Bayesian Masking: Sparse Bayesian Estimation with Weaker Shrinkage Bias

Yohei Kondo YKONDO@SYS.I.KYOTO-U.AC.JP
Shin-ichi Maeda ICHI@SYS.I.KYOTO-U.AC.JP
Graduate School of Informatics, Kyoto University, Kyoto, Japan

Kohei Hayashi HAYASHI.KOHEI@GMAIL.COM
National Institute of Informatics, Tokyo, Japan
Kawarabayashi Large Graph Project, ERATO, JST

Editor: Geoffrey Holmes and Tie-Yan Liu

Abstract

A common strategy for sparse linear regression is to introduce regularization, which eliminates irrelevant features by letting the corresponding weights be zeros. However, regularization often shrinks the estimator for relevant features, which leads to incorrect feature selection.

Motivated by the above-mentioned issue, we propose Bayesian masking (BM), a sparse estimation method which imposes no regularization on the weights. The key concept of BM is to introduce binary latent variables that randomly mask features. Estimating the masking rates determines the relevance of the features automatically. We derive a variational Bayesian inference algorithm that maximizes the lower bound of the factorized information criterion (FIC), which is a recently developed asymptotic criterion for evaluating the marginal log-likelihood. In addition, we propose reparametrization to accelerate the convergence of the derived algorithm. Finally, we show that BM outperforms Lasso and automatic relevance determination (ARD) in terms of the sparsity-shrinkage trade-off.

Keywords: Sparse estimation, Factorized information criterion, Lasso, Automatic relevance determination

1. Introduction

In sparse linear regression, various approaches impose sparsity by implementing regularization on a weight parameter. For example, Lasso (Tibshirani, 1994) introduces sparsity by regularizing the weights by L1 norm. Automatic relevance determination (ARD) (MacKay, 1994; Neal, 1996) regularizes the weights by a prior distribution, with hyperparameters indicating the relevance of the input features. Empirical Bayes estimation of the hyperparameters thus eliminates irrelevant features automatically. Although ARD is notorious for its slow convergence, several authors have improved the algorithm (e.g., (Wipf and Nagarajan, 2008)).

The trade-off between sparsity and shrinkage is a crucial issue in sparse regularization methods (Aravkin et al., 2014). In Lasso, for example, a large regularization constant incorporates strong sparsity and is more likely to estimate the weights of irrelevant features as zero, which is desirable in terms of interpretability. However, it also shrinks the weights of relevant features and may eliminate them. ARD suffers from the same problem, although the bias of ARD is weaker than that of Lasso (Aravkin et al., 2014). Because both sparsity and shrinkage are caused by regularization, the shrinkage effects are inevitable as long as we use the regularization scheme.

To address this issue, we propose an alternative method for sparse estimation, namely, Bayesian masking (BM), which differs from existing methods in that it does not impose any regularization on the weights. Our contributions can be summarized as follows.

- The BM model (Section 4). The BM model introduces binary latent variables into a linear regression model. The latent variables randomly mask features to be zero at each sample according to the priors that are defined for each feature, but shared among samples. The estimation of the priors on the masking rates determines the relevance of the features.
- A variational Bayesian inference algorithm for the BM model (Section 4.2). The EM-like coordinate ascent algorithm maximizes the lower bound of the factorized information criterion (FIC). The convergence of the algorithm is accelerated by combining gradient ascent and reparametrization (Section 4.4), which are motivated by previous studies on convergence analysis of coordinate ascent and information geometry.
- Analytic forms of the one-dimensional (1D) estimators of Lasso, ARD (Section 3), and BM (Section 4.3). The analytic estimators of these methods provide insights into their shrinkage mechanisms.

Through numerical experiments, we empirically show that the proposed method outperforms Lasso and ARD in terms of the sparsity-shrinkage trade-off.

1.1. Notation

Hereafter, $\boldsymbol{x}_n$ denotes a column vector of the n-th row of a matrix X.

2. Background

2.1. Linear Regression and Least Squares

Consider a linear regression model:

$$\boldsymbol{y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{1}$$

where $\boldsymbol{y} \in \mathbb{R}^N$ is a vector of target values, $X \in \mathbb{R}^{N\times K}$ is a matrix of explanatory variables, $\boldsymbol{\beta} \in \mathbb{R}^K$ is a vector of weight parameters, and $\boldsymbol{\epsilon} \sim N(0, \lambda^{-1}I)$ denotes observation noise. Further, N is the number of samples and K is the number of features. Because the noise is i.i.d. Gaussian, the maximum likelihood estimator (MLE) is given as the least-squares (LS) solution:

$$\hat{\boldsymbol{\beta}}_{\mathrm{LS}} = \arg\min_{\boldsymbol{\beta}} \frac{\lambda}{2}\|\boldsymbol{y} - X\boldsymbol{\beta}\|_2^2 = (X^\top X)^{-1}X^\top \boldsymbol{y}. \tag{2}$$

2.2. Lasso

Lasso is formulated as the L1-penalized regression problem, and the estimator is given as

$$\hat{\boldsymbol{\beta}}_{\text{Lasso}} = \arg\min_{\boldsymbol{\beta}} \frac{\lambda}{2}\|\boldsymbol{y} - X\boldsymbol{\beta}\|_2^2 + \alpha\|\boldsymbol{\beta}\|_1, \tag{3}$$

where $\alpha(> 0)$ is a regularization constant.

2.3. ARD

Consider the prior distribution of $\boldsymbol{\beta}$:

$$p(\boldsymbol{\beta}|\boldsymbol{\gamma}) = N(\boldsymbol{\beta}|0, \Gamma), \tag{4}$$

where $\Gamma = \text{diag}(\boldsymbol{\gamma})$ and $\boldsymbol{\gamma}$ is the hyperparameter determining the variance of the prior. ARD determines $\boldsymbol{\gamma}$ by the empirical Bayes principle, i.e., by maximizing the marginal log-likelihood: $\hat{\boldsymbol{\gamma}} = \arg\max_{\gamma} \int p(\boldsymbol{y}|\boldsymbol{\beta})p(\boldsymbol{\beta}|\boldsymbol{\gamma}, \lambda)\mathrm{d}\boldsymbol{\beta}$. Then, the estimator of $\boldsymbol{\beta}$ is then often given by the posterior mean with plugged-in $\hat{\boldsymbol{\gamma}}$ (Wipf and Nagarajan, 2008; Aravkin et al., 2014):

$$\hat{\boldsymbol{\beta}}_{\text{ARD}} = \hat{\Gamma}X^{\top}(\lambda^{-1}I + X\hat{\Gamma}X^{\top})^{-1}\boldsymbol{y}. \tag{5}$$

Clearly, for $\lambda^{-1} > 0$, $\hat{\gamma}_k = 0$ results in $\beta_k = 0$ for any k.

