Bayesian Analysis (2022) **17**, Number 3, pp. 879–900

An Ensemble EM Algorithm for Bayesian Variable Selection

Jin Wang*, Yunbo Ouyang†, Yuan Ji‡, and Feng Liang§

Abstract. We study the Bayesian approach to variable selection for linear regression models. Motivated by a recent work by Ročková and George (2014), we propose an EM algorithm that returns the MAP estimator of the set of relevant variables. Due to its particular updating scheme, our algorithm can be implemented efficiently without inverting a large matrix in each iteration and therefore can scale up with big data. We also have showed that the MAP estimator returned by our EM algorithm achieves variable selection consistency even when p diverges with n. In practice, our algorithm could get stuck with local modes, a common problem with EM algorithms. To address this issue, we propose an ensemble EM algorithm, in which we repeatedly apply our EM algorithm to a subset of the samples with a subset of the covariates, and then aggregate the variable selection results across those bootstrap replicates. Empirical studies have demonstrated the superior performance of the ensemble EM algorithm.

Keywords: Bayesian variable selection, EM, Bayesian bootstrap, asymptotic consistency.

1 Introduction

Consider a simple linear regression model with Gaussian noise:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e}, \tag{1.1}$$

where $\mathbf{y} = (y_1, \ldots, y_n)^T$ is the $n \times 1$ response vector, $\mathbf{X}$ is the $n \times p$ design matrix, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ is the unknown regression coefficient vector, and $\mathbf{e} = (e_1, \ldots, e_n)^T$ is a vector of i.i.d. Gaussian random variables with mean zero and variance σ^2. In many real applications, such as bioinformatics and image analysis, where linear regression models have been routinely used, the number of potential predictors (i.e., p) is large but only a small fraction of them are believed to be relevant. Therefore model (1.1) is often assumed to be "sparse" in the sense that most of the coefficients β_j are zero. Estimating the set of relevant variables, $S = \{j : \beta_j \neq 0\}$, is an important problem in modern statistical analysis. Many variable selection algorithms have been proposed in the framework of penalized likelihood, such as LASSO (Tibshirani, 1996), SCAD (Fan and Li, 2001) and MCP (Zhang et al., 2010), just to name a few; for a review of this

*Amazon.com, Inc., Seattle, WA, USA, jinwangmls@gmail.com
†LinkedIn Corporation, Sunnyvale, CA, USA, youyang@linkedin.com
‡Department of Public Health Sciences, University of Chicago, Chicago, IL, USA, koaeraser@gmail.com
§Department of Statistics, University of Illinois at Urbana-Champaign, Champaign, IL, USA, liangf@illinois.edu

 https://doi.org/10.1214/21-BA1275

area, see the book by Bühlmann and van de Geer (2011) and a selective review article by Fan and Lv (2010).

The Bayesian approach to variable selection is conceptually simple and straightforward. First introduce a p-dimensional binary vector $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)^T$ to index all the 2^p subsets of variables, where $\gamma_j = 1$ if the jth variable is included in the model and $\gamma_j = 0$ if excluded. Usually γ_js are modeled by independent Bernoulli distributions. Given $\boldsymbol{\gamma}$, a popular prior choice for $\boldsymbol{\beta}$ is the "spike and slab" prior (Mitchell and Beauchamp, 1988):

$$\pi(\beta_j \mid \gamma_j) = \begin{cases} \delta_0(\beta_j), & \text{if } \gamma_j = 0; \\ g(\beta_j), & \text{if } \gamma_j = 1, \end{cases} \tag{1.2}$$

where $\delta_0(\cdot)$ is the Kronecker delta function corresponding to the density function of a point mass at 0 and g is a continuous density function. After specifying priors on all the unknowns, one needs to calculate the posterior distribution. Most algorithms for Bayesian variable selection rely on MCMC algorithms, such as Gibbs or Metropolis-Hasting, to obtain the posterior distribution; for a review of recent developments in this area, see O'Hara and Sillanpää (2009). MCMC algorithms, however, are insufficient to meet the growing demand for scalability from real applications. Since the primary goal here is variable selection, we focus on efficient algorithms that return the MAP estimator of $\boldsymbol{\gamma}$, as an alternative to these MCMC-based sampling methods that return the whole posterior distribution of all the unknown parameters.

Recently, Ročková and George (2014) proposed a simple, elegant EM algorithm for Bayesian variable selection. They adopted a continuous version of the "spike and slab" prior in which the spike and the slab components in (1.2) are two normal distributions with different variances (George and McCulloch, 1993), and proposed an EM algorithm to obtain the MAP estimator of the regression coefficients $\boldsymbol{\beta}$. The MAP estimator $\hat{\boldsymbol{\beta}}_{\text{MAP}}$, however, is not sparse, so an additional thresholding step is needed to estimate $\boldsymbol{\gamma}$.

In this paper, we develop an EM algorithm that directly returns the MAP estimator of $\boldsymbol{\gamma}$. We adopt the same continuous "spike and slab" prior as do Ročková and George (2014), but while their algorithm returns $\hat{\boldsymbol{\beta}}_{\text{MAP}}$ by treating $\boldsymbol{\gamma}$ as latent, our approach treats $\boldsymbol{\beta}$ as latent and returns $\hat{\boldsymbol{\gamma}}_{\text{MAP}}$, the MAP estimator of the model index. The special structure of our EM algorithm allows us to use a computational trick to avoid inverting a big matrix at each iteration, which seems unavoidable when using the algorithm of Ročková and George (2014). Further we can show that $\hat{\boldsymbol{\gamma}}_{\text{MAP}}$ returned by our EM algorithm achieves asymptotic consistency even when p diverges to infinity with increasing sample size n.

Although shown to achieve selection consistency, in practice, our EM algorithm could get stuck at a local mode due to the large discrete space in which $\boldsymbol{\gamma}$ lies. Borrowing the idea of bagging, we propose an ensemble version of our EM algorithm (which we call BBEM): apply our EM algorithm to multiple Bayesian bootstrap (BB) copies of the data, and then aggregate the variable selection results. Bayesian bootstrap for variable selection was explored before by Clyde and Lee (2001) for the purpose of prediction, where models built on different bootstrap copies are combined to predict the response. But the focus of our approach is to summarize the evidence for variable relevance from

multiple BB copies, which is similar in nature to several frequentist ensemble methods for variable selection, such as the AIC ensemble (Zhu and Chipman, 2006), stability selection (Meinshausen and Bühlmann, 2010), and random Lasso (Wang et al., 2011).

The rest of the paper is organized as follows. Section 2 describes our method in detail, Section 3 presents the asymptotic results, and Section 4 describes the BBEM algorithm. Empirical studies are presented in Section 5, conclusions in Section 6, and technical proofs in Section 7.

2 Method

2.1 Prior Specification

We adopt the continuous version of "spike and slab" prior for each β_j, i.e., a mixture of two normal distributions with mean zero and different variances:

$$\pi(\beta_j \mid \sigma, \gamma_j) = \begin{cases} \mathsf{N}(0, \sigma^2 v_0), & \text{if } \gamma_j = 0; \\ \mathsf{N}(0, \sigma^2 v_1), & \text{if } \gamma_j = 1, \end{cases} \tag{2.1}$$

where $v_1 > v_0 > 0$. Alternatively, we can write the prior of $\boldsymbol{\beta}$ as a product of

$$\pi(\beta_j \mid \sigma^2, \gamma_j) = \mathsf{N}(0, \sigma^2 d_{\gamma_j}), \quad j = 1, \ldots, p,$$

where

$$d_{\gamma_j} = \gamma_j v_1 + (1 - \gamma_j) v_0.$$

We shall discuss the choice of the tuning parameters v_0 and v_1 in our asymptotic analysis in Section 3 and in our empirical studies in Section 5.

For the remaining parameters, we specify independent Bernoulli priors for elements of $\boldsymbol{\gamma}$, and conjugate Beta and Inverse Gamma priors for θ and σ^2 as follows:

$$\begin{aligned} \pi(\boldsymbol{\gamma} \mid \theta) &= \mathsf{Bern}(\theta), \\ \pi(\theta) &= \mathsf{Beta}(a_0, b_0), \\ \pi(\sigma^2) &= \mathsf{IG}(\nu_0/2, \nu_0 \lambda_0/2). \end{aligned}$$

For hyper-parameters $(a_0, b_0, \nu_0, \lambda_0)$, we suggest the following non-informative choices unless prior knowledge is available:

$$a_0 = b_0 = 1.1, \quad \nu_0 = \lambda_0 = 1. \tag{2.2}$$

2.2 The EM Algorithm

With the Gaussian model and prior distributions specified above, we write the full posterior distribution as follows:

$$\pi(\boldsymbol{\gamma}, \boldsymbol{\beta}, \theta, \sigma^2 \mid \mathbf{y}) \;\propto\; p(\mathbf{y} \mid \boldsymbol{\beta}, \sigma^2) \times \pi(\boldsymbol{\beta} \mid \sigma, \boldsymbol{\gamma}) \times \pi(\boldsymbol{\gamma} \mid \theta) \times \pi(\theta) \times \pi(\sigma^2).$$

Treating $\boldsymbol{\beta}$ as the latent variable, we derive an EM algorithm that returns the MAP estimator of parameters $\Theta = (\boldsymbol{\gamma}, \sigma^2, \theta)$; note that we have switched the roles of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ compared to the approach of Ročková and George (2014).

