta}_1^\top, \ldots, \boldsymbol{\eta}_n^\top)^\top \in \mathbb{R}^{nq}$, and $\boldsymbol{r} = (\boldsymbol{r}_1^\top, \ldots, \boldsymbol{r}_n^\top)^\top \in \mathbb{R}^N$, we can also express (9) for all N observations as

$$\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{\gamma} + \boldsymbol{Z}\boldsymbol{\eta} + \boldsymbol{r}, \quad \boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n \otimes \boldsymbol{\Omega}), \quad \boldsymbol{r} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}_N). \tag{10}$$

Before introducing the NVC-SSL model, we make a few remarks about our model setup. First, although we focus on continuous responses with Gaussian errors for concreteness, our method can easily be extended to NVC models with discrete or non-Gaussian responses by recasting our model into the generalized linear mixed model (GLMM) framework. In this case, we would employ a monotonically increasing link function g to relate the conditional expectation of $y_i(t_{ij})$ given the p covariates $\boldsymbol{x}_i(t_{ij}) = (x_{i1}(t_{ij}), \ldots, x_{ip}(t_{ij}))^\top$ to the varying coefficients $\beta_k(t)$'s as

$$\begin{aligned} \mathbb{E}[y_i(t_{ij}) \mid \boldsymbol{x}_i(t_{ij})] &= g^{-1}\left(\sum_{k=1}^p x_{ik}(t_{ij})\beta_k(t_{ij}) + \alpha_i(t_{ij})\right) \\ &= g^{-1}\left(\boldsymbol{u}_{ij}^\top\boldsymbol{\gamma} + \boldsymbol{z}_{ij}^\top\boldsymbol{\eta}_i\right), \quad \boldsymbol{\eta}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Omega}), \end{aligned} \tag{11}$$

where $\boldsymbol{u}_{ij}$ is the jth row of $\boldsymbol{U}_i$ in (7) and $\boldsymbol{z}_{ij}$ is the jth row of $\boldsymbol{Z}_i$ in (8). For example, if the response variables are binary, we can assume that $y_i(t_{ij}) \mid \boldsymbol{x}_i(t_{ij}) \sim \text{Bernoulli}(p_{ij})$ and employ the logit link function $g(p_{ij}) = \log(p_{ij}/(1-p_{ij}))$ to obtain a logistic NVC model. By putting the same priors on $(\boldsymbol{\gamma}, \boldsymbol{\Omega})$ in (11) as those introduced in Section 2.3, we can implement NVC-SSL for logistic NVC models. In particular, the ECM algorithm in Section 3 can be extended to logistic NVC models using the approach in Bai (2023a), and the MCMC algorithms in Section 4 can also be extended to GLMMs straightforwardly using Pólya-gamma data augmentation (Polson et al., 2013).

Secondly, our method can also be easily extended to NVC models where the varying coefficients are *multivariate* functions, e.g. spatial models where $\beta_k := \beta_k(\boldsymbol{s}), k = 1, \ldots, p$, and $\boldsymbol{s} \in \mathcal{S}$ where $\mathcal{S}$ is a spatial domain. If the varying coefficients are multivariate functions, we can replace the univariate basis functions $B_{kl}(t)$ and $\widetilde{B}_{il}(t)$ in (3) with tensor products of basis functions (see e.g., Bai et al. (2022)). For example, if the varying coefficients are functions of two variables, $\beta_k(u,v)$, we can approximate the varying coefficients as

$$\beta_k(u,v) \approx \sum_{l=1}^{d_u} \sum_{m=1}^{d_v} \gamma_{klm} B_{kl}(u) B_{km}(v),$$

and the functional random effects as

$$\alpha_i(u,v) \approx \sum_{l=1}^{d_u} \sum_{m=1}^{d_v} \widetilde{B}_{il}(u) \widetilde{B}_{im}(v) \eta_{ilm}.$$

We would then proceed to estimate the model parameters, e.g. the basis coefficients, exactly the same way as we would in the case of univariate varying coefficient functions $\beta_k(t)$.

2.3 Prior specification for NVC-SSL

Having rewritten our NVC model (2) in matrix form (10), parameter estimation reduces to estimating $(\boldsymbol{\gamma}, \boldsymbol{\Omega}, \sigma^2)$. We take a Bayesian approach and endow these parameters with suitable priors. In particular, estimating the varying coefficient functions $\beta_k(t)$'s in (2) are straightforward once we have estimates of the basis coefficients $\boldsymbol{\gamma}$. By (3), we can estimate $\widehat{\beta}_k(t) = \sum_{l=1}^{d} \widehat{\gamma}_{kl} B_{kl}(t), k = 1, \ldots, p$, once we have an estimate $\widehat{\boldsymbol{\gamma}}$.

As discussed in Section 1.2, we are interested in not only estimating the varying coefficient functions in (2), but *also* performing variable selection from them. Under the assumption of sparsity, most of the $\beta_k(t)$'s in (2) should equal zero. To facilitate variable selection, we endow the vector of basis coefficients $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1^\top, \ldots, \boldsymbol{\gamma}_p^\top)^\top$ in (10) with the *spike-and-slab group lasso* (SSGL) prior of Bai et al. (2022),

$$\pi(\boldsymbol{\gamma} \mid \theta) = \prod_{k=1}^{p} \left[(1-\theta) \boldsymbol{\Psi}(\boldsymbol{\gamma}_k \mid \lambda_0) + \theta \boldsymbol{\Psi}(\boldsymbol{\gamma}_k \mid \lambda_1) \right], \tag{12}$$

where $\theta \in (0,1)$ is a mixing proportion, or the expected proportion of nonzero $\boldsymbol{\gamma}_k$'s, and $\boldsymbol{\Psi}(\cdot \mid \lambda)$ denotes a multivariate Laplace density indexed by hyperparameter λ,

$$\boldsymbol{\Psi}(\boldsymbol{\gamma}_k \mid \lambda) = \frac{\lambda^d e^{-\lambda \|\boldsymbol{\gamma}_k\|_2}}{2^d \pi^{(d-1)/2} \Gamma((d+1)/2)}, \quad k = 1, \ldots, p.$$

The SSGL prior (12), which we denote as $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$ going forward, can be considered a two-group refinement of the group lasso (Yuan and Lin, 2006). Under the prior (12), the posterior mode for $\boldsymbol{\gamma}$ gives *exact* sparsity (i.e. some of the $\boldsymbol{\gamma}_k$ vectors will be exactly $\mathbf{0}$). This allows $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$ to perform joint estimation and variable selection (Bai et al., 2022). In the present context, if the posterior mode for $\boldsymbol{\gamma}_k$ is $\widehat{\boldsymbol{\gamma}}_k = \mathbf{0}$, then the kth function will be estimated as $\widehat{\beta}_k(t) = \sum_{l=1}^{d_k} \widehat{\gamma}_{kl} B_{kl}(t) = 0$ and thus thresholded out of the model.

We typically set $\lambda_0 \gg \lambda_1$ in (12), so that the first mixture component $\boldsymbol{\Psi}(\cdot \mid \lambda_0)$ (the spike) is heavily concentrated around the d-dimensional zero vector $\mathbf{0}$ for each $k = 1, \ldots, p$. Meanwhile, the slab component $\boldsymbol{\Psi}(\cdot \mid \lambda_1)$ stabilizes the posterior estimates of large coefficients, *preventing* them from being downward biased. One of the chief advantages of SSGL over other group penalties such as group lasso, group smoothly clipped absolute deviation (SCAD), or group minimax concave penalty (MCP) (Yuan and Lin, 2006; Breheny and Huang, 2015) is the SSGL's ability to perform *adaptive* shrinkage. Group lasso, group SCAD, and group MCP all contain a *single* regularization parameter $\lambda > 0$ controlling the sparsity of the solution. Consequently, if λ is large, then all groups may be overshrunk. In contrast, the slab hyperparameter λ_1 in $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$ applies minimal shrinkage to groups with larger coefficients, allowing these groups to escape the pull of the spike.

To model the uncertainty of the mixing proportion θ in (12), we endow θ with a beta prior,

$$\theta \sim \mathcal{B}(a, b), \tag{13}$$

where $a > 0$ and $b > 0$ are fixed positive constants. Unlike the group lasso, group SCAD, and group MCP, this prior (13) on θ ultimately renders our Bayesian penalty *non*-separable in the sense that the groups $\boldsymbol{\gamma}_k, k = 1, \ldots, p$ are a priori *dependent*. This non-separability provides several benefits. First, the prior on θ allows the NVC-SSL model to *share* information across functional components and self-adapt to ensemble information about sparsity. Second, with appropriate choices for the hyperparameters in $\theta \sim \mathcal{B}(a, b)$, namely $a = 1, b = p$, our prior performs an automatic multiplicity adjustment (Scott and Berger, 2010) and favors parsimonious models in high dimensions. This helps NVC-SSL to avoid the curse of dimensionality for large p.

To complete the NVC-SSL prior specification, we place independent conditionally conjugate priors on the parameters $(\boldsymbol{\Omega}, \sigma^2)$ in (10). Namely, we endow $\boldsymbol{\Omega}$ with the prior,

$$\boldsymbol{\Omega} \sim \text{Inverse-Wishart}(\nu, \boldsymbol{\Phi}), \tag{14}$$

where the degrees of freedom $\nu > q - 1$ and the scale matrix $\boldsymbol{\Phi}$ is positive-definite. Finally, we endow the measurement error variance σ^2 with the prior,

$$\sigma^2 \sim \text{Inverse-Gamma}(c_0/2, d_0/2), \tag{15}$$

where $c_0 > 0, d_0 > 2$.

2.4 Theoretical considerations

In the literature on Bayesian asymptotics, a common theme is to study the posterior contraction rate, or the speed at which the posterior distribution converges to a point mass at the true parameter as sample size N grows to infinity. Recently, in the "fixed p" regime, posterior contraction rates have been derived for Bayesian NVC models by Deshpande et al. (2020), Guhaniyogi et al. (2022), and Guhaniyogi et al. (2023). However, these papers do *not* consider the case where p is allowed to diverge with n.

In a follow-up paper to this article, Bai (2023b) derives sufficient conditions for posterior contraction in high-dimensional Bayesian NVC models when $p \gg n$ and p grows subexponentially with n. To summarize briefly, the prior distribution is required to be heavily concentrated near zero (to capture sparsity) and to have a sufficiently heavy tail (to capture the true nonzero varying coefficients). With appropriately chosen hyperparameters, Bai (2023b) shows that the NVC-SSL prior can achieve adaptive posterior contraction to the true varying coefficients. The NVC-SSL prior is adaptive in the sense that it adapts to the unknown sparsity level *and* the unknown smoothness of the varying coefficients. These sufficient conditions are not specific to the NVC-SSL prior; other multivariate priors that satisfy the conditions in Bai (2023b) would also theoretically achieve adaptive posterior contraction. This general theory for Bayesian NVC models when $p > n$ is described in detail in Bai (2023b).

3. Scalable MAP estimation for variable selection

3.1 ECM algorithm

We now detail how to implement NVC-SSL, i.e. the model (5) with priors (12)-(15), for variable selection. We first present a very fast ECM algorithm which targets the posterior mode. Once we have obtained the MAP estimator $\widehat{\boldsymbol{\gamma}}$, the varying coefficients can then be estimated as $\widehat{\beta}_k(t) = \sum_{l=1}^{d} \widehat{\gamma}_{kl} B_{kl}(t), k = 1, \ldots, p$. As discussed in Section 2.3, the MAP estimator under NVC-SSL is *exactly* sparse, with many $\boldsymbol{\gamma}_k$'s thresholded to zero. This enables the MAP estimator to perform *automatic* variable selection, since $\widehat{\beta}_k(t) = 0$ if $\widehat{\boldsymbol{\gamma}}_k = \mathbf{0}$.

Let $\boldsymbol{\Xi}$ denote the collection $\boldsymbol{\Xi} = \{\boldsymbol{\gamma}, \theta, \boldsymbol{\eta}, \boldsymbol{\Omega}, \sigma^2\}$. Based on (10) and the prior densities (12)-(15), the log-posterior density for $\boldsymbol{\Xi}$ (up to an additive constant) is given by

$$\begin{aligned}
\log \pi(\boldsymbol{\Xi} \mid \boldsymbol{Y}) = & -\frac{N}{2}\log\sigma^2 - \frac{\|\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{\gamma} - \boldsymbol{Z}\boldsymbol{\eta}\|^2}{2\sigma^2} + \frac{n}{2}\log(\det(\boldsymbol{\Omega}^{-1})) - \frac{1}{2}\sum_{i=1}^{n} \boldsymbol{\eta}_i^\top \boldsymbol{\Omega}^{-1}\boldsymbol{\eta}_i \\
& + \sum_{k=1}^{p} \log\left((1-\theta)\lambda_0^d e^{-\lambda_0 \|\boldsymbol{\gamma}_k\|_2} + \theta \lambda_1^d e^{-\lambda_1 \|\boldsymbol{\gamma}_k\|_2} \right) \\
& + (a-1)\log\theta + (b-1)\log(1-\theta) \\
& + \frac{\nu+q+1}{2}\log\left(\det(\boldsymbol{\Omega}^{-1})\right) - \frac{1}{2}\mathrm{tr}(\boldsymbol{\Phi}\boldsymbol{\Omega}^{-1}) - \left(\frac{c_0+2}{2}\right)\log\sigma^2 - \frac{d_0}{2\sigma^2}. \quad (16)
\end{aligned}$$

Our objective is to maximize the log-posterior (16) with respect to $\boldsymbol{\Xi}$. We first introduce latent 0-1 indicators, $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_p)^\top$, i.e. $\tau_k \in \{0, 1\}$ for $k = 1, \ldots, p$. Then the $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$ prior (12) can be expressed as the marginal prior under a hierarchical beta-Bernoulli prior,

$$\begin{aligned}\pi(\boldsymbol{\gamma} \mid \boldsymbol{\tau}) = & \prod_{k=1}^{p} \left[(1-\tau_k)\boldsymbol{\Psi}(\boldsymbol{\gamma}_k \mid \lambda_0) + \tau_k \boldsymbol{\Psi}(\boldsymbol{\gamma}_k|\lambda_1)\right], \\ \pi(\boldsymbol{\tau} \mid \theta) = & \prod_{k=1}^{p} \theta^{\tau_k}(1-\theta)^{1-\tau_k}.\end{aligned} \tag{17}$$

With the augmented log-posterior $\log \pi(\boldsymbol{\Xi}, \boldsymbol{\tau} \mid \boldsymbol{Y})$, we can now implement an ECM algorithm to find the MAP estimator $\widehat{\boldsymbol{\Xi}} = \{\widehat{\boldsymbol{\gamma}}, \widehat{\theta}, \widehat{\boldsymbol{\eta}}, \widehat{\boldsymbol{\Omega}}, \widehat{\sigma}^2\}$. We first initialize the parameters $\boldsymbol{\Xi}^{(0)}$, and then in each tth iteration, we iterate between the E-step and the CM-steps until convergence. In the E-step, we treat the latent indicator variables $\boldsymbol{\tau}$ in (17) as missing data and compute $F^{(t)}(\boldsymbol{\gamma}, \theta, \boldsymbol{\eta}, \boldsymbol{\Omega}, \sigma^2) = \mathbb{E}_{\boldsymbol{\tau}}\left[\log(\boldsymbol{\Xi}, \boldsymbol{\tau} \mid \boldsymbol{Y}) \mid \boldsymbol{\Xi}^{(t-1)}\right]$. In the CM-step, we then optimize $F^{(t)}(\boldsymbol{\gamma}, \theta, \boldsymbol{\eta}, \boldsymbol{\Omega}, \sigma^2)$ with respect to $\boldsymbol{\Xi}$ by performing two iterative updates:

1. Update $(\theta, \boldsymbol{\eta})$, holding $(\boldsymbol{\gamma}, \boldsymbol{\Omega}, \sigma^2)$ fixed at their previous values, i.e. solve

$$(\theta^{(t)}, \boldsymbol{\eta}^{(t)}) = \arg\max_{\theta, \boldsymbol{\eta}} F^{(t)}(\boldsymbol{\gamma}^{(t-1)}, \theta, \boldsymbol{\eta}, \boldsymbol{\Omega}^{(t-1)}, \sigma^{2(t-1)}).$$

2. Update $(\boldsymbol{\gamma}, \boldsymbol{\Omega}, \sigma^2)$, holding $(\theta, \boldsymbol{\eta})$ fixed at their current values, i.e. solve

$$(\boldsymbol{\gamma}^{(t)}, \boldsymbol{\Omega}^{(t)}, \sigma^{2(t)}) = \arg\max_{\boldsymbol{\gamma}, \boldsymbol{\Omega}, \sigma^2} F^{(t)}(\boldsymbol{\gamma}, \theta^{(t)}, \boldsymbol{\eta}^{(t)}, \boldsymbol{\Omega}, \sigma^2).$$

To be more specific, in the E-step, we compute $\mathbb{E}_{\boldsymbol{\tau}}[\tau_k \mid \boldsymbol{Y}, \boldsymbol{\Xi}^{(t-1)}] = p_k^\star(\boldsymbol{\gamma}_k^{(t-1)}, \theta^{(t-1)}), k = 1, \dots, p$, where

$$p_k^\star(\boldsymbol{\gamma}_k, \theta) = \frac{\theta \boldsymbol{\Psi}(\boldsymbol{\gamma}_k|\lambda_1)}{\theta \boldsymbol{\Psi}(\boldsymbol{\gamma}_k|\lambda_1) + (1-\theta)\boldsymbol{\Psi}(\boldsymbol{\gamma}_k|\lambda_0)}, \tag{18}$$

is the conditional posterior probability that $\boldsymbol{\gamma}_k$ is drawn from the slab distribution rather than from the spike. We then compute $\lambda_k^\star, k = 1, \dots, p$, where

$$\lambda_k^\star = \mathbb{E}_{\boldsymbol{\tau}}\left[\log\left((1-\theta)\lambda_0^d e^{-\lambda_0\|\boldsymbol{\gamma}_k\|_2} + \theta\lambda_1^d e^{-\lambda_1\|\boldsymbol{\gamma}_k\|_2}\right) \mid \boldsymbol{Y}, \boldsymbol{\Xi}\right] = \lambda_1 p_k^\star + \lambda_0(1-p_k^\star). \tag{19}$$

Based on (18)-(19),

$$\begin{aligned}F^{(t)}(\boldsymbol{\gamma}, \theta, \boldsymbol{\eta}, \boldsymbol{\Omega}, \sigma^2) &= \mathbb{E}_{\boldsymbol{\tau}}\left[\log \pi(\boldsymbol{\Xi} \mid \boldsymbol{Y}) \mid \boldsymbol{\Xi}^{(t-1)}\right] \\ &= -\frac{N}{2}\log\sigma^2 - \frac{\|\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{\gamma} - \boldsymbol{Z}\boldsymbol{\eta}\|_2^2}{2\sigma^2} + \frac{n}{2}\log\left(\det(\boldsymbol{\Omega}^{-1})\right) - \frac{1}{2}\sum_{i=1}^{n}\boldsymbol{\eta}_i^\top\boldsymbol{\Omega}^{-1}\boldsymbol{\eta}_i \\ &\quad + \sum_{k=1}^{p}\lambda_k^\star\|\boldsymbol{\gamma}_k\|_2 + \left(a - 1 + \sum_{k=1}^{p}p_k^\star\right)\log\theta + \left(b - 1 + p - \sum_{k=1}^{p}p_k^\star\right)\log(1-\theta) \\ &\quad + \frac{\nu+q+1}{2}\log\left(\det(\boldsymbol{\Omega}^{-1})\right) - \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Phi}\boldsymbol{\Omega}^{-1}\right) - \left(\frac{c_0+2}{2}\right)\log\sigma^2 - \frac{d_0}{2\sigma^2}.\end{aligned} \tag{20}$$

The CM-step maximizes the objective (20). First, holding $(\boldsymbol{\gamma}^{(t-1)}, \boldsymbol{\Omega}^{(t-1)}, \sigma^{2(t-1)})$ fixed, θ has the following closed form update,

$$\theta^{(t)} = \frac{a - 1 + \sum_{k=1}^{p} p_k^{\star}}{a + b + p - 2}. \tag{21}$$

Meanwhile, each $\boldsymbol{\eta}_i, i = 1, \ldots, n$ in $\boldsymbol{\eta}$ can be updated individually in closed form as

$$\boldsymbol{\eta}_i^{(t)} = \boldsymbol{B}_{\boldsymbol{Z}_i} \left(\boldsymbol{Y}_i - \boldsymbol{U}_i \gamma^{(t-1)} \right), \tag{22}$$

where

$$\boldsymbol{B}_{\boldsymbol{Z}_i} = \left(\boldsymbol{Z}_i^{\top} \boldsymbol{Z}_i + \sigma^{2(t-1)} \boldsymbol{\Omega}^{(t-1)} \right)^{-1} \boldsymbol{Z}_i^{\top},$$

and $\boldsymbol{Y}_i$, $\boldsymbol{U}_i$, and $\boldsymbol{Z}_i$ are as in (9).