3. Trade-off between Sparsity and Shrinkage

Concerning the trade-off between sparsity and shrinkage, Aravkin et al. (2014) derived the upper bounds of the estimators for $K = 2$ and showed that undesirable shrinkage occurs for both Lasso and ARD; specifically, the shrinkage bias of Lasso is larger than that of ARD when an unnecessary feature is correctly pruned.

In this section, we revisit the above-mentioned issue. To understand how sparse regularization works, we derive the exact forms of the estimators for $K = 1$ and show that ARD is better than Lasso in terms of the sparsity-shrinkage trade-off. Although our analysis is much simpler than the earlier study by Aravkin et al. (2014), it is meaningful because our derived estimators

- are exact and analytically written (no approximation is needed) and
- highlight the significant differences between ARD and Lasso.

3.1. 1D Estimators

LS When $K = 1$, the matrix inverse in Eq. (2) becomes a scalar inverse and $\hat{\beta}_{\text{LS}}$ is simply written as

$$\hat{\beta}_{\text{LS}} = \frac{\boldsymbol{x}^{\top}\boldsymbol{y}}{\boldsymbol{x}^{\top}\boldsymbol{x}}, \tag{6}$$

where we let $\boldsymbol{x}$ represent X to emphasize the dimensionality.

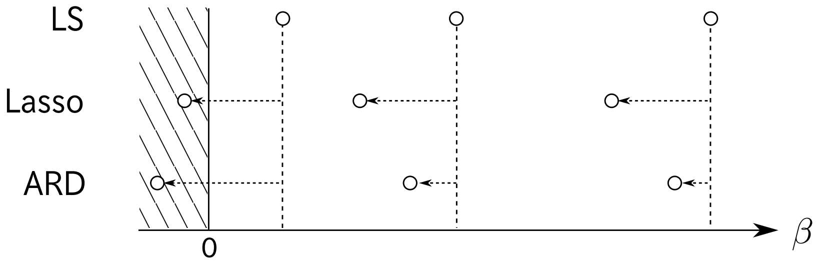

Figure 1: Illustrative comparison of the shrinkage effect with Lasso and ARD in the 1D case. Lasso shifts the estimator to zero by the same value, whereas the shrinkage of ARD is larger when the LS estimator is smaller.

Lasso For $\beta \geq 0$, the L1-penalty becomes $\alpha\beta$. Thus, Eq. (3) yields a stationary point $\hat{\beta}_{\mathrm{LS}} - \alpha/\boldsymbol{x}^\top\boldsymbol{x}$ where $\hat{\beta}_{\mathrm{LS}} = \boldsymbol{x}^\top\boldsymbol{y}/\boldsymbol{x}^\top\boldsymbol{x}$. For $\beta < 0$, the solution is the same except that the sign is reversed. Combining both cases yields the solution for all β:

$$\hat{\beta}_{\mathrm{Lasso}} = \mathrm{sign}(\hat{\beta}_{\mathrm{LS}}) \max(0, |\hat{\beta}_{\mathrm{LS}}| - \frac{2\alpha}{\lambda\boldsymbol{x}^\top\boldsymbol{x}}). \tag{7}$$

ARD Similar to Lasso, the 1D estimator of ARD is analytically written as[1]

$$\hat{\beta}_{\mathrm{ARD}} = \mathrm{sign}(\hat{\beta}_{\mathrm{LS}}) \max\big(0, |\hat{\beta}_{\mathrm{LS}}| - \frac{1}{\lambda|\boldsymbol{x}^\top\boldsymbol{y}|}\big). \tag{8}$$

Note that we assumed that λ is known.[2]

3.2. Comparison of LS, Lasso and ARD

Although their regularizations are different, Lasso and ARD have the same shrinkage mechanism — subtracting the constant from $\hat{\beta}_{\mathrm{LS}}$ and cropping $\hat{\beta}_{\mathrm{LS}}$ to zero if its magnitude is smaller than that of the constant. Since the constants in Eqs. (7) and (8) are both larger than zero except for the noiseless case (i.e., $\lambda = \infty$), the shrinkage bias is inevitable in both Lasso with $\alpha > 0$ and ARD. On the other hand, this retraction to zero is necessary for sparsity because it prunes the irrelevant features.

It is worth noting that the bias of ARD is much weaker than that of Lasso when the scale of $\hat{\beta}_{\mathrm{ARD}}$ is large. This is easily confirmed by transforming the constant in Eq. (8) as $(\lambda|\boldsymbol{x}^\top\boldsymbol{y}|)^{-1} = (\lambda\boldsymbol{x}^\top\boldsymbol{x}|\hat{\beta}_{\mathrm{LS}}|)^{-1}$, which indicates that the shrinkage is weak when $|\hat{\beta}_{\mathrm{LS}}|$ is large but strong when $|\hat{\beta}_{\mathrm{LS}}|$ is close to zero. Compared to Lasso, this behavior of ARD is desirable, as it maintains sparsity for weak features while alleviating unnecessary shrinkage. Figure 1 shows how shrinkage occurs in Lasso and ARD.

[1] The full derivation of Eq. (8) is shown in Appendix A.

[2] Practically, λ is set to the unbiased version of MLE (Aravkin et al., 2014).

4. BM Model and Inference Algorithms

4.1. BM Model

Obviously, the shrinkage bias of Lasso and ARD comes from their imposition regularization on the weights. For example, if $\alpha = 0$ in Lasso, the loss function becomes equivalent to that of LS, and of course, no shrinkage occurs. Using $\boldsymbol{\gamma} = \infty$ yields the same result in ARD.

Hence, we introduce a new estimation method that maintains sparsity by using latent variables instead of regularization. Let $Z \in \{0, 1\}^{N \times K}$ be binary latent variables having the same dimensionality of X. We insert Z between X and $\boldsymbol{\beta}$, i.e.,

$$\boldsymbol{y} = (X \circ Z)\boldsymbol{\beta} + \boldsymbol{\epsilon}, \tag{9}$$

where '$\circ$' denotes the Hadamard product (i.e., element-wise multiplication).

Z masks the features randomly at each sample. For Bayesian treatment, we introduce prior distributions. We assume that Z follows a Bernoulli prior distribution as $z_{nk} \sim \mathrm{Bern}(\pi_k)$, where π_k indicates how feature k is relevant. Then, estimating the priors on the masking rates automatically determines the relevance of the features.