E Step

The objective function Q at the $(t+1)$th iteration in an EM algorithm is defined as the integrated logarithm of the full posterior with respect to $\boldsymbol{\beta}$, given $\mathbf{y}$ and $\Theta^{(t)} = (\boldsymbol{\gamma}^{(t)}, \sigma^2_{(t)}, \theta^{(t)})$, the parameter values from the previous iteration:

$$\begin{aligned} Q(\Theta \mid \Theta^{(t)}) &= \mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}} \log \pi(\Theta, \boldsymbol{\beta} \mid \mathbf{y}) \\ &= -\frac{1}{2\sigma^2} \mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}} \Big[\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^{p} \frac{\beta_j^2}{d_{\gamma_j}} \Big] + F(\Theta), \end{aligned}$$

where

$$\begin{aligned} F(\Theta) \quad = \quad & -\frac{n+p}{2} \log \sigma^2 - \frac{1}{2} \sum_{j=1}^{p} \log d_{\gamma_j} + \log \pi(\boldsymbol{\gamma}|\theta) \\ & + \log \pi(\theta) + \log \pi(\sigma^2) + \textit{constant} \end{aligned}$$

is a function of Θ not depending on $\boldsymbol{\beta}$.

It is easy to show that given $\Theta^{(t)}$ and $\mathbf{y}$, $\boldsymbol{\beta}$ follows a normal distribution with mean $\mathbf{m}$ and covariance matrix $\sigma^2_{(t)}\mathbf{V}$, where

$$\begin{aligned} \mathbf{m} \quad &= \quad \mathbf{V}^{-1}\mathbf{X}^T\mathbf{y}, \quad \mathbf{V} = \big(\mathbf{X}^T\mathbf{X} + D^{-1}_{\boldsymbol{\gamma}^{(t)}}\big)^{-1}, \\ D_{\boldsymbol{\gamma}^{(t)}} \quad &= \quad \operatorname{diag}\Big(d_{\gamma_j^{(t)}}\Big)_{j=1}^{p} = \operatorname{diag}\Big(\gamma_j^{(t)} v_1 + (1-\gamma_j^{(t)}) v_0\Big)_{j=1}^{p}. \end{aligned} \tag{2.3}$$

Then the two expectation terms in (2.3) can be expressed as:

$$\mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}} \left\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\right\|^2 = \sigma^2_{(t)} \operatorname{tr}(\mathbf{X}\mathbf{V}\mathbf{X}^T) + \left\|\mathbf{y} - \mathbf{X}\mathbf{m}\right\|^2, \tag{2.4}$$

$$\mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}} \sum_{j=1}^{p} \frac{\beta_j^2}{d_{\gamma_j}} = \sum_{j=1}^{p} \frac{\sigma^2_{(t)} V_{jj} + m_j^2}{(1-\gamma_j^{(t)}) v_0 + \gamma_j^{(t)} v_1}. \tag{2.5}$$

M Step

We update each parameter in $(\boldsymbol{\gamma}, \theta, \sigma)$ sequentially by holding others fixed to maximize the objective function Q, as in the ECM algorithm (Meng and Rubin, 1993).

1. **Update γ_js.** The terms involving γ_j in (2.3) are

$$-\frac{1}{2\sigma^2_{(t)}} \mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}} \left[\frac{\beta_j^2}{d_{\gamma_j}}\right] - \frac{1}{2} \log d_{\gamma_j} + \log \pi(\gamma_j \mid \theta^{(t)}). \tag{2.6}$$

Plugging in $\gamma_j = 0$ and $\gamma_j = 1$ to (2.6) respectively, we have

$$\gamma_j^{(t+1)} = 1, \quad \text{if} \quad \mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}} \big[\beta_j^2\big] > r^{(t)}, \tag{2.7}$$

where $\mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}}[\beta_j^2]$ can be computed based on (2.3) and

$$r^{(t)} = \frac{\sigma_{(t)}^2}{1/v_0 - 1/v_1}\Big(\log\frac{v_1}{v_0} - 2\log\frac{\theta^{(t)}}{1-\theta^{(t)}}\Big).$$

2. **Update** (σ^2, θ)**.** Given $\boldsymbol{\gamma}^{(t+1)}$, the updating equations for the other two parameters are given by

$$\sigma_{(t+1)}^2 = \frac{\mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}}\Big[\|\mathbf{y}-\mathbf{X}\boldsymbol{\beta}\|^2 + \sum_{j=1}^p \beta_j^2/d_{\gamma_j^{(t+1)}}\Big] + v_0\lambda_0}{n+p+v_0}, \tag{2.8}$$

$$\theta^{(t+1)} = \frac{\sum_{j=1}^p \gamma_j^{(t+1)} + a_0 - 1}{p + a_0 + b_0 - 2}. \tag{2.9}$$

Due to the conjugate prior (2.1) specified on $\boldsymbol{\beta}$, the update equation (2.8) for σ^2 has a weighted sum of squares of $\boldsymbol{\beta}$ in the numerator and p in the denominator, which may cause σ^2 to be underestimated when p is large and $\boldsymbol{\beta}$ is sparse. As shown in our asymptotic analysis in Section 3, however, variable selection accuracy is not sensitive to the estimation accuracy of σ^2. In practice, to alleviate this problem we suggest that the linear model be refitted with selected variables. Alternatively, as one reviewer suggested, one could use a non-conjugate prior for $\boldsymbol{\beta}$ leaving out σ^2 from the prior variance.

Stopping Rule

The EM algorithm alternates between the E-step and M-step until convergence. A natural stopping criterion is to check whether the change of the objective function Q has become small. Evaluating the Q function, however, is time consuming. To reduce the computational cost for evaluating the Q function, we adopt a different stopping rule, as our main focus is $\boldsymbol{\gamma}$: halt when the estimate $\boldsymbol{\gamma}^{(t)}$ stays the same for k_0 iterations. In practice, we suggest setting $k_0 = 3$. The pseudo code of this EM algorithm is summarized in Algorithm 1.

2.3 Computational Cost

At each E-step, updating the posterior of $\boldsymbol{\beta}$ given other parameters in (2.3) requires inverting a $p \times p$ matrix

$$\mathbf{V}_{(t)} = (\mathbf{X}^T\mathbf{X} + D_{\boldsymbol{\gamma}^{(t)}}^{-1})^{-1}, \tag{2.10}$$

which is the major computational burden of our algorithm. When $p > n$, we can use the Sherman-Morrison-Woodbury formula to compute the inverse of an $n \times n$ matrix. So the computational cost at each iteration is of order $O(\min(n,p)^3)$, which, however, is still time-consuming when both n and p are large.

Note that the only change in (2.10) from iteration to iteration is $D_{\boldsymbol{\gamma}^{(t)}}$, a diagonal matrix depending on the binary vector $\boldsymbol{\gamma}^{(t)}$. From our experience, only a small fraction

Algorithm 1: EM

Input: $\mathbf{X}, \mathbf{y}, v_0, v_1, a_0, b_0, \nu_0, \lambda_0$
Initialize $\Theta^{(0)}$;
E-step: Calculate the two expectations in (2.4) and (2.5), denoted as $EE^{(0)}$;
for $t = 1 : maxIter$ **do**
  M-step: Update $\Theta^{(t)}$ using (2.7), (2.8), (2.9);
  E-step: Update $EE^{(t)}$ using (2.4), (2.5);
  if $\boldsymbol{\gamma}^{(t)}$ *stays the same for* $k_0 = 3$ *iterations* **then**
    break;
  end
end
Return $\boldsymbol{\gamma}$;

of $\{\gamma_j^{(t)}\}_{j=1}^p$ are changed at each iteration after the first few iterations. So we propose to use the following recursive formula to compute (2.10):

$$\begin{aligned} \mathbf{V}_{(t)} &= (\mathbf{X}^T\mathbf{X} + D^{-1}_{\boldsymbol{\gamma}^{(t-1)}} + D^{-1}_{\boldsymbol{\gamma}^{(t)}} - D^{-1}_{\boldsymbol{\gamma}^{(t-1)}})^{-1} \\ &= (\mathbf{V}^{-1}_{(t-1)} + D^{-1}_{\boldsymbol{\gamma}^{(t)}} - D^{-1}_{\boldsymbol{\gamma}^{(t-1)}})^{-1}, \end{aligned} \tag{2.11}$$

where $D^{-1}_{\boldsymbol{\gamma}^{(t)}} - D^{-1}_{\boldsymbol{\gamma}^{(t-1)}}$ is a diagonal matrix with the jth diagonal entry being non-zero only if the inclusion/exclusion status, i.e., the value of γ_j, is changed from the previous iteration. Let l denote the number of variables whose γ_j values are changed from iteration $(t-1)$ to t. Then $D^{-1}_{\boldsymbol{\gamma}^{(t)}} - D^{-1}_{\boldsymbol{\gamma}^{(t-1)}}$ is a rank l matrix. We can further reduce the computational complexity from $O(\min(n,p)^3)$ to $O(l^3)$ by applying the Woodbury formula to (2.11).