Next, we update $(\boldsymbol{\gamma}, \boldsymbol{\Omega}, \sigma^2)$ holding $(\theta^{(t)}, \boldsymbol{\eta}^{(t)})$ fixed. First, in order to update $\boldsymbol{\gamma}$, we solve the following optimization:

$$\boldsymbol{\gamma}^{(t)} = \arg\max_{\boldsymbol{\gamma}} -\frac{1}{2} \| \widetilde{\boldsymbol{Y}} - \boldsymbol{U}\boldsymbol{\gamma} \|_2^2 - \sum_{k=1}^{p} \sigma^{2(t-1)} \lambda_k^{\star} \| \boldsymbol{\gamma}_k \|_2, \tag{23}$$

where $\widetilde{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\eta}^{(t)}$ and the $\lambda_k^{\star}$'s are as in (19). It can be seen that (23) is an adaptive group lasso problem with group-specific weights $\sigma^{2(t-2)}\lambda_k^{\star}$. This optimization can be solved with any standard group lasso algorithm (Yuan and Lin, 2006; Breheny and Huang, 2015). We opt to use the coordinate ascent algorithm of Breheny and Huang (2015) due to its speed and numerical stability.

Although each CM step requires solving the optimization (23), it should be reiterated that $\lambda_k^{\star} = \lambda_1 p_k^{\star} + \lambda_0 (1 - p_k^{\star})$ in (23). Under sparsity, most of the $p_k^{\star}$'s in (18) are very close to zero (i.e. most of the $\boldsymbol{\gamma}_k$'s come from the spike density in (17)), and thus, most of the $\lambda_k^{\star}$'s satisfy $\lambda_k^{\star} \approx \lambda_0$. Therefore, as long as the spike hyperparameter λ_0 is large, *most* of the group-specific weights in (23) will *also* be large. Since a larger penalty is applied these respective groups, most of them will be thresholded to zero very early on and then remain at zero in the group optimization algorithm. This ensures that the coordinate ascent algorithm for solving (23) converges very rapidly. At the same time, for the few nonzero varying coefficients, we have $p_k^{\star} \approx 1$ and $\lambda_k^{\star} \approx \lambda_1$ (where $\lambda_1 \ll \lambda_0$). We are therefore able to apply a *weaker* penalty to basis coefficients $\boldsymbol{\gamma}_k$ with *larger* entries.

Finally, the updates for $\boldsymbol{\Omega}$ and σ^2 have the closed forms,

$$\boldsymbol{\Omega}^{(t)} = \frac{1}{n + \nu + q + 1} \left(\boldsymbol{\Phi} + \sum_{i=1}^{n} \boldsymbol{\eta}_i^{(t)} (\boldsymbol{\eta}_i^{(t)})^{\top} \right), \tag{24}$$

and

$$\sigma^{2(t)} = \frac{\| \boldsymbol{Y} - \boldsymbol{U}\boldsymbol{\gamma}^{(t)} - \boldsymbol{Z}\boldsymbol{\eta}^{(t)} \|_2^2 + d_0}{N + c_0 + 2}. \tag{25}$$

In particular, as long as the scale matrix $\boldsymbol{\Phi}$ in (24) is chosen to be positive-definite, the update for $\boldsymbol{\Omega}$ is also guaranteed to be positive-definite.

Algorithm 1 ECM algorithm for MAP estimation under NVC-SSL

```
Input: Initial values $\boldsymbol{\gamma}^{(0)}$, $\theta^{(0)}$, $\boldsymbol{\Omega}^{(0)}$, $\sigma^{2(0)}$, $t = 0$
Output: Estimated varying coefficients $\widehat{\beta}_k(\boldsymbol{t}),\ k = 1, \ldots, p$
while diff $> \epsilon$ do
    1. Increment $t$
    2. E-step:
       for $k = 1, \ldots, p$ do
           (a) Compute $p_k = p^\star(\boldsymbol{\gamma}_k^{(t-1)}, \theta^{(t-1)})$ as in (18)
           (b) Set $\lambda_k^\star = \lambda_1 p_k + \lambda_0(1 - p_k)$
    3. M-step:
           (a) Update $\theta^{(t)}$ according to (21)
           (b) For $i = 1, \ldots, n$, update $\boldsymbol{\eta}_i^{(t)}$ according to (22)
           (c) Update $\boldsymbol{\gamma}^{(t)}$ by solving (23)
           (d) Update $\boldsymbol{\Omega}^{(t)}$ according to (24)
           (e) Update $\sigma^{2(t)}$ according to (25)
    4. Set $\mathsf{diff} = \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^{(t-1)}\|_2^2 / \|\boldsymbol{\gamma}^{(t-1)}\|_2^2$
return $\widehat{\beta}_k(\boldsymbol{t}) = \sum_{l=1}^d \widehat{\gamma}_{kl} B_{kl}(\boldsymbol{t})$, $k = 1, \ldots, p$
```

The complete algorithm for the NVC-SSL model is given in Algorithm 1. Convergence can be assessed using the criterion $\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^{(t-1)}\|_2^2 / \|\boldsymbol{\gamma}^{(t-1)}\|_2^2 < 10^{-6}$. Let $\boldsymbol{t} = (t_{11}, \ldots, t_{1m_1}, \ldots, t_{n1}, \ldots, t_{nm_n})^\top$ be the vector of all observation times for all subjects. Once we have obtained the final MAP estimate $\widehat{\boldsymbol{\gamma}}$, we can estimate the varying coefficients as $\widehat{\beta}_k(\boldsymbol{t}) = \sum_{l=1}^d \widehat{\gamma}_{kl} B_{kl}(\boldsymbol{t}), k = 1, \ldots, p$, where $\widehat{\beta}_k(\boldsymbol{t})$ is the $N \times 1$ vector of $\widehat{\beta}_k$ evaluated at all N time points in $\boldsymbol{t}$.

Since the ECM algorithm has the ascent property, our algorithm is guaranteed to converge to a local mode. However, the NVC-SSL log-posterior (16) is a nonconvex function of its parameters, and hence, Algorithm 1 is not guaranteed to converge to the global mode. Nevertheless, we have not found local convergence to be a practical problem. The (local) MAP estimate under the NVC-SSL model performs very well in practice, as demonstrated in Sections 5 and 6. In addition, the ECM algorithm is also relatively fast. When there are $dp = 40{,}000$ parameters in $\boldsymbol{\gamma}$, the ECM algorithm takes 44 seconds on average to finish running for a well-chosen spike hyperparameter λ_0 (see Figure 5).

An alternative way to perform variable selection is to fit NVC-SSL with MCMC and to use the MCMC samples to estimate the posterior inclusion probabilities $P(\tau_k = 1 \mid \boldsymbol{Y}), k = 1, \ldots, p$. Selection can then be performed by thresholding these probabilities. For example, we can use the median probability model (MPM) (Barbieri and Berger, 2004; Barbieri et al.,

2021) and select $\beta_k(t)$ if $P(\tau_k = 1 \mid \boldsymbol{Y}) \geq 0.5$. In Appendix A, we explore the use of MPM for selection. We found that the MPM approach was inferior to using the MAP estimator for variable selection, and the MAP estimator was better able to detect weak signals. Thus, if *variable selection* is a primary objective of the data analyst, we recommend using the ECM algorithm presented in this section to select the varying coefficient functions. If inference is also desirable, then the analyst can additionally use the MCMC algorithms in Section 4 to obtain uncertainty intervals.

The ECM algorithm that we introduced in this section is based specifically on a beta-Bernoulli hierarchical construction (17) for the $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$ prior (12), and therefore, it is not applicable to other NVC models. On the other hand, if one is able to obtain good estimates for $(\boldsymbol{\Omega}, \sigma^2)$ in (10), then it may be possible to use the profile likelihood approach of Fan and Li (2012) to perform (non-Bayesian) variable selection under a penalized regression framework for (10). When there is *no* sparsity in the $\beta_k(t)$'s, restricted maximum likelihood (REML) can be used to estimate $(\boldsymbol{\Omega}, \sigma^2)$ (Guo, 2002). However, under sparsity and/or high dimensions, it is not as straightforward to estimate variance components in the penalized regression framework (Reid et al., 2016). In contrast, estimating $(\boldsymbol{\Omega}, \sigma^2)$ is fairly simple under a fully Bayesian framework, where we can endow these parameters with appropriate priors (14)-(15) and use our ECM algorithm to estimate them.

3.2 Computational complexity

Recall that m_i is the number of repeated measurements for subject i, and for n subjects, there are a total of $N = \sum_{i=1}^{n} m_i$ observations. With p varying coefficients, each represented by a basis expansion (3) with d basis functions, the number of unknown parameters in $\boldsymbol{\gamma}$ is dp. Meanwhile, we have nq unknown parameters in $\boldsymbol{\eta}$ and q^2 unknown parameters in $\boldsymbol{\Omega}$, where q is the number of basis functions in (4). The computational cost of performing the E-step is $\mathcal{O}(d^2p)$ operations, which arises from computing p ℓ_2-norms $\|\boldsymbol{\gamma}_k\|_2, k = 1, \ldots, p$, in $\boldsymbol{\Psi}(\boldsymbol{\gamma}_k \mid \lambda_1) \propto \exp(-\lambda_1 \|\boldsymbol{\gamma}_k\|_2)$ and $\boldsymbol{\Psi}(\boldsymbol{\gamma}_k \mid \lambda_0) \propto \exp(-\lambda_0 \|\boldsymbol{\gamma}_k\|_2)$ to obtain the $p_k^{\star}$'s in (18). In the M-step, the costs of updating θ, $\boldsymbol{\eta}$, $\boldsymbol{\gamma}$, $\boldsymbol{\Omega}$, and σ^2 are respectively $\mathcal{O}(p)$, $\mathcal{O}(Ndp + Nq^2 + q^3)$, $\mathcal{O}(Ndpr)$, $\mathcal{O}(nq^2)$, and $\mathcal{O}(Ndp + Nnq + N^2)$, where r is the number of iterations it takes for the group coordinate ascent algorithm of Breheny and Huang (2015) to converge.

Assuming that $\max\{d, q\} \ll n$ and $dp > N$, the most expensive step in the ECM algorithm is therefore solving for $\boldsymbol{\gamma}$ in (23). This gives Algorithm 1 an overall computational complexity of $\mathcal{O}(Ndpr)$. However, as discussed in Section 3.1, r (i.e. the number of iterations it takes for the group coordinate ascent algorithm to solve (23)) tends to be small provided that λ_0 is sufficiently large. Thus, our ECM algorithm not only scales linearly in both N and p, but it is also quite efficient in practice. We verify the speed and efficiency of our ECM algorithm in Section 5.3.

3.3 Choice of hyperparameters

We now provide our recommendations for setting the hyperparameters in the NVC-SSL prior (12)-(15). We fix the slab hyperparameter λ_1 in the SSGL prior (12) to be $\lambda_1 = 1$. This allows the slab density $\boldsymbol{\Psi}(\cdot \mid \lambda_1)$ to be fairly diffuse so that it is able to prevent overshrinkage of important covariates. We also set the shape parameters in the prior (13) on the mixing proportion θ to be $a = 1$ and $b = p$. This ensures that θ is small with high probability, and therefore, most of the $\boldsymbol{\gamma}_k$'s will belong to the spike density $\boldsymbol{\Psi}(\cdot \mid \lambda_0)$ in the SSGL prior (12). Finally, to ensure that the priors on the variance parameters $\boldsymbol{\Omega}$ and σ^2 are weakly informative, we set $\nu = q + 2$ and $\boldsymbol{\Phi} = \boldsymbol{I}_q$ in (14) and $c_0 = 1$ and $d_0 = 1$ in (15).

The spike hyperparameter λ_0 in the SSGL prior (12) plays the role of a regularization parameter on $\boldsymbol{\gamma}$, with larger values of λ_0 leading to more basis coefficients being thresholded to zero. Hence, the practical performance of NVC-SSL is governed heavily by the choice of λ_0. While one could fix λ_0 *a priori* to a positive value where $\lambda_0 \gg \lambda_1$, the speed of the ECM algorithm that we introduced in Section 3.1 makes it computationally feasible to determine a more optimal choice of λ_0 from a set of candidate values.

To this end, we fit NVC-SSL using several choices of λ_0 from a grid of L decreasing λ_0 values $\lambda_0^1 > \lambda_0^2 > \ldots > \lambda_0^L$. Following Wei et al. (2011), we choose $\lambda_0^l, 1 \leq l \leq L$, using the Bayesian information criterion (BIC) of Schwarz (1978). We have from (10) that the marginal likelihood of $\boldsymbol{Y}$ is $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{U}\boldsymbol{\gamma}, \boldsymbol{Z}(\boldsymbol{I}_n \otimes \boldsymbol{\Omega})\boldsymbol{Z}^\top + \sigma^2 \boldsymbol{I}_N)$. Let $\ell(\boldsymbol{\gamma}, \boldsymbol{\Omega}, \sigma^2)$ denote the log-marginal likelihood of $\boldsymbol{Y}$. For a given λ_0, the BIC in our present context is

$$\text{BIC}(\lambda_0) = -2\ \ell(\widehat{\boldsymbol{\gamma}}_{\lambda_0}, \widehat{\boldsymbol{\Omega}}_{\lambda_0}, \widehat{\sigma}^2_{\lambda_0}) + \log N \times \#\text{ of nonzero elements in } \widehat{\boldsymbol{\gamma}}_{\lambda_0}, \tag{26}$$

where $(\widehat{\boldsymbol{\gamma}}_{\lambda_0}, \widehat{\boldsymbol{\Omega}}_{\lambda_0}, \widehat{\sigma}^2_{\lambda_0})$ are the MAP estimates for $(\boldsymbol{\gamma}, \boldsymbol{\Omega}, \sigma^2)$ with λ_0 as the spike hyperparameter in (12). We select the $\lambda_0 \in \{\lambda_0^1, \ldots, \lambda_0^L\}$ which minimizes the BIC (26). An alternative to minimizing BIC is to choose λ_0 using cross-validation (CV). However, CV requires refitting the model KL times, where K is the number of folds. In contrast, BIC only requires solving L optimization problems, one for each λ_0 in the grid. Thus, using CV is roughly K times slower than minimizing BIC.

In order to accelerate the computation of the L optimizations for $\lambda_0 \in \{\lambda_0^1, \ldots, \lambda_0^L\}$, we employ a warm starting strategy, where for each $\lambda_0^l, 2 \leq l \leq L$, we initialize the ECM algorithm with $\boldsymbol{\gamma}^{(0)} = \widehat{\boldsymbol{\gamma}}(\lambda_0^{l-1})$, where $\widehat{\boldsymbol{\gamma}}(\lambda_0^{l-1})$ denotes the MAP estimator obtained from fitting NVC-SSL with the previous λ_0 in the grid. Since $\widehat{\boldsymbol{\gamma}}(\lambda_0^{l-1})$ serves as a reasonable initialization for $\boldsymbol{\gamma}^{(0)}$, the ECM algorithm for each λ_0^l converges very quickly to a local mode. As shown in Section 5.3, the ECM algorithm typically converges in 10 or fewer iterations. In all of our numerical experiments and real data applications, we found that tuning λ_0 from the equispaced grid $\{300, 290, \ldots, 20, 10\}$ worked well in practice.

Finally, using B-splines as the basis functions in (3)-(4), we determined that it is sufficient to fix the basis dimensions to be $d = q = 8$. While it is possible to further tune these values (for instance, we could use the BIC (26) to select (λ_0, d, q) from a grid of triplets), we found that increasing d and q to be greater than eight offered little to no benefits in terms of improved estimation or variable selection. Thus, we recommend only tuning the spike hyperparameter λ_0, while keeping all other hyperparameters in the priors (12)-(15) and the basis dimensions in the model (5) fixed at the default values suggested in this section.

4. MCMC for scalable uncertainty quantification

Apart from the ease with which one can incorporate unknown within-subject covariances through appropriate prior distributions, another advantage of Bayesian NVC models over penalized frequentist NVC models is their ability to provide natural uncertainty quantification through their posterior distributions. However, posterior sampling can be very challenging if p is large. In this section, we demonstrate how NVC-SSL is amenable to fast posterior sampling for uncertainty quantification.

We emphasize that the algorithms in this section are intended to be used in cases where $p \gg N$. If p is small and N is very large, then it is more advisable to use the scalable approaches introduced in Guhaniyogi et al. (2022) and Guhaniyogi et al. (2023) instead. In Section 7, we discuss some avenues for future work where *both* N and p could be very large.

4.1 Exact Gibbs sampling algorithm

We first introduce an exact Gibbs sampling algorithm for fitting the NVC-SSL model. In order to obtain closed form updates in the Gibbs sampler, it will be convenient to reparameterize the $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$ prior (17) as a Gaussian scale mixture density. First, note that for $\boldsymbol{\gamma}_k \sim \boldsymbol{\Psi}(\boldsymbol{\gamma}_k \mid \lambda) \propto \lambda^d \exp(-\lambda \|\boldsymbol{\gamma}_k\|_2)$, $\boldsymbol{\gamma}_k$ is the marginal density of the scale mixture,

$$\boldsymbol{\gamma}_k \mid \xi_k \sim \mathcal{N}(\mathbf{0}, \xi_k \boldsymbol{I}_d), \quad \xi_k \sim \text{Gamma}\left(\frac{d+1}{2}, \frac{\lambda^2}{2}\right).$$

Consequently, for $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_p)^\top$, we can rewrite the prior (17) as the hierarchical model,

$$\begin{aligned}
&\boldsymbol{\gamma} \mid \boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{D}_{\boldsymbol{\xi}}), && \text{where } \boldsymbol{D}_{\boldsymbol{\xi}} = \text{Bdiag}(\xi_1 \boldsymbol{I}_d, \ldots, \xi_p \boldsymbol{I}_d),\\
&\xi_k \mid \tau_k \sim \text{Gamma}\left(\tfrac{d+1}{2}, \tfrac{(\lambda_k^\star)^2}{2}\right), && \text{where } \lambda_k^\star = \tau_k \lambda_1 + (1-\tau_k)\lambda_0,\\
&\tau_k \mid \theta \sim \text{Bernoulli}(\theta), && k = 1, \ldots, p,
\end{aligned} \tag{27}$$

and Bdiag denotes a block-diagonal matrix. With the likelihood function for (10), the hierarchical priors in (27) for $\boldsymbol{\gamma}$, and the priors (14)-(15) on $(\boldsymbol{\Omega}, \sigma^2)$, we obtain as the joint posterior for all parameters,

$$\begin{aligned}
\pi(\boldsymbol{\gamma}, \boldsymbol{\xi}, \boldsymbol{\tau}, \theta, \boldsymbol{\eta}, \boldsymbol{\Omega}, \sigma^2 \mid \boldsymbol{Y}) \propto{}& (\sigma^2)^{-N/2} \exp\left(-\frac{\|\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{\gamma} - \boldsymbol{Z}\boldsymbol{\eta}\|_2^2}{2\sigma^2}\right)\\
&\times (\det(\boldsymbol{\Omega}))^{-1/2} \exp\left(-\frac{\boldsymbol{\eta}^\top \boldsymbol{\Omega}^{-1} \boldsymbol{\eta}}{2}\right)\\
&\times \pi(\boldsymbol{\gamma} \mid \boldsymbol{\xi}) \times \pi(\boldsymbol{\xi} \mid \boldsymbol{\tau}) \times \pi(\boldsymbol{\tau} \mid \theta) \times \pi(\boldsymbol{\Omega}) \times \pi(\sigma^2),
\end{aligned} \tag{28}$$

where the terms in the last line of the display denote the prior densities for the respective parameters in (27), (14), and (15). Based on (28), we can derive an exact Gibbs sampling algorithm where all conditional distributions are available in closed form. This MCMC algorithm is given in Algorithm 2.