We also introduce the priors for $\boldsymbol{\beta}$ and λ; however, we set them to be as weak as possible so that their effects are negligible when N is sufficiently large. Thus, we simply employ them as constants as they do not depend on N, i.e., $\log p(\boldsymbol{\beta}, \lambda) = O(1)$.

4.2. FAB-EM Algorithm

Our approach is based on the concept of Bayesian model selection; the central task is to evaluate the marginal log-likelihood. However, in our case, the marginal likelihood is intractable.

Thus, we adopt FIC, a recently proposed approximation for the marginal log-likelihood (Fujimaki and Morinaga, 2012; Hayashi and Fujimaki, 2013; Hayashi et al., 2015). We also adopt the factorized asymptotic Bayesian inference (FAB), which provides a tractable algorithm for parameter inference and model pruning by optimizing the lower bound of FIC. The FAB algorithm alternately maximizes the lower bound in an EM-like manner.

To obtain the lower bound of FIC, we introduce a mean-field approximation on the posterior distribution of Z as $q(\boldsymbol{z}_n) = \prod_k q(z_{nk}) = \prod_k \mathrm{Bern}(\mu_{nk})$. Then we obtain the objective function as

$$\begin{aligned}\mathcal{G}(\{\boldsymbol{\mu}_n\}, \boldsymbol{\beta}, \lambda, \boldsymbol{\pi}) \quad = \quad & \mathbb{E}_q[\log p(\boldsymbol{y}|X, Z, \boldsymbol{\beta}, \lambda)] + \mathbb{E}_q[\log p(Z|\boldsymbol{\pi})] \\ & -\frac{1}{2}\sum_k \left(\log(N\pi_k) + \frac{\sum_n \mathbb{E}_q[z_{nk}]/N - \pi_k}{\pi_k}\right) - \frac{K+1}{2}\log N \\ & + \sum_{n,k} H(q(z_{nk})),\end{aligned} \tag{10}$$

where $\mathbb{E}_q$ means the expectation under $q = q(Z)$, and H is the entropy. The derivation of Eq. (10) is described in Appendix B.

FAB E-step In the FAB E-step, we update $\{\mu_{nk}\}$. By taking the gradient of $\mathcal{G}$ w.r.t. μ_{nk} and setting it to zero, we obtain the following fixed-point equations:

$$\mu_{nk} = \sigma\left(c_{nk} + \log\frac{\pi_k}{1-\pi_k} - \frac{1}{2N\pi_k}\right) \tag{11}$$

where $\sigma(\cdot)$ is the sigmoid function and $c_{nk} = x_{nk}\beta_k\lambda(y_n - \frac{1}{2}x_{nk}\beta_k - \sum_{l\neq k}\mu_{nl}x_{nl}\beta_l)$. Updating Eq. (11) several times give us $\{\mu_{nk}\}$ at a local maximum of $\mathcal{G}$.

FAB M-step Since only the first and second terms in Eq. (10) are relevant, the FAB M-step is equivalent to the M-step in the EM algorithm. We have the closed-form solutions to update $\boldsymbol{\beta}$, λ, and $\boldsymbol{\pi}$ as

$$\begin{aligned}\boldsymbol{\beta} &= \Omega^{-1}(X\circ M)^\top\boldsymbol{y}, &(12)\\ \frac{1}{\lambda} &= \frac{\sum_n(y_n^2 - 2y_n(\boldsymbol{x}_n\circ\boldsymbol{\mu}_n)^\top\boldsymbol{\beta} + (\boldsymbol{x}_n\circ\boldsymbol{\beta})^\top\mathbb{E}_q[\boldsymbol{z}_n\boldsymbol{z}_n^\top](\boldsymbol{x}_n\circ\boldsymbol{\beta}))}{N} &(13)\\ \pi_k &= \sum_n\frac{\mu_{nk}}{N}, &(14)\end{aligned}$$

where $\Omega = \sum_n(\boldsymbol{x}_n\boldsymbol{x}_n^\top)\circ\mathbb{E}_q[\boldsymbol{z}_n\boldsymbol{z}_n^\top]$ and $M = (\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_N)^\top$. We note that $\mathbb{E}_q[\boldsymbol{z}_n\boldsymbol{z}_n^T] = \boldsymbol{\mu}_n\boldsymbol{\mu}_n^T + \mathrm{diag}(\boldsymbol{\mu}_n - \boldsymbol{\mu}_n\circ\boldsymbol{\mu}_n)$.

Pruning step As noted in previous papers, the third term in $\mathcal{G}$ penalizes $\{\mu_{nk}\}$ and automatically eliminates irrelevant features. Then, when $\pi_k = \sum_n\mu_{nk}/N < \delta$, we remove the corresponding feature from the model. The pseudo-code of the resulting algorithm is given in Algorithm 1.

Algorithm 1 Bayesian masking by FAB-EM algorithm

```
Initialize $(\{\boldsymbol{\mu}_n\}, \boldsymbol{\beta}, \lambda, \boldsymbol{\pi})$
repeat
    Update $\{\mu_{nk}\}$ by Eq. (11)
    for $k = 1, ..., K$ do
        if $\sum_n \mu_{nk}/N < \delta$ then
            Remove $k$-th dimension from the model
        end if
    end for
    Update $(\boldsymbol{\beta}, \lambda, \boldsymbol{\pi})$ by Eqs. (12-14)
until termination criterion is met
```

4.3. Analysis of FAB Estimator

As in Section 3, we investigate the FAB estimator of $\boldsymbol{\beta}$. If $\boldsymbol{y}$ follows the linear model (1) with $\boldsymbol{\beta} = \boldsymbol{\beta}^*$, the FAB estimator is expectedly obtained as

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\epsilon}}[\hat{\boldsymbol{\beta}}_{\mathrm{FAB}}] &= \Omega^{-1}\tilde{X}^\top X\boldsymbol{\beta}^*\\ &= \boldsymbol{\beta}^* + \Omega^{-1}\boldsymbol{b} \qquad (15)\end{aligned}$$

for any $q(Z)$, where $\tilde{X} = X \circ M$ and $b_k = (\boldsymbol{x}_k \circ \boldsymbol{\mu}_k)^\top \sum_{l \neq k} \beta_l^* (\boldsymbol{x}_l \circ (\mathbf{1} - \boldsymbol{\mu}_l))$.