For example, without loss of generality, suppose only the first l covariates have their γ_j values changed. Then, we can write

$$D^{-1}_{\boldsymbol{\gamma}^{(t)}} - D^{-1}_{\boldsymbol{\gamma}^{(t-1)}} = U_{p\times l} A_{l\times l} U^T,$$

where $A = \left(\frac{1}{v_0} - \frac{1}{v_1}\right)\mathrm{diag}(2\gamma_j^{(t)} - 1)_{j=1}^l$ and U consists of the first l columns from $\mathbf{I}_p$. Applying the Woodbury formula, we have

$$\mathbf{V}_{(t)} = \mathbf{V}_{(t-1)} - \mathbf{V}_{(t-1)}U(A^{-1} + U^T\mathbf{V}_{(t-1)}U)^{-1}U^T\mathbf{V}_{(t-1)},$$

where we need to invert only an $l \times l$ matrix $(A^{-1} + U^T\mathbf{V}_{(t-1)}U)$.

3 Asymptotic Consistency

In this section, we study the asymptotic property of the MAP estimator returned by our EM algorithm. Assume that the data are generated from a Gaussian regression model:

$$\mathbf{y}_n \sim \mathsf{N}_n\big(\mathbf{X}_n\boldsymbol{\beta}_n^*, \sigma^2\mathbf{I}_n\big).$$

Here we allow the dimension $p = p_n$ to diverge with n, and also, to vary with n, the true coefficient vector $\boldsymbol{\beta}_n^*$ and the true model index $\boldsymbol{\gamma}_n^*$, where $\gamma_{nj}^* = 1$ if $\beta_{nj}^* \neq 0$ and $\gamma_{nj}^* = 0$ if $\beta_{nj}^* = 0$. Next we show that, asymptotically, our EM algorithm can return us the correct model index $\boldsymbol{\gamma}_n^*$ with probability approaching unity.

First we list some technical conditions needed in our proof.

(A1) Condition on the design matrix: $\lambda_{n1}(\mathbf{X}_n^T\mathbf{X}_n)^{-1} = O(n^{-\eta_1})$, $0 < \eta_1 \leq 1$, where $\lambda_{n1}(A)$ denotes the smallest eigenvalue of matrix A.

(A2) Condition on the sparsity level: $\|\boldsymbol{\beta}_n^*\|_2 = O(n^{\eta_2})$, $0 < \eta_2 < \eta_1$. This condition controls the L_2 norm of the true regression coefficient vector. Similar conditions on the L_2 or L_1 norm of $\boldsymbol{\beta}_n^*$ have been used in other work, such as Shao and Deng (2012) and Loh et al. (2017).

(A3) Beta-min condition:

$$\liminf_n \frac{\min\left\{|\beta_{nj}^*|, \gamma_{nj}^* = 1\right\}}{n^{(\eta_3-1)/2}} \geq M, \ 0 \leq \eta_3 < 1,$$

where M is a positive constant. This is a common condition in the literature of selection consistency (Bühlmann and van de Geer, 2011); it requires that the minimal non-zero coefficient not goes to zero at a rate faster than $1/\sqrt{n}$. In the traditional asymptotic setting where $\boldsymbol{\beta}_n^*$ is fixed, we have $\eta_3 = 0$.

(A4) Condition on hyper-parameters: assume that $\log[\hat{\theta}_n/(1-\hat{\theta}_n)]$ and $\hat{\sigma}_n^2$ are bounded.

(A5) Condition on tuning parameters: assume that v_1 is fixed at some constant and that v_0 satisfies

$$0 < v_0 = O(n^{-r_0}), \quad 1 - \eta_3 < r_0 < \min\left\{\eta_1 - \alpha, \frac{2}{3}(\eta_1 - \eta_2)\right\},$$

where $0 < \alpha < 1$ is the rate of the dimension $p = O(n^\alpha)$.

Theorem 3.1. *Assume (A1)–(A5) and $p = O(n^\alpha)$ where $0 \leq \alpha < 1$; then the model returned by our EM algorithm, $\hat{\gamma}_n$, achieves model selection consistency, namely,*

$$\mathbb{P}(\hat{\boldsymbol{\gamma}}_n = \boldsymbol{\gamma}_n^*) \to 1, \quad \text{as } n \to \infty. \tag{3.1}$$

Proof. See Section 7.1. □

Remark 1. Our condition on the design matrix (A1) is much weaker than the *Irrepresentable Condition* needed for LASSO (Wainwright, 2009; Zhao and Yu, 2006). In the classical asymptotic setting for linear regression when p is fixed, it is common to assume that $(\mathbf{X}_n^T\mathbf{X}_n)/n$ converges to a full-rank matrix, which satisfies our condition with $\eta_1 = 1$. Having a design matrix satisfying $\eta_1 = 1$ represents an ideal setting where the smallest eigenvalue of $(\mathbf{X}_n^T\mathbf{X}_n)/n$ is still lower-bounded by a positive constant. But this may not hold when p diverges with n. Our condition (A1) allows the smallest eigenvalue of $(\mathbf{X}_n^T\mathbf{X}_n)/n$ to go to zero at a rate slower than $\frac{1}{n^{1-\eta_1}}$.

Remark 2. To satisfy our condition on hyper-parameters (A4), we could either fix $\hat{\theta}_n$ and $\hat{\sigma}_n$ to be, or upper-bounded by, some constants in our algorithm so that they will not take extreme values. In simulations, we recommend (2.2) as the choice for hyper-parameters unless p is large. Note that our recommendation differs from the ones in Castillo et al. (2015) and Narisetty et al. (2014) because the notion of selection consistency studied in the two aforementioned papers is different from ours. In a nutshell, for the MAP estimator to achieve selection consistency defined in (3.1), we only need the posterior probability on the true model γ_n^* to be the largest among all 2^p models, while selection consistency in Castillo et al. (2015) and Narisetty et al. (2014) requires the posterior probability on γ_n^* to go to unity.

Remark 3. Our proof can be easily extended to cover the case when $p \gg n$; we discuss the proof and related assumptions in Section 7.2. Although our EM algorithm is shown to achieve selection consistency theoretically, in practice, we find it not performing well with high-dimensional data. This is why we propose a variation of our algorithm in the next section.

4 Bayesian Bootstrap

A common issue with the EM algorithm is that it could be trapped at local modes. Standard remedies are available for dealing with this issue—for instance, trying a set of different initial values or utilizing more advanced optimization procedures at the M-step. Since our EM algorithm is searching for the optimal $\boldsymbol{\gamma}$ over a big discrete space (all p-dimensional binary vectors), these remedies are less useful when p is large.

When performing optimization with respect to $\boldsymbol{\gamma}$, a discrete vector, the resulting solution is often not stable. Bagging is an easy but powerful method (Breiman, 1996) to alleviate this problem; it consists of applying the same algorithm to multiple bootstrap copies of the data and then aggregating the final results. We propose the following ensemble EM algorithm, in which we repeatedly run our EM algorithm, Algorithm 1 from Section 2.2, on Bayesian bootstrap replicates.