As shown in Algorithm 2, the main computational bottleneck in the exact Gibbs sampler is sampling the basis coefficients $\boldsymbol{\gamma} \in \mathbb{R}^{dp}$ in Step 6, which overwhelms the cost of sampling from any of the other parameters when $dp > N$. Note that sampling from the $\boldsymbol{\eta}_i$ vectors and $\boldsymbol{\Omega}$ is not expensive, since we only need to sample q-dimensional vectors and a $q \times q$ matrix respectively in this case. Typically, q (the number of basis functions) is not overwhelming large. On the other hand, if the number of covariates p is large, then it can be computationally demanding to sample a dp-dimensional random Gaussian vector. Typical methods based on Cholesky decomposition require $\mathcal{O}(d^3p^3)$ operations to compute the Cholesky decomposition for the covariance matrix $\boldsymbol{\Sigma}_\gamma$ (Bhattacharya et al., 2016).

Algorithm 2 MCMC algorithm for NVC-SSL

```
Input: Initial values $\boldsymbol{\gamma}^{(0)}$, $\theta^{(0)}$, $\boldsymbol{\Omega}^{(0)}$, $\boldsymbol{\xi}^{(0)}$, $\sigma^{2(0)}$, $T$ (number of MCMC samples to run),
       $B$ (number of samples to discard as burn-in)
Output: MCMC samples for varying coefficients $\beta_k(\boldsymbol{t}),\ k = 1, \ldots, p$

for $t = 1, \ldots, T$ do
  1. for $i = 1, \ldots, n$ do
       Sample $\boldsymbol{\eta}_i^{(t)} \sim \mathcal{N}(\boldsymbol{\zeta}_i,\ \boldsymbol{\Xi}_i)$, where $\boldsymbol{\Xi}_i = \left\{ \boldsymbol{Z}_i^\top \boldsymbol{Z}_i / \sigma^{2(t-1)} + (\boldsymbol{\Omega}^{(t-1)})^{-1} \right\}^{-1}$ and
       $\boldsymbol{\zeta}_i = \boldsymbol{\Xi}_i \boldsymbol{Z}_i^\top (\boldsymbol{Y}_i - \boldsymbol{U}_i \boldsymbol{\gamma}^{(t-1)}) / \sigma^{2(t-1)}$
  2. Sample $\boldsymbol{\Omega}^{(t)} \sim$ Inverse-Wishart $\left( \nu + n,\ \boldsymbol{\Phi} + \sum_{i=1}^n \boldsymbol{\eta}_i^{(t)} (\boldsymbol{\eta}_i^{(t)})^\top \right)$
  3. Sample $\sigma^{2(t)} \sim$ Inverse-Gamma $\left( \frac{N+c_0}{2},\ \frac{\| \boldsymbol{Y} - \boldsymbol{U}\boldsymbol{\gamma}^{(t-1)} - \boldsymbol{Z}\boldsymbol{\eta}^{(t)} \|_2^2 + d_0}{2} \right)$
  4. for $k = 1, \ldots, p$ do
     (a) Sample $\tau_k^{(t)} \sim$ Bernoulli $\left( \frac{\pi_1}{\pi_1 + \pi_0} \right)$, where $\pi_1 = \theta^{(t-1)} \lambda_1^{d+1} \exp\{ -\lambda_1^2 \xi_k^{(t-1)} / 2 \}$ and
         $\pi_0 = (1 - \theta^{(t-1)}) \lambda_0^{d+1} \exp\{ -\lambda_0^2 \xi_k^{(t-1)} / 2 \}$
     (b) Sample $\xi_k^{(t)} \sim$ Generalized-Inverse-Gaussian $\left( \frac{1}{2}, \| \boldsymbol{\gamma}_k^{(t-1)} \|_2^2, (\lambda_k^\star)^2 \right)$, where
         $\lambda_k^\star = \tau_k^{(t)} \lambda_1 + (1 - \tau_k^{(t)}) \lambda_0$
  5. Sample $\theta^{(t)} \sim$ Beta $\left( a + \sum_{k=1}^p \tau_k^{(t)},\ b + p - \sum_{k=1}^p \tau_k^{(t)} \right)$
  6. Sample $\boldsymbol{\gamma}^{(t)} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\gamma}}, \boldsymbol{\Sigma}_{\boldsymbol{\gamma}})$, where $\boldsymbol{\mu}_{\boldsymbol{\gamma}} = \boldsymbol{\Sigma}_{\boldsymbol{\gamma}} \boldsymbol{U}^\top \left( \boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\eta}^{(t)} \right) / \sigma^{2(t)}$,
     $\boldsymbol{\Sigma}_{\boldsymbol{\gamma}} = \{ \boldsymbol{U}^\top \boldsymbol{U} / \sigma^{2(t)} + \boldsymbol{D}_{\boldsymbol{\xi}}^{-1} \}^{-1}$, and $\boldsymbol{D}_{\boldsymbol{\xi}} = \text{Bdiag}\ (\xi_1^{(t)} \boldsymbol{I}_d, \ldots, \xi_p^{(t)} \boldsymbol{I}_d)$
return $\beta_k^{(B+1)}(\boldsymbol{t}) = \sum_{l=1}^d \gamma_{kl}^{(B+1)} B_{kl}(\boldsymbol{t}), \ldots, \widehat{\beta}_k^{(T)}(\boldsymbol{t}) = \sum_{l=1}^d \gamma_{kl}^{(T)} B_{kl}(\boldsymbol{t})$ for $k = 1, \ldots, p$
```

In order to alleviate the cost of directly sampling from $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\gamma}}, \boldsymbol{\Sigma}_{\boldsymbol{\gamma}})$ in Step 6 of Algorithm 2 when $dp > N$, we can employ the fast sampling method of Bhattacharya et al. (2016). The algorithm of Bhattacharya et al. (2016) indirectly samples from $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\gamma}}, \boldsymbol{\Sigma}_{\boldsymbol{\gamma}})$ by solving a system of linear equations with N equations and is much more efficient than methods based on Cholesky decomposition. This algorithm for sampling $\boldsymbol{\gamma}$ is given in Algorithm 3.

The computational complexity of Algorithm 3 is $\mathcal{O}(N^2 dp)$ when $dp > N$. Here, the main bottleneck is computing the matrix product $\boldsymbol{U}\boldsymbol{D}_{\boldsymbol{\xi}}\boldsymbol{U}^\top / \sigma^2$ in Step 3 which requires $\mathcal{O}(N^2 dp)$ operations and is more expensive than even inverting the matrix $\boldsymbol{K} = \boldsymbol{U}\boldsymbol{D}_{\boldsymbol{\xi}}\boldsymbol{U}^\top / \sigma^2 + \boldsymbol{I}_N$ in Step 4, which requires $\mathcal{O}(N^3)$ operations.

Using Algorithm 3 to sample from $\boldsymbol{\gamma}$ enables NVC-SSL to scale linearly in p per MCMC iteration rather than cubically. However, the fact that this exact algorithm scales quadratically in terms of total sample size N may still be problematic. In particular, the multi-

Algorithm 3 Exact algorithm for sampling $\boldsymbol{\gamma}$ in Step 6 of Algorithm 2 when $dp > n$

```
Input: Most recent MCMC samples of $\boldsymbol{\xi}$, $\boldsymbol{\eta}$, and $\sigma$
Output: An exact sample of $\boldsymbol{\gamma}$ from Step 6 of Algorithm 2

  1. Sample $\boldsymbol{m} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{D}_{\boldsymbol{\xi}})$ and $\boldsymbol{\delta} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_N)$ independently
  2. Set $\boldsymbol{v} = (\boldsymbol{U}/\sigma)\boldsymbol{m} + \boldsymbol{\delta}$ and $\boldsymbol{v}^{\star} = (\boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\eta})/\sigma - \boldsymbol{v}$
  3. Set $\boldsymbol{K} = \boldsymbol{U}\boldsymbol{D}_{\boldsymbol{\xi}}\boldsymbol{U}^{\top}/\sigma^2 + \boldsymbol{I}_N$
  4. Set $\boldsymbol{w} = \boldsymbol{K}^{-1}\boldsymbol{v}^{\star}$
  5. Set $\boldsymbol{\gamma} = \boldsymbol{m} + \boldsymbol{D}_{\boldsymbol{\xi}}\boldsymbol{U}^{\top}\boldsymbol{w}/\sigma$

return $\boldsymbol{\gamma}$
```

plicative factor of N^2 in $\mathcal{O}(N^2dp)$ might still render it costly to run the exact NVC-SSL Gibbs sampler if p is large. For many large-scale problems, it is desirable to sample from the marginal posteriors in *linear* time with respect to sample size N. This motivates us to develop an approximate MCMC algorithm in the next section that has a computational complexity of $\mathcal{O}(Ndp)$ per iteration.

4.2 Approximate Gibbs sampling algorithm and its computational complexity

Before introducing our approximate MCMC algorithm, we first review several other recently proposed approaches for sampling from spike-and-slab models that also scale linearly in both the number of covariates and the sample size. Biswas et al. (2022) proposed an exact sampling method that has order $\max\{N^2p_t, Np\}$, where p_t is the number of covariates that switch between the spike and the slab states between iterations $t-1$ and t. Typically, p_t is much smaller than p, potentially offering substantial speed-ups over Algorithm 3. However, the algorithm of Biswas et al. (2022) *requires* both the spike and slab densities to be Gaussian and *cannot* be used for Laplace densities (or other scale-mixture densities) like the ones employed by $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$. For NVC-SSL, we use multivariate Laplace densities in the prior (12) so that the MAP estimator is *exactly* sparse (a feature that is not the case for Gaussian spike-and-slab priors). This rules out the approach of Biswas et al. (2022).

In another line of work, Narisetty et al. (2019) proposed the skinny Gibbs algorithm. In each iteration of skinny Gibbs, the components of the regression coefficients vector $\boldsymbol{\gamma}$ are partitioned as $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_S^{\top}, \boldsymbol{\gamma}_{S^c}^{\top})^{\top}$, where S denotes the set of variables belonging to the slab density and S^c denotes the set of those belonging to the spike density. Suppose that the set S is of size s. Skinny Gibbs decreases the cost of sampling $\boldsymbol{\gamma}$ to $\mathcal{O}(Np)$ by "sparsifying" the covariance matrix for the conditional distribution of $\boldsymbol{\gamma}$. That is, the cross-covariances in the conditional distribution of $\boldsymbol{\gamma}$ are set to be zero, i.e. $\text{cov}(\boldsymbol{\gamma}_S, \boldsymbol{\gamma}_{S^c}) = \mathbf{0}$, and the off-diagonal entries of $\text{cov}(\boldsymbol{\gamma}_{S^c})$ are also set to be zero. Thus, skinny Gibbs independently samples $\boldsymbol{\gamma}_S$ from an s-dimensional multivariate Gaussian distribution and then independently samples the $p-s$ the entries in $\boldsymbol{\gamma}_{S^c}$ from univariate Gaussian distributions.

While the skinny Gibbs algorithm has a computational complexity that is linear in both N and p, ignoring the correlations between $\boldsymbol{\gamma}_S$ and $\boldsymbol{\gamma}_{S^c}$ changes the posterior in a very nontrivial way. In particular, Theorem 1 of Narisetty et al. (2019) shows that the conditional distribution of $\boldsymbol{\gamma}$ under skinny Gibbs is separable in $\boldsymbol{\gamma}_S$ and $\boldsymbol{\gamma}_{S^c}$ (i.e. it can be written as a product of two functions $f_1(\boldsymbol{\gamma}_S)$ and $f_2(\boldsymbol{\gamma}_{S^c})$), which is not the case for the conditional distribution in Step 6 of Algorithm 2. In the present context, we want to use MCMC to conduct *inference* rather than variable selection. Therefore, we aim to approximate the transition kernel of the NVC-SSL Markov chain so that we *preserve* the correlations between the active set and the inactive set in the conditional distribution of $\boldsymbol{\gamma}$.

To achieve an order N speed-up, we adopt a similar idea as Johndrow et al. (2020) who devised an approximate MCMC algorithm for the horseshoe prior (Carvalho et al., 2010) in univariate Gaussian regression. However, unlike spike-and-slab priors, the horseshoe prior does *not* naturally partition the covariates into "significant" and "insignificant" groups. Consequently, Johndrow et al. (2020) require artificially segregating the covariates into two groups by using a user-specified threshold $\delta > 0$ to determine significant groups (i.e. regression coefficients with magnitude larger than δ are deemed to be "significant"). In practice, it can be difficult to specify an appropriate threshold for δ. In contrast, the binary indicators $\boldsymbol{\tau}$ in our spike-and-slab prior (17) naturally partition the groups of basis coefficients $\boldsymbol{\gamma}_k$'s in the (10) as either belonging to the spike or the slab. The automatic partitioning scheme allows us to construct a suitable approximate MCMC algorithm.

First, for the indicator variables $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_p)^\top$ in (27), define the set $S = \{k : \tau_k = 1\}$, i.e. S is the set of indices of the varying coefficients belonging to the slab density in (17). Suppose that the cardinality of S is $|S| = s$. Then $S^c = \{1, \ldots, p\} \setminus S$ denotes the indices of the varying coefficients belonging to the spike density in (17), and $|S^c| = p - s$. Let $\boldsymbol{U}_S$ denote the $N \times ds$ submatrix of $\boldsymbol{U}$ in (7) whose columns correspond to S, i.e. for each $k \in S$, $\boldsymbol{U}_S$ contains the d columns of $\boldsymbol{U}$ that correspond to the kth varying coefficient. Similarly, let $\boldsymbol{D}_S = \text{Bdiag}\{\xi_k \boldsymbol{I}_d\}_{k \in S}$ denote the $ds \times ds$ block-diagonal submatrix of $\boldsymbol{D}_{\boldsymbol{\xi}}$ in (27) whose diagonal blocks correspond to the s indices of S.

When $\tau_k = 0$ in (27), $\lambda_k^\star \approx \lambda_0$ and $\|\boldsymbol{\gamma}_k\|_2 \approx 0$ in step 4(b) of Algorithm 2. Hence, $\xi_k \approx 0$ for $k \in S^c$. As a result, the $N \times N$ matrix product $\boldsymbol{U}\boldsymbol{D}_{\boldsymbol{\xi}}\boldsymbol{U}^\top$ can be well-approximated by $\boldsymbol{U}_S\boldsymbol{D}_S\boldsymbol{U}_S^\top$ (Johndrow et al., 2020). Likewise, if we let $\widetilde{\boldsymbol{D}}_{\boldsymbol{\xi}} = \text{Bdiag}\{\xi_k \boldsymbol{I}_d \mathbb{I}(\tau_k = 1)\}$ denote the block-diagonal matrix where we replace the $\xi_k, k \in S$, in $\boldsymbol{D}_{\boldsymbol{\xi}}$ with zero, then the matrix product $\boldsymbol{D}_{\boldsymbol{\xi}}\boldsymbol{U}^\top$ is well-approximated by $\widetilde{\boldsymbol{D}}_{\boldsymbol{\xi}}\boldsymbol{U}^\top$. This suggests that we can make the following replacements in Algorithm 3 to obtain an *approximate* MCMC algorithm:

- We can approximate $\boldsymbol{K}$ in Step 3 with $\widetilde{\boldsymbol{K}}$, where $\widetilde{\boldsymbol{K}} = \boldsymbol{U}_S\boldsymbol{D}_S\boldsymbol{U}_S^\top/\sigma^2 + \boldsymbol{I}_N$.
- We can approximate $\boldsymbol{K}^{-1}$ in Step 4 with $\widetilde{\boldsymbol{K}}^{-1}$.
- We can approximate $\boldsymbol{D}_{\boldsymbol{\xi}}$ in Step 5 with $\widetilde{\boldsymbol{D}}_{\boldsymbol{\xi}}$.

In particular, using the Woodbury matrix identity, we have that

$$\widetilde{\boldsymbol{K}}^{-1} = \boldsymbol{I}_N - \boldsymbol{U}_S\left(\boldsymbol{U}_S^\top\boldsymbol{U}_S/\sigma^2 + \boldsymbol{D}_S^{-1}\right)^{-1}\boldsymbol{U}_S^\top/\sigma^2, \tag{29}$$

i.e. computing the inverse of $\widetilde{\boldsymbol{K}}$ requires inverting only an $ds \times ds$ matrix now instead of an $N \times N$ matrix. Under sparsity and a relatively small basis dimension d, we typically have

```
**Algorithm 4** Approximate algorithm for sampling $\boldsymbol{\gamma}$ in Step 6 of Algorithm 2 when $dp > n$
**Input:** Most recent MCMC samples of $\boldsymbol{\xi}$, $\boldsymbol{\eta}$, $\boldsymbol{\tau}$ and $\sigma$
**Output:** An approximate sample of $\boldsymbol{\gamma}$ from Step 6 of Algorithm 2

    1. Sample $\boldsymbol{m} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{D}_{\boldsymbol{\xi}})$ and $\boldsymbol{\delta} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_N)$ independently
    2. Set $\boldsymbol{v} = (\boldsymbol{U}/\sigma)\boldsymbol{m} + \boldsymbol{\delta}$ and $\boldsymbol{v}^\star = (\boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\eta})/\sigma - \boldsymbol{v}$
    3. Set $\widetilde{\boldsymbol{K}} = \boldsymbol{I}_N - \boldsymbol{U}_S(\boldsymbol{U}_S^\top \boldsymbol{U}_S/\sigma^2 + \boldsymbol{D}_S^{-1})^{-1}\boldsymbol{U}_S^\top/\sigma^2$
    4. Set $\boldsymbol{w} = \widetilde{\boldsymbol{K}}^{-1}\boldsymbol{v}^\star$, where $\widetilde{\boldsymbol{K}}^{-1}$ is computed as in (29)
    5. Set $\boldsymbol{\gamma} = \boldsymbol{m} + \widetilde{\boldsymbol{D}}_{\boldsymbol{\xi}}\boldsymbol{U}^\top\boldsymbol{w}/\sigma$, where $\widetilde{\boldsymbol{D}}_{\boldsymbol{\xi}} = \text{Bdiag}\{\xi_k \boldsymbol{I}_d \mathbb{I}(\tau_k = 1)\}$

**return** $\boldsymbol{\gamma}$
```

that $ds \ll N$. In addition, the matrix multiplication $\widetilde{\boldsymbol{D}}_{\boldsymbol{\xi}}\boldsymbol{U}^\top \mathbf{w}/\sigma$ costs $\mathcal{O}(Nds)$ operations, instead of the $\mathcal{O}(Ndp)$ operations that are required to compute the product $\boldsymbol{D}_{\boldsymbol{\xi}}\boldsymbol{U}^\top\boldsymbol{w}/\sigma$ in Algorithm 3.

The complete approximate MCMC algorithm for approximately sampling from $\boldsymbol{\gamma}$ is given in Algorithm 4. By examining Algorithm 4, we see that the most expensive operation is now computing the matrix-vector product $(\boldsymbol{U}/\sigma)\boldsymbol{m}$ in Step 2, which requires $\mathcal{O}(Ndp)$ operations. If dp is very large, then $\mathcal{O}(Ndp)$ represents a *substantial* cost reduction from the $\mathcal{O}(N^2dp)$ cost of the exact sampling scheme in Algorithm 3.