Eq. (15) immediately suggests that $\hat{\boldsymbol{\beta}}_{\mathrm{FAB}}$ is biased by $\Omega^{-1}\boldsymbol{b}$. The bias consists of the two cross terms: $(\boldsymbol{x}_k, \boldsymbol{x}_l)$ and $(\boldsymbol{\mu}_k, \mathbf{1} - \boldsymbol{\mu}_l)$. Thus, the bias increases when $\boldsymbol{x}_k$ and $\boldsymbol{x}_l$ are correlated and $\boldsymbol{z}_k$ and $\boldsymbol{z}_l$ are negatively correlated. On the other hand, the cross terms become zero when $\mu_{nk} = 0$ or $\mu_{nl} = 1$ for all n. This also implies that the bias is weakened by an appropriately estimated q. In Section 5.2, we numerically evaluate the bias of FAB with Lasso and ARD, showing that FAB achieves the lowest bias.

Remarkably, when $K = 1$, no cross term appears and the bias vanishes for any q satisfying $\pi > 0$. Furthermore, the 1D estimator is simply written as

$$\hat{\beta}_{\mathrm{FAB}} = \frac{\tilde{\boldsymbol{x}}^\top \boldsymbol{y}}{\tilde{\boldsymbol{x}}^\top \boldsymbol{x}}. \tag{16}$$

Again, if $\mu_{nl} = 1$ for all n, $\hat{\beta}_{\mathrm{FAB}}$ recovers $\hat{\beta}_{\mathrm{LS}}$.

4.4. FAB-EG and Hybrid FAB-EM-EG Algorithms

Hayashi and Fujimaki (2013) reported that model pruning in FAB is slow, and we find that our algorithm suffers from the same drawback. In our case, $\{\pi_k\}$ for irrelevant features requires many iterations until convergence to zero although the weights $\boldsymbol{\beta}$ and $\{\pi_k\}$ for relevant features converge rapidly. To overcome the problem of slow convergence, we combine gradient ascent and reparametrization as discussed below.

First, we replace the FAB-M step by gradient ascent, which is motivated by previous studies (Salakhutdinov et al., 2003; Maeda and Ishii, 2007) on convergence analysis of the EM algorithm. Thus, we find that gradient ascent certainly helps; however, the convergence remains slow. This is because the model distribution is insensitive to the direction of π_k when β_k is small. This means that the gradient for π_k would be shallow for an irrelevant feature k, since the estimator of β_k takes a small value for the feature.

The natural gradient (NG) method (Amari, 1998) can effectively overcome the insensitivity in the model distribution. The key concept of the NG method is to adopt the Fisher information matrix as a metric on a parameter space (the so-called the Fisher information metric) in order to define the distance by the Kullback-Leibler (KL) divergence between model distributions instead of the Euclidean distance between parameter vectors. Thus, parameter updates by the NG method can make steady changes in the model distribution at each iteration. Amari and his co-workers showed that the NG method is especially effective when the parameter space contains singular regions, i.e., regions where the Fisher information matrix degenerates (Amari et al., 2006; Wei et al., 2008). This is because the problem of shallow gradient is severe around a singular region, since gradient completely vanishes along with the region. Hence, learning by ordinary gradient is often trapped around singular regions and remains trapped for many iterations, even when the models in the regions show poor agreement with data. In contrast, learning by the NG method is free from such a slow down.

In our case, the model has two singular regions for each feature, i.e., $\beta_k = 0$ and $\pi_k = 0$. In particular, singular region $\beta_k = 0$ causes the slow convergence. Hence, the NG method should be effective; however, the evaluation of the Fisher information metric is computationally expensive in our case. Therefore, as an alternative, we propose a simple reparametrization that approximates the Fisher information metric. Our strategy is to perform a block diagonal approximation of the full matrix, as has been performed recently in (Desjardins et al., 2015) for neural networks.

Toward this end, we examine the Fisher information metric in the single-parameter case ($K = 1$), which approximates diagonal blocks of the full metric. Then, the model of interest here is simply $y_n \sim \pi N(y_n|x_n\beta, \lambda^{-1}) + (1-\pi)N(y_n|0, \lambda^{-1})$. We focus on the case of small β because the slow learning of π is prominent in this region as noted above. Although the exact form of the metric is difficult to compute, our focus on the small-β case allows further approximation. Taylor expansion around $\beta = 0$ and lowest-order approximation give us a metric tensor:

$$G \equiv \begin{pmatrix} G_{\beta\beta} & G_{\beta\pi} \\ G_{\beta\pi} & G_{\pi\pi} \end{pmatrix} = \lambda N \langle x^2 \rangle \begin{pmatrix} \lambda\beta^2 N\langle x^2\rangle f(\pi) + \pi^2 & \beta\pi \\ \beta\pi & \beta^2 \end{pmatrix}., \tag{17}$$

where $f(\pi)$ represent polynomials of π and $\langle x^2 \rangle \equiv \sum_n x_n^2/N$. The key factor in G is $G_{\pi\pi} \propto \beta^2$ because this represents the fact that a smaller value of β makes the model more insensitive to changes in π.

The approximated metric obtained above is complicated and difficult to handle. Hence, we consider the following simple reparametrization for $k = 1, ..., K$:

$$(\beta_k, \pi_k) \to (\beta_k, s_k = \beta_k\pi_k). \tag{18}$$

Let us show what metric is introduced in (β, π)-space through the reparametrization. Toward this end, we recall that considering usual gradient ascent on (β, s)-space corresponds to introducing a metric $J^\top J$ in the original space, where J is the Jacobian matrix for the reparametrization. Then, the reparametrization corresponds to introducing a block diagonal metric tensor in which each diagonal block is given by

$$G' = \begin{pmatrix} 1 + \pi_k^2 & \beta_k\pi_k \\ \beta_k\pi_k & \beta_k^2 \end{pmatrix}. \tag{19}$$

The similarity between G and G' is clear. Although they are not identical, G' is simpler and shares the key factor as $G'_{\pi\pi} \propto \beta^2$.

FAB-G step In the FAB-G step, we replace the FAB-M step by gradient ascent. We calculate the gradient on the parameter space after reparametrization (β_k, s_k), and project it onto the original space. Then, we obtain the following update rule for (β_k, π_k):

$$\begin{pmatrix} \beta_k^{t+1} \\ \pi_k^{t+1} \end{pmatrix} = \begin{pmatrix} \beta_k^{t} \\ \pi_k^{t} \end{pmatrix} + \eta_t \begin{pmatrix} 1 & -\frac{\pi_k}{\beta_k} \\ -\frac{\pi_k}{\beta_k} & \frac{1+\pi_k^2}{\beta_k^2} \end{pmatrix} \begin{pmatrix} \frac{\partial \mathcal{G}}{\partial \beta_k} \\ \frac{\partial \mathcal{G}}{\partial \pi_k} \end{pmatrix}, \tag{20}$$

where t is the number iterations and $\{\eta_t\}$ are the learning coefficients. We note that the update of π_k by $\frac{\partial \mathcal{G}}{\partial \pi_k}$ is scaled by β_k^{-2}, and then accelerated when β_k is small, as expected. When $\pi_k = 1$, we update only β_k by $\eta \frac{\partial \mathcal{G}}{\partial \beta_k}$. Since the singularity problem comes from $\boldsymbol{\beta}$ and $\boldsymbol{\pi}$, not λ, updating λ retains the closed-form solution Eq. (14).