The original bootstrap repeatedly samples data from the original data set $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with replacement, i.e., each observation $(\mathbf{x}_i, y_i)$ is sampled with probability $1/n$. In Bayesian bootstrap (Rubin, 1981), instead of sampling a subset of the data, we assign a random weight w_i to the ith observation and then fit a weighted least squares regression model on the whole data set. In particular, following Rubin (1981), we generate the weights $\mathbf{w} = (w_1, \ldots, w_n)$ from an n-category Dirichlet distribution:

$$\mathbf{w}_{n\times 1} \sim \mathsf{Dir}(1, \cdots, 1). \tag{4.1}$$

When applying Algorithm 1 with a weighted linear regression model, all the updating equations stay the same, except the updating equations (2.3) for the posterior of $\boldsymbol{\beta}$, which become

$$\mathbf{m} = \mathbf{V}\mathbf{X}^T \mathrm{diag}(\mathbf{w})\mathbf{y}, \quad \mathbf{V} = (\mathbf{X}^T \mathrm{diag}(\mathbf{w})\mathbf{X} + D^{-1}_{\boldsymbol{\gamma}^{(t)}})^{-1}. \tag{4.2}$$

Equation (2.4), the expectation of the weighted residual sum of squares, needs to be updated accordingly:

$$\mathbb{E}_{\boldsymbol{\beta}|\Theta^{(t)},\mathbf{y}} \left\|\mathbf{y}-\mathbf{X}\boldsymbol{\beta}\right\|_{\mathbf{w}}^2 = \sigma_{(t)}^2 \mathrm{tr}(\mathrm{diag}(\mathbf{w})\mathbf{X}\mathbf{V}\mathbf{X}^T) + (\mathbf{y}-\mathbf{X}\mathbf{m})^T \mathrm{diag}(\mathbf{w})(\mathbf{y}-\mathbf{X}\mathbf{m}). \quad (4.3)$$

It is well-known that in order to make the aggregation work, we should control the correlation among estimates from bootstrap replicates. For example, in Random Forest (Breiman, 2001), the number of variables used for choosing the optimal split of a tree is restricted to a subset of the variables, instead of all p variables. A similar idea was implemented in Random Lasso (Wang et al., 2011), an ensemble algorithm for variable selection with Lasso. In the same spirit, we apply our EM algorithm with only a subset of variables at each Bayesian bootstrap iteration. A naive way is to pick a subset from the p variables randomly. This, however, will be inefficient when p is large and the true model is sparse, since it is likely that most random subsets will contain no relevant variables. So we employ a biased sampling procedure: sample the p variables based on a weight vector $\tilde{\pi}$ that is defined as

$$\tilde{\pi}_{p\times 1} \propto |\mathbf{X}^T\mathbf{y}|/\mathrm{diag}(\mathbf{X}^T\mathbf{X}), \quad (4.4)$$

that is, variables are sampled based on their marginal effect in a simple linear regression.

The ensemble EM algorithm operates as follows. First we sample a random set of L variables according to the probability vector $\tilde{\pi}$, and then draw an $n \times 1$ bootstrap weight vector $\mathbf{w}$ from (4.1). Let $\tilde{\mathbf{X}}$ be the new data matrix with the L columns. Then apply our EM algorithm to the bootstrap replicate $\tilde{\mathbf{X}}$ with weight $\mathbf{w}$. Let $\boldsymbol{\gamma}_k$ denote the model returned by the kth Bayesian bootstrap iteration, where the jth position γ_{kj} is 1 if the jth variable is selected and zero otherwise; of course, the jth position is zero if the jth variable is not included in the initial L variables. Define the final variable selection frequency for the p variables as

$$\boldsymbol{\phi}_{p\times 1} = \frac{1}{K}\sum_{k=1}^{K} \boldsymbol{\gamma}_k. \quad (4.5)$$

We can report the final variable selection result by thresholding ϕ_j at some fixed number, for example, one half. Or we can produce a path-plot of $\boldsymbol{\phi}$ as v_0 varies, which could be a useful tool to investigate the importance of each variable. We illustrate the latter in our simulation study in Section 5.

As for the computational cost, the inversion of the $L \times L$ matrix in (4.2) is a big improvement compared with that of a $p \times p$ matrix. By the same reasoning as in Section 2.3, the computation can be further reduced by inverting an l-by-l matrix, where l is the number of the L variables that change their inclusion status. The complete BBEM algorithm is summarized in Algorithm 2.

5 Empirical Study

In this section, we first compare the proposed EM algorithm (Algorithm 1) with other popular methods on a widely used benchmark data set. Then we compare BBEM (Algorithm 2) with other methods on two more challenging data sets of larger dimensions.

Algorithm 2: BBEM

Input: $\mathbf{X}, \mathbf{y}, v_0, v_1, a_0, b_0, \nu_0, \lambda_0, K, L$
Compute the variable weight vector $\tilde{\pi}$ from (4.4);
for $k = 1 : K$ **do**
  Generate a subset of L variables according to $\tilde{\pi}$;
  Create the replicate $\tilde{\mathbf{X}}^k$ with the L variables;
  Initialize $\Theta_k^{(0)}$;
  Generate the bootstrap weight vector $\mathbf{w}$ from (4.1);
  E-step: Calculate the two expectations in (2.5), denoted as $EE_k^{(0)}$;
  for $t = 1 : maxIter$ **do**
    M-step: Update $\Theta_k^{(t)}$ using (2.7), (2.8), (2.9);
    E-step: Update $EE_k^{(t)}$ using (4.2), (4.3);
    if $\boldsymbol{\gamma}_k^{(t)}$ *stays the same for* $k_0 = 3$ *iterations* **then**
      break;
    end
  end
  Record $\boldsymbol{\gamma}_k^{(t)}$;
end
Return $\boldsymbol{\phi}$ from (4.5);

Finally, we apply BBEM to a restaurant revenue data set from a Kaggle competition, and show that our algorithm outperforms the benchmark given by Random Forest.

For the hyper-parameters v_0 and v_1, we set $v_1 = 100$ as fixed and tune an appropriate value for v_0 based either on 5-fold cross-validation or on BIC. For the initial value of θ, we suggest that $1/2$ be used for ordinary problems, but $\sqrt{n}/p$ for large-p problems. Given $\theta^{(0)}$, the initial value of the binary vector $\boldsymbol{\gamma}^{(0)}$ is randomly generated from Bernoulli distribution with parameter $\theta^{(0)}$. The initial value of σ^2 is set as 1. In addition, there are two bootstrap parameters: the total number of replicates K, and the number of variables used in each bootstrap L. For efficiency, the number of variables in each bootstrap replicate should not exceed the sample size n. We use $K = 100$, and $L = n/2$ if p is large and $L = p$ is p is small.

5.1 Performance on a Widely Used Benchmark

First we apply our EM algorithm on a widely used benchmark data set (Tibshirani, 1996), which has $p = 8$ variables, each from a standard normal distribution with pairwise correlation $\rho(\mathbf{x}_i, \mathbf{x}_j) = 0.5^{|i-j|}$. The response variable is generated from

$$\mathbf{y} = 3\mathbf{x}_1 + 1.5\mathbf{x}_2 + 2\mathbf{x}_5 + \epsilon,$$

where $\epsilon \sim \mathsf{N}(0, \sigma^2)$.

Following Fan and Li (2001), we repeat the experiment 100 times under two scenarios: (1) $n = 40, \sigma = 3$ and (2) $n = 60, \sigma = 1$. The result is summarized in Table 1, which reports the average number of zero-coefficients (i.e., no selection) among signal variables $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_5)$ and among noise variables, respectively. The results for SCAD1 (tuning parameter selected by cross-validation), SCAD2 (tuning parameter fixed) and LASSO are taken from Fan and Li (2001). In the first "small sample-size high noise" scenario, our EM algorithm has the highest number of zero-coefficients among noise variables, i.e., the lowest type I error. The average number of signal variables missed by EM is slightly higher than SCAD1 (where the tuning parameter is chosen by cross-validation) but lower than SCAD2 (where the tuning parameter is pre-fixed). But overall, our EM algorithm and the two SCAD methods perform the best. In the second "large sample-size low noise" scenario, no signal variables are missed by any method, but EM has the lowest type I error.

Method	$\mathbf{x}_j \in$ Noise $(j = 3, 4, 6, 7, 8)$	$\mathbf{x}_j \in$ Signal $(j = 1, 2, 5)$
$n = 40$, $\sigma = 3$		
EM	4.55	0.24
SCAD1	4.20	0.21
SCAD2	4.31	0.27
LASSO	3.53	0.07
Oracle	5.00	0.00
$n = 60$, $\sigma = 1$		
EM	4.72	0.00
SCAD1	4.37	0.00
SCAD2	4.42	0.00
LASSO	3.56	0.00
Oracle	5.00	0.00

Table 1: Performance on a widely used benchmark ($n = 40, 60$). The average number of zero-coefficients (i.e., no selection) out of 100 simulations for each types of variable (Signal or Noise) are shown. The results other than EM (Algorithm 1) are from Fan and Li (2001).

Following Wang et al. (2011) and Xin and Zhu (2012), we repeat the experiment 100 times with the same sample size $n = 50$ but two different noise levels: low noise level ($\sigma = 3$) and high noise level ($\sigma = 6$). Table 2 reports, for the signal and the noise variables, respectively, the minimum, median, maximum of being selected out of 100 simulations. Both Lasso and random Lasso have a higher chance of selecting the signal variables, but at the price of mistakenly including many noise variables. Overall, our EM algorithm performs the best, along with PGA and stability selection, two frequentist ensemble methods for variable selection.