In short, Algorithm 4 allows us to *approximately* sample from the basis coefficients $\boldsymbol{\gamma}$ in the NVC-SSL model with a runtime per MCMC iteration that is linear in *both* the number of covariates p *and* the total sample size N. The approximate MCMC algorithm has an even faster per iteration runtime than the ECM algorithm introduced in Section 3.1, which has time complexity of $\mathcal{O}(Ndpr)$, $r > 1$, per iteration. However, the ECM algorithm typically converges after a few iterations, whereas we may need to run MCMC for a much larger number of iterations to obtain enough posterior samples for good inference.

4.3 Trade-offs between exact and approximate Gibbs sampling algorithms

Algorithm 4 reduces the per iteration cost of posterior sampling by a factor of N. In Section 5.3, we demonstrate that as dp increases, this offers a very significant reduction in MCMC runtime. However, the trade-off for faster computation is that the pointwise uncertainty intervals for the varying coefficients are slightly more conservative. This finding stands in contrast to that of Johndrow et al. (2020) who claimed that inference from their approximate MCMC algorithm for the horseshoe prior was "virtually indistinguishable" from inference under the exact MCMC algorithm. In this section, we precisely quantify this trade-off.

Specifically, we investigate the implications of Algorithm 4 on the posterior mean and variance of the conditional distribution for $\boldsymbol{\gamma}$. Without loss of generality, assume that $S = \{1, \ldots, s\}$ and $S^c = \{s+1, \ldots, p\}$. Then we can partition the basis coefficients $\boldsymbol{\gamma}$ as $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_S^\top, \boldsymbol{\gamma}_{S^c}^\top)^\top$, where $\boldsymbol{\gamma}_S$ consists of the first s groups (ds entries) in $\boldsymbol{\gamma}$, while $\boldsymbol{\gamma}_{S^c}$ consists of the last $p-s$ groups ($d(p-s)$ entries) in $\boldsymbol{\gamma}$. Instead of exactly sampling from $\mathcal{N}(\boldsymbol{\mu}_{\gamma}, \boldsymbol{\Sigma}_{\boldsymbol{\gamma}})$ in Step 6 of Algorithm 2, the steps in Algorithm 4 amount to sampling $\boldsymbol{\gamma}$ from the conditional distribution $\mathcal{N}(\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\gamma}}, \widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}})$ (Johndrow et al., 2020), where

$$\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\gamma}} = \begin{pmatrix} \widetilde{\boldsymbol{\mu}}_{\boldsymbol{\gamma}_S} \\ \widetilde{\boldsymbol{\mu}}_{\boldsymbol{\gamma}_{S^c}} \end{pmatrix} = \begin{pmatrix} \left(\boldsymbol{U}_S^\top \boldsymbol{U}_S/\sigma^2 + \boldsymbol{D}_S^{-1}\right)^{-1} \boldsymbol{U}_S^\top (\boldsymbol{Y} - \boldsymbol{Z}\boldsymbol{\eta})/\sigma^2 \\ \mathbf{0}_{(dp-ds)\times 1} \end{pmatrix}, \tag{30}$$

and

$$\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}} = \begin{pmatrix} \widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S} & \widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S,S^c}} \\ \widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S,S^c}}^\top & \widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} \end{pmatrix} = \begin{pmatrix} \left(\boldsymbol{U}_S^\top \boldsymbol{U}_S/\sigma^2 + \boldsymbol{D}_S^{-1}\right)^{-1} & -\boldsymbol{D}_S \boldsymbol{U}_S^\top \widetilde{\boldsymbol{K}}^{-1} \boldsymbol{U}_{S^c} \boldsymbol{D}_{S^c}/\sigma^2 \\ -\boldsymbol{D}_{S^c} \boldsymbol{U}_{S^c}^\top \widetilde{\boldsymbol{K}}^{-1} \boldsymbol{U}_S \boldsymbol{D}_S/\sigma^2 & \boldsymbol{D}_{S^c} \end{pmatrix}. \tag{31}$$

We see from (30) that the marginal conditional distribution of $\boldsymbol{\gamma}_S$ has the same mean as the mean of the conditional distribution for $\boldsymbol{\gamma}$ if we were to fit NVC-SSL to *only* the first s varying coefficients. Meanwhile, the conditional mean for $\boldsymbol{\gamma}_{S^c}$ is a zero vector of length $d(p-s)$. Under sparsity, the exact MCMC algorithm will also result in MCMC samples for $\boldsymbol{\gamma}_{S^c}$ that are centered around a mean that is very close to zero. Based on these observations, we expect fairly negligible differences between the estimated posterior mean of $\boldsymbol{\gamma}$ under the exact and approximate MCMC algorithms. This is confirmed in our numerical experiments in Section 5.2.

Unlike the skinny Gibbs algorithm (Narisetty et al., 2019), we also observe from (31) that Algorithm 4 preserves the cross-correlations between $\boldsymbol{\gamma}_S$ and $\boldsymbol{\gamma}_{S^c}$, i.e. $\text{cov}(\boldsymbol{\gamma}_S, \boldsymbol{\gamma}_{S^c}) = \widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S,S^c}} \neq \mathbf{0}_{ds\times(dp-ds)}$. However, while there is negligible bias for the posterior *means*, the marginal posterior *variances* for $\boldsymbol{\gamma}$ under the approximate MCMC algorithm are slightly inflated. This is formalized in the next proposition.

Proposition 1 *Let $\boldsymbol{\Sigma}_{\boldsymbol{\gamma}_S}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}}$ denote the marginal covariance matrices for the conditional distributions of $\boldsymbol{\gamma}_S$ and $\boldsymbol{\gamma}_{S^c}$ respectively under the exact MCMC algorithm (Algorithm 3). Meanwhile, let $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S}$ and $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}}$ denote the marginal covariance matrices for the conditional distributions of $\boldsymbol{\gamma}_S$ and $\boldsymbol{\gamma}_{S^c}$ under the approximate MCMC algorithm (Algorithm 4), as defined in* (31). *Then $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S} \geq \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_S}$ and $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} \geq \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}}$.*

The proof of Proposition 1 is given in Appendix B, which also gives precise expressions for $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_S}$ and $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}}$. The implication of Proposition 1 is that the variances in the marginal posteriors for the entries in $\boldsymbol{\gamma}$ will tend to be *larger* under the approximate MCMC algorithm than under the exact MCMC algorithm. As a consequence, the pointwise posterior credible intervals for the varying coefficients $\beta_k(t)$ will also tend to be wider. To see this, suppose that, based on the posterior samples for $\boldsymbol{\gamma}$, we form the credible intervals $[\gamma_{kl}^L, \gamma_{kl}^U], k = 1, \ldots, p, l = 1, \ldots, d$ with a prescribed level of probability $1-\alpha, \alpha \in (0,1)$. Then the $(1-\alpha)\times 100\%$ posterior credible intervals for $\beta_k(t)$ will be $[\sum_{l=1}^d \gamma_{kl}^L B_{kl}(t), \sum_{l=1}^d \gamma_{kl}^U B_{kl}(t)]$. As a result of Proposition 1, the endpoints $[\gamma_{kl}^L, \gamma_{kl}^U]$ will tend to be further apart under Algorithm 4, leading to wider pointwise credible intervals for the varying coefficients $\beta_k(t)$'s than those under Algorithm (3).

Nevertheless, our approximate MCMC algorithm for NVC-SSL is still a practical choice if computational time is a primary concern. In Section 5.3, we show that when $dp = 40{,}000$, the approximate MCMC algorithm reduces the average runtime for 1000 MCMC iterations from 5.6 hours (for the exact algorithm) to 18.9 minutes. This is a substantial computational gain and suggests that Algorithm 4 is much more scalable for large p than Algorithm 3. Sparse and/or low-rank approximations like Algorithm 4 are also routinely employed in practice to improve the scalability and computational feasibility of fully Bayesian inference, at the expense of not performing exact inference. See, for example, the Gaussian process and spatial statistics literature (Rasmussen and Williams, 2006; Banerjee et al., 2013; Datta et al., 2016; Hughes and Haran, 2013).

Moreover, if one is interested in *simultaneous* coverage of the varying coefficient functions rather than pointwise coverage, then wider credible intervals are actually preferred. We show in Section 5.2 that the approximate MCMC algorithm has higher simultaneous coverage of the true varying coefficients than the exact MCMC algorithm. The approximate MCMC algorithm also manages to produce uncertainty intervals that capture the true shape of the varying coefficients, as shown in the right three panels of Figure 3.

5. Simulation studies

Here, we conduct simulation studies for NVC-SSL to validate its variable selection and estimation performance, inferential capabilities, and scalability. All of the methods in this section were implemented in the publicly available R package `NVCSSL`, which can be found on the Comprehensive R Archive Network.

5.1 Variable selection and estimation performance

We first assessed the performance of the NVC-SSL MAP estimator obtained from the ECM algorithm in Section 3. In particular, we investigated the MAP estimator's ability to:

1. capture different shapes for the varying coefficients, including nonzero but constant (i.e. non-time varying) functions;
2. detect weak signals, i.e. varying coefficients with small magnitudes for $\|\beta_k(t)\|_\infty$;
3. perform well under a variety of unknown within-subject covariance functions, including those that allow for long-range correlation and zero covariance functions (i.e. zero within-subject correlations).

We generated data for $n = 50$ subjects from model (2) as follows. To simulate the observation times t_{ij}'s, we first sampled from $\{1, 2, \ldots, 20\}$, where each time point has a 60 percent chance of being skipped. This way, we had very irregularly spaced data, with m_i being different for different subjects. We then added random perturbation from $\mathcal{U}(-0.5, 0.5)$ to the non-skipped time points.

To model the high-dimensional scenario, we set $p = 500$, with the first six variables $x_{i1}, \ldots, x_{i6}$ being the relevant ones. The first covariate x_{i1} was simulated from $\mathcal{U}(t/10, 2 + t/10)$ for any given time point t. The covariates $x_{ik}, k = 2, \ldots, 5$, conditioned on x_{i1}, were i.i.d. drawn from a normal distribution with mean zero and variance $(1 + x_{i1})/(2 + x_{i1})$. The covariate x_{i6}, independent of $x_{ik}, k = 1, \ldots, 5$, was normal with mean $1.5 \exp(t/40)$ and variance 1. Finally, for $k = 7, \ldots, 500$, each x_{ik}, independent of the others, was drawn from a multivariate normal distribution with covariance structure $\text{cov}(x_{ik}(t), x_{ik}(s)) = \rho^{-|t-s|}$, with $\rho = 0.5$. The true coefficient functions were

$$\beta_1(t) = 10\sin\left(\frac{\pi t}{15}\right), \quad \beta_2(t) = -0.6t + 6, \quad \beta_3(t) = -1 + 2\sin\left(\frac{\pi(t-25)}{8}\right),$$
$$\beta_4(t) = 1 + 2\cos\left(\frac{\pi(t-25)}{15}\right), \quad \beta_5(t) = 2 + \frac{10}{1+e^{10-t}}, \quad \beta_6(t) = -5,$$
$$\beta_7(t) = \cdots = \beta_p(t) = 0.$$

In particular, β_3 and β_4 are weak signals with small magnitudes for $\|\beta_k(t)\|_\infty$. Meanwhile, β_2 is a linear function, β_5 is a sigmoid curve with flat regions, and β_6 is a nonzero constant (non-time varying) function. For the measurement error term in (2), we fixed the noise variance $\sigma^2 = 1$. For the unknown within-subject covariances, we considered the following five experimental settings for the covariance function $k(t, t')$ in (2):

- Experiment 1: first-order autoregressive (AR(1)). We set $k_i(t_{ij}, t_{ij'}) = s_i^2 \rho_i^{|t_{ij} - t_{ij'}|}$, $1 \leq j, j' \leq m_i$.
- Experiment 2: compound symmetry (CS). We set $k_i(t_{ij}, t_{ij'}) = s_i^2\{\mathbb{1}(t_{ij} = t_{ij'}) + \rho_i \mathbb{1}(t_{ij} \neq t_{ij'})\}, 1 \leq j, j' \leq m_i$.
- Experiment 3: squared exponential (SE). We set $k_i(t_{ij}, t_{ij'}) = s_i^2 \exp(-(t_{ij} - t_{ij'})^2/\ell_i^2)$, $1 \leq j, j' \leq m_i$.
- Experiment 4: periodic. We set $k_i(t_{ij}, t_{ij'}) = s_i^2 \exp(-2[\sin^2(\pi|t_{ij} - t_{ij'}|/p_i)]/\ell_i^2)$, $1 \leq j, j' \leq m_i$.
- Experiment 5: zero (i.id. errors). We set $k_i(t_{ij}, t_{ij'}) = 0, 1 \leq j, j' \leq m_i$.

We sampled the within-subject variance hyperparameters $s_i^2 \in \{0.5, 0.75, 1, 1.25, 1.5\}$, the autocorrelation hyperparameters $\rho_i \in \{0.2, 0.4, 0.6, 0.8\}$, the lengthscale hyperparameters $\ell_i \in \{0.3, 0.6, 0.9, 1.2, 1.5, 1.8\}$, and the period hyperparameters $p_i \in \{0.5, 1, 1.5, 2\}$.

We briefly discuss our choices of covariance functions. The AR(1) and SE covariance functions in Experiments 1 and 3 embody the belief that correlations between two time points t and t' decreases exponentially as $|t - t'|$ increases. The behavior implied by these kernel functions is not appropriate when there could be strong correlations between far apart time points. In contrast, the CS covariance function in Experiment 2 implies that the correlation between t and t' is the *same* for all $t \neq t'$, making it suitable for capturing long-range correlations. The periodic covariance function in Experiment 4 exhibits periodic oscillation, meaning that as $|t - t'|$ increases, the correlation could be *either* weak *or* strong between t and t'. Finally, the zero covariance function in Experiment 5 simply means that there are no within-subject correlations, and we are operating under the assumption of i.i.d. errors.

To implement the NVC-SSL method, we tuned the spike hyperparameter λ_0 from the grid $\{300, 290, \ldots, 10\}$ and selected λ_0 using the BIC criterion (26). Meanwhile, we fixed all the other hyperparameters to the ones recommended in Section 3.3 and fixed the basis dimensions as $d = q = 8$ in (5). We compared our approach to the group lasso (gLASSO), group smoothly clipped absolute deviation (gSCAD), and group minimax concave penalty (gMCP) (Yuan and Lin, 2006; Breheny and Huang, 2015). For high-dimensional NVC models, these methods solve the following optimization problem:

$$\widehat{\boldsymbol{\gamma}} = \arg\max_{\boldsymbol{\gamma}} \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{\gamma}\|_2^2 + \sum_{k=1}^{p} pen_{\lambda}(\boldsymbol{\gamma}_k),$$

where $\boldsymbol{U}$ is defined as in (7) and $pen_{\lambda}(\cdot)$ is a penalty function that depends on a tuning parameter λ. These penalized approaches, which we refer to NVC-gLASSO, NVC-gSCAD, and NVC-gMCP respectively, have been considered by numerous authors in the varying coefficient literature (Wang and Xia, 2009; Wang et al., 2008; Wei et al., 2011). For these methods, we also fixed the basis dimension $d = 8$ and tuned λ using the BIC criterion.

While Wang et al. (2008), Wang et al. (2008), and Wei et al. (2011) demonstrated competitive performance of their methods even when failing to account for within-subject correlations, we wanted to see whether explicitly incorporating estimation of unknown within-subject correlations (as with the NVC-SSL model) could improve estimation and variable selection. Moreover, Experiment 5 (i.e. zero covariance function) was conducted in order to fairly compare NVC-SSL to NVC-gLASSO, NVC-gSCAD, and NVC-gMCP, since the setting of Experiment 5 is the exact NVC model implied by the latter three methods.

To compare these methods, we evaluated the estimation error, out-of-sample prediction error, and variable selection performance. For estimation error, we computed the mean squared error (MSE),

$$\text{MSE} = \frac{1}{Np} \sum_{k=1}^{p} \sum_{i=1}^{n} \sum_{j=1}^{m_i} \left[\widehat{\beta}_k(t_{ij}) - \beta_{0k}(t_{ij}) \right]^2.$$

For out-of-sample prediction, we generated 30 new observations $(\boldsymbol{Y}_{new}, \boldsymbol{t}_{new}, \boldsymbol{X}_{new})$, calculated a new $\boldsymbol{U}$ matrix (7), and computed the mean squared prediction error (MSPE),

$$\text{MSPE} = \frac{1}{N}\|\boldsymbol{Y}_{new} - \boldsymbol{U}_{new}\widehat{\gamma}\|_2^2.$$

Finally, to evaluate variable selection performance, we compared the sensitivity (Sens), specificity (Spec), and Matthews correlation coefficient (MCC), given by

$$\text{Sens} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{Spec} = \frac{\text{TN}}{\text{TN} + \text{FP}},$$
$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP}+\text{FP})(\text{TP}+\text{FN})(\text{TN}+\text{FP})(\text{TN}+\text{FN})}},$$

where TP, TN, FP, and FN are the number of true positives, true negatives, false positives, and false negatives respectively. MCC takes values between -1 and 1, with higher values indicating better overall variable selection performance.

We repeated Experiments 1-5 for 200 replications each. In Table 1, we report our results averaged across the 200 replicates. We see that under all of the different unknown within-subject covariance structures, NVC-SSL had the lowest MSE and the lowest MSPE, indicating the best estimation and predictive performance. For variable selection, NVC-SSL had the highest average sensitivity, indicating the best ability to detect the true nonzero

Table 1: Simulation results for NVC-SSL, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP, averaged across 200 replicates. To better highlight the differences in estimation performance, we rescale the MSE by 100, i.e. we report MSE $\times$ 100 in the first column. The empirical standard error is reported in parentheses following the average.

Experiment 1: AR(1) covariance function

	MSE $\times$ 100	MSPE	Sens	Spec	MCC
NVC-SSL	**0.114** (0.049)	**4.057** (1.976)	**0.988** (0.052)	0.999 (0.002)	**0.948** (0.069)
NVC-gLASSO	0.989 (0.128)	8.559 (2.948)	0.688 (0.057)	**1** (0)	0.827 (0.033)
NVC-gSCAD	0.291 (0.030)	4.752 (2.043)	0.667 (0)	**1** (0)	0.815 (0)
NVC-gMCP	0.284 (0.025)	4.729 (2.050)	0.667 (0)	**1** (0)	0.815 (0)

Experiment 2: CS covariance function

	MSE $\times$ 100	MSPE	Sens	Spec	MCC
NVC-SSL	**0.113** (0.046)	**4.255** (2.254)	**0.989** (0.041)	0.999 (0.002)	**0.947** (0.067)
NVC-gLASSO	0.990 (0.116)	8.884 (3.175)	0.681 (0.050)	0.999 (0.001)	0.823 (0.029)
NVC-gSCAD	0.291 (0.029)	4.956 (2.266)	0.667 (0)	**1** (0)	0.815 (0)
NVC-gMCP	0.284 (0.024)	4.930 (2.274)	0.667 (0)	**1** (0)	0.815 (0)

Experiment 3: SE covariance function

	MSE $\times$ 100	MSPE	Sens	Spec	MCC
NVC-SSL	**0.112** (0.056)	**4.448** (2.536)	**0.989** (0.053)	0.999 (0.002)	**0.946** (0.071)
NVC-gLASSO	0.980 (0.129)	9.347 (3.784)	0.681 (0.050)	0.999 (0.001)	0.823 (0.029)
NVC-gSCAD	0.287 (0.031)	5.056 (2.486)	0.667 (0)	**1** (0)	0.815 (0)
NVC-gMCP	0.280 (0.024)	5.030 (2.498)	0.667 (0)	**1** (0)	0.815 (0)

Experiment 4: Periodic covariance function

	MSE $\times$ 100	MSPE	Sens	Spec	MCC
NVC-SSL	**0.106** (0.042)	**4.062** (2.134)	**0.981** (0.045)	0.999 (0.002)	**0.965** (0.054)
NVC-gLASSO	0.996 (0.128)	8.706 (3.213)	0.688 (0.055)	0.999 (0.001)	0.827 (0.033)
NVC-gSCAD	0.291 (0.029)	4.737 (2.106)	0.667 (0)	**1** (0)	0.815 (0)
NVC-gMCP	0.286 (0.026)	4.722 (2.096)	0.667 (0)	**1** (0)	0.815 (0)

Experiment 5: Zero covariance function (i.i.d. errors)

	MSE $\times$ 100	MSPE	Sens	Spec	MCC
NVC-SSL	**0.062** (0.024)	**3.027** (2.269)	**0.999** (0.012)	0.999 (0.002)	**0.962** (0.055)
NVC-gLASSO	0.966 (0.111)	7.784 (3.123)	0.680 (0.045)	**1** (0)	0.823 (0.026)
NVC-gSCAD	0.268 (0.023)	3.798 (2.284)	0.667 (0)	**1** (0)	0.815 (0)
NVC-gMCP	0.261 (0.017)	3.775 (2.278)	0.667 (0)	**1** (0)	0.815 (0)

varying coefficient functions. NVC-gSCAD and NVC-gMCP had higher specificity, but NVC-SSL's average specificity was still quite good, at 0.999. Moreover, NVC-SSL had the highest average MCC, indicating the best overall ability to correctly select the true nonzero

Table 2: Simulation results for estimation and variable selection of the nonzero varying coefficients $\beta_1(t)$, $\beta_2(t)$, $\beta_3(t)$, $\beta_4(t)$, $\beta_5(t)$, $\beta_6(t)$ for NVC-SSL, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP, averaged across 200 replicates. "Proportion" gives the proportion of replicates that selected the varying coefficient.

