Fast Laplace Approximation for Sparse Bayesian Spike and Slab Models

Syed Abbas Zilqurnain Naqvi
Purdue University
naqvi@purdue.edu

Shandian Zhe
Purdue University
szhe@purdue.edu

Yuan Qi
Purdue University
alanqi@cs.purdue.edu

Jieping Ye
University of Michigan, Ann Arbor
jpye@umich.edu

Abstract

We propose a simple yet effective fast approximate inference algorithm based on Laplace's method for Bayesian spike-and-slab models. Unlike previous variational Bayes (VB) and expectation propagation (EP) approximations that fit a factorized distribution to the exact joint posterior, our method computes approximate posterior marginals without taking any factorization assumption, and enjoys a linear computational cost in the number of features. Moreover, our method provides numerical estimation of selection probabilities and indicator variables to quantify selection uncertainty for each individual feature. The simulation and real-world data verify the inference quality, computational efficiency and predictive performance of the proposed algorithm.

1 Introduction

We consider the Bayesian spike-and-slab models for very high dimensional feature selection problems. For very high dimensional problems, existing Monte Carlo methods [9] converge slowly with tens of thousands of features in data; and the variational Bayes (VB) and expectation propagation (EP) approaches [3, 4, 5] either need a fully factorized approximation to obtain a linear cost but at the price of a reduced approximation quality, or have a quadratic cost, making them impractical for large data. By contrast, the frequentist L_1-type methods have fast solvers developed over years, making them a practical tool. To address the computational issue associated with the spike-and-slab model, we develop the Fast Laplace Approximation for Spike-and-slab (FLAS) model. FLAS not only maintains the benefits of the Bayesian treatment (e.g., uncertainty quantification) but also is computationally as efficient as the frequentist methods.

Specifically, we apply the Laplace approximation to the marginal posterior distribution of each weight parameter. We exploit two efficient optimization methods, GIST [2] and L-BFGS [10], to obtain the mode of the posterior distribution. Then we use ensemble Nyström to calculate the diagonal of the inverse Hessian over the mode to obtain the approximate posterior marginals. The theoretical analysis of the error bound is also provided. With the posterior marginals of model weights, we use quadrature integration to estimate the marginal posteriors of selection probabilities and indicator variables for all features, which quantify the selection uncertainty. While a factorized joint posterior assumption is usually not true, VB and EP often adopt it for computational efficiency. By contrast, our method is free of this assumption but still enjoys a linear cost $O(np)$, where n and p are the numbers of samples and features, respectively.

On simulated data, the new method FLAS performs feature selection better than or comparable to the alternative approximate methods with less running time, and provides higher prediction accuracy than various sparse methods including VB, EP, automatic relevance determination (ARD), lasso, elastic net and a capped-L_1 method. On large real benchmark datasets, our method often achieves improved prediction accuracy with a comparable speed than the alternative scalable sparse methods, such as ARD and capped-L_1.

2 Spike-and-Slab Models

We first present sparse linear models with spike-and-slab priors. Suppose we have n independent and identically distributed samples $\mathcal{D} = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_n, t_n)\}$, where $\mathbf{x}_i$ is the p dimensional feature vector of the i-th sample, and t_i is its response. We aim at predicting the response vector $\mathbf{t} = [t_1, \ldots, t_n]^\top$ based on the feature set $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top$ and selecting a small number of features relevant to the prediction. For real-world applications, we often have $n \ll p$.

For regression, the Gaussian data likelihood is used: $p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \tau) = \prod_{i=1}^n \mathcal{N}(t_i|\mathbf{x}_i^\top \mathbf{w}, \tau^{-1})$ where $\mathbf{w}$ are regression weights, and τ is the precision parameter; for classification, the logistic likelihood is used: $p(\mathbf{t}|\mathbf{X}, \mathbf{w}) = \prod_{i=1}^n \sigma(\mathbf{x}_i^\top \mathbf{w})^{t_i}[1 - \sigma(\mathbf{x}_i^\top \mathbf{w})]^{1-t_i}$ where $t_i \in \{0, 1\}$, $\mathbf{w}$ are classifier weights, and $\sigma(a) = 1/(1 + \exp(-a))$.

A set of latent binary variables $\{z_j\}$ are introduced to indicate the feature selection: $z_j = 1$ means the j-th feature is selected; otherwise, it is not. Then a spike-and-slab prior [9] over $\mathbf{w}$ is assigned:

$$p(\mathbf{w}|\mathbf{z}) = \prod_{j=1}^p \mathcal{N}(w_j|0, r_0)^{(1-z_j)} \mathcal{N}(w_j|0, r_1)^{z_j}, \tag{1}$$

$$p(z_j = 1|s_j) = s_j \quad (1 \le j \le p) \tag{2}$$

where r_0 and r_1 are the variances of the two Gaussian components and $s_j \in [0, 1]$ represents the selection probability for the j-feature. We set $r_1 \gg r_0$ so that if the j-th feature is selected, the prior over w_j has a large variance r_1 (as a regular L_2 penalty in the frequentist framework) and, if not, the zero-mean prior has a very small variance r_0, leading to aggressive shrinkage of the irrelevant feature. We further assign a uninformative Beta prior over s_j: $p(s_j) = \text{Beta}(1, 1)$.

3 Algorithm

3.1 Optimization of the Model

To obtain the Laplace approximation, we need to compute the mode and the second-order derivative of the log posterior distribution at the mode. Due to the nonconvexity of the spike-and-slab model, we use two scalable nonconvex optimization methods, L-BFGS [10] and GIST [2]. For L-BFGS, we marginalize out both $\mathbf{z}$ and $\mathbf{s}$ and optimize $\mathbf{w}$; for GIST, we only marginalize out $\mathbf{z}$ and jointly optimize $\mathbf{w}$ and $\mathbf{s}$. Both methods are efficient and have computational costs linear in p. The details are given in the appendix.