5.2 Performance on a Highly-Correlated Data Set

Next we demonstrate our two algorithms on an example of highly-correlated variables from Wang et al. (2011). The data set has $p = 40$ variables and the response $\mathbf{y}$ is

Method	$\mathbf{x}_j \in$ Signal ($j = 1, 2, 5$)			$\mathbf{x}_j \in$ Noise ($j = 3, 4, 6, 7, 8$)		
	Min	Median	Max	Min	Median	Max
$n = 50$, $\sigma = 3$						
EM	91	97	100	3	6	12
Lasso	99	100	100	48	55	61
Random Lasso	95	99	100	33	40	48
ST2E	89	96	100	4	12	20
PGA	82	98	100	4	7	11
Stability selection						
$\lambda_{min} = 1$	81	83	100	0	2	9
$\lambda_{min} = 0.5$	90	98	100	4	8	22
$n = 50$, $\sigma = 6$						
EM	53	67	91	6	10	14
Lasso	76	85	99	47	49	53
Random Lasso	92	94	100	40	48	58
ST2E	68	69	96	9	13	21
PGA	54	76	94	9	14	16
Stability selection						
$\lambda_{min} = 1$	59	61	92	4	8	18
$\lambda_{min} = 0.5$	76	84	100	30	42	50

Table 2: Performance on a widely used benchmark ($n = 50$). The min, median, max number of being selected out of 100 simulations for each types of variable (Signal or Noise) are shown. The results other than EM (Algorithm 1) are from Xin and Zhu (2012).

generated from

$$\mathbf{y} = 3\mathbf{x}_1 + 3\mathbf{x}_2 - 2\mathbf{x}_3 + 3\mathbf{x}_4 + 3\mathbf{x}_5 - 2\mathbf{x}_6 + \epsilon,$$

where $\epsilon \sim \mathsf{N}(0, \sigma^2)$ and $\sigma = 6$. Each $\mathbf{x}_i$ is generated from a standard normal with the following correlation structure among the first six signal variables: the signal variables are divided into two groups, $V_1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ and $\mathbf{V}_2 = \{\mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$; the within group correlation is 0.9 and the between-group correlation is 0.

We repeat the simulation 100 times with $n = 50$ and $n = 100$, and summarize the results in Table 3. For this example, due to the high correlation among features, we expect ensemble methods to perform better. Indeed, BBEM has the best performance in terms of selecting true signal variables while excluding noise variables. The performance of EM algorithm, although not the best, is also comparable with other top ensemble methods like random Lasso from Wang et al. (2011), and T2E and PGA from Xin and Zhu (2012).

For illustration purpose, we apply BBEM on a data set with $n = 50$ and v_0 varying from 10^{-4} to 1. Figure 1 shows the path-plot of the selection frequency from BBEM. There is clearly a gap between the signal variables and the noise ones. For a range of v_0, from 0.001 to 0.02, BBEM can successfully select the six true variables $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_6\}$ if we threshold the selection frequency ϕ_j at 0.5.

Method	$\mathbf{x}_j \in$ Signal ($j = 1:6$)			$\mathbf{x}_j \in$ Noise		
	Min	Median	Max	Min	Median	Max
$n = 50$, $\sigma = 6$						
Lasso	11	70	77	12	17	25
Random Lasso	84	96	97	11	21	30
ST2E	85	96	100	18	25	34
PGA	55	87	90	14	23	32
EM	65	85.5	89	4	10	13
BBEM	89	96	100	4	8	15
$n = 100$, $\sigma = 6$						
Lasso	8	84	88	12	22	31
Random Lasso	89	99	99	8	14	21
ST2E	93	100	100	14	21	27
PGA	40	85	92	13	22	33
EM	84	91	95	1	7	16
BBEM	95	99	100	4	9	14

Table 3: Performance on a highly-correlated data set. The min, median, max number of times being selected (i.e., no selection) out of 100 simulations for each type of variables (Signal and Noise) are shown. The results other than EM (Algorithm 1) and BBEM (Algorithm 2) are from Xin and Zhu (2012).

5.3 Performance on a Large-p Small-n Example

Finally we apply BBEM to a large-p small-n example from Ročková and George (2014), where $p = 1000$ and $n = 100$. Each of the p features is generated from a standard normal with pairwise correlation to be $0.6^{|i-j|}$ and the response $\mathbf{y}$ is generated from the following linear model:

$$\mathbf{y} = \mathbf{x}_1 + 2\mathbf{x}_2 + 3\mathbf{x}_3 + \epsilon,$$

where $\epsilon \sim \mathsf{N}(0, 3)$.

For this large p example, we set the parameters in the BBEM algorithm as follows: the initial value of θ is $\sqrt{n}/p$, the number of variables used in each bootstrap iteration $L = n/2 = 50$ and the total number of replicates $K = 100$. It is well known that cross-validation based on prediction accuracy tends to include more noise variables (Bühlmann and van de Geer, 2011; Meinshausen, 2007; Wang et al., 2007). Following Wang et al. (2007), we choose to tune v_0 via BIC for this example where the true model is known to be sparse. For illustration purpose, we also include BBEM with a fixed tuning parameter $v_0 = 0.03$ in the comparison group. We compare BBEM with the EMVS algorithm from Ročková and George (2014), which is implemented by us using the annealing technique for $\boldsymbol{\beta}$'s initialization, and fixed $v_0 = 0.5, v_1 = 1000$ as suggested in Ročková and George (2014).

Table 4 reports the average number of signal and noise variables being selected over 100 iterations for each method. BBEM with BIC tuning performs the best: it selects 2.99 signal variables out of 3 on average—only miss $\mathbf{x}_1$, the variable with the weakest signal, once in all 100 iterations, and meanwhile has the smallest type I error. The

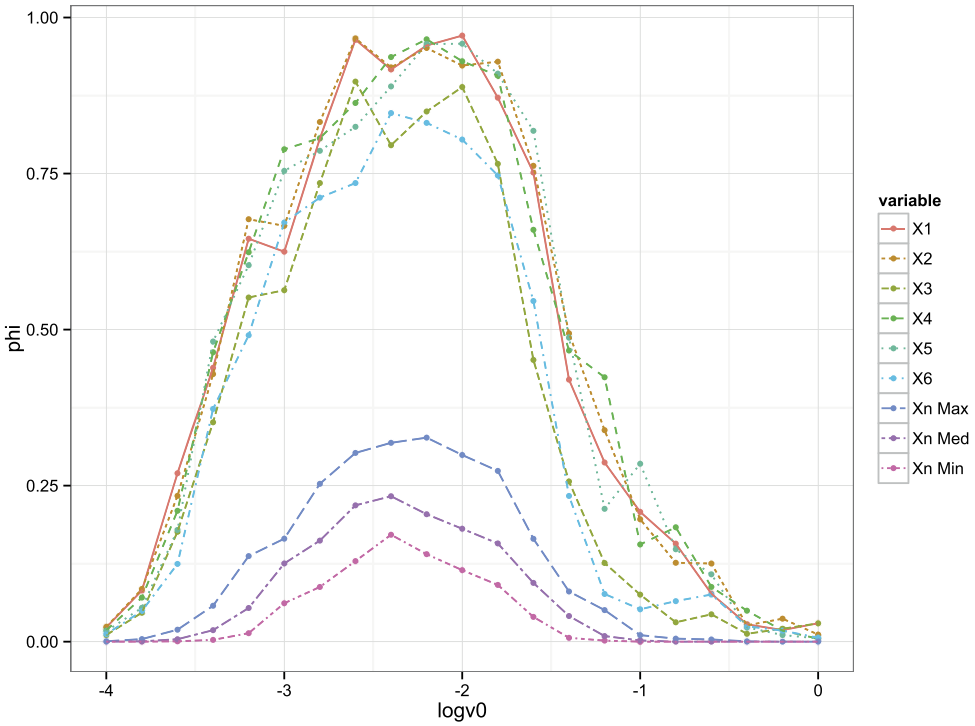

Figure 1: Highly-correlated data ($n = 50$). A path-plot of the average selection frequency when v_0 varies in the logarithm scale of base 10. Top 6 lines represent the true variables $\mathbf{x}_{1:6}$ and the bottom 3 lines represent the maximum, median and minimum among the noise variables $\mathbf{x}_{7:40}$.

BBEM algorithm with a fixed tuning parameter has a similar result as EMVS but is much faster. The computation advantage for BBEM comes from two aspects: the computation trick that reduces the computation cost on matrix inversion, and the sub-sampling step in Bayesian bootstrap that allows us to work with a subset of variables of size smaller than p.

5.4 A Real Example

For TFI, a company that owns some of the world's most well-known brands like Burger King and Arby's, decisions on where to open new restaurants are crucial. It usually takes a big investment of both time and capital at the beginning to set up a new restaurant. If a wrong location is chosen, likely the restaurant will soon be closed and all the initial investment will be lost. TFI hosted a prediction competition on Kaggle,[1] where the goal is to build a mathematical model to predict the revenue of a restaurant based on a set

[1] https://www.kaggle.com/c/restaurant-revenue-prediction

	$x_j \in$ Signal	$x_j \in$ Noise
BBEM (BIC)	2.99	0.24
BBEM ($v_0 = 0.03$)	2.96	0.27
EMVS	2.97	0.29
Oracle	3	0

Table 4: Performance on a large-p small-n example. The table shows the average number of signal and noise variables being selected out of 100 iterations. In BBEM (Algorithm 2), v_0 is either chosen by BIC or fixed at 0.03. EMVS is the algorithm proposed by Ročková and George (2014).

of demographic, real estate, and commercial information. There are 137 restaurants in the training data set and 1000 in the test data set. Features include the Open Date, City, City Group, Restaurant Type, and three categories of obfuscated data (P1–P37, numeric): demographic data, real estate data, and commercial data. The response is the transformed restaurant revenue in a given year.