Experiment 1: AR(1) covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
NVC-SSL	**0.060**	**0.081**	**0.093**	**0.103**	**0.113**	**0.042**	1	1	**0.97**	**0.96**	1	1
NVC-gLASSO	0.870	1.082	0.468	0.565	1.068	0.894	1	1	0.045	0.08	1	1
NVC-gSCAD	0.074	0.136	0.476	0.577	0.136	0.058	1	1	0	0	1	1
NVC-gMCP	0.073	0.105	0.476	0.577	0.134	0.057	1	1	0	0	1	1

Experiment 2: CS covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
NVC-SSL	**0.066**	**0.080**	**0.094**	**0.094**	**0.114**	**0.045**	1	1	**0.965**	**0.97**	1	1
NVC-gLASSO	0.890	1.068	0.472	0.566	1.036	0.916	1	1	0.025	0.06	1	1
NVC-gSCAD	0.085	0.129	0.475	0.577	0.132	0.056	1	1	0	0	1	1
NVC-gMCP	0.084	0.096	0.475	0.577	0.131	0.056	1	1	0	0	1	1

Experiment 3: SE covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
NVC-SSL	**0.063**	**0.076**	**0.087**	**0.098**	**0.106**	**0.043**	1	1	**0.975**	**0.96**	1	1
NVC-gLASSO	0.848	1.059	0.473	0.562	1.051	0.908	1	1	0.015	0.07	1	1
NVC-gSCAD	0.078	0.128	0.474	0.574	0.122	0.058	1	1	0	0	1	1
NVC-gMCP	0.076	0.100	0.474	0.574	0.121	0.056	1	1	0	0	1	1

Experiment 4: Periodic covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
NVC-SSL	**0.065**	**0.079**	**0.092**	**0.098**	**0.110**	**0.042**	1	1	**0.97**	**0.975**	1	1
NVC-gLASSO	0.899	1.047	0.471	0.562	1.122	0.878	1	1	0.035	0.09	1	1
NVC-gSCAD	0.089	0.125	0.476	0.578	0.133	0.056	1	1	0	0	1	1
NVC-gMCP	0.089	0.101	0.476	0.578	0.132	0.056	1	1	0	0	1	1

Experiment 5: Zero covariance function (i.i.d. errors)

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
NVC-SSL	**0.039**	**0.041**	**0.045**	**0.048**	**0.082**	**0.026**	1	1	**1**	**0.995**	1	1
NVC-gLASSO	0.839	1.032	0.470	0.570	1.010	0.907	1	1	0.045	0.035	1	1
NVC-gSCAD	0.054	0.093	0.475	0.576	0.099	0.040	1	1	0	0	1	1
NVC-gMCP	0.053	0.064	0.475	0.576	0.097	0.039	1	1	0	0	1	1

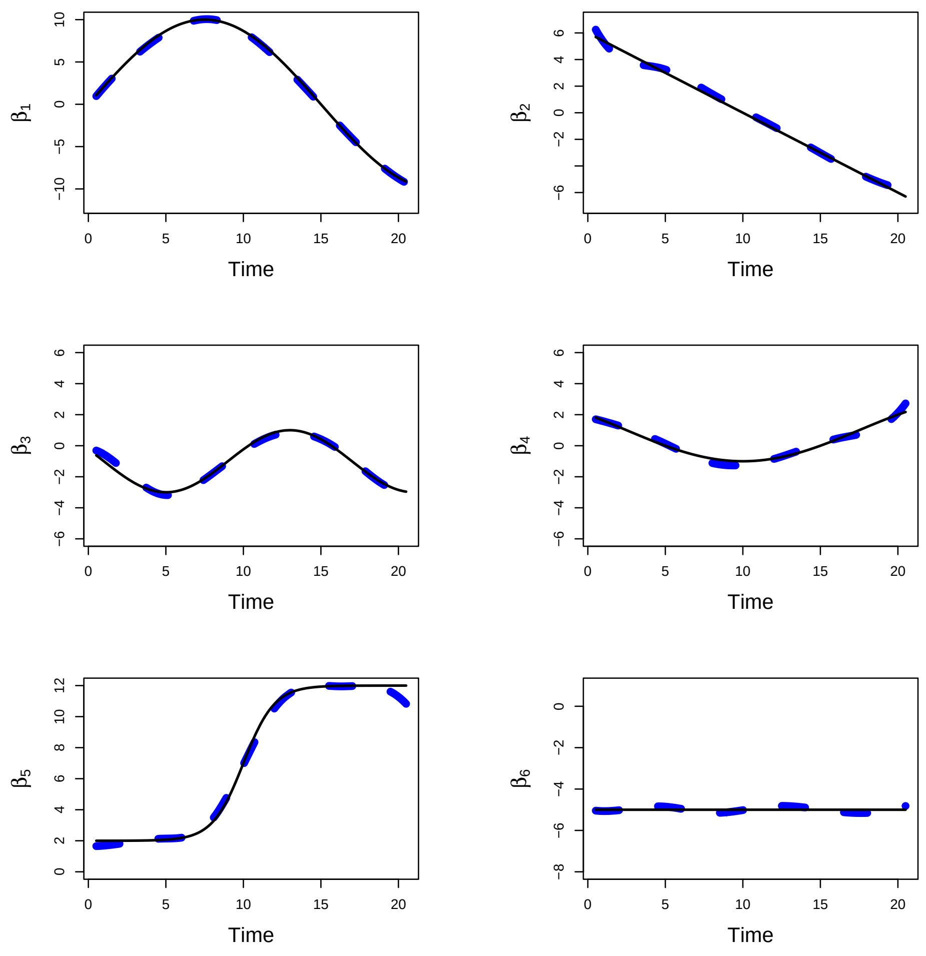

Figure 1: Plots of the MAP estimates (dashed blue lines) for $\beta_k(t), k = 1, \ldots, 6$, under the NVC-SSL model from one replication of Experiment 1 (i.e. AR(1) within-subject covariance function). The true functions are the solid black lines.

functions while excluding the spurious ones. Our results suggest that accounting for within-subject correlations can greatly improve estimation, prediction, and variable selection.

An interesting thing to point out is that in Experiment 5 (i.e. zero within-subject correlations), NVC-SSL *still* managed to outperform NVC-gLASSO, NVC-gSCAD, and NVC-gMCP. One might assume that the latter three methods would be competitive in this setting, since the implied model under these methods is i.i.d. errors with noise variance $\sigma^2 = 1$. However, NVC-SSL continued to have lower MSE and MSPE and higher MCC in this scenario. This suggests the following two things. First, the NVC-SSL method is flexible enough to *also* estimate covariance functions equal to zero, and thus, NVC-SSL can work well in the case where the residual errors truly are i.i.d. Secondly, our results also point to the practical benefit having a slab density $\mathbf{\Psi}(\cdot \mid \lambda_1)$, in addition to a spike density $\mathbf{\Psi}(\cdot \mid \lambda_0)$

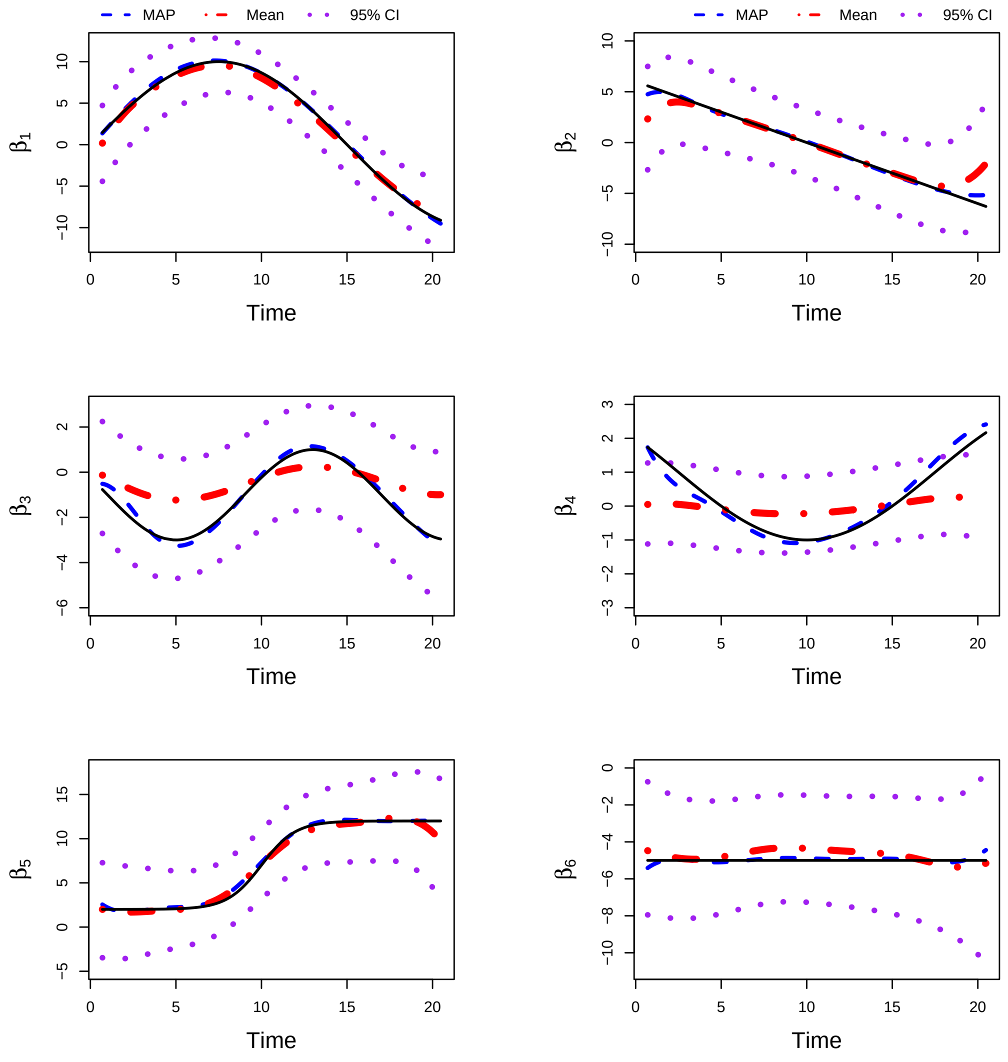

Figure 2: Plots of the MAP estimates (dashed blue line), posterior mean estimates (dashed red line with dots), and 95% posterior credible intervals (dotted purple lines) for $\beta_k(t), k = 1, \ldots, 6$, under the NVC-SSL model from one replication of Experiment 2 (i.e. CS within-subject covariance function). The true functions are the solid black lines.

in the $\mathcal{SSGL}(\lambda_0, \lambda_1, \theta)$ prior (12). Meanwhile, gLASSO, gMCP, and gSCAD only have one regularization parameter $\lambda > 0$ controlling the level of sparsity. As a result, these other methods may overshrink many of the functions to zero when λ is large.

We also compared how well NVC-SSL, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP were able to estimate and select the true nonzero varying coefficients $\beta_1, \ldots, \beta_6$. These results are reported in Table 2. In this case, we computed the MSE for the individual functions as MSE $= N^{-1} \sum_{i=1}^{N} \sum_{j=1}^{m_i} [\beta_k(t_{ij}) - \beta_{0k}(t_{ij})]^2$. Table 2 shows that in all our experiments, NVC-SSL had the lowest average MSE, indicating that it estimated the nonzero functions the best. We also kept track of the proportion of the 200 replicates that selected $\beta_1, \ldots, \beta_6$. As shown in Table 2, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP were unable to select the weak signals β_3 and β_4 in almost all replications, which explains their lower average sensitivity and MCC in Table 1. In contrast, NVC-SSL selected β_3 and β_4 in 96% to 100% of the replicates. This verifies that NVC-SSL is especially well-suited for detecting weak signals. Our results in Table 2 further reinforce the benefit of having a slab density in the $\mathcal{SSGL}(\lambda_1, \lambda_0, \theta)$ prior (12) for NVC-SSL, which helps smaller signals to escape the pull of the spike and thus be detected.

Figure 1 plots the estimated varying coefficients (dashed line) against the true varying coefficients (solid line) for $\beta_k(t), k = 1, \ldots, 6$ from one replication of Experiment 1. Figure 1 shows that the MAP estimator under NVC-SSL is able to capture the true shapes of these functions, including the weaker signals (β_3 and β_4). Moreover, NVC-SSL was also able to capture the linear trend in β_2, the flat regions of the function β_5, and the constant nonzero (non time-varying) function β_6. This demonstrates the flexibility of our model and justifies the use of B-splines as the basis functions in (5).

In Appendix A, we report additional results for our Experiments 1-5 where we used the MCMC algorithm in Section 4.1 to fit the NVC-SSL model. We conclude that the MAP estimator is superior as a *point estimator* for function selection and estimation in high dimensions. This is demonstrated in Figure 2 where the MAP estimator (dashed line) is shown to better capture the shapes of the weak signals β_3 and β_4 than the posterior mean (dotted and dashed line). However, posterior *inference* from the 95% credible intervals (dotted lines in Figure 2) is still quite good, even if the MAP estimator is preferred for point estimation.

5.2 Performance of MCMC for inference

In this section, we compare the performance of the exact and approximate MCMC algorithms introduced in Section 4 for inference. We simulated data from $n = 100$ subjects. For each ith subject, $m_i = 8$ time points were randomly sampled from $\mathcal{U}(0, 20)$, leading to a total of $N = 800$ observations. We set $p = 1000$, with the true varying coefficients,

$$\beta_1(t) = 10 \sin\left(\frac{\pi t}{15}\right), \quad \beta_2(t) = 8 \cos\left(\frac{\pi(t-20)}{5}\right), \quad \beta_3(t) = 2 + \frac{10}{1 + e^{10-t}},$$
$$\beta_4(t) = \cdots = \beta_p(t) = 0,$$

i.e. β_1, β_2, and β_3 are nonzero functions and the rest of the varying coefficients are zero. We generated the time-varying covariates $x_k(t)$ the same way as we did in Section 5.1, and we simulated the response variables $y(t)$ from the model (2) under the following within-subject covariance structures (see Experiments 1-5 in Section 5.1 for details):

- Experiment 6: AR(1) covariance function
- Experiment 7: CS covariance function
- Experiment 8: SE covariance function

- Experiment 9: periodic covariance function
- Experiment 10: zero covariance function (i.e. no within-subject correlations)

In Experiments 6 through 10, the hyperparameters in the within-subject covariance functions $k_i(t_{ij}, t_{ij'})$ for each ith subject were simulated the same way as those described in Section 5.2.

Each experiment was repeated 200 times. In all replications, we ran both the exact and approximate MCMC algorithms introduced in Section 4 for 2000 iterations, discarding the first 500 iterations as burnin. The remaining 1500 MCMC samples were used to approximate the posteriors and perform uncertainty quantification. Our MCMC algorithms were initialized with the MAP estimator obtained from the ECM algorithm, and all hyperparameters and basis dimensions were the same as those used for the ECM algorithm. With the MAP estimator as our choice of initialization, the effective sample size (prior to burnin) was very close to 2000 for each of the basis coefficients in $\boldsymbol{\gamma}$, suggesting that 2000 MCMC iterations was sufficient.

We used the posterior samples of $\boldsymbol{\gamma}$ to estimate the posterior mean for each kth varying coefficient as $\widetilde{\beta}_k(t) = \sum_{l=1}^{d} \widetilde{\gamma}_{kl} B_{kl}(t)$, where $\widetilde{\boldsymbol{\gamma}}$ denotes the posterior mean of $\boldsymbol{\gamma}$. In order to obtain the pointwise 95% posterior credible intervals for each varying coefficient, we used the 2.5th and 97.5 sample quantiles of the MCMC samples for $\boldsymbol{\gamma}$. That is, the 95% credible interval for each varying coefficient function $\beta_k(t)$ at time t was $[\beta_k^L(t), \beta_k^U(t)]$, where $\beta_k^L(t) = \sum_{l=1}^{d} \gamma_{kl}^L B_{kl}(t)$ and $\beta_k^U(t) = \sum_{l=1}^{d} \gamma_{kl}^U B_{kl}(t)$, and γ_{kl}^L and γ_{kl}^U were the 2.5 and 97.5 quantiles for the MCMC samples of γ_{kl}.

We compared the MSE for the posterior mean functions $\widetilde{\beta}_k(t)$'s obtained from the exact MCMC and the approximate MCMC algorithms. We also compared the average width and the empirical coverage probability (ECP) of the 95% posterior credible intervals. We looked at both the pointwise ECP (i.e. the proportion of pointwise credible intervals that contained the true value of $\beta_k(t_{ij})$ for each observed time point t_{ij} $1 \leq i \leq n, 1 \leq j \leq m_i$) and the simultaneous ECP. Here, the simultaneous ECP was determined by the proportion of simulations where *all* of the posterior credible intervals covered *all* of the true varying coefficient functions in the entire time domain. It is important to note that in high dimensions, a Bayesian credible set is not necessarily a confidence set (van der Pas et al., 2017). Nevertheless, investigating the ECP is a good way to gauge whether the Bayesian uncertainty intervals are reasonable or not.

Our results are reported in Table 3. We can see that the average MSE for the posterior means under the exact MCMC and approximate MCMC algorithms were practically identical, which aligns with our theoretical analysis in Section 4.3. However, the average width of credible intervals were slightly larger for the approximate MCMC algorithm than for the exact algorithm. We theoretically quantified this trade-off in Section 4.3.

While the *pointwise* ECP was comparable for both algorithms, Table 3 shows that the *simultaneous* ECP was considerably higher for the approximate MCMC algorithm. Namely, 99 to 100 percent of simulations in each experiment had credible intervals which contained *all* of the true varying coefficient functions. This can be attributed to the larger size of the uncertainty intervals produced by the approximate MCMC algorithm.

Figure 3 plots the posterior mean varying coefficients for $\beta_1(t)$, $\beta_2(t)$, and $\beta_3(t)$ from one replication of Experiment 9. The ground truth is plotted as a solid line, while the

Table 3: Simulation results for the exact MCMC algorithm and the approximate MCMC algorithm, averaged across 200 replicates. The rescaled MSE (i.e. MSE $\times$ 100) is reported for the posterior mean. The standard errors for the rescaled MSE and the average width of the pointwise 95% posterior intervals are reported in parentheses.