3.2 Laplace Approximation for Marginal Posterior of Weights

Standard Laplace approximation requires to invert the Hessian matrix of the negative log probability at the mode, via which we can obtain a joint approximate posterior: $p(\mathbf{w}|\mathbf{X}, \mathbf{t})$. For prediction and feature selection, however, we only need marginal posterior of each weight w_j, which only requires the diagonal entry of the inverse Hessian. Nevertheless, we still have to invert the Hessian matrix, which has time complexity of $O(p^3)$ and is unacceptable for large problems. To resolve this issue, we resort to Nyström method. Specifically, let us denote the mode of the model weights by $\tilde{\mathbf{w}}$ and consider the Hessian matrix in regression case first, $H = \tau \mathbf{X}^\top \mathbf{X} + \text{diag}(\mathbf{v})$ where $v_j = -\frac{d^2 \log(p(w_j))}{dw_j^2}\Big|_{w_j = \tilde{w}_j}$, where $p(w_j)$ is the prior probability after marginalizing z_j and s_j. For GIST, we take the second derivative of $p(w_j, \tilde{s}_j)$ to approximate v_j. Then the Nyström approach is used to approximate $\mathbf{X}^\top \mathbf{X}$: A subset of columns of $\mathbf{X}$ are sampled to form a low-rank $n \times k$ matrix $\mathbf{X}_k = [\mathbf{f}_{i_1}, \ldots, \mathbf{f}_{i_k}]$, where $\mathbf{f}_{i_t}$ is the i_t-th column of $\mathbf{X}$; and $\mathbf{X}^\top \mathbf{X} \approx \mathbf{X}^\top \mathbf{X}_k (\mathbf{X}_k^\top \mathbf{X}_k)^\dagger \mathbf{X}_k^\top \mathbf{X}$ where $(\cdot)^\dagger$ is the generalized inverse operation. The inverse of Hessian is then approximated by

$$\mathbf{H}^{-1} \approx \tilde{\mathbf{H}}^{-1}, \quad \tilde{\mathbf{H}} = \tau \mathbf{X}^\top \mathbf{X}_k (\mathbf{X}_k^\top \mathbf{X}_k)^\dagger \mathbf{X}_k^\top \mathbf{X} + \text{diag}(\mathbf{v}). \tag{3}$$

Applying Woodbury matrix identity [14], we can readily reduce the complexity to $O(nkp)$:
$\tilde{\mathbf{H}}^{-1} = \mathrm{diag}(\mathbf{v})^{-1} - \mathrm{diag}(\mathbf{v})^{-1}\mathbf{X}^\top\mathbf{X}_k(\tau^{-1}\mathbf{X}_k^\top\mathbf{X}_k + \mathbf{X}_k^\top\mathbf{X}\mathrm{diag}(\mathbf{v})^{-1}\mathbf{X}^\top\mathbf{X}_k)^{-1}\mathbf{X}_k^\top\mathbf{X}\mathrm{diag}(\mathbf{v})^{-1}$.

Since we can choose $k \ll p$, the inversion cost will still be linear in p. We can then read off the diagonal of $\tilde{\mathbf{H}}^{-1}$ to calculate the marginal posterior approximation of each w_j: a Gaussian with mean m_j being the posterior mode $\tilde{w}_j$ and variance σ_j^2 equal to $\tilde{\mathbf{H}}^{-1}(j,j)$.

For classification, the Hessian matrix has a slight different form: $\mathbf{H} = \tilde{\mathbf{X}}^\top\tilde{\mathbf{X}} + \mathrm{diag}(\mathbf{v})$ where $\tilde{\mathbf{X}} = \mathbf{X}\mathrm{diag}(\sqrt{\mathbf{b}})$, and $b_i = \sigma(\mathbf{x}_i^\top\tilde{\mathbf{w}})(1-\sigma(\mathbf{x}_i^\top\tilde{\mathbf{w}}))$. Then we follow the same way as in regression case to calculate the Laplace approximation for each w_j.

Using Nyström approach to estimate the diagonal of inverse Hession will inevitably bring some approximation error. To improve the accuracy, a simple ensemble approach is proposed. Specifically, we first sample d disjoint sets of columns of $\mathbf{X}$, each set is of the same size k. For each set r, we can calculate an approximate inverse Hessian $\tilde{\mathbf{H}}_r^{-1}$. The estimation of the j-th diagonal entry of inverse Hession is then obtained by $\mathbf{H}^{-1}(j,j) \approx \frac{1}{d}\sum_{r=1}^{d}\tilde{\mathbf{H}}_r^{-1}(j,j)$. Using Taylor expansion and error bounds of Nyström approximations[7], we can prove that the proposed ensemble approach can have a smaller estimation error. This is expressed in the following theorems (the proof details are given in the appendix).

Theorem 1. Define $\Omega = \{\mathbf{A} \in \mathbb{R}^{p\times p} | \mathbf{A} \succ \mathbf{0}, \lambda_{min}(\mathbf{A}) \geq c, \lambda_{max}(\mathbf{A}) < \infty\}$. Assume Hessian $\mathbf{H}$ and approximate Hessian $\tilde{\mathbf{H}}$ both belong to $\mathbf{\Omega}$. Consider a function $f(\mathbf{A}) = \mathbf{e}_j^\top\mathbf{A}^{-1}\mathbf{e}_j, \mathbf{A} \in \mathbf{\Omega}$. Then, $\|\nabla f(\mathbf{A})\|_F \leq L$, $(1-\eta)\mathbf{H} + \eta\tilde{\mathbf{H}} \in \mathbf{\Omega}\ \forall\ \eta \in [0,1]$, and with high probability,

$$|\mathbf{H}^{-1}(j,j) - \tilde{\mathbf{H}}^{-1}(j,j)| \leq L \cdot D_0$$

where c is a small positive constant, and $L = p/c^2$. $\mathbf{e}_j$ is a standard basis vector with 1 in j-th coordinate and 0's elsewhere, and D_0 is the standard Nyström error bound based on Frobenius norm [7].