We refer to the FAB algorithm as the FAB-EG algorithm when the G step replaces the M step. Even though the FAB-EG algorithm shows faster convergence, the fast initial progress in the FAB-EM algorithm remains an attractive feature. Thus, to exploit both benefits, we propose a hybrid algorithm in which the learning progresses by FAB-EM initially and by FAB-EG later. The pseudo-code of the hybrid algorithm is given in Algorithm 2.

Algorithm 2 Bayesian masking by Hybrid FAB-EM-EG algorithm

```
Initialize ($\{\boldsymbol{\mu}_n\}, \boldsymbol{\beta}, \lambda, \boldsymbol{\pi}$)
$t \leftarrow 0$
repeat
    Update $\{\boldsymbol{\mu}_n\}$ by FAB-E step
    for $k = 1, ..., K$ do
        if $\sum_n \mu_{nk}/N < \delta$ then
            Remove $k$-th dimension from the model
        end if
    end for
    if $t < T$ then
        Update ($\boldsymbol{\beta}, \lambda, \boldsymbol{\pi}$) by FAB-M step
    else
        Update ($\boldsymbol{\beta}, \lambda, \boldsymbol{\pi}$) by FAB-G step
    end if
    $t \leftarrow t + 1$
until termination criterion is met
```

5. Experiments

5.1. Overview

First, we evaluate BM (i.e., the proposed method), Lasso, and ARD with a simple example where $K = 2$, and we show that BM outperforms the other two in terms of the sparsity-shrinkage trade-off. Second, using the same example, we show how the parameters are updated by the FAB-EM and FAB-EG algorithms. Specifically, we highlight how the use of gradient ascent and the introduction of the reparametrization help to avoid being trapped in the singular region. Third, we demonstrate that the hybrid FAB-EM-EG algorithm converges faster than the FAB-EM algorithm. Finally, we evaluate BM, Lasso, and ARD again using larger values of K.

5.2. Experiment with Two Features

For demonstration purposes, we borrowed a simple $K = 2$ setting from Aravkin et al. (2014) who considered that

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0.5 & 1 \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \end{pmatrix}, \tag{21}$$

where ϵ_1 and ϵ_2 are sampled from $N(0, 0.005)$. Assume that the true parameter values are $\bar{\beta}_1 = 0$ and $\bar{\beta}_2 = 1$, i.e., the first feature is irrelevant and the second feature is relevant. Note that the variance is supposed to be known for simplicity.

5.2.1. Comparison of BM, Lasso, and ARD

In our setting, we generated 500 datasets, each containing 2×20 samples of y. Note that, in BM (Algorithm 1), we adopted zero tolerance for model pruning: we pruned the k-th

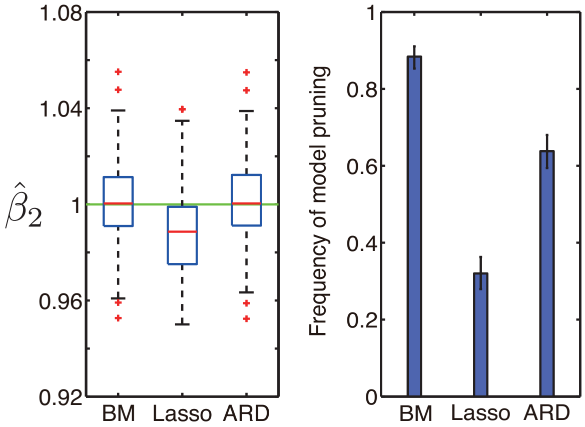

Figure 2: Estimation results on synthetic data from Eq. (21). (Left) Box plots of estimated values of β_2. The green line indicates the true value $\bar{\beta}_2 = 1$. (Right) Frequency of pruning of the irrelevant feature. The error bars represent the 95% confidence interval from fitting of the binomial distribution.

feature only when $\hat{\pi}_k$ was smaller than the machine epsilon. In Lasso, we determined α by 2-fold cross validation.

The estimation results are summarized in Figure 2; the left panel shows $\hat{\beta}_2$ when the irrelevant feature was pruned and the right panel shows the frequency of pruning of the irrelevant feature in the 500 trials. Note that the relevant feature was not pruned in any of the methods. We can easily see that BM achieved the highest sparsity without shrinkage of $\hat{\beta}_2$. On the other hand, ARD displayed no visible shrinkage as in BM; however, its sparsity was lower than that of BM. Lasso displayed shrinkage bias and the lowest sparsity.

5.2.2. Learning Trajectory of FAB-EM and FAB-EG Algorithms

Using the same simple example, we show how the parameters are updated by the FAB-EM and FAB-EG algorithms. For comparison, we also performed the FAB-EG algorithm without reparametrization. We fixed the learning coefficient as $\eta_t = 2 \times 10^{-6}$ for FAB-EG and $= 2 \times 10^{-4}$ for FAB-EG without reparametrization.

Figure 3 shows typical learning trajectories of β_1 and π_1 with 100 samples. We considered the 10 initial points of β_1 and π_1 located diagonally in the upper right. The initial values of β_2 and π_2 are set to the true values. In FAB-EM, the learning trajectories were trapped around $\beta_1 = 0$. In FAB-EG without reparametrization, the trapping was mitigated but still occurred, especially when the initial values of β_1 were small. Intuitively speaking, this is because the gradient for π_1 is shallow with small β_1. Thus, the learning trajectory approached to smaller β_1 since the feature was irrelevant, and then, the learning of π_1 became slower. In contrast, the learning trajectories of FAB-EG with the reparametrization approached to $\pi_1 = 0$ with fewer iterations regardless of the initial points, which means that the irrelevant feature was pruned quickly. This result empirically demonstrates that using