Experiment 6: AR(1) covariance function

	MSE $\times$ 100	Width	Pointwise ECP (%)	Simultaneous ECP (%)
Exact	0.202 (0.049)	2.128 (0.003)	99.9	85
Approximate	0.202 (0.049)	2.423 (0.009)	100	100

Experiment 7: CS covariance function

	MSE $\times$ 100	Width	Pointwise ECP (%)	Simultaneous ECP (%)
Exact	0.174 (0.046)	2.128 (0.003)	99.9	89
Approximate	0.174 (0.046)	2.422 (0.008)	99.9	99

Experiment 8: SE covariance function

	MSE $\times$ 100	Width	Pointwise ECP (%)	Simultaneous ECP (%)
Exact	0.199 (0.044)	2.128 (0.003)	99.9	91
Approximate	0.199 (0.044)	2.423 (0.008)	99.9	99

Experiment 9: Periodic covariance function

	MSE $\times$ 100	Width	Pointwise ECP (%)	Simultaneous ECP (%)
Exact	0.190 (0.046)	2.129 (0.004)	99.9	87
Approximate	0.190 (0.046)	2.423 (0.008)	100	100

Experiment 10: Zero covariance function (i.i.d. errors)

	MSE $\times$ 100	Width	Pointwise ECP (%)	Simultaneous ECP (%)
Exact	0.136 (0.038)	2.127 (0.003)	99.9	89
Approximate	0.136 (0.038)	2.421 (0.008)	99.9	99

posterior mean and 95% posterior credible intervals are plotted as dashed lines. The results for the exact algorithm are shown in the left panels and the results for the approximate algorithms are shown in the right panels. Figures 2 and 3 show that despite having wider credible intervals, the approximate algorithm still captures the shape of the true varying coefficients. Running the exact algorithm for 2000 iterations also took 2.3 hours for the one replicate in Figure 3, whereas the approximate algorithm only took only 6.2 minutes on an 11th Gen Intel Core i5-1135G7 processor. In short, the approximate MCMC algorithm gives slightly more conservative pointwise uncertainty intervals but it also provides better *simultaneous* coverage *and* it is much faster and more scalable than the exact algorithm.

5.3 Timing and efficiency comparisons

Here, we report results for the timing and efficiency of the algorithms that we introduced in Sections 3 and 4. To conduct our experiments, we modified Experiment 8 from Section 5.2 (i.e. SE within-subject covariance function for $n = 100$ subjects and $N = 800$ total observations). Namely, we varied $p \in \{500, 1000, \dots, 5000\}$, but kept all the other simulation settings the same. We again used $d = 8$ basis functions for each varying coefficient, leading to a total of $dp \in \{4000, 8000, \dots, 40{,}000\}$ unknown basis coefficients in $\boldsymbol{\gamma}$. We report timing

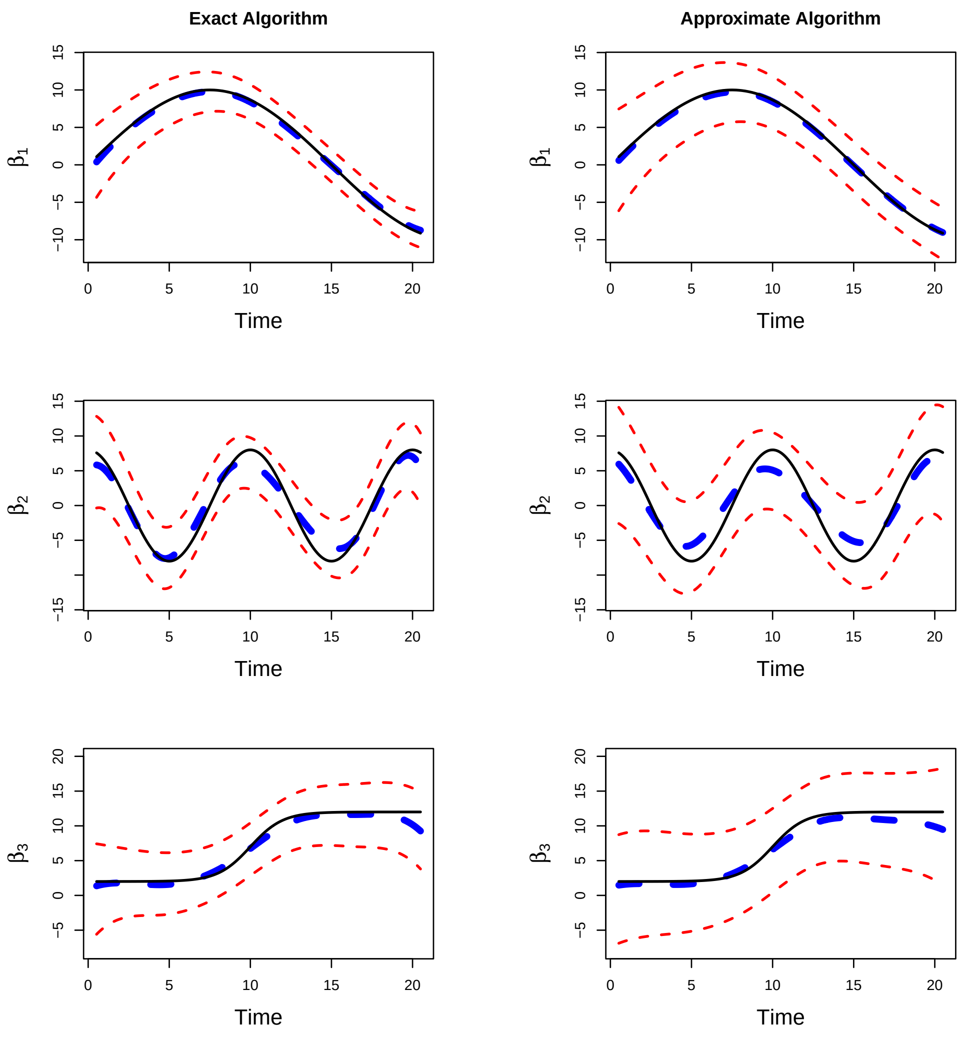

Figure 3: Plots of the results from one replication of Experiment 9 (i.e. periodic within-subject covariance function) under the exact MCMC algorithm (left three figures) and the approximate MCMC algorithm (right three figures). The true functions $\beta_1(t)$, $\beta_2(t)$, and $\beta_3(t)$ are the solid black lines, the posterior mean estimates are the thick dashed blue lines, and the 95% posterior credible intervals are the thin red dashed lines.

results for the optimal λ_0 chosen from BIC (26). To stress that we are in fact dealing with a very high-dimensional problem, we report our results using dp instead of merely p. All of our experiments were performed on an Intel Xeon 8358 Platinum processor with 2.6GHz CPU and 128 GB memory.

We first compared the average per-iteration runtime for the ECM algorithm of Section 3 and the exact MCMC and approximate MCMC algorithms of Section 4 across 50 replicates. Figure 4 plots the average runtime per iteration for these three methods against dp. We see

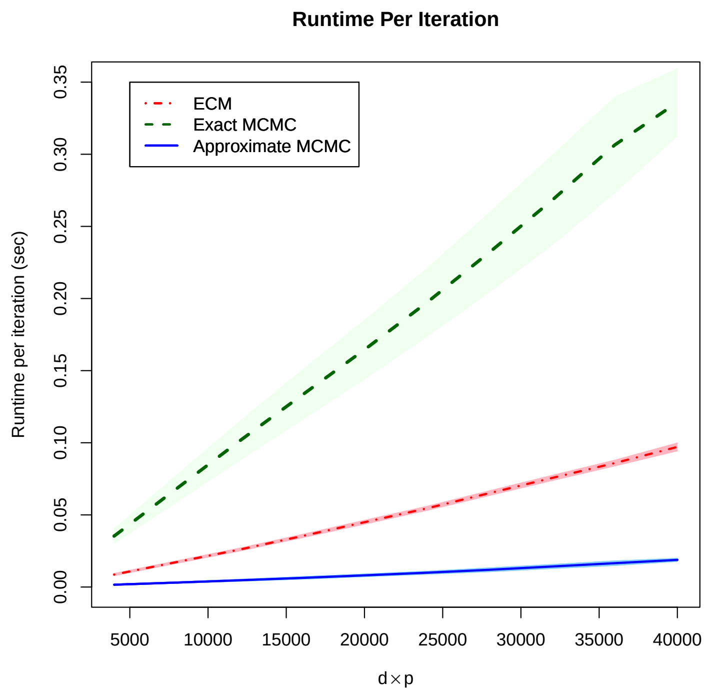

Figure 4: Plots of the average runtime per iteration (seconds) across 50 replications for the ECM algorithm, exact MCMC algorithm, and approximate MCMC algorithm against the dimension dp of the basis coefficients vector. The lightly shaded areas indicate the regions within one standard deviation of the mean.

that all methods scale linearly with p. In particular, for $dp = 40{,}000$, the average runtime for one iteration was 5.8 seconds for the ECM algorithm, 20.2 seconds for the exact MCMC algorithm, and 1.1 seconds for the approximate MCMC algorithm.

As shown in Figure 4, the approximate MCMC algorithm had on average the fastest runtime per iteration, the ECM algorithm was the second fastest, and the exact MCMC algorithm was the slowest. This matches our earlier complexity analysis, since the computational complexity of the ECM algorithm is $\mathcal{O}(Ndpr)$, where r is the number of iterations it takes to numerically solve for $\boldsymbol{\gamma}$ in (23). Meanwhile, the approximate MCMC algorithm in Section 4.2 has complexity of $\mathcal{O}(Ndp)$. The extra factor of $r > 1$ in the ECM algorithm accounts for its slower per-iteration runtime than the approximate MCMC algorithm. With time complexity of $\mathcal{O}(N^2 dp)$, the exact MCMC algorithm has the slowest runtime, indicating that in general, $N \gg r$.

Although the per iteration cost was the fastest for the approximate MCMC algorithm, the ECM algorithm also terminated very quickly. In all of experiments, the ECM algorithm converged within 10 iterations, even when $dp = 40{,}000$. On the other hand, we would typically run MCMC algorithms for much more than 10 iterations – usually for at least a

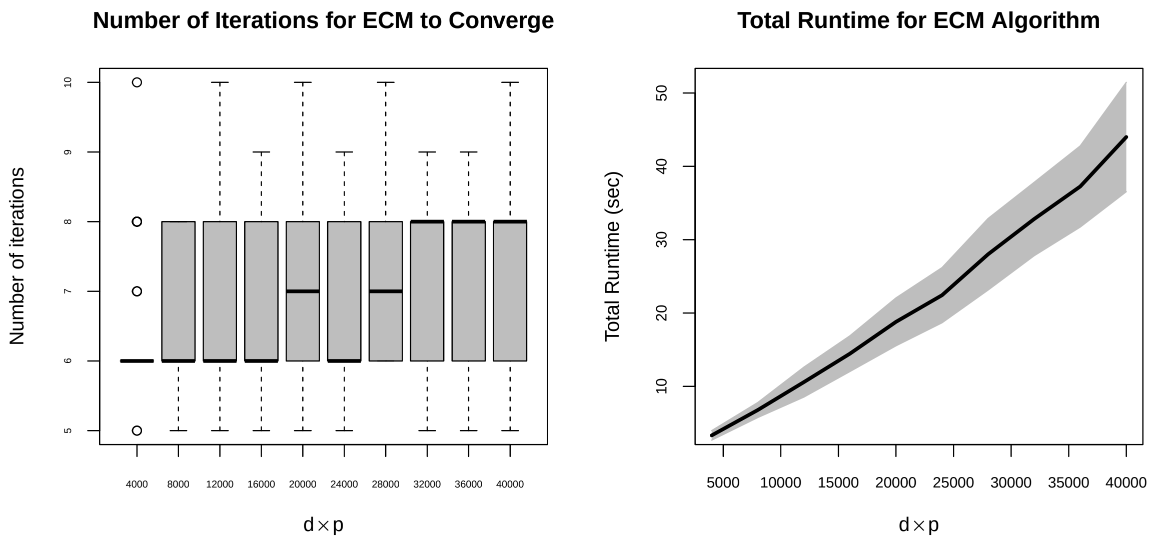

Figure 5: Left panel: Box plots of the number of iterations it took for the ECM algorithm to converge (50 replications). Right panel: Plot of the average total runtime (seconds) across 50 replications for the ECM algorithm to finish running against the dimension dp of the basis coefficients vector. The lightly shaded area in the right plot indicates the region within one standard deviation of the mean.

couple hundred iterations. As a result, the *overall* time to complete the MCMC algorithm may still be greater than that for the MAP estimation algorithm.

In the left panel of Figure 5, we plot the box plots for the number of iterations that it took for the ECM algorithm to finish running for each $dp \in \{4000, 8000, \ldots, 40{,}000\}$. The right panel of Figure 5 reports the *total* runtime for the ECM algorithm as a function of dp. For $dp = 40{,}000$, it took on average 7.54 iterations and 44 seconds for the ECM algorithm to converge. These results demonstrate the scalability and computational feasibility of finding the MAP estimator for NVC-SSL.

In the left panel of Figure 6, we compare the *total* runtime of the exact MCMC and the approximate MCMC algorithms for 1000 iterations. Both algorithms were initialized with the MAP estimator for $\boldsymbol{\gamma}$ obtained from the ECM algorithm. For $dp = 4000$, the average total runtime was 1.6 minutes for the approximate MCMC algorithm vs. 34.5 minutes for the exact MCMC algorithm. For $dp = 40{,}000$, the average total runtime was 18.8 minutes for the approximate MCMC algorithm vs. 338.9 minutes (or 5.6 hours) for the exact MCMC algorithm. It is clear that the approximate MCMC algorithm provides orders of magnitude speedup, especially when dp is very large.

Perhaps a more transparent way to compare the MCMC algorithms is their efficiency, or their effective sample size (ESS) per second. For correlated MCMC samples, the ESS estimates the number of independent samples that would have given the same precision (or

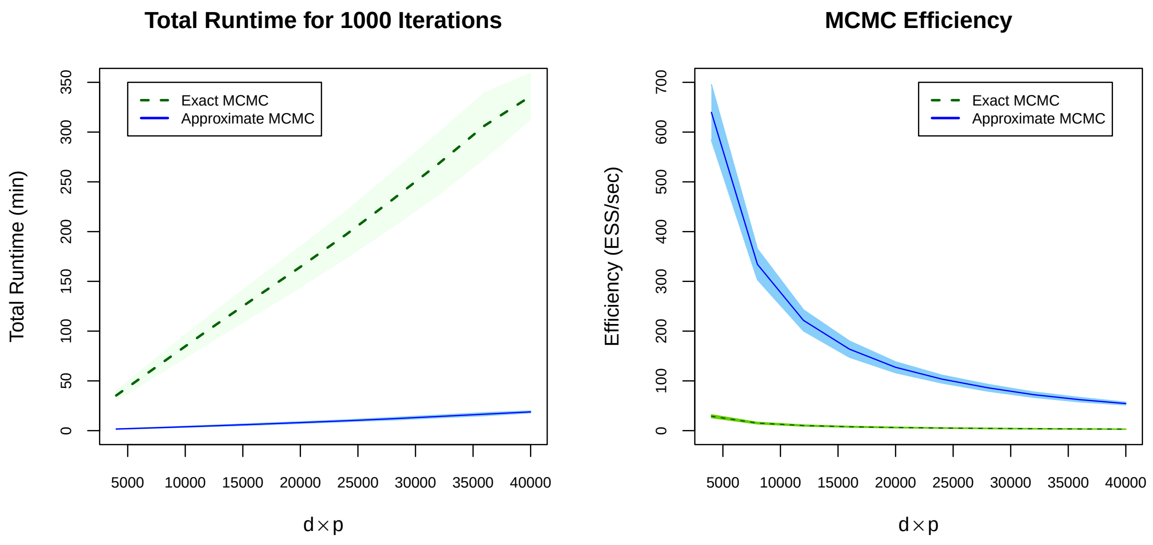

Figure 6: Left panel: Plots comparing the average total runtime (minutes) across 50 replications for the exact MCMC and approximate MCMC algorithms to run 1000 iterations. Right panel: Plot of the MCMC efficiency (ESS per second) across 50 replications for the exact MCMC and approximate MCMC algorithms based on 1000 iterations. In both plots, the lightly shaded areas are the regions within one standard deviation of the mean.

variance) as the MCMC samples. Thus, a higher ESS per second indicates greater MCMC efficiency. We used the R package `sns` to estimate the ESS for all dp entries in $\boldsymbol{\gamma}$ and then took the average ESS for these parameters in $\boldsymbol{\gamma}$. In the right panel of Figure 6, we plot the average efficiency against dp for the exact MCMC and the approximate MCMC algorithms. Figure 6 shows that the approximate MCMC algorithm has much higher efficiency. In particular, when $dp = 4000$, the average efficiency was 639.7 samples per second vs. only 28.9 samples per second for the exact algorithm. For $dp = 40{,}000$, the average efficiency was 54.4 samples per second for the approximate algorithm vs. only 3.1 samples per second for the exact algorithm. Thus, even though we have only approximated the MCMC transition kernel in the approximate algorithm, we have not done so at the expense of efficiency – in fact, we significantly *increased* the efficiency of our MCMC samples.

6. Yeast cell cycle data analysis

The cell cycle is a tightly regulated set of processes by which cells grow, replicate their DNA, segregate their chromosomes, and divide into daughter cells. Transcription factors (TFs) are sequence-specific DNA binding proteins which regulate the transcription of genes from DNA to mRNA by binding specific DNA sequences. To better understand how TFs regulate the cell cycle, we applied our proposed NVC-SSL procedure to a dataset of cell-cycle regulated yeast genes and associated TFs.

The data that we used comes from the α-factor synchronized cultures of Spellman et al. (1998) and the CHIP-chip data of Lee et al. (2002). Spellman et al. (1998) measured genome-wide mRNA levels for 6178 yeast open reading frames (ORFs) over approximately two cell cycle periods, with measurements at 7-minute intervals for 119 minutes (for a total of 18 time points). The data of Lee et al. (2002) contains binding information of 96 TFs which elucidates which TFs bind to promoter sequences of genes across the yeast genome. We aimed to fit the varying coefficient model to these 96 TFs and an intercept function $\beta_0(t)$ representing the baseline change in mRNA over time, i.e.

$$y_i(t_{ij}) = \beta_0(t_{ij}) + \sum_{k=1}^{96} x_{ik}\beta_k(t_{ij}) + \varepsilon_i(t_{ij}), \quad i = 1, \ldots, n, \quad j = 1, \ldots, 18. \tag{32}$$

where $y_i(t_{ij})$ denotes the mRNA level for the ith gene at the jth time point. Thus, including the intercept function, we have $p = 97$ varying coefficients. Like other authors (Wang et al., 2008; Xue and Qu, 2012), we also penalized $\beta_0(t)$ in order to ensure the identifiability of all varying coefficients.

Previous works for fitting (32) assumed that the error terms $\varepsilon_i(t_{ij})$'s were i.i.d. for all i and j (Wang et al., 2008; Wei et al., 2011). However, de Lichtenberg et al. (2005) identified 113 yeast genes most likely to be periodically expressed (or to display periodicities over time) in small-scale experiments, including 104 genes used by Spellman et al. (1998). This suggests that at least some genes display temporal correlation, and the independence assumptions previously used are not appropriate. The NVC-SSL model allows us to flexibly model the genes' temporal correlations by decomposing the error $\varepsilon_i(t_{ij})$ into a functional random effect (where all m_i random effects $\alpha_i(t_{i1}), \ldots, \alpha_i(t_{im_i})$ for the ith subject are correlated) and a measurement error term r_{ij}, as in (2).

Using the datasets in Spellman et al. (1998) and Lee et al. (2002), we extracted the 104 genes identified as periodically expressed by de Lichtenberg et al. (2005). After excluding genes with missing values in either of the experiments, we were left with $n = 47$ genes, for a total of $N = 846$ observations.

6.1 Results for variable selection and out-of-sample prediction

We compared the NVC-SSL, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP models. BIC was used to select the spike hyperparameter λ_0 in NVC-SSL and the penalty parameter λ in NVC-gLASSO, NVC-gSCAD, and NVC-gMCP. To assess their variable selection performance, we fit these models using all $n = 47$ genes. We also examined the models' predictive power. To do so, we randomly divided the dataset into 37 training observations and 10 test observations. We fit the NVC models to the training data and then used our fitted models to predict the trajectories of mRNA level $\widehat{y}(t)$ for the 10 test observations and compute the out-of-sample MSPE. We repeated this procedure 200 times, so that we had 200 different test sets on which to evaluate these different methods.