We first transform the "Open Date" to a numeric feature called "Year Since 1900" and merge the "City" column into the "City Group" column, which now contains four categories: Istanbul, Izmir, Ankara, and others (small cities). Then we create dummy variables for the categorical features like "City Group" and "Restaurant Type", and keep all the obfuscated numeric columns P1–P37. The final training set has 43 features and 137 samples.

After standardizing the data, we fix v_1 at 100 and tune v_0 from $10^{-4.5}$ to $10^{-0.5}$ for the BBEM algorithm, where each bootstrap sample uses $L = 15$ variables, and the total number of replicates is $K = 300$. The path-plot of selection frequency for important features is shown in Figure 2. It is not surprising that "City Group", "Years Since 1900" and "Restaurant Type" are important predictors for the revenue. Quite a few obfuscated features are also selected as important predictors. Although we do not know their meanings, they should provide valuable information for TFI to choose their next restaurant's location.

Since the evaluation metric for this specific competition is based on the rooted mean square error (RMSE), we use the same metric in our 5-fold cross-validation. We tuned v_0 from the set $\{0.0001, 0.0002, 0.0005, 0.001, 0.002, 0.005, 0.01\}$, and found $v_0 = 0.002$ has the smallest RMSE score. Then we fix v_0 at 0.002, and re-run BBEM on the whole training data. Let $\mathbf{m}$ denote the averaged posterior mean of $\boldsymbol{\beta}$ from L bootstrap iterations, and $\boldsymbol{\gamma}$ the averaged selection frequency for p variables. We then use $\mathbf{m} * \boldsymbol{\gamma}$ (where $*$ denotes element-wise product) for prediction in the same spirit as the Bayesian model averaging. Our final Kaggle score is 1989762.52, which outperforms the random forest benchmark (RMSE = 1998014.94) provided by Kaggle.[2] It is remarkable for BBEM to outperform random forest considering that BBEM does not use any nonlinear features but random forest does.

[2]At Kaggle, each team can submit their prediction and see the corresponding performance on the test data many times, so one can easily obtain a good score by keep tweaking the model to overfit the test data. For this reason, we did not compare our result with those "low" scores on the leaderboard provided by individual teams.

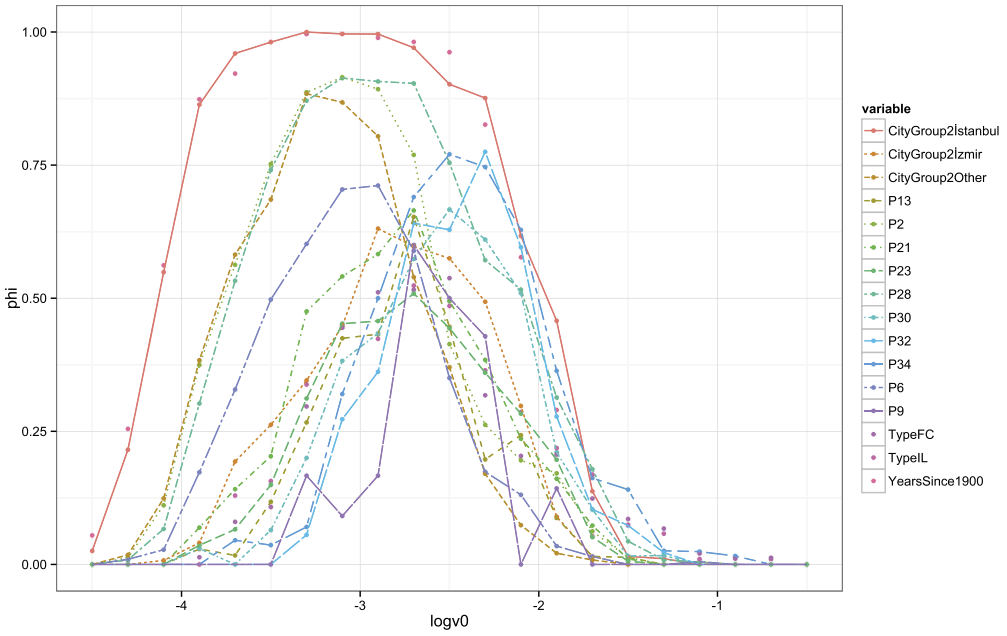

Figure 2: Restaurant data. The path plot of selection frequency when v_0 varies in the logarithm scale of base 10. Only a subset of variables with high selection frequencies are displayed.

6 Further Discussion

Variable selection is an important problem in modern statistics. In this paper, we study the Bayesian approach to variable selection in the context of multiple linear regression. We proposed an EM algorithm that returns the MAP estimator of the set of relevant variables. The algorithm can be operated very efficiently and therefore can scale up with big data. In addition, we have shown that the MAP estimator from our EM algorithm provides a consistent estimator of the true variable set even when the model dimension diverges with the sample size.

Further, we propose an ensemble version of our EM algorithm based on Bayesian bootstrap, which, as demonstrated via real and simulated examples, substantially increases accuracy while maintaining computation efficiency. A further investigation is needed for our ensemble EM algorithm to address questions like the optimal choice of bootstrap parameters and variable selection consistency after resampling.

Although we restrict our discussion for the linear model, the two algorithm we proposed can be easily extended to other generalized linear models by using latent variables (Polson and Scott, 2013), another interesting topic for future research.

7 Proofs

7.1 Proof of Theorem 3.1

Proof. Recall the EM algorithm returns

$$\hat{\gamma}_{nj} = 1, \quad \text{if} \quad \mathbb{E}_{\boldsymbol{\beta}_n|\Theta^{(t)},\mathbf{y}}\left[\beta_{nj}^2\right] > r_n,$$

where the threshold

$$r_n = \frac{\hat{\sigma}_n^2}{1/v_0 - 1/v_1}\Big(\log\frac{v_1}{v_0} - 2\log\frac{\hat{\theta}_n}{1-\hat{\theta}_n}\Big) = O(n^{-r_0}\log n)$$

and the conditional second moment of β_{nj} is equal to $m_j^2 + \hat{\sigma}_n^2 V_{jj}$ with

$$\begin{aligned}
\mathbf{m} &= (\mathbf{X}_n^T\mathbf{X}_n + D^{-1})^{-1}\mathbf{X}_n^T(\mathbf{X}_n\boldsymbol{\beta}_n^* + \mathbf{e}_n) \\
&= \boldsymbol{\beta}_n^* - (\mathbf{X}_n^T\mathbf{X}_n + D^{-1})^{-1}D^{-1}\boldsymbol{\beta}_n^* + (\mathbf{X}_n^T\mathbf{X}_n + D^{-1})^{-1}\mathbf{X}_n^T\mathbf{e}_n \\
&= \boldsymbol{\beta}_n^* - \mathbf{b}_n + \mathbf{W}_n, \\
\mathbf{V} &= (\mathbf{X}_n^T\mathbf{X}_n + D^{-1})^{-1}, \quad D^{-1} = \operatorname{diag}\left(\frac{1-\hat{\gamma}_{nj}}{v_0} + \frac{\hat{\gamma}_{nj}}{v_1}\right).
\end{aligned}$$

Here we represent the posterior mean of $\boldsymbol{\beta}_n$ as three separate terms: the true coefficient vector $\boldsymbol{\beta}_n^*$, the bias term $\mathbf{b}_n$ and the random error term $\mathbf{W}_n$. The event $\{\hat{\boldsymbol{\gamma}}_n = \boldsymbol{\gamma}_n^*\}$ is equivalent to

$$\left\{\min_{j:\gamma_{nj}^*=1} m_j^2 + \hat{\sigma}_n^2 V_{jj} > r_n\right\} \cap \left\{\max_{j:\gamma_{nj}^*=0} m_j^2 + \hat{\sigma}_n^2 V_{jj} < r_n\right\}. \tag{7.1}$$

Throughout the proof, for two sequences $\{a_n\}$ and $\{b_n\}$, we write $a_n \prec b_n$ if $a_n/b_n \to 0$. For a matrix A, denote the matrix L_∞ norm by $\|A\|_\infty$ which is equal to the maximum absolute row sum of A, and denote the matrix L_2 norm by $\|A\|_2$ which is equal to its largest eigenvalue (singular value) when A is symmetric (non-symmetric). For a vector v, denote its L_2 norm by $\|v\|_2$, and max norm by $\|v\|_\infty = \max_j |v_j|$.

First we prove the following results that quantify m_j^2 and V_{jj}.