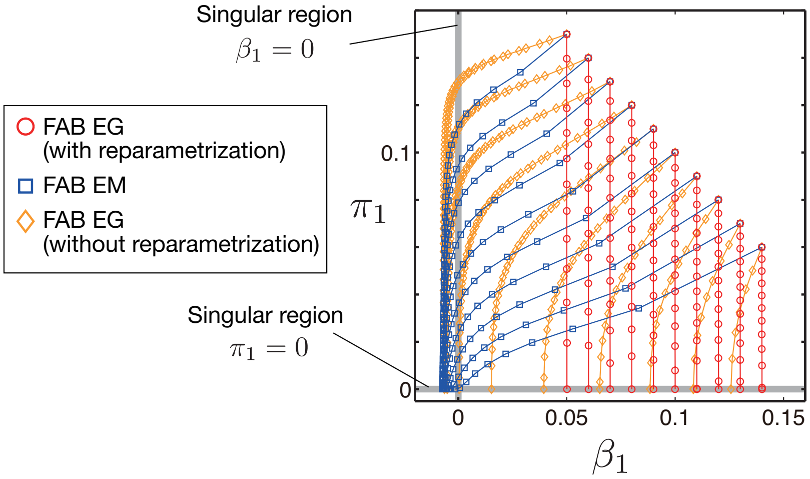

Figure 3: Typical learning trajectories on β_1-π_1 plane from 10 different initial points by FAB-EM steps and FAB-EG steps with/without reparametrization.

gradient ascent alone improves the convergence only slightly, but combining it with the reparametrization accelerates the convergence sharply.

5.3. Experiment with Larger Number of Features

Next, we explain the results with larger examples ($10 \leq K \leq 100$). We generated $\boldsymbol{\beta}$ and X from the uniform distribution in $[0, 1]$, and half of the elements in $\boldsymbol{\beta}$ were set as zero. $\boldsymbol{y}$ was generated by Eq. (1) with $\lambda^{-1} = 0.2$. N was set as $20K$. We controlled $\{\eta_t\}$ as described in Appendix C.

5.3.1. Performance Validation of Hybrid FAB-EM-EG algorithm

With $K = 50$, we demonstrate that the hybrid FAB-EM-EG algorithm converges faster than the FAB-EM algorithm. Toward this end, we counted the number of correctly pruned features and plotted it against the elapsed time for the algorithms. We set T in Algorithm 2 to 200 iterations. Figure 4 shows the number of correctly pruned features against the elapsed time. We can clearly see the faster convergence of the hybrid FAB-EM-EG algorithm. Note that the faster convergence is not attributable to over-pruning because the number of wrongly pruned features at termination were 2.0 ± 1.3 (hybrid FAB-EM-EG) and 1.8 ± 1.3 (FAB-EM-EG), which were nearly equal.

5.3.2. Precision and Recall

For $K > 2$, we used two performance measures, Recall and Precision, defined as Precision $= m_3/m_2$ and Recall $= m_3/m_1$, where m_1 and m_2 are the numbers of true and estimated irrelevant features, respectively, and m_3 is the number of correctly pruned features. We examined $K = 10$, 30, 50, and 100, and for each K, we generated 100 datasets. Figure 5

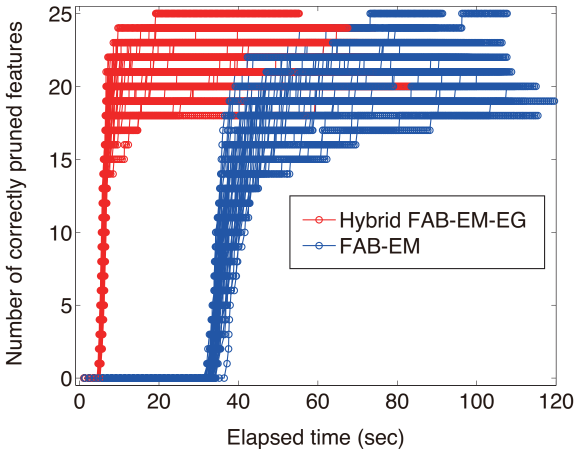

Figure 4: Performance validation of the hybrid FAB-EM-EG algorithm. For the hybrid FAB-EM-EG and the FAB-EM algorithms, the number of correctly pruned features is plotted against the elapsed time. Different lines of the same color represent different datasets.

summarizes the estimation results for BM (Algorithm 2), Lasso, and ARD. We set the algorithm switching point T as 500 iterations. In Lasso, α was determined by 10-fold cross validation. As shown in the left and middle panels, although BM displayed slightly lower Precision than the others, it achieved the highest Recall. We also computed the F_1 score, the harmonic mean of Precision and Recall, which can be interpreted as a metric for evaluating the performance in terms of the sparsity-shrinkage trade-off. As shown in the right panel, BM attained the highest F_1 score for all K values in this range. Thus, we concluded that BM achieved the best performance for the larger values of K.

6. Conclusion

In this paper, we proposed a new sparse estimation method, BM, whose key feature is that it does not impose direct regularization on the weights. Our strategy was to introduce binary latent variables that randomly mask features and to perform a variational Bayesian inference based on FIC. In addition, we introduced gradient ascent and the reparametrization to accelerate the convergence. Our analysis of the estimators of BM, Lasso, and ARD highlighted how their sparsity mechanisms are different from one another. Finally, experimental comparisons of the three methods demonstrated that BM achieves the best performance in terms of the sparsity-shrinkage trade-off.

Note that augmenting a statistical model by random masking variables itself is not a new idea. For example, van der Maaten et al. (2013) used random masking to generate virtual training data. However, our approach is distinguished from those studies by its purpose. Namely, we aim to identify whether the features are relevant or not, rather than improving prediction performance. In the augmented model, the FAB algorithm penalizes the masking

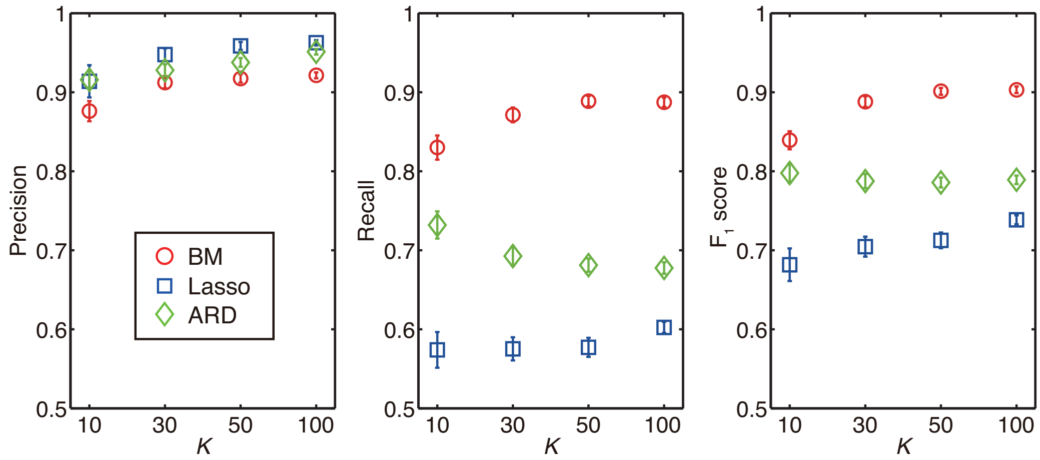

Figure 5: Performance measures are plotted with standard errors: (Left) Precision, (Middle) Recall, and (Right) F_1 score.

rates, i.e., existence probability of the features, unlike the sparse regularization techniques where the weight values of the features are penalized. Applying the BM to real-world tasks where model identification is crucial, e.g., causal network inference, is a promising future work.