Theorem 2. Define $\Omega = \{\mathbf{A} \in \mathbb{R}^{p\times p} | \mathbf{A} \succ \mathbf{0}, \lambda_{min}(\mathbf{A}) \geq c, \lambda_{max}(\mathbf{A}) < \infty\}$. Assume Hessian $\mathbf{H}$ and a set of approximate Hessians $\{\tilde{\mathbf{H}}_1, \ldots, \tilde{\mathbf{H}}_d\}$ all belong to $\mathbf{\Omega}$, then with high probability,

$$|\mathbf{H}^{-1}(j,j) - \frac{1}{d}\sum_{r=1}^{d}\tilde{\mathbf{H}}_r^{-1}(j,j)| \leq L \cdot D_1$$

where D_1 the error bound for ensemble Nyström based on Frobenius norm. Because $D_1 < D_0$ (see [7]), the ensemble approach for diagonal entry estimation of $\mathbf{H}^{-1}$ has a smaller error bound.

3.3 Estimation of Selection Probabilities

Given the Laplace approximation, we can estimate the probability s_j for selecting the j-th feature, without assuming these probabilities are factorized over the features. In contrast, both VB and EP approximations use the factorized approximation over the probabilities of the binary selection indicators $\mathbf{z}$.

We first invert the conditional relationship between s_j and w_j based on Bayes rule, $p(s_j|w_j) = \frac{s_j\mathcal{N}(w_j|0,r_1)+(1-s_j)\mathcal{N}(w_j|0,r_0)}{\frac{1}{2}\mathcal{N}(w_j|0,r_1)+\frac{1}{2}\mathcal{N}(w_j|0,r_0)}$. Then the marginal posterior of s_j can be computed by $p(s_j|\mathbf{t},\mathbf{X}) = \int p(s_j|w_j)\mathcal{N}(w_j|m_j,\sigma_j^2)\mathrm{d}w_j$ where $\mathcal{N}(w_j|m_j,\sigma_j^2)$ is the estimated posterior marginal of w_j. Then, the posterior mean and variance of s_j are calculated by

$$\mathrm{E}[s_j] = \int \frac{2\mathcal{N}_1(w_j) + \mathcal{N}_0(w_j)}{3(\mathcal{N}_1(w_j) + \mathcal{N}_0(w_j))}q(w_j)\mathrm{d}w_j, \quad \mathrm{Var}[s_j] = \int \frac{3\mathcal{N}_1(w_j) + \mathcal{N}_0(w_j)}{6(\mathcal{N}_1(w_j) + \mathcal{N}_0(w_j))}q(w_j)\mathrm{d}w_j - \mathrm{E}^2$$

where $\mathcal{N}_g(w_j)$ (for $g = 0, 1$) and $q(w_j)$ are the shorthand for $\mathcal{N}(w_j|0,r_g)$ and $\mathcal{N}(w_j|m_j,\sigma_j^2)$ respectively. A similar procedure can be used to calculate the posterior moments of z_j—the selection indicator of j-th feature. The integrations to calculate the means and variances of s_j and z_j do not have a closed-form solution. So we apply Gauss-Hermite quadrature method [8] to obtain an estimation. Since the integration is one dimensional and smooth, the quadrature method is computationally efficient and accurate: With only 5 quadrature nodes (or function evaluations), we can estimate $\mathrm{E}(s_j)$, $\mathrm{E}(z_j)$, $\mathrm{Var}(s_j)$, and $\mathrm{Var}(z_j)$ with high accuracy (e.g., the numerical difference from the true integration is often on the order of 10^{-4}). It is easy to see that both time and space complexity of our algorithm are $O(np)$, including estimating the posterior mean and variance of $\mathbf{w}$, $\mathbf{s}$, and $\mathbf{z}$. The linear cost makes our algorithm scalable for large data.

4 Experiments

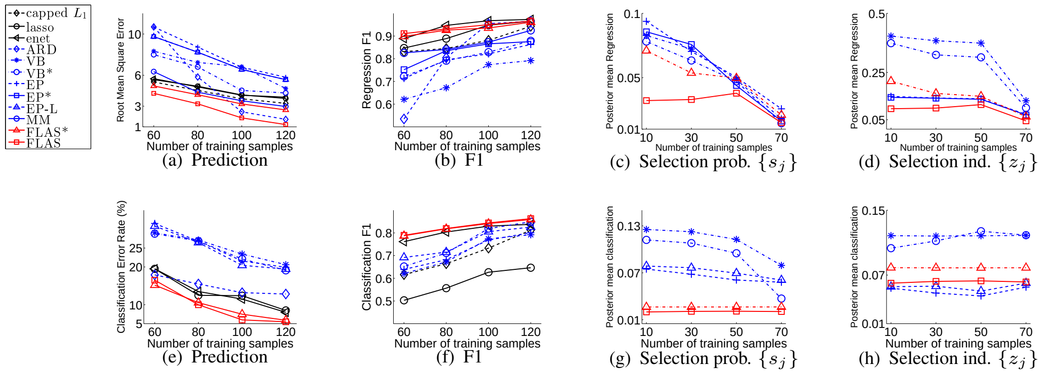

Figure 1: Simulation results, including the prediction accuracy, the F1 score of feature selection, and the root mean squared error for the posterior mean estimation of $\{s_j\}$ and $\{z_j\}$. Results are averaged over 50 runs.