(R1) V_{jj} is upper bounded by the largest eigenvalue of $\mathbf{V}$,

$$V_{jj} \le \frac{1}{\lambda_{n1} + 1/v_1} = O(n^{-\eta_1}) \prec O(n^{-r_0}\log n) = r_n. \tag{7.2}$$

(R2) The bias term $\mathbf{b}_n$ is bounded by

$$\begin{aligned}
\|\mathbf{b}_n\|_\infty \le \|\mathbf{b}_n\|_2 &\le \|(\mathbf{X}_n^T\mathbf{X}_n + D^{-1})^{-1}\|_2 \cdot \|D^{-1}\boldsymbol{\beta}_n^*\|_2 \\
&\le \frac{1/v_0}{\lambda_{n1} + 1/v_1}\|\boldsymbol{\beta}_n^*\|_2 = O(n^{r_0-\eta_1+\eta_2}).
\end{aligned} \tag{7.3}$$

When $r_0 < 2(\eta_1 - \eta_2)/3$, $\max_j |b_{nj}|^2 \prec O(n^{-r_0}\log n) = r_n$.

(R3) Note that $\mathbf{W}_n$ is not a Gaussian random vector due to the dependence between D and $\mathbf{e}_n$, but it can be rewritten as

$$\mathbf{W}_n \quad = \quad (\mathbf{X}_n^T\mathbf{X}_n + D^{-1})^{-1}(\mathbf{X}_n^T\mathbf{X}_n)(\mathbf{X}_n^T\mathbf{X}_n)^{-1}\mathbf{X}_n^T\mathbf{e}_n = A\tilde{\mathbf{W}}_n,$$

where $A = \left(\mathbf{X}_n^T\mathbf{X}_n + D^{-1}\right)^{-1}\left(\mathbf{X}_n^T\mathbf{X}_n\right)$ and $\tilde{\mathbf{W}}_n = (\mathbf{X}_n^T\mathbf{X}_n)^{-1}\mathbf{X}_n^T\mathbf{e}_n$. Since A is a matrix with norm bounded by 1, we have

$$\|\mathbf{W}_n\|_\infty \le \|A\|_\infty\|\tilde{\mathbf{W}}_n\|_\infty \le \sqrt{p}\|A\|_2\|\tilde{\mathbf{W}}_n\|_\infty \le \sqrt{p}\|\tilde{\mathbf{W}}_n\|_\infty. \tag{7.4}$$

(R4) $\tilde{\mathbf{W}}_n = (\mathbf{X}_n^T\mathbf{X}_n)^{-1}\mathbf{X}_n^T\mathbf{e}_n$ is a Gaussian random vector with covariance $\sigma^2(\mathbf{X}_n^T\mathbf{X}_n)^{-1}$ and mean $\mathbf{0}$. So the variance for W_{nj} is upper bounded by $\sigma^2\lambda_{n1}^{-1}$. Recall the tail bound for Gaussian variables: for any $Z \sim \mathsf{N}(0, \tau^2)$,

$$\mathbb{P}(|Z| > t) = \mathbb{P}(|Z|/\tau > t/\tau) \le \frac{\tau}{t}e^{-\frac{t^2}{2\tau^2}}.$$

With Result (R3) and the Bonferroni's inequality, we can find a constant $M > 0$ such that

$$\begin{aligned}
\mathbb{P}(\max_j |W_{nj}| > \sqrt{r_n}) &\le \mathbb{P}(\max_j |\tilde{W}_{nj}| > \sqrt{r_n/p}) \\
&\le p \cdot \mathbb{P}(|\tilde{W}_{nj}| > \sqrt{r_n/p}) \\
&\le \frac{p\sqrt{p}\sigma}{\sqrt{r_n\lambda_{n1}}}e^{-\frac{r_n\lambda_{n1}}{2p\sigma^2}} = O\big(e^{-Mn^{\eta_1-r_0-\alpha}}\big),
\end{aligned}$$

which goes to 0 when $r_0 < \eta_1 - \alpha$. So with probability going to 1, $\max_j |W_{nj}|$ is upper bounded by $\sqrt{r_n}$.

(R5) When $1 - \eta_3 < r_0$, $\min_{j:\gamma_j^*=1} |\beta_{nj}^*|^2 \sim n^{\eta_3-1} \succ O(n^{-r_0}\log n) = r_n$.

Now we prove (7.1). Given $1 - \eta_3 < r_0 < \min\{\eta_1 - \alpha, 2(\eta_1 - \eta_2)/3\}$, we have

$$\begin{aligned}
\mathbb{P}\left(\max_{j:\gamma_{nj}^*=0}(m_j^2 + \hat{\sigma}_n^2 V_{jj}) > r_n\right) &\le \mathbb{P}\left(\big(\max_j |b_{nj}| + \max_j |W_{nj}|\big)^2 + \hat{\sigma}_n^2 \max_j V_{jj} > r_n\right) \\
&\le \mathbb{P}\left(\max_j |W_{nj}| > \sqrt{r_n}\right) = O\big(e^{-Mn^{\eta_1-r_0-\alpha}}\big), \\
\mathbb{P}\left(\min_{j:\gamma_{nj}^*=1}(m_j^2 + \hat{\sigma}_n^2 V_{jj}) < r_n\right) &\le \mathbb{P}\left(\min_{j:\gamma_{nj}^*=1} |\beta_{nj}^*|^2 - \big(\max_j |b_{nj}| + \max_j |W_{nj}|\big)^2 < r_n\right) \\
&\le \mathbb{P}\left(\max_j |W_{nj}| > \sqrt{r_n}\right) = O\big(e^{-Mn^{\eta_1-r_0-\alpha}}\big).
\end{aligned}$$

So (7.1) holds with probability $1 - O(e^{-Mn^{\eta_1-r_0-\alpha}}) \to 1$. □

7.2 Selection Consistency when $p \gg n$

Our proof for Theorem 3.1 can be easily extended to cover the case when p grows exponentially with n. However, when $p > n$, the coefficient vector $\boldsymbol{\beta}_n^*$ is typically not identifiable. For more discussions on identifiability under deterministic designs, see Shao and Deng (2012).

In order to discuss selection consistency, we first need to impose some identifiability condition. Let $\mathbf{Q}$ be a $p \times r$ matrix with columns being a set of orthonormal basis of $\mathcal{R}$, the subspace spanned by rows of $\mathbf{X}_n$, where $r \leq n$ denotes the rank of $\mathbf{X}_n$. Let $\mathbf{Q}_\perp$ be a $p \times (p-r)$ matrix with columns being a set of orthonormal basis of $\mathcal{R}^\perp$, the orthogonal complement of $\mathcal{R}$. There are infinitely many p-dimensional vectors $\boldsymbol{\beta}$ satisfying $\mathbf{X}_n\boldsymbol{\beta} = \mathbf{X}_n\boldsymbol{\beta}_n^*$; we can only identify their projections onto $\mathcal{R}$, which are uniquely determined, but not their projections onto $\mathcal{R}^\perp$. We assume that the projection of the sparse true coefficient vector $\boldsymbol{\beta}_n^*$ onto $\mathcal{R}^\perp$ is of a small magnitude:

$$\|\boldsymbol{\beta}_n^* - \mathbf{Q}\mathbf{Q}^T\boldsymbol{\beta}_n^*\|_\infty = \|\mathbf{Q}_\perp\mathbf{Q}_\perp^T\boldsymbol{\beta}_n^*\|_\infty = o(n^{-r_0}\log n). \tag{7.5}$$

Note that this condition (7.5) is satisfied automatically when $\boldsymbol{\beta}_n$ is in the row span of the design matrix $\mathbf{X}_n$ or when $\mathbf{X}_n$ is of full rank as in Theorem 3.1, while the Irrepresentable Condition could be violated even when $\mathbf{X}_n$ is of full rank. Similar conditions are used in Shao and Deng (2012) and Zhang et al. (2008).

The other conditions, (A1)–(A5), in Section 3 are almost the same, except that (i) λ_{n1} in (A1) now denotes the smallest non-zero eigenvalue of $\mathbf{X}_n^T\mathbf{X}_n$ since the smallest eigenvalue of $\mathbf{X}_n^T\mathbf{X}_n$ is zero, and (ii) in (A5), α is no longer needed or equivalently $\alpha = 0$, and $\log p = O(n^{\eta_1 - r_0})$.

To achieve selection consistency when $p \gg n$, we need another condition that is not needed for Theorem 3.1: the EM algorithm must start with all variables being excluded, i.e., $\boldsymbol{\gamma}^{(0)}$ is an all zero vector. Then we have $D^{-1} = \frac{1}{v_0}\mathbf{I}_p$ being a constant diagonal matrix.