Acknowledgments

Y.K. was supported by the Platform Project for Supporting in Drug Discovery and Life Science Research (Platform for Dynamic Approaches to Living Systems) from Japan Agency for Medical Research and development (AMED). S.M. and K.H. were supported by JSPS KAKENHI Grant Number 15K15210 and 15K16055, respectively.

References

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10: 251–276, 1998.

Shun-ichi Amari, Hyeyoung Park, and Tomoko Ozeki. Singularities affect dynamics of learning in neuromanifolds. *Neural Computation*, 18(5):1007–1065, 2006.

Aleksandr Aravkin, James V. Burke, Alessandro Chiuso, and Gianluigi Pillonetto. Convex vs non-convex estimators for regression and sparse estimation: the mean squared error properties of ARD and GLasso. *Journal of Machine Learning Research*, 15:217–252, 2014.

G. Desjardins, K. Simonyan, R. Pascanu, and K. Kavukcuoglu. Natural Neural Networks. *ArXiv e-prints*, 2015.

Ryohei Fujimaki and Satoshi Morinaga. Factorized asymptotic bayesian inference for mixture modeling. In *AISTATS*, 2012.

Kohei Hayashi and Ryohei Fujimaki. Factorized asymptotic bayesian inference for latent feature models. In *27th Annual Conference on Neural Information Processing Systems (NIPS)*, 2013.

Kohei Hayashi, Shin-ichi Maeda, and Ryohei Fujimaki. Rebuilding factorized information criterion: Asymptotically accurate marginal likelihood. In *International Conference on Machine Learning (ICML)*, 2015.

DavidJ MacKay. Bayesian Methods for Backpropagation Networks. In *Models of Neural Networks III*, pages 211–254. Springer, 1994.

Shin-ichi Maeda and Shin Ishii. Convergence analysis of the EM algorithm and joint minimization of free energy. In *IEEE Workshop on Machine Learning for Signal Processing*, 2007.

Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer, 1996.

K. B. Petersen and M. S. Pedersen. The matrix cookbook, 2012.

Ruslan Salakhutdinov, Sam Roweis, and Zoubin Ghahramani. Optimization with EM and expectation-conjugate-gradient. In *International Conference on Machine Learning (ICML)*, 2003.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

Laurens van der Maaten, Minmin Chen, Stephen Tyree, and Kilian Q. Weinberger. Learning with marginalized corrupted features. In *International Conference on Machine Learning (ICML)*, 2013.

Haikun Wei, Jun Zhang, Florent Cousseau, Tomoko Ozeki, and Shun-ichi Amari. Dynamics of learning near singularities in layered networks. *Neural Computation*, 20(3):813–843, 2008.

David P. Wipf and Srikantan S. Nagarajan. A new view of automatic relevance determination. In *Advances in Neural Information Processing Systems 20*, 2008.

Appendix A. The One-Dimensional ARD Estimator

According to Wipf and Nagarajan (2008), the negative marginal log-likelihood when $K = 1$ is given by

$$\log|\lambda^{-1}I + \gamma \boldsymbol{x}\boldsymbol{x}^\top| + \boldsymbol{y}^\top(\lambda^{-1}I + \gamma\boldsymbol{x}\boldsymbol{x}^\top)^{-1}\boldsymbol{y} \tag{22}$$

$$= \log(\lambda^{-1} + \gamma\boldsymbol{x}^\top\boldsymbol{x}) + \lambda\boldsymbol{y}^\top\boldsymbol{y} - \boldsymbol{y}^\top\frac{\lambda^2\gamma\boldsymbol{x}\boldsymbol{x}^\top}{1+\lambda\gamma\boldsymbol{x}^\top\boldsymbol{x}}\boldsymbol{y} \tag{23}$$

$$= \log(\lambda^{-1} + \gamma\boldsymbol{x}^\top\boldsymbol{x}) + \lambda\boldsymbol{y}^\top\boldsymbol{y} - \frac{\lambda\gamma(\boldsymbol{x}^\top\boldsymbol{y})^2}{\lambda^{-1}+\gamma\boldsymbol{x}^\top\boldsymbol{x}}. \tag{24}$$

In the second line, we use the matrix determinant lemma (Petersen and Pedersen, 2012, Eq. (24)) for the first term and the variant of the Sherman-Morrison relation (Petersen and Pedersen, 2012, Eq. (160)) for the second term. The derivative is

$$\frac{\partial \text{ Eq. (24)}}{\partial \gamma} = \frac{\boldsymbol{x}^\top \boldsymbol{x}}{\lambda^{-1} + \gamma \boldsymbol{x}^\top \boldsymbol{x}} - \frac{\lambda(\boldsymbol{x}^\top \boldsymbol{y})^2}{\lambda^{-1} + \gamma \boldsymbol{x}^\top \boldsymbol{x}} + \frac{\lambda\gamma(\boldsymbol{x}^\top \boldsymbol{y})^2 \boldsymbol{x}^\top \boldsymbol{x}}{(\lambda^{-1} + \gamma \boldsymbol{x}^\top \boldsymbol{x})^2} \tag{25}$$

$$= \frac{\boldsymbol{x}^\top \boldsymbol{x}(\lambda^{-1} + \gamma \boldsymbol{x}^\top \boldsymbol{x}) - \lambda(\boldsymbol{x}^\top \boldsymbol{y})^2(\lambda^{-1} + \gamma \boldsymbol{x}^\top \boldsymbol{x}) + \lambda\gamma(\boldsymbol{x}^\top \boldsymbol{y})^2 \boldsymbol{x}^\top \boldsymbol{x}}{(\lambda^{-1} + \gamma \boldsymbol{x}^\top \boldsymbol{x})^2} \tag{26}$$

$$= \frac{\lambda^{-1}\boldsymbol{x}^\top \boldsymbol{x} + \gamma(\boldsymbol{x}^\top \boldsymbol{x})^2 - (\boldsymbol{x}^\top \boldsymbol{y})^2}{(\lambda^{-1} + \gamma \boldsymbol{x}^\top \boldsymbol{x})^2}. \tag{27}$$