Table 1: The training time (seconds) on simulated data ($p = 1000$).

(a) Regression

method	60	80	100	120
capped L_1	**0.0054**	0.0705	**0.0103**	**0.0108**
lasso	0.0312	0.0313	0.0321	0.0329
elastic net	0.0360	0.0346	0.0352	0.0349
ARD	0.03	0.17	0.20	0.67
VB	2.4161	2.3999	2.4794	2.4404
VB*	2.7544	3.0118	2.6012	2.7795
EP	0.9345	1.0478	1.1160	1.1468
EP*	0.505	0.681	1.047	1.936
MM	2.5230	1.1047	0.4314	0.5282
FLAS*	0.0664	0.0642	0.0704	0.0855
FLAS	0.0140	**0.0154**	0.0216	1.4526

(b) Classification

method	60	80	100	120
capped L_1	0.0180	0.0499	0.0427	0.0559
lasso	0.1033	0.1289	0.1555	0.1821
elastic net	0.08690	0.1009	0.1163	0.1356
ARD	0.06	0.07	0.15	0.45
VB	10.3312	11.2570	12.3317	13.3366
VB*	0.0812	0.1570	2.8915	3.0194
EP	1.1165	1.1695	1.2400	1.3090
EP-L	0.0132	0.0581	0.0598	0.1631
FLAS*	0.0344	0.0736	0.0794	0.1929
FLAS	**0.0097**	**0.0111**	**0.0139**	**0.0152**

We compared our approach with alternative approximate inference algorithms for the spike-and-slab model, including EP, VB and MM [15]. We used three versions of EP algorithms, where one is based on continuous spikes [5](EP), another is based on delta spikes (EP*), and the third one [4] uses fully factorized posterior assumption for model weights to obtain a linear cost $O(np)$ in classification context (EP-L). For VB, we used two versions: [16](VB) having cubic cost $O(p^3)$ but without a factorized posterior assumption over model weights, and [13](VB*) using a fully factorized posterior assumption with reduced cost $O(np^2)$. We also tested other popular sparse learning methods, including ARD, lasso, elastic net, and capped L_1. Our algorithms based on L-BFGS and GIST are denoted by FLAS and FLAS* respectively. The solution of L_2 regularization was used as initialization. We set $k = 5, d = 5$ for ensemble Nyström.

We first examined our algorithm in a simulation study. The study aimed to evaluate our algorithm in four aspects: (i) the predictive performance when $p \gg n$, (ii) the capability to select relevant features and (iii) the accuracy of the estimated posterior of the selection probabilities, (iii) the running time. To do so, we simulated data where $p = 1000$ and only 20 features are relevant to the responses. The relevant features were generated from a multi-variate Gaussian distribution with a block diagonal covariance matrix. We fixed the number of test samples to 200 and vary the number of training samples n from $\{60, 80, 100, 120\}$. For each n, we randomly generated 50 datasets and reported the average results. To evaluate the accuracy of posterior inference, we ran a similar simulation with $p = 100$, because the ground truth was generated by Gibbs sampling, which converges slowly for high dimensional problems. The details are given in the appendix.

As we can see from Figure 1 and Table 1, in most cases our algorithm exhibits better performance than competing methods not only in prediction accuracy, but also in feature selection and uncertainty quantification. Furthermore, our algorithm converges faster than EP and VB and and is comparable to ARD and L_1 type methods in terms of running speed.

Finally, on real-world large benchmark data, our algorithm often obtained higher prediction accuracy, with running time comparable to the efficient sparse methods, such as capped L_1 and ARD. The results are provided in the appendix.

References

[1] Nicholas Francis Arcolano. *Approximation of Positive Semidefinite Matrices Using the Nystrom Method*. PhD thesis, HARVARD UNIVERSITY, 2011.

[2] Pinghua Gong, Changshui Zhang, Zhaosong Lu, Jianhua Huang, and Jieping Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *The 30th International Conference on Machine Learning*, pages 37–45, 2013.

[3] Daniel Hernández-Lobato, José Miguel Hernández-Lobato, Thibault Helleputte, and Pierre Dupont. Expectation propagation for bayesian multi-task feature selection. In *Machine Learning and Knowledge Discovery in Databases*, pages 522–537. Springer, 2010.

[4] Daniel Hernández-Lobato, José Miguel Hernández-Lobato, and Alberto Suárez. Expectation propagation for microarray data classification. *Pattern recognition letters*, 31(12):1618–1626, 2010.

[5] José Miguel Hernández-Lobato. *Balancing flexibility and robustness in machine learning semi-parametric methods and sparse linear models*. PhD thesis, Ph. D. Thesis, Universidad Autó De Madrid, 2010.

[6] Shimon Kogan et al. Predicting risk from financial reports with regression. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics*, pages 272–280, 2009.

[7] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Ensemble nystrom method. In *Advances in Neural Information Processing Systems*, pages 1060–1068, 2009.

[8] Thomas P Minka. Deriving quadrature rules from Gaussian processes. Technical report, Statistics Department, Carnegie Mellon University, 2000.

[9] Toby J Mitchell and John J Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

[10] Jorge Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[11] Andreas Rosenwald et al. The use of molecular profiling to predict survival after chemotherapy for diffuse large-b-cell lymphoma. *New England Journal of Medicine*, 346(25):1937–1947, 2002.

[12] Todd E Scheetz et al. Regulation of gene expression in the mammalian eye and its relevance to eye disease. *Proceedings of the National Academy of Sciences*, 103(39):14429–14434, 2006.