All four methods selected the intercept function $\beta_0(t)$. These intercept functions are plotted in Figure 7. We see that all four methods concluded that there is a baseline periodic trend in mRNA levels over time. However, the NVC-SSL curve for $\beta_0(t)$ is a bit more

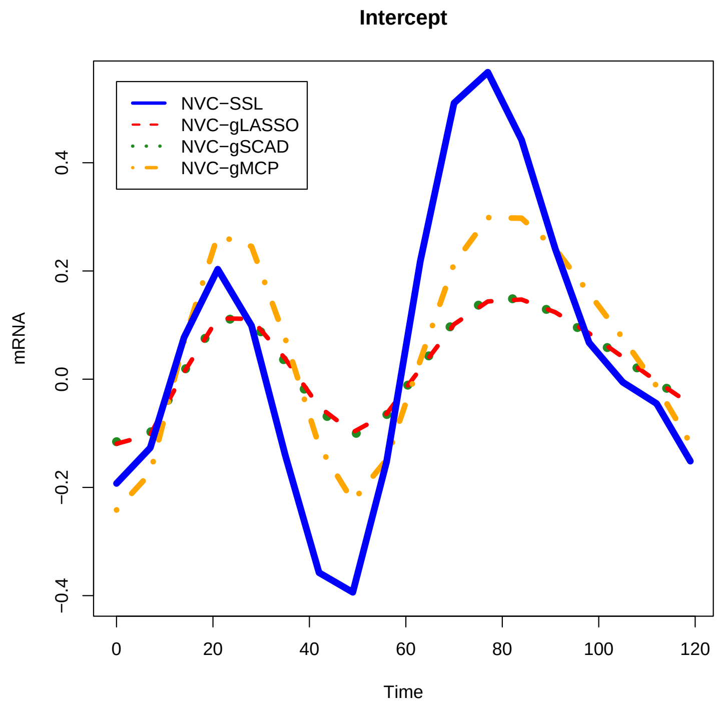

Figure 7: Plots of the estimated intercept function $\beta_0(t)$ representing the baseline trend in mRNA level over time for NVC-SSL, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP.

pronounced and less smooth, with a higher amplitude. The smaller amplitudes for NVC-gLASSO's, NVC-gSCAD's, and NVC-gMCP's estimates of $\beta_0(t)$ indicate that these methods all penalized the baseline trend more heavily than NVC-SSL.

Table 4 shows our results for the number of TFs selected and the out-of-sample prediction error. The NVC-SSL model selected the most TFs and had the second lowest average MSPE. Figure 8 gives the names of the 17 TFs selected by NVC-SSL and plots their estimated transcriptional effects over time. NVC-gLASSO, NVC-gSCAD, and NVC-gMCP all selected more parsimonious models, with NVC-gMCP selecting the sparsest model with only three TFs. In addition, NVC-gMCP had the lowest average MSPE. However, the signals in this dataset were rather weak to begin with, so it may not be surprising that the most parsimonious model also had the best predictive accuracy – a model that always selects the null model on this dataset would likely give a similar predictive performance. We saw from our simulations in Section 5.1 that NVC-SSL was better able to detect weak signals, and that may also be the case here.

The NVC-SSL model was able to detect meaningful biological signal in the data. The cell cycle is an ordered set of events, culminating in cell growth and division into two daughter cells. Stages of the cell cycle are commonly divided into G1-S-G2-M. The G1 stage stands for "GAP 1." The S stage stands for "Synthesis" and is the stage when DNA replication

Table 4: Average MSPE on 200 test sets (standard errors in parentheses) and number of transcription factors selected by NVC-SSL, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP.

	MSPE	Number of TFs Selected
NVC-SSL	0.512 (0.141)	17
NVC-gLASSO	0.515 (0.129)	7
NVC-gSCAD	0.554 (0.290)	7
NVC-gMCP	0.433 (0.167)	3

occurs. The G2 stage stands for "GAP 2." The M stage stands for "mitosis," when nuclear (chromosomes separate) and cytoplasmic (cytokinesis) division occur. The NVC-SSL model selected several TFs that have also been shown to be significant at various stages of the cell cycle in the literature. In particular, the NVC-SSL method selected SWI5 and ACE2. Simon et al. (2001) found that the SWI5 and ACE2 proteins activate genes at the end of M and early G1.

The TFs selected by NVC-SSL also included several pairs of syneristic, or "cooperative," TFs that have been reported in the literature (Banerjee and Zhang, 2003; Tsai et al., 2005). These pairs of TFs are thought to cooperate together to regulate transcription in the yeast cell cycle. Among the 17 TFs selected by NVC-SSL, seven of them (ACE2, HIR1, HIR2, STB15, SUM1, SWI5) belonged to cooperative pairs of TFs identified by Banerjee and Zhang (2003), including the complete cooperative pairs HIR1-HIR2 and ACE-SWI5. On the other hand, NVC-gLASSO and NVC-gSCAD only found four genes belonging to cooperative pairs (HIR1, HIR2, SWI5, SWI6) and one complete cooperative pair HIR1-HIR2, while NVC-gMCP found three genes belonging to cooperative pairs (HIR1, STB1, and SWI5) but no complete cooperative pairs.

6.2 Variable selection performance with added synthetic noise variables

In order to investigate the performance and stability of our variable selection approach in high dimensions, we artificially added 1000 noise variables so that $p = 1097$. For each ith gene, these noise variables were randomly generated from a uniform distribution Uniform$(x_{i,\min}, x_{i,\max})$, where $x_{i,\min}$ and $x_{i,\max}$ denote the minimum and maximum binding information values for the ith gene. We then fit the NVC-SSL, NVC-gLASSO, NVC-gSCAD, and NVC-gMCP models with $p = 1097$ varying coefficients. With $d = 8$ basis functions, we thus had to estimate 8776 unknown basis coefficients $\boldsymbol{\gamma}$ in (5).

We repeated the above procedure 200 times, adding 1000 artificial noise variables to the original dataset each time. For each of the 200 replications, we recorded the number of true TFs selected and the number of noise variables selected. We also kept track of the TFs that were *always* selected by each method in all 200 experiments.

Our results are shown in Table 5. The NVC-SSL selected on average 6.55 real TFs and 1.685 noise variables. NVC-gSCAD and NVC-gMCP selected more noise variables on average than NVC-SSL. In particular, NVC-gMCP selected an average of 8.75 noise variables and only 4.40 real TFs, indicating that NVC-gMCP performed the worst in terms

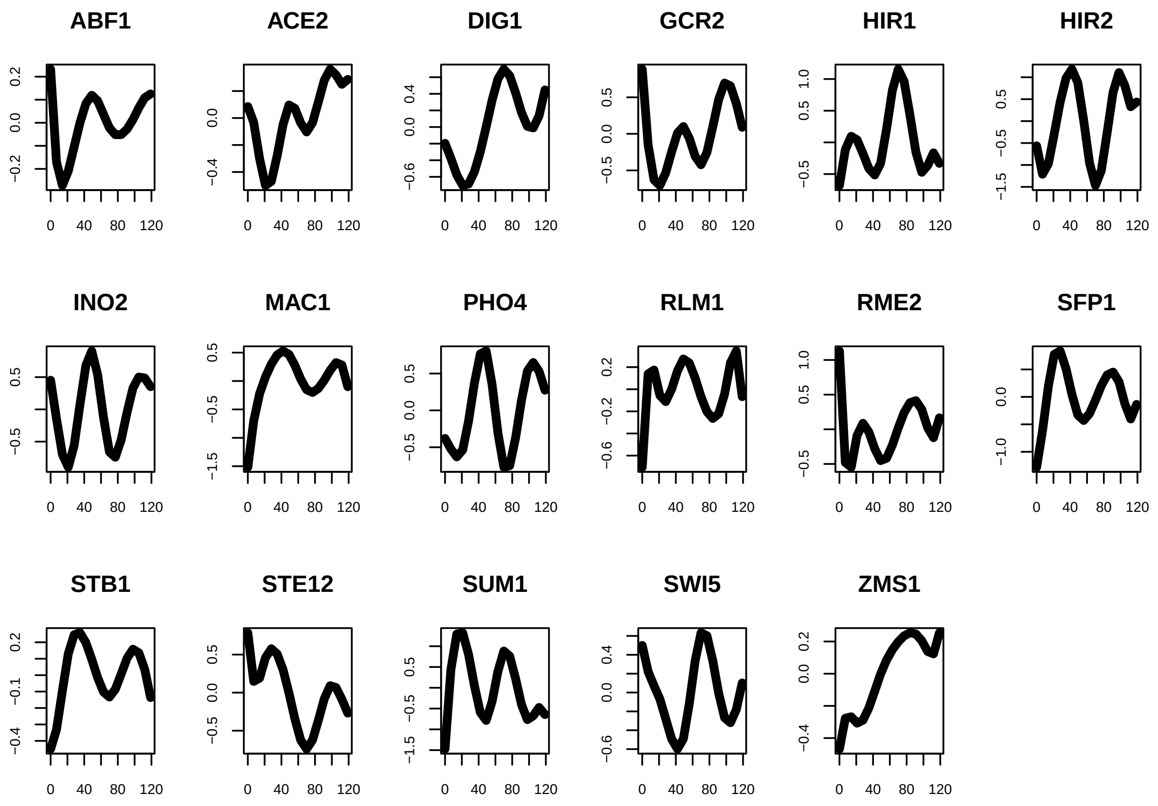

Figure 8: Plots of the estimated transcriptional effects over time for the 17 TFs selected by NVC-SSL.

of being able to exclude noise variables. On this particular dataset, NVC-gLASSO tended to select the most parsimonious model (an average of 5.24 real TFs and 0.09 noise variables), with typically the least number of noise variables selected.

However, Table 5 also shows that NVC-gLASSO only selected one real TF (HIR1) in all 200 experiments, indicating that variable selection was not as consistent for NVC-gLASSO across the replicates. Although NVC-SSL selected on average 1.685 noise variables, NVC-SSL also exhibited the greatest overall variable selection stability, selecting four real TFs (HIR1, RME1, SFP1, and SWI5) in all 200 replications, compared to three for NVC-gSCAD and two for NVC-gMCP. The four TFs that NVC-SSL selected in all 200 experiments were also selected by NVC-SSL on the original dataset with only $p = 97$. Our results demonstrate that variable selection for NVC-SSL is fairly stable in the presence of many known noise variables.

Table 5: Variable selection results after adding 1000 noise variables to the dataset. The first two columns report the average number of real TFs and the average number of noise variables selected across 200 replications, with the empirical standard error in parentheses. The third column lists the TFs that were selected in all 200 experiments.

	Real TFs	Noise Variables	TFs Always Selected
NVC-SSL	6.55 (1.76)	1.69 (1.15)	HIR1, RME1, SFP1, SWI5
NVC-gLASSO	5.24 (1.75)	0.09 (0.28)	HIR1
NVC-gSCAD	6.89 (3.03)	1.82 (11.68)	HIR1, RME1, SWI5
NVC-gMCP	4.40 (3.72)	8.75 (14.16)	HIR1, SWI5

7. Discussion

In this paper, we have introduced the nonparametric varying coefficient spike-and-slab lasso, a new Bayesian approach for estimation and variable selection in high-dimensional NVC models. The NVC-SSL extends the spike-and-slab lasso methodology (Ročková and George, 2018) to the functional regression setting with dependent responses. NVC-SSL performs simultaneous estimation and variable selection of the functional components. Moreover, the NVC-SSL flexibly models the unknown within-subject covariance structure. This is in sharp contrast to frequentist penalized approaches to NVC models which have ignored these temporal correlations entirely or previous Bayesian approaches which have required the prespecification of a parametric covariance structure. Unlike frequentist approaches, the NVC-SSL model also employs a *non*-separable penalty which allows for automatic model complexity control and self-adaptivity to the true sparsity in the data.

For variable selection and estimation, we introduced an efficient ECM algorithm to rapidly obtain MAP estimates. For uncertainty quantification, we proposed an approximate MCMC algorithm. Both our ECM and approximate MCMC algorithms scale linearly in p and in N. We demonstrated through extensive simulation studies and a real data application that our method provides reliable variable selection, function estimation, and uncertainty quantification under a variety of within-subject correlation structures. NVC-SSL is also able to detect weak signals and capture many different function shapes, including functions with flat regions and non-time varying (constant) functions.

The NVC-SSL enjoys strong theoretical support. However, we have deferred the theoretical treatment of our method to a follow-up paper (Bai, 2023b). Bai (2023b) gives general sufficient conditions for adaptive posterior contraction in high-dimensional $p \gg n$ Bayesian NVC models (adaptive in the sense that the posterior can adapt to the unknown sparsity level and the unknown smoothness of the varying coefficient functions). The NVC-SSL prior is one particular choice of prior that can be shown to satisfy these sufficient conditions with well-chosen hyperparameters.

This paper has focused solely on the "large p" problem, where we implicitly assumed that N was not too large. The "small N, large p" scenario arises in many practical settings such as GWAS studies (Li et al., 2015; Johndrow et al., 2020). However, it is also worthwhile to explore scalable Bayesian NVC models in the "large N, large p" setting, where *both* N and

p could be massive. One possible direction is the prior-preconditioned conjugate gradient (PPCG) method of Nishimura and Suchard (2022). One of the benefits of the approach of Nishimura and Suchard (2022) is the fact that it bypasses matrix inversions entirely. In preliminary work, we did attempt to implement a version of the PPCG method for NVC-SSL. However, we found that when $N \ll p$, PPCG was actually *slower* than the *exact* MCMC algorithm that we introduced in Section 4. This is because *each* MCMC iteration of the PPCG method requires iteratively solving a linear system using conjugate gradient descent (CGD), and the number of iterations it took for the CGD to converge was often greater than N. Nevertheless, we believe that PPCG is a useful avenue to pursue if N and p are both large, and the cost of iteratively solving a linear system with CGD is minimal compared to the cost of using direct methods such as Cholesky decomposition.

Another possible direction for scalable uncertainty quantification is to extend the weighted Bayesian bootstrap (WBB) (Newton et al., 2021; Nie and Ročková, 2022) to the NVC setting. Roughly speaking, WBB methods approximate the posterior by performing MAP estimation on many independently perturbed datasets. Recently, in linear regression with i.i.d. errors, Nie and Ročková (2022) employed WBB to approximately sample from the posterior distribution under spike-and-slab lasso priors. The approach of Nie and Ročková (2022) is shown to scale favorably in both N and p. However, WBB requires the observations to be independent, which is *not* the case for the NVC models in this paper. An extension of WBB to the *dependent* data setting considered in this paper is also of interest.

Acknowledgments

The authors are grateful to the Action Editor and the anonymous referee for very constructive feedback which greatly improved this paper. This work was initiated when the first listed author was a postdoc at the Perelman School of Medicine, University of Pennsylvania (UPenn) mentored by the last two authors and the second listed author was an Assistant Professor at UPenn. Dr. Ray Bai was funded in part by generous funding from the National Science Foundation (DMS-2015528). Dr. Mary Boland was funded in part by generous funding from the Perelman School of Medicine. Dr. Yong Chen was supported in part by National Institutes of Health grants 1R01LM012607 and 1R01AI130460.

Appendix A. Additional simulation results

In this section, we compare the results obtained from MCMC for Experiments 1-5 in Section 5.1 to those obtained from the ECM algorithm. Namely, we assessed the performance of the estimated posterior mean as a point estimate. We also used MPM (Barbieri and Berger, 2004; Barbieri et al., 2021) to perform variable selection. Unlike the MAP estimator, the posterior mean under the NVC-SSL model is *not* exactly sparse. However, for spike-and-slab models, one can threshold the posterior inclusion probabilities $P(\tau_k = 1 \mid \boldsymbol{Y}), k = 1, \ldots, p$, to select variables. These posterior inclusion probabilities can be estimated as

$$\widehat{P}(\tau_k = 1 \mid \boldsymbol{Y}) = \frac{1}{T - B} \sum_{t=B+1}^{T} \tau_k^{(t)},$$

Table 6: Simulation results for NVC-SSL using the exact MCMC algorithm in Section 4.1 vs. the ECM algorithm in Section 3.1, averaged across 200 replicates. To better highlight the differences in estimation performance, we rescale the MSE by 100, i.e. we report MSE × 100 in the first column. The empirical standard error is reported in parentheses following the average. For MCMC, the posterior mean is used to compute the MSE and MSPE, and the MPM is used to select variables and compute Sens, Spec, and MCC. For ECM, the MAP estimator is used to compute all performance metrics.

Experiment 1: AR(1) covariance function

	MSE × 100	MSPE	Sens	Spec	MCC
MCMC	0.670 (0.095)	6.579 (2.297)	0.667 (0)	**1** (0)	0.815 (0)
ECM	**0.144** (0.049)	**4.057** (1.976)	**0.988** (0.052)	0.999 (0.002)	**0.948** (0.069)

Experiment 2: CS covariance function

	MSE × 100	MSPE	Sens	Spec	MCC
MCMC	0.679 (0.102)	6.597 (2.710)	0.667 (0)	**1** (0)	0.815 (0)
ECM	**0.113** (0.046)	**4.255** (2.254)	**0.989** (0.041)	0.999 (0.002)	**0.947** (0.067)

Experiment 3: SE covariance function

	MSE × 100	MSPE	Sens	Spec	MCC
MCMC	0.663 (0.098)	6.318 (2.187)	0.667 (0)	**1** (0)	0.815 (0)
ECM	**0.112** (0.056)	**4.448** (2.536)	**0.989** (0.053)	0.999 (0.002)	**0.946** (0.071)

Experiment 4: Periodic covariance function

	MSE × 100	MSPE	Sens	Spec	MCC
MCMC	0.663 (0.110)	6.511 (2.324)	0.667 (0)	**1** (0)	0.815 (0)
ECM	**0.106** (0.042)	**4.062** (2.134)	**0.991** (0.045)	0.999 (0.002)	**0.965** (0.054)

Experiment 5: Zero covariance function (i.i.d. errors)

	MSE × 100	MSPE	Sens	Spec	MCC
MCMC	0.619 (0.098)	5.198 (2.306)	0.667 (0)	**1** (0)	0.815 (0)
ECM	**0.062** (0.024)	**3.027** (2.269)	**0.999** (0.012)	0.999 (0.002)	**0.962** (0.055)

where $\tau_k^{(t)}$ is the tth MCMC sample drawn for τ_k in Step 4(a) of Algorithm 2, T is the total number of MCMC iterations, and B is the number of burnin samples. In the present context, MPM selects the kth varying coefficient $\beta_k(t)$ if $P(\tau_k = 1 \mid \boldsymbol{Y}) \geq 0.5$.

We repeated Experiments 1-5 from Section 5.1 for 200 replications each, where we used the exact MCMC algorithm in Section 4.1 to estimate the varying coefficients. The MAP estimator was used to initialize the MCMC algorithm, and the hyperparameters were the same as those in the ECM algorithm. We ran the algorithm for 2000 iterations, with a burnin period of 500 samples. The effective sample size prior to burnin was very close to 2000 for each of the basis coefficients in $\boldsymbol{\gamma}$, suggesting that the number of MCMC iterations we used was sufficient.

Table 7: Simulation results for estimation and variable selection of the nonzero varying coefficients $\beta_1(t)$, $\beta_2(t)$, $\beta_3(t)$, $\beta_4(t)$, $\beta_5(t)$, $\beta_6(t)$ using the exact MCMC algorithm in Section 4.1 vs. the ECM algorithm in Section 3.1, averaged across 200 replicates. "Proportion" gives the proportion of replicates that selected the varying coefficient.