The stationary point is then given as

$$\hat{\gamma} = \max(0, \frac{(\boldsymbol{x}^\top \boldsymbol{y})^2 - \lambda^{-1}\boldsymbol{x}^\top \boldsymbol{x}}{(\boldsymbol{x}^\top \boldsymbol{x})^2}) \tag{28}$$

$$= \max(0, \hat{\beta}_{\text{LS}}^2 - (\lambda \boldsymbol{x}^\top \boldsymbol{x})^{-1}). \tag{29}$$

Note that we use the max operator since γ is the variance and it must be non-negative. By substituting the result of $\hat{\gamma} > 0$ into Eq. (5), we obtain

$$\hat{\beta}_{\text{ARD}} = \hat{\gamma}\boldsymbol{x}^\top(\lambda^{-1}I + \hat{\gamma}\boldsymbol{x}\boldsymbol{x}^\top)^{-1}\boldsymbol{y} \tag{30}$$

$$= \lambda\hat{\gamma}\boldsymbol{x}^\top \boldsymbol{y} - \frac{\lambda\hat{\gamma}^2\boldsymbol{x}^\top\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{y}}{\lambda^{-1} + \hat{\gamma}\boldsymbol{x}^\top\boldsymbol{x}} \tag{31}$$

$$= \lambda\hat{\beta}_{\text{LS}}^2\boldsymbol{x}^\top\boldsymbol{y} - \hat{\beta}_{\text{LS}} - \frac{\lambda\hat{\beta}_{\text{LS}}^4\boldsymbol{x}^\top\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{y} - 2\hat{\beta}_{\text{LS}}^2\boldsymbol{x}^\top\boldsymbol{y} + \frac{\lambda\boldsymbol{x}^\top\boldsymbol{y}}{\lambda^2\boldsymbol{x}^\top\boldsymbol{x}}}{\lambda^{-1} + \hat{\beta}_{\text{LS}}^2\boldsymbol{x}^\top\boldsymbol{x} - \lambda^{-1}} \tag{32}$$

$$= \lambda\hat{\beta}_{\text{LS}}^2\boldsymbol{x}^\top\boldsymbol{y} - \hat{\beta}_{\text{LS}} - \frac{\lambda\hat{\beta}_{\text{LS}}^4\boldsymbol{x}^\top\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{y} - 2\hat{\beta}_{\text{LS}}^2\boldsymbol{x}^\top\boldsymbol{y} + \lambda^{-1}\hat{\beta}_{\text{LS}}}{\hat{\beta}_{\text{LS}}^2\boldsymbol{x}^\top\boldsymbol{x}} \tag{33}$$

$$= -\hat{\beta}_{\text{LS}} + 2\hat{\beta}_{\text{LS}} - \frac{1}{\lambda\hat{\beta}_{\text{LS}}\boldsymbol{x}^\top\boldsymbol{x}} \tag{34}$$

$$= \hat{\beta}_{\text{LS}} - \frac{1}{\lambda\boldsymbol{x}^\top\boldsymbol{y}}. \tag{35}$$

Recall that $\hat{\beta}_{\text{ARD}} = 0$ when $\hat{\gamma} \leq 0$. Since $\hat{\beta}_{\text{LS}}^2$ and $(\lambda\boldsymbol{x}^\top\boldsymbol{x})^{-1}$ are both non-negative, the condition $\hat{\gamma} \leq 0$ is written as $\hat{\beta}_{\text{LS}}^2 \leq (\lambda\boldsymbol{x}^\top\boldsymbol{x})^{-1}$, or equivalently, $|\hat{\beta}_{\text{LS}}| \leq |\lambda\boldsymbol{x}^\top\boldsymbol{y}|^{-1}$. Substituting this condition into the above equation yields Eq. (8).

Appendix B. Derivation of The Lower Bound of FIC

Hayashi et al. (2015) obtained a general representation of the lower bound of FIC, and in this case, we have

$$\text{FIC} \geq \mathbb{E}_q[\log p(\boldsymbol{y}, Z|X, \boldsymbol{\beta}, \lambda, \boldsymbol{\pi})] - \frac{1}{2}\mathbb{E}_q[\log|F_{\boldsymbol{\beta}}|] - \frac{K+1}{2}\log N + H(q), \tag{36}$$

where $q = q(Z)$ and $F_{\boldsymbol{\beta}}$ denotes the Hessian matrix of the negative log-likelihood w.r.t. $\boldsymbol{\beta}$. Note that the priors for $\boldsymbol{\beta}$ and λ do not appear in the lower bound, since the priors do not depend on N, as assumed in Section 4.1. In order to derive the FAB algorithm, we compute the lower bound of the second term on the right-hand side.

The Hessian matrix $F_{\boldsymbol{\beta}}$ is represented as

$$\begin{aligned} F_{\boldsymbol{\beta}} &= -\nabla_{\boldsymbol{\beta}}\nabla_{\boldsymbol{\beta}} \log p(\boldsymbol{y}, Z|X, \boldsymbol{\beta}, \lambda, \boldsymbol{\pi}) & (37) \\ &= (X \circ Z)^\top (X \circ Z). & (38) \end{aligned}$$

Then, using Hadamard's inequality for a positive-semidefinite matrix yields

$$\begin{aligned} -\log|F_{\boldsymbol{\beta}}| &\geq -\sum_k \log \sum_n x_{nk}^2 z_{nk} & (39) \\ &\geq -\sum_k \log(\sum_n z_{nk})(\sum_n x_{nk}^2) & (40) \\ &= -\sum_k \log \sum_n z_{nk} + \text{const.} & (41) \end{aligned}$$

As stated in Fujimaki and Morinaga (2012), since $-\log \sum_n z_{nk}$ is a concave function, its linear approximation at $N\pi_k > 0$ yields the lower bound:

$$-\mathbb{E}_q[\log \sum_n z_{nk}] \geq -\left(\log N\pi_k + \frac{\sum_n \mathbb{E}_q[z_{nk}]/N - \pi_k}{\pi_k}\right). \tag{42}$$

Thus, we obtain Eq. (10) in the main text.

Appendix C. Control of Learning Coefficient

We explain how to set the learning coefficients $\{\eta_t\}$ in Section 5.3. We set η_t to a constant value η; however, when the maximum of the update of $\{\pi_k\}$ is greater than 0.05, η_t is modified such that the maximum is 0.05. We chose the constant η as $2 \times 10^{-2}/N$, where N is the number of samples.