[13] Michalis K Titsias and Miguel Lázaro-Gredilla. Spike and slab variational inference for multi-task and multiple kernel learning. In *NIPS*, volume 24, pages 2339–2347, 2011.

[14] Max A Woodbury. Inverting modified matrices. *Memorandum report*, 42:106, 1950.

[15] Tso-Jung Yen. A majorization–minimization approach to variable selection using spike and slab priors. *The Annals of Statistics*, 39(3):1748–1775, 2011.

[16] Shandian Zhe, Syed AZ Naqvi, Yifan Yang, and Yuan Qi. Joint network and node selection for pathway-based genomic data analysis. *Bioinformatics*, 2013.

5 Appendix

5.1 Optimization of the Model

To obtain the Laplace approximation, we need to compute the mode and the second-order derivative of the log posterior distribution at the mode. Due to the nonconvexity of the spike-and-slab model, we use two scalable nonconvex optimization methods as described in the following paragraphs.

5.1.1 L-BFGS Optimization of the Marginalized Model

For the first approach, denoted by FLAS, we marginalize out both $\mathbf{z}$ and $\mathbf{s}$, and the negative log probability of the marginalized model is

$$\mathcal{F}(\mathbf{w}) = L(\mathbf{w}) - \sum_{j=1}^{p} \log\Big(\frac{1}{2}\mathcal{N}(w_j|0, r_1) + \frac{1}{2}\mathcal{N}(w_j|0, r_0)\Big),$$

where $L(\mathbf{w})$ is the negative log likelihood for regression or classification. To minimize the negative log probability, we choose to use the L-BFGS method [10] because of its low computational and memory cost. As a quasi-Newton method, the L-BFGS method uses last M function/gradient pairs to approximate the inverse Hessian matrix of the parameters $\mathbf{w}$. Because M is set to be much smaller than p, often as small as 3-10, the computational cost is linear in p.

To use L-BFGS, we need to compute the gradient over $\mathbf{w}$:

$$\frac{\mathrm{d}\mathcal{F}}{\mathrm{d}w_j} = \frac{\mathrm{d}L(\mathbf{w})}{\mathrm{d}w_j} + \frac{r_0 + r_1 g(w_j)}{r_0 r_1 (1 + g(w_j))} w_j$$

where $g(w_j) = \sqrt{\frac{r_1}{r_0}} \exp(\frac{1}{2}(\frac{1}{r_1} - \frac{1}{r_0})w_j^2)$, $\frac{\mathrm{d}L(\mathbf{w})}{\mathrm{d}\mathbf{w}} = \tau \mathbf{X}^\top(\mathbf{X}\mathbf{w} - \mathbf{t})$ for linear regression, and $\frac{\mathrm{d}L(\mathbf{w})}{\mathrm{d}\mathbf{w}} = \sum_{n=1}^{N}\big(\frac{t_n}{1+\exp(\mathbf{x}_n^\top \mathbf{w})} - \frac{1-t_n}{1+\exp(-\mathbf{x}_n^\top \mathbf{w})}\big)\mathbf{w}$ for logistic regression.

Using the gradient in the L-BFGS method, we can compute the mode of w_j efficiently. Then we can approximate the posteriors of w_j, s_j and z_j, as explained in Section 3.2.

5.1.2 GIST Optimization of the Joint Model

For the second approach, denoted by FLAS*, we only marginalize out $\mathbf{z}$ (without marginalizing $\mathbf{z}$, the optimization requires discrete optimization strategies which are prohibitively expensive) and jointly optimize over both the weights $\mathbf{w}$ and the selection probability $\mathbf{s}$. From a Bayesian perspective, we prefer the first approach in the above because, by marginalizing out $\mathbf{s}$, it essentially takes all possible values of $\mathbf{s}$ into account. But the second approach can provide a more pronounced selective shrinkage effect than the first approach. The reason is that while the marginalized model always has a mixture of two penalizers, the joint model explicitly switches its regularization level—using either the spike component or the slab component—based on the optimal value of $\mathbf{s}$.

For the joint optimization, we use GIST, a recently developed nonconvex optimization method [2], which converges to a local optimum with a cost linear in p. Specifically, we minimize the negative log joint probability

$$\min_{\mathbf{w},\mathbf{s}} \mathcal{F}(\mathbf{w}, \mathbf{s}) = \min_{\mathbf{w}} L(\mathbf{w}) + \min_{\mathbf{s}} R(\mathbf{w}, \mathbf{s})$$

where $R(\mathbf{w}, \mathbf{s}) = \sum_{j=1}^{p} R_j(w_j, s_j)$, and

$$R_j(w_j, s_j) = -\log\big((1 - s_j)\mathcal{N}(w_j|0, r_0) + s_j \mathcal{N}(w_j|0, r_1)\big)$$

which is obtained by marginalizing out z_j in the model.

GIST iteratively optimizes $\mathbf{w}$ and $\mathbf{s}$ using the following two steps:

$$\begin{aligned}\mathbf{w}^{(k+1)} &= \operatorname*{argmin}_{\mathbf{w}}\ L(\mathbf{w}^{(k)}) + \langle \nabla L(\mathbf{w}^{(k)}), \mathbf{w} - \mathbf{w}^{(k)} \rangle \\ &\quad + \frac{\rho^k}{2}||\mathbf{w} - \mathbf{w}^{(k)}||^2 + \min_{\mathbf{s}} R(\mathbf{w}, \mathbf{s}), \\ \mathbf{s}^{(k+1)} &= \operatorname*{argmin}_{\mathbf{s}} R(\mathbf{w}^{(k+1)}, \mathbf{s})\end{aligned}$$

where $\mathbf{w}^{(k)}$ is the value of $\mathbf{w}$ at step k, and $\rho^{(k)}$ is the stepsize at step k.

Due to the form of $R(\mathbf{w}, \mathbf{s})$, the original minimization problem can be cast into p independent univariate proximal operator problems:

$$\begin{aligned} w_j^{(k+1)} &= \underset{w_j}{\operatorname{argmin}} \frac{1}{2}(w_j - u_j^{(k)})^2 + \frac{1}{\rho^{(k)}} \min_{s_j} R_j(w_j, s_j), \\ s_j^{(k+1)} &= \underset{s_j}{\operatorname{argmin}} R_j(w_j^{(k+1)}, s_j) \end{aligned}$$

where $j = 1, \ldots, p$, and $u_j^{(k)} = w_j^{(k)} - \nabla L(w_j^{(k)})/\rho^{(k)}$. To solve the univariate optimization problem, we calculate the value of w_j for the following two cases. For the first case $s_j = 1$, the function has its minimal at $w_j = b_1$ if $|b_1| > a$ and $w_j = \operatorname{sign}(b_1)a$ otherwise, where $b_1 = \frac{u^{(k)}}{1+1/(r_1\rho^{(k)})}$ and $a = \sqrt{\left(\frac{2r_0r_1}{r_1-r_0}\right)\log\sqrt{\frac{r_1}{r_0}}}$; for the second case $s_j = 0$, the function has its minimal at $w_j = b_0$ if $|b_0| < a$ and $w_j = \operatorname{sign}(b_0)a$ if otherwise, where $b_0 = \frac{u^{(k)}}{1+1/(r_0\rho^{(k)})}$. Then comparing the minimal values for these two cases and taking the smaller one, we can easily obtain the new $w_j^{(k+1)}$ and $s_j^{(k+1)}$. Note that, when $w = a$, s_j can be either 1 or 0, which gives the same function values.

5.2 Proof for approximation errors of inverse Hessian diagonal estimation

Theorem 1. Define $\Omega = \{\mathbf{A} \in \mathbb{R}^{p\times p} | \mathbf{A} \succ \mathbf{0}, \lambda_{min}(\mathbf{A}) \geq c, \lambda_{max}(\mathbf{A}) < \infty\}$. Assume Hessian $\mathbf{H}$ and approximate Hessian $\tilde{\mathbf{H}}$ both belong to $\mathbf{\Omega}$. Consider a function $f(\mathbf{A}) = \mathbf{e}_j^\top \mathbf{A}^{-1}\mathbf{e}_j, \mathbf{A} \in \mathbf{\Omega}$. Then, $\|\nabla f(\mathbf{A})\|_F \leq L$, $(1-\eta)\mathbf{H} + \eta\tilde{\mathbf{H}} \in \mathbf{\Omega}\ \forall\ \eta \in [0,1]$, and with high probability,

$$|\mathbf{H}^{-1}(j,j) - \tilde{\mathbf{H}}^{-1}(j,j)| \leq L \cdot D_0$$

where c is a small positive constant, and $L = p/c^2$. $\mathbf{e}_j$ is a standard basis vector with 1 in j-th coordinate and 0's elsewhere, and D_0 is the standard Nyström error bound based on Frobenius norm [7].

Proof. The derivative of $f(\mathbf{A})$ can be calculated by

$$\nabla f(\mathbf{A}) = -\mathbf{A}^{-1}\mathbf{e}_j\mathbf{e}_j^\top\mathbf{A}^{-1}.$$

Now since $\mathbf{A} \succ \mathbf{0}$ and consequently $\mathbf{A}^{-1} \succ \mathbf{0}$, $\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^p \lambda_i^2}$ and $\|\mathbf{A}^{-1}\|_F = \sqrt{\sum_{i=1}^p \frac{1}{\lambda_i^2}}$. Since $\lambda_{max}(\mathbf{A}) < \infty$ and $\lambda_{max}(\mathbf{A}^{-1}) < \infty$, $\|\mathbf{A}\|_F < \infty$ and $\|\mathbf{A}^{-1}\|_F < \infty$ which implies $\|\nabla f(\mathbf{A})\|_F < \infty$. Now $\|\nabla f(\mathbf{A})\|_F \leq \|\mathbf{A}^{-1}\|_F^2 = \sum_{i=1}^p \frac{1}{\lambda_i^2} \leq p(1/\lambda_{min}^2(\mathbf{A})) \leq p/c^2 = L$.

Since $f(\mathbf{A}) = \mathbf{e}_j^\top\mathbf{A}^{-1}\mathbf{e}_j$ for $\mathbf{A} \in \mathbf{\Omega}$, we can write $\mathbf{H}^{-1}(j,j) = f(\mathbf{H})$. Now, let us consider $|\mathbf{H}^{-1}(j,j) - \tilde{\mathbf{H}}^{-1}(j,j)| = |f(\tilde{\mathbf{H}}) - f(\mathbf{H})|$. We define $\mathbf{\Delta} = \tilde{\mathbf{H}} - \mathbf{H}$. Then $f(\tilde{\mathbf{H}}) = f(\mathbf{H} + \mathbf{\Delta})$. Now for any $0 \leq \eta \leq 1$, we have $(\mathbf{H} + \eta\mathbf{\Delta}) = ((1-\eta)\mathbf{H} + \eta\tilde{\mathbf{H}}) \succ \mathbf{0}$, and since, based on Weyl's inequality, $\lambda_{max}((1-\eta)\mathbf{H} + \eta\tilde{\mathbf{H}}) \leq (1-\eta)\lambda_{max}(\mathbf{H}) + \eta\lambda_{max}(\tilde{\mathbf{H}}) < \infty$, and $\lambda_{min}((1-\eta)\mathbf{H} + \eta\tilde{\mathbf{H}}) \geq (1-\eta)\lambda_{min}(\mathbf{H}) + \eta\lambda_{max}(\tilde{\mathbf{H}}) \geq (1-\eta)c + \eta c = c$, $\mathbf{H} + \eta\mathbf{\Delta} \in \mathbf{\Omega}$. This implies that $\|\nabla f(\mathbf{H} + \eta\mathbf{\Delta})\|_F \leq L$ for any $0 \leq \eta \leq 1$. Since $\frac{df(\mathbf{H}+\eta\mathbf{\Delta})}{d\eta} = \operatorname{tr}(\nabla f(\mathbf{H} + \eta\mathbf{\Delta})^\top \cdot \mathbf{\Delta})$, it is defined and bounded for all $\eta \in [0,1]$, hence it is continuous with respect to η. Therefore, by mean value theorem, there exist a number $t \in [0,1]$ such that:

$$f(\mathbf{H} + \mathbf{\Delta}) = f(\mathbf{H}) + \operatorname{tr}(\nabla f(\mathbf{H} + t\mathbf{\Delta})^\top \cdot \mathbf{\Delta}). \tag{4}$$

Thus by cauchy schwarz inequality,

$$|f(\mathbf{H} + \mathbf{\Delta}) - f(\mathbf{H})| \leq \|\nabla f(\mathbf{H} + t\mathbf{\Delta})\|_F \cdot \|\mathbf{\Delta}\|_F \leq L \cdot \|\mathbf{\Delta}\|_F. \tag{5}$$

Note that $\|\mathbf{\Delta}\|_F$ is the Nyström approximation error for $\mathbf{X}^\top\mathbf{X}$ and therefore we can readily apply the standard Nyström error bound D_0. The detailed form of D_0 can be found in [7] (Theorem 2).

Theorem 2. Define $\Omega = \{\mathbf{A} \in \mathbb{R}^{p\times p} | \mathbf{A} \succ \mathbf{0}, \lambda_{min}(\mathbf{A}) \geq c, \lambda_{max}(\mathbf{A}) < \infty\}$. Assume Hessian $\mathbf{H}$ and a set of approximate Hessians $\{\tilde{\mathbf{H}}_1, \ldots, \tilde{\mathbf{H}}_d\}$ all belong to $\mathbf{\Omega}$, then with high probability,

$$|\mathbf{H}^{-1}(j,j) - \frac{1}{d}\sum_{r=1}^d \tilde{\mathbf{H}}_r^{-1}(j,j)| \leq L \cdot D_1$$

where D_1 the error bound for ensemble Nyström based on Frobenius norm.

Proof. First, we have

$$
\begin{aligned}
|\mathbf{H}^{-1}(j,j) - \frac{1}{d}\sum_r \tilde{\mathbf{H}}_r^{-1}(j,j)| &= \frac{1}{d}|\sum_r f(\mathbf{H}) - f(\tilde{\mathbf{H}}_r)| \\
&\le \frac{1}{d}\sum_r |f(\mathbf{H}) - f(\tilde{\mathbf{H}}_r)|. \qquad (6)
\end{aligned}
$$

Following (5), we have

$$|\mathbf{H}^{-1}(j,j) - \frac{1}{d}\sum_r \tilde{\mathbf{H}}_r^{-1}(j,j)| \le L \cdot \frac{1}{d}\sum_{r=1}^{d} \|\mathbf{\Delta}_r\|_F \qquad (7)$$

where $\mathbf{\Delta}_r = \tilde{\mathbf{H}}_r - \mathbf{H}$. From the proof of Theorem 3 in [7], we can see that the error bound for ensemble Nyström is obtained by calculating the bound for $\frac{1}{d}\sum_{r=1}^{d} \|\mathbf{\Delta}_r\|_F$ (note that the error for ensemble Nyström is upper bounded by $\frac{1}{d}\sum_{r=1}^{d} \|\mathbf{\Delta}_r\|_F$). Therefore, we can directly apply the resulting error bound D_1 (the detailed form can be found in Theorem 3 [7]) to obtain

$$|\mathbf{H}^{-1}(j,j) - \frac{1}{d}\sum_r \tilde{\mathbf{H}}_r^{-1}(j,j)| \le L \cdot D_1.$$

Proposition 1. Assume that $\lambda_{max}(\mathbf{X}^\top\mathbf{X}) < \infty$, and $\forall j\ b \le v_j < \infty$, where b is a small positive constant. Then both Hessian $\mathbf{H}$ and any approximate Hessian $\tilde{\mathbf{H}}$ based on Nyström method belong to $\mathbf{\Omega}_0 = \{\mathbf{A} \in \mathbb{R}^{p\times p} | \mathbf{A} \succ \mathbf{0}, \lambda_{min}(\mathbf{A}) \ge b, \lambda_{max}(\mathbf{A}) < \infty\}$, and hence theorems 1 and 2 are satisfied with $L = p/b^2$

Proof. Since $\tau\mathbf{X}^\top\mathbf{X} \succeq \mathbf{0}$ and $\text{diag}(\mathbf{v}) \succ \mathbf{0}$, $H = \tau\mathbf{X}^\top\mathbf{X} + \text{diag}(\mathbf{v}) \succ \mathbf{0}$. Now by Weyl's inequality, $\lambda_{max}(\tau\mathbf{X}^\top\mathbf{X} + \text{diag}(\mathbf{v})) \le \lambda_{max}(\tau\mathbf{X}^\top\mathbf{X}) + \lambda_{max}(\text{diag}(\mathbf{v})) < \infty$, and $\lambda_{min}(\tau\mathbf{X}^\top\mathbf{X} + \text{diag}(\mathbf{v})) \ge \lambda_{min}(\tau\mathbf{X}^\top\mathbf{X}) + \lambda_{min}(\text{diag}(\mathbf{v})) \ge b$, $\lambda_{min}(\tau\mathbf{X}^\top\mathbf{X}) \ge 0$. Therefore, $\mathbf{H} \in \mathbf{\Omega}_0$.

Using theorem 3.5 in [1] we can conclude that $\tau\mathbf{X}^\top\mathbf{X}_k(\mathbf{X}_k^\top\mathbf{X}_k)^\dagger\mathbf{X}_k^\top\mathbf{X} \succeq \mathbf{0}$, therefore $\tilde{\mathbf{H}} \succ \mathbf{0}$. Based on theorem 3.8 in [1] $\lambda_{max}(\tau\mathbf{X}^\top\mathbf{X}_k(\mathbf{X}_k^\top\mathbf{X}_k)^\dagger\mathbf{X}_k^\top\mathbf{X}) \le \lambda_{max}(\tau\mathbf{X}^\top\mathbf{X}) < \infty$, and $\lambda_{min}(\tau\mathbf{X}^\top\mathbf{X}_k(\mathbf{X}_k^\top\mathbf{X}_k)^\dagger\mathbf{X}_k^\top\mathbf{X}) \ge 0$. Therefore, combined with Weyl's inequality, $\lambda_{max}(\tilde{\mathbf{H}}) < \infty$, and $\lambda_{min}(\tilde{\mathbf{H}}) \ge b$. Therefore, $\tilde{\mathbf{H}} \in \mathbf{\Omega}_0$.

5.3 Simulation Details

Data Generation for 1000 dimensions. For simulation, we first set the feature dimension p to 1000. We assumed 20 out of the 1000 features are relevant to the response. The irrelevant features were generated independently from the standard Gaussian distribution. The relevant features were generated from a multi-variate Gaussian distribution with a block diagonal covariance matrix. The covariance matrix consisted of two 10 by 10 sub-covariance matrices on the main diagonal. In each sub-covariance matrix, the diagonal elements were set to 1 and the off-diagonal elements were set to 0.81. Therefore, the 20 features were generated from two different groups. The weights $\mathbf{w}$ were set as

$$\mathbf{w} = [\underbrace{0, \ldots, 0}_{980}, \mathbf{v}, \mathbf{v}/\sqrt{10}, -\mathbf{v}, -\mathbf{v}/\sqrt{10}]$$

where $\mathbf{v} = [5, 5, 5, 5, 5]$. Given the sampled $\mathbf{X}$, for regression the response vector $\mathbf{t}$ were generated by $\mathbf{t} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon}$, where each ϵ_i was sampled independently from the standard Gaussian. For classification, we generated each response by $t_i = -1 \cdot \delta(\mathbf{x}_i^\top\mathbf{w} < 0) + 1 \cdot \delta(\mathbf{x}_i^\top\mathbf{w} > 0)$, where $\delta(x) = 1$ if $x = 1$ and 0 otherwise.

Data Generation for 100 dimensions. We chose a relatively small number of features in this simulation because we need to evaluate the accuracy of posterior inference results via comparing with Gibbs sampling, which converges slowly for very high dimensional problems. We made 20 out of the 100 features relevant to the response. The irrelevant features were generated independently from the standard Gaussian distribution. The relevant features were generated from a multi-variate Gaussian distribution with a block diagonal covariance matrix. The covariance matrix consisted of two 10 by 10 sub-covariance matrices on the main diagonal. In each sub-covariance matrix, the diagonal elements were set to 1 and the off-diagonal elements were set to 0.81. Therefore, the 20 features were generated from two different groups. The weights $\mathbf{w}$ were set as

$$\mathbf{w} = [\underbrace{0, \ldots, 0}_{80}, \underbrace{5+v, \ldots, 5+v}_{10}, \underbrace{-5+v, \ldots, -5+v}_{10}]$$

where v was generated from $\mathrm{Uniform}[-1, 1]$. Given the sampled $\mathbf{X}$, the response vector $\mathbf{t}$ was generated by $\mathbf{t} = \mathbf{X}\mathbf{w} + \epsilon$, where ϵ was sampled from the standard Gaussian for regression. For classification, we generated each response by $t_i = -1 \cdot \delta(\mathbf{x}_i^\top \mathbf{w} < 0) + 1 \cdot \delta(\mathbf{x}_i^\top \mathbf{w} > 0)$, where $\delta(x) = 1$ if $x = 1$ and 0 otherwise. We fixed the number of test samples to be 2000 and varied the number of training samples n from $\{10, 30, 50, 70\}$. For each n, we randomly generated 50 datasets and reported the average results.

5.4 Large Real Benchmark Data

We examined all the algorithms on 14 published large real datasets, including 8 classification datasets[1] and 6 regression datasets: Diffuse large B cell lymphoma (DLBCL) [11], GSE5680 [12], Yearprediction[2](Year), House-census[3](House), 10K corpus [6] and TIED[4]. Among the 14 datasets, the feature numbers are often at tens of thousands, while the sample sizes are often at hundreds or thousands. The information about these datasets is provided in Table 2.

We compared our algorithms, FLAS* and FLAS, with lasso, elastic net, capped L_1, ARD and EP-L. For the intractable EP and VB methods—the ones with computational cost $O(p^3)$ or $O(np^2)$, we first reduced the dimensionality of the datasets, by running FLAS and pruning all features with posterior mean selection probability less than 0.5, and then performed the comparison. We randomly split each dataset into two parts—10