Experiment 1: AR(1) covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
MCMC	0.252	0.693	0.344	0.440	0.657	0.177	1	1	0	0	1	1
ECM	**0.060**	**0.081**	**0.093**	**0.103**	**0.113**	**0.042**	1	1	**0.97**	**0.96**	1	1

Experiment 2: CS covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
MCMC	0.254	0.728	0.349	0.440	0.656	0.181	1	1	0	0	1	1
ECM	**0.066**	**0.080**	**0.094**	**0.094**	**0.114**	**0.045**	1	1	**0.965**	**0.97**	1	1

Experiment 3: SE covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
MCMC	0.256	0.680	0.345	0.435	0.622	0.179	1	1	0	0	1	1
ECM	**0.063**	**0.076**	**0.087**	**0.098**	**0.106**	**0.043**	1	1	**0.975**	**0.96**	1	1

Experiment 4: Periodic covariance function

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
MCMC	0.246	0.681	0.343	0.438	0.623	0.183	1	1	0	0	1	1
ECM	**0.065**	**0.079**	**0.092**	**0.098**	**0.110**	**0.042**	1	1	**0.97**	**0.975**	1	1

Experiment 5: Zero covariance function (i.i.d. errors)

	MSE						Proportion					
	β_1	β_2	β_3	β_4	β_5	β_6	β_1	β_2	β_3	β_4	β_5	β_6
MCMC	0.251	0.708	0.347	0.440	0.587	0.171	1	1	0	0	1	1
ECM	**0.039**	**0.041**	**0.045**	**0.048**	**0.082**	**0.026**	1	1	**0.97**	**0.975**	1	1

Tables 6 and 7 report our results from using MCMC to perform estimation and variable selection, contrasted with the results from using the ECM algorithm. Our results show that the MAP estimator obtained from the ECM algorithm gave superior variable selection, both overall (Table 6) *and* with respect to the six true nonzero varying coefficient functions (Table 7). In particular, the MAP estimator had lower average MSE (both overall and for $\beta_k(t), k = 1, \ldots, p$) across all the different scenarios. Table 7 shows that the MPM method consistently failed to select the weak signals β_3 and β_4, similar to the competing methods

NVC-gLASSO, NVC-gSCAD, and NVC-gMCP (Table 2). This is demonstrated in Figure 2, which shows that the MAP estimator is better able to detect and capture the true shape of smaller magnitude varying coefficient functions than the posterior mean.

On the other hand, uncertainty quantification from the 95% posterior credible intervals obtained from MCMC was quite good, with a pointwise ECP of 99.9% in all simulations. This is illustrated by the credible bands in Figure 2. We therefore conclude that the NVC-SSL MAP estimator is preferable for *variable selection* – especially in the presence of weak signals, while the MCMC algorithm is very useful for *uncertainty quantification.*

Appendix B. Proof of Proposition 1

Under the exact MCMC algorithm, the conditional distribution of $\boldsymbol{\gamma}$ in Step 6 of Algorithm 2 has the covariance matrix,

$$\boldsymbol{\Sigma}_{\boldsymbol{\gamma}} = \left(\boldsymbol{U}^\top \boldsymbol{U}/\sigma^2 + \boldsymbol{D}_{\boldsymbol{\xi}}^{-1}\right)^{-1} = \begin{pmatrix} \boldsymbol{U}_S^\top \boldsymbol{U}_S/\sigma^2 + \boldsymbol{D}_S^{-1} & \boldsymbol{U}_S^\top \boldsymbol{U}_{S^c}/\sigma^2 \\ \boldsymbol{U}_{S^c}^\top \boldsymbol{U}_S/\sigma^2 & \boldsymbol{U}_{S^c}^\top \boldsymbol{U}_{S^c}/\sigma^2 + \boldsymbol{D}_{S^c}^{-1} \end{pmatrix}^{-1} \triangleq \begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B}^\top & \boldsymbol{C} \end{pmatrix}^{-1}.$$

Since $\boldsymbol{A} = \boldsymbol{U}_S^\top \boldsymbol{U}_S/\sigma^2 + \boldsymbol{D}_S^{-1}$ and $\boldsymbol{C} = \boldsymbol{U}_{S^c}^\top \boldsymbol{U}_{S^c}/\sigma^2 + \boldsymbol{D}_{S^c}^{-1}$ are both positive-definite (with smallest eigenvalues greater than or equal to $[\max_{1 \le k \le p}\{\xi_k\}]^{-1} > 0$), we can write

$$\boldsymbol{\Sigma}_{\boldsymbol{\gamma}} = \begin{pmatrix} \boldsymbol{A}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^\top \boldsymbol{A}^{-1} & -\boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1} \\ -(\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^\top \boldsymbol{A}^{-1} & (\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1} \end{pmatrix}, \tag{33}$$

and the Schur complement $(\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1}$ is also positive-definite. Thus, noting that $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S} = \boldsymbol{A}^{-1}$ by (31), we have that

$$\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_S} = \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^\top \boldsymbol{A}^{-1}.$$

But for any $\boldsymbol{x} \in \mathbb{R}^{ds}$,

$$\begin{aligned} \boldsymbol{x}^\top(\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_s} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_s})\boldsymbol{x} &= \boldsymbol{x}^\top \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{x} \\ &= \|(\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1/2}\boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{x}\|_2^2 \ge 0. \end{aligned}$$

Thus, $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_S}$ is non-negative definite, i.e. $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_S} \ge \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_S}$.

Now, using the facts that $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} = \boldsymbol{D}_{S^c}^{-1}$, $\boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}} = (\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B})^{-1}$, and $\boldsymbol{B} = \boldsymbol{U}_S^\top \boldsymbol{U}_{S^c}/\sigma^2$ by (31) and (33), two applications of the Woodbury matrix identity give that

$$\begin{aligned} &\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}} \\ &\quad = \boldsymbol{D}_{S^c}\frac{\boldsymbol{U}_{S^c}^\top}{\sigma}\left(\boldsymbol{I}_N + \frac{\boldsymbol{U}_{S^c}\boldsymbol{D}_{S^c}\boldsymbol{U}_{S^c}^\top}{\sigma^2}\right)^{-1}\frac{\boldsymbol{U}_{S^c}}{\sigma}\boldsymbol{D}_{S^c} + \boldsymbol{C}^{-1}\boldsymbol{B}^\top(\boldsymbol{A}^{-1} + \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\boldsymbol{C}^{-1}. \end{aligned}$$

Now, for any $\boldsymbol{y} \in \mathbb{R}^{d(p-s)}$, we have

$$\begin{aligned} &\boldsymbol{y}^\top(\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}})\boldsymbol{y} \\ &= \boldsymbol{y}^\top \boldsymbol{D}_{S^c}\frac{\boldsymbol{U}_{S^c}^\top}{\sigma}\left(\boldsymbol{I}_N + \frac{\boldsymbol{U}_{S^c}\boldsymbol{D}_{S^c}\boldsymbol{U}_{S^c}^\top}{\sigma^2}\right)^{-1}\frac{\boldsymbol{U}_{S^c}}{\sigma}\boldsymbol{D}_{S^c}\boldsymbol{y} + \boldsymbol{y}^\top \boldsymbol{C}^{-1}\boldsymbol{B}^\top(\boldsymbol{A}^{-1} + \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{B}^\top)^{-1}\boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{y} \\ &= \left\| \left(\boldsymbol{I}_N + \frac{\boldsymbol{U}_{S^c}\boldsymbol{D}_{S^c}\boldsymbol{U}_{S^c}^\top}{\sigma^2}\right)^{-1/2}\frac{\boldsymbol{U}_{S^c}}{\sigma}\boldsymbol{D}_{S^c}\boldsymbol{y}\right\|_2^2 + \|(\boldsymbol{A}^{-1} + \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{B}^\top)^{-1/2}\boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{y}\|_2^2 \ \ge 0 + 0. \end{aligned}$$

Thus, $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} - \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}}$ is also non-negative definite, i.e. $\widetilde{\boldsymbol{\Sigma}}_{\boldsymbol{\gamma}_{S^c}} \ge \boldsymbol{\Sigma}_{\boldsymbol{\gamma}_{S^c}}$. □

References

Ray Bai. Bayesian group regularization in generalized linear models with a continuous spike-and-slab prior. *arXiv pre-print arXiv: 2007.07021*, 2023a.

Ray Bai. Adaptive posterior contraction for high-dimensional Bayesian varying coefficient models under shrinkage priors. *preprint*, 2023b.

Ray Bai, Gemma E. Moran, Joseph L. Antonelli, Yong Chen, and Mary R. Boland. Spike-and-slab group lassos for grouped regression and sparse generalized additive models. *Journal of the American Statistical Association*, 117(537):184–197, 2022.

Anjishnu Banerjee, David B. Dunson, and Surya T. Tokdar. Efficient Gaussian process regression for large datasets. *Biometrika*, 100(1):75–89, 2013.

Nilanjana Banerjee and Michael Q. Zhang. Identifying cooperativity among transcription factors controlling the cell cycle in yeast. *Nucleic Acids Research*, 31(23):7024–7031, 2003.

Maria M. Barbieri, James O. Berger, Edward I. George, and Veronika Ročková. The median probability model and correlated variables. *Bayesian Analysis*, 16(4):1085 – 1112, 2021.

Maria Maddalena Barbieri and James O. Berger. Optimal predictive model selection. *The Annals of Statistics*, 32(3):870 – 897, 2004.

Anirban Bhattacharya, Antik Chakraborty, and Bani K Mallick. Fast sampling with Gaussian scale mixture priors in high-dimensional regression. *Biometrika*, 103(4):985–991, 2016.

Clemens Biller and Ludwig Fahrmeir. Bayesian varying-coefficient models using adaptive regression splines. *Statistical Modelling*, 1(3):195–211, 2001.

Niloy Biswas, Lester Mackey, and Xiao-Li Meng. Scalable spike-and-slab. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 2021–2040, 17–23 Jul 2022.

Patrick Breheny and Jian Huang. Group descent algorithms for nonconvex penalized linear and logistic regression models with grouped predictors. *Statistics and Computing*, 25(2): 173–187, 2015.

Carlos M. Carvalho, Nicholas G. Polson, and James G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

Huaihou Chen and Yuanjia Wang. A penalized spline approach to functional mixed effects model analysis. *Biometrics*, 67(3):861–870, 2011.

Abhirup Datta, Sudipto Banerjee, Andrew O. Finley, and Alan E. Gelfand. Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514):800–812, 2016.

Ulrik de Lichtenberg, Lars Juhl Jensen, Anders Fausbøll, Thomas S. Jensen, Peer Bork, and Søren Brunak. Comparison of computational methods for the identification of cell cycle-regulated genes. *Bioinformatics*, 21(7):1164–1171, 2005.

Sameer K. Deshpande, Veronika Ročková, and Edward I. George. Simultaneous variable and covariance selection with the multivariate spike-and-slab lasso. *Journal of Computational and Graphical Statistics*, 28(4):921–931, 2019.

Sameer K. Deshpande, Ray Bai, Cecilia Balocchi, Jennifer E. Starling, and Jordan Weiss. VCBART: Bayesian trees for varying coefficients. *arXiv pre-print arXiv: 2003.06416*, 2020.

Jianqing Fan and Jin-Ting Zhang. Two-step estimation of functional linear models with applications to longitudinal data. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 62(2):303–322, 2000.

Jianqing Fan and Wenyang Zhang. Statistical methods with varying coefficient models. *Statistics and Its Interface*, 1(1):179–195, 2008.

Yingying Fan and Runze Li. Variable selection in linear mixed effects models. *The Annals of Statistics*, 40(4):2043 – 2068, 2012.

Lingrui Gan, Naveen N. Narisetty, and Feng Liang. Bayesian regularization for graphical models with unequal shrinkage. *Journal of the American Statistical Association*, 114 (527):1218–1231, 2019a.

Lingrui Gan, Xinming Yang, Naveen Narisetty, and Feng Liang. Bayesian joint estimation of multiple graphical models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 9799–9809. Curran Associates, Inc., 2019b.

Rajarshi Guhaniyogi, Cheng Li, Terrance D. Savitsky, and Sanvesh Srivastava. Distributed Bayesian varying coefficient modeling using a Gaussian process prior. *Journal of Machine Learning Research*, 23(84):1–59, 2022.

Rajarshi Guhaniyogi, Laura Baracaldo, and Sudipto Banerjee. Bayesian data sketching for varying coefficient regression models. *preprint*, 2023.

Wensheng Guo. Functional mixed effects models. *Biometrics*, 58(1):121–128, 2002.

Trevor Hastie and Robert Tibshirani. Varying-coefficient models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 55(4):757–796, 1993.

Donald R. Hoover, John A. Rice, Colin O. Wu, and Li-Ping Yang. Nonparametric smoothing estimates of time-varying coefficient models with longitudinal data. *Biometrika*, 85(4): 809–822, 1998.

Jianhua Z. Huang, Colin O. Wu, and Lan Zhou. Polynomial spline estimation and inference for varying coefficient models with longitudinal data. *Statistica Sinica*, 14:763–788, 2004.

Zhipeng Huang, Jialiang Li, David Nott, Lei Feng, Tze-Pin Ng, and Tien-Yin Wong. Bayesian estimation of varying-coefficient models with missing data, with application to the singapore longitudinal aging study. *Journal of Statistical Computation and Simulation*, 85(12):2364–2377, 2015.

John Hughes and Murali Haran. Dimension reduction and alleviation of confounding for spatial generalized linear mixed models. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 75(1):139–159, 2013.

James Johndrow, Paulo Orenstein, and Anirban Bhattacharya. Scalable approximate MCMC algorithms for the horseshoe prior. *Journal of Machine Learning Research*, 21 (73):1–61, 2020.

Robert T. Krafty, Phyllis A. Gimotty, David Holtz, George Coukos, and Wensheng Guo. Varying coefficient model with unknown within-subject covariance for analysis of tumor growth curves. *Biometrics*, 64:1023–1031, 2008.

Tong Ihn Lee, Nicola J. Rinaldi, François Robert, Duncan T. Odom, Ziv Bar-Joseph, Georg K. Gerber, Nancy M. Hannett, Christopher T. Harbison, Craig M. Thompson, Itamar Simon, Julia Zeitlinger, Ezra G. Jennings, Heather L. Murray, D. Benjamin Gordon, Bing Ren, John J. Wyrick, Jean-Bosco Tagne, Thomas L. Volkert, Ernest Fraenkel, David K. Gifford, and Richard A. Young. Transcriptional regulatory networks in saccharomyces cerevisiae. *Science*, 298(5594):799–804, 2002.

Jiahan Li, Zhong Wang, Runze Li, and Rongling Wu. Bayesian group lasso for nonparametric varying-coefficient models with application to functional genome-wide association studies. *The Annals of Applied Statistics*, 9(2):640–664, 06 2015.

Zehang Li, Tyler Mccormick, and Samuel Clark. Bayesian joint spike-and-slab graphical lasso. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3877–3885, Long Beach, California, USA, 09–15 Jun 2019.

K Y Liang and S L Zeger. Regression analysis for correlated data. *Annual Review of Public Health*, 14(1):43–68, 1993.

Shelley H. Liu, Jennifer F. Bobb, Birgit Claus Henn, Chris Gennings, Lourdes Schnaas, Martha Tellez-Rojo, David Bellinger, Manish Arora, Robert O. Wright, and Brent A. Coull. Bayesian varying coefficient kernel machine regression to assess neurodevelopmental trajectories associated with exposure to complex mixtures. *Statistics in Medicine*, 25 (3):665–683, 2018.

Gemma E. Moran, Veronika Ročková, and Edward I. George. Spike-and-slab lasso biclustering. *The Annals of Applied Statistics*, 15(1):148 – 173, 2021.

Naveen N. Narisetty, Juan Shen, and Xuming He. Skinny Gibbs: A consistent and scalable Gibbs sampler for model selection. *Journal of the American Statistical Association*, 114 (527):1205–1217, 2019.

Michael A. Newton, Nicholas G. Polson, and Jianeng Xu. Weighted Bayesian bootstrap for scalable posterior distributions. *Canadian Journal of Statistics*, 49(2):421–437, 2021.

Lizhen Nie and Veronika Ročková. Bayesian bootstrap spike-and-slab lasso. *Journal of the American Statistical Association (to appear)*, 2022.

Akihiko Nishimura and Marc A. Suchard. Prior-preconditioned conjugate gradient method for accelerated Gibbs sampling in "large n, large p" Bayesian sparse regression. *Journal of the American Statistical Association (to appear)*, 2022.

Nicholas G Polson, James G Scott, and Jesse Windle. Bayesian inference for logistic models using Pólya–gamma latent variables. *Journal of the American Statistical Association*, 108 (504):1339–1349, 2013.

Annie Qu and Runze Li. Quadratic inference functions for varying-coefficient models with longitudinal data. *Biometrics*, 62:379–391, 2006.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning.* MIT Press, 2006.

Stephen Reid, Robert Tibshirani, and Jerome Friedman. A study of error variance estimation in lasso regression. *Statistica Sinica*, 26(1):35–67, 2016.

John A. Rice. Functional and longitudinal data analysis: Perspectives on smoothing. *Statistica Sinica*, 14(3):631–647, 2004.

Veronika Ročková and Edward I. George. Fast Bayesian factor analysis via automatic rotations to sparsity. *Journal of the American Statistical Association*, 111(516):1608–1622, 2016.

Veronika Ročková and Edward I. George. The spike-and-slab LASSO. *Journal of the American Statistical Association*, 113(521):431–444, 2018.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461 – 464, 1978.

James G Scott and James O Berger. Bayes and empirical-Bayes multiplicity adjustment in the variable-selection problem. *The Annals of Statistics*, 38(5):2587–2619, 2010.

Itamar Simon, John Barnett, Nancy Hannett, Christopher T Harbison, Nicola J Rinaldi, Thomas L Volkert, John J Wyrick, Julia Zeitlinger, David K Gifford, Tommi S Jaakkola, and Richard A Young. Serial regulation of transcriptional regulators in the yeast cell cycle. *Cell*, 106(6):697 – 708, 2001.

Paul T. Spellman, Gavin Sherlock, Michael Q. Zhang, Vishwanath R. Iyer, Kirk Anders, Michael B. Eisen, Patrick O. Brown, David Botstein, and Bruce Futcher. Comprehensive identification of cell cycle–regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, 9(12):3273–3297, 1998.

William T. Stephenson, Soumya Ghosh, Tin D. Nguyen, Mikhail Yurochkin, Sameer Deshpande, and Tamara Broderick. Measuring the robustness of Gaussian processes to kernel choice. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 3308–3331, 28–30 Mar 2022.

Zaixiang Tang, Yueping Shen, Xinyan Zhang, and Nengjun Yi. The spike-and-slab lasso generalized linear models for prediction and associated genes detection. *Genetics*, 205(1): 77–88, 2017.

Huai-Kuang Tsai, Henry Horng-Shing Lu, and Wen-Hsiung Li. Statistical methods for identifying yeast cell cycle transcription factors. *Proceedings of the National Academy of Sciences*, 102(38):13532–13537, 2005.

Stéphanie van der Pas, Botond Szabó, and Aad van der Vaart. Uncertainty quantification for the horseshoe (with discussion). *Bayesian Analysis*, 12(4):1221 – 1274, 2017.

Hansheng Wang and Yingcun Xia. Shrinkage estimation of the varying coefficient model. *Journal of the American Statistical Association*, 104(486):747–757, 2009.

Lifeng Wang, Hongzhe Li, and Jianhua Z. Huang. Variable selection in nonparametric varying-coefficient models for analysis of repeated measurements. *Journal of the American Statistical Association*, 103(484):1556–1569, 2008.

Fengrong Wei, Jian Huang, and Hongzhe Li. Variable selection and estimation in high-dimensional varying-coefficient models. *Statistica Sinica*, 21:1515–1540, 2011.

Colin O. Wu and Chin-Tsang Chiang. Kernel smoothing on varying coefficient models with longitudinal dependent variable. *Statistica Sinica*, 10(2):433–456, 2000.

Lan Xue and Annie Qu. Variable selection in high-dimensional varying-coefficient models with global optimality. *Journal of Machine Learning Research*, 13(1):1973–1998, 2012.

William Weimin Yoo and Subhashis Ghosal. Supremum norm posterior contraction and credible sets for nonparametric multivariate regression. *The Annals of Statistics*, 44(3): 1069–1102, 2016.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.