Now, we are ready to prove selection consistency when $p \gg n$. The proof is similar to the one in Section 7.1 with the following changes:

(R1) the new bound for V_{jj} is given by

$$V_{jj} \leq \frac{1}{0 + 1/v_0} = O(n^{-r_0}) \prec O(n^{-r_0}\log n) = r_n. \tag{7.6}$$

(R2) We now bound $\max_j |b_{nj}|$ as follows.

$$\begin{aligned}
\|\mathbf{b}_n\|_\infty &= \frac{1}{v_0}\|(\mathbf{X}_n^T\mathbf{X}_n + \frac{1}{v_0}\mathbf{I})^{-1}\boldsymbol{\beta}_n^*\|_\infty \\
&\leq \frac{1}{v_0}\|(\mathbf{X}_n^T\mathbf{X}_n + \frac{1}{v_0}\mathbf{I})^{-1}\mathbf{Q}\mathbf{Q}^T\boldsymbol{\beta}_n^*\|_\infty + \frac{1}{v_0}\|(\mathbf{X}_n^T\mathbf{X}_n + \frac{1}{v_0}\mathbf{I})^{-1}\mathbf{Q}_\perp\mathbf{Q}_\perp^T\boldsymbol{\beta}_n^*\|_\infty \\
&\leq \frac{1}{v_0}\|(\mathbf{X}_n^T\mathbf{X}_n + \frac{1}{v_0}\mathbf{I})^{-1}\mathbf{Q}\mathbf{Q}^T\boldsymbol{\beta}_n^*\|_2 + \|\mathbf{Q}_\perp\mathbf{Q}_\perp^T\boldsymbol{\beta}_n^*\|_\infty
\end{aligned}$$

$$\leq \quad \frac{1/v_0}{\lambda_{n1}+1/v_0}\|\boldsymbol{\beta}_n^*\|_2 + o(n^{-r_0}\log n) \prec O(n^{-r_0}\log n) = r_n.$$

(R3) $\mathbf{W}_n$ now is a Gaussian random vector since D is a constant matrix. So we no longer need $\tilde{\mathbf{W}}_n$, and consequently can ignore the $\sqrt{p}$ factor from (7.4) in our later proof.

(R4) $\mathbf{W}_n = (\mathbf{X}_n^T\mathbf{X}_n + \frac{1}{v_0}\mathbf{I})^{-1}\mathbf{X}_n^T\mathbf{e}_n$ is a Gaussian random vector with maximum variance upper bounded by $\sigma^2\lambda_{n1}^{-1}$. Applying the union bound, we have

$$\begin{aligned}\mathbb{P}(\max_j |W_{nj}| > \sqrt{r_n}) \quad &\leq \quad p\cdot\mathbb{P}(|W_{nj}| > \sqrt{r_n}) \\ &\leq \quad p\sqrt{\frac{\sigma^2}{r_n\lambda_{n1}}}e^{-\frac{r_n\lambda_{n1}}{2\sigma^2}} = O\big(e^{-Mn^{\eta_1-r_0}+\log p}\big)\end{aligned}$$

which goes to 0 when $\log p = o(n^{\eta_1-r_0})$.

Combining all the results, we can conclude that our EM algorithm can achieve selection consistency even when $\log p = O(n^{\eta_1-r_0})$.

References

Breiman, L. (1996). "Bagging predictors." *Machine Learning*, 24(2): 123–140. 886

Breiman, L. (2001). "Random forests." *Machine Learning*, 45(1): 5–32. MR3874153. 887

Bühlmann, P. and van de Geer, S. (2011). *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media. MR2807761. doi: https://doi.org/10.1007/978-3-642-20192-9. 880, 885, 891

Castillo, I., Schmidt-Hieber, J., Van der Vaart, A., et al. (2015). "Bayesian linear regression with sparse priors." *The Annals of Statistics*, 43(5): 1986–2018. MR3375874. doi: https://doi.org/10.1214/15-AOS1334. 886

Clyde, M. A. and Lee, H. K. H. (2001). "Bagging and Bayesian bootstrap." In Richardson, T. and Jaakkola, T. (eds.), *Artificial Intelligence and Statistics*, 169–174. 880

Fan, J. and Li, R. (2001). "Variable selection via nonconcave penalized likelihood and its oracle properties." *Journal of the American Statistical Association*, 96(456): 1348–1360. MR1946581. doi: https://doi.org/10.1198/016214501753382273. 879, 889

Fan, J. and Lv, J. (2010). "A selective overview of variable selection in high dimensional feature space." *Statistica Sinica*, 20(1): 101. MR2640659. 880

George, E. I. and McCulloch, R. E. (1993). "Variable selection via Gibbs sampling." *Journal of the American Statistical Association*, 88(423): 881–889. 880

Loh, P.-L., Wainwright, M. J., et al. (2017). "Support recovery without incoherence: A case for nonconvex regularization." *The Annals of Statistics*, 45(6): 2455–2482. MR3737898. doi: https://doi.org/10.1214/16-AOS1530. 885

Meinshausen, N. (2007). "Relaxed Lasso." *Computational Statistics & Data Analysis*, 52(1): 374–393. MR2409990. doi: https://doi.org/10.1016/j.csda.2006.12.019. 891

Meinshausen, N. and Bühlmann, P. (2010). "Stability selection." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4): 417–473. MR2758523. doi: https://doi.org/10.1111/j.1467-9868.2010.00740.x. 881

Meng, X.-L. and Rubin, D. B. (1993). "Maximum likelihood estimation via the ECM algorithm: A general framework." *Biometrika*, 80(2): 267–278. MR1243503. doi: https://doi.org/10.1093/biomet/80.2.267. 882

Mitchell, T. J. and Beauchamp, J. J. (1988). "Bayesian variable selection in linear regression." *Journal of American Statistical Association*, 83(404): 1023–1032. MR0997578. 880

Narisetty, N. N., He, X., et al. (2014). "Bayesian variable selection with shrinking and diffusing priors." *The Annals of Statistics*, 42(2): 789–817. MR3210987. doi: https://doi.org/10.1214/14-AOS1207. 886

O'Hara, R. B. and Sillanpää, M. J. (2009). "A review of Bayesian variable selection methods: What, how and which." *Bayesian Analysis*, 4(1): 85–118. MR2486240. doi: https://doi.org/10.1214/09-BA403. 880

Polson, N. G. and Scott, J. G. (2013). "Data augmentation for non-Gaussian regression models using variance-mean mixtures." *Biometrika*, 100: 459–471. MR3068446. doi: https://doi.org/10.1093/biomet/ass081. 894

Ročková, V. and George, E. I. (2014). "EMVS: the EM approach to Bayesian variable selection." *Journal of the American Statistical Association*, 109(506): 828–847. MR3223753. doi: https://doi.org/10.1080/01621459.2013.869223. 879, 880, 881, 891, 893

Rubin, D. B. (1981). "The Bayesian bootstrap." *The Annals of Statistics*, 9(1): 130–134. MR0600538. 886

Shao, J. and Deng, X. (2012). "Estimation in high-dimensional linear models with deterministic design matrices." *The Annals of Statistics*, 40(2): 812–831. MR2933667. doi: https://doi.org/10.1214/12-AOS982. 885, 897

Tibshirani, R. (1996). "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 58(1): 267–288. MR1379242. 879, 888

Wainwright, M. J. (2009). "Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (lasso)." *IEEE transactions on information theory*, 55(5): 2183–2202. MR2729873. doi: https://doi.org/10.1109/TIT.2009.2016018. 885

Wang, H., Li, R., and Tsai, C.-L. (2007). "Tuning parameter selectors for the smoothly clipped absolute deviation method." *Biometrika*, 94(3): 553–568. MR2410008. doi: https://doi.org/10.1093/biomet/asm053. 891

Wang, S., Nan, B., Rosset, S., and Zhu, J. (2011). "Random Lasso." *The Annals of Applied Statistics*, 5(1): 468–485. MR2810406. doi: https://doi.org/10.1214/10-AOAS377. 881, 887, 889, 890

Xin, L. and Zhu, M. (2012). "Stochastic Stepwise Ensembles for Variable Selection." *Journal of Computational and Graphical Statistics*, 21(2): 275–294. MR2945467. doi: https://doi.org/10.1080/10618600.2012.679223. 889, 890, 891

Zhang, C.-H. et al. (2010). "Nearly unbiased variable selection under minimax concave penalty." *The Annals of Statistics*, 38(2): 894–942. MR2604701. doi: https://doi.org/10.1214/09-AOS729. 879

Zhang, J., Jeng, X. J., and Liu, H. (2008). "Some two-step procedures for variable selection in high-dimensional linear regression." *arXiv preprint* arXiv:0810.1644. 897

Zhao, P. and Yu, B. (2006). "On model selection consistency of lasso." *Journal of Machine Learning Research*, 7(Nov): 2541–2563. MR2274449. 885

Zhu, M. and Chipman, H. A. (2006). "Darwinian evolution in parallel universes: A parallel genetic algorithm for variable selection." *Technometrics*, 48(4): 491–502. MR2328618. doi: https://doi.org/10.1198/004017006000000093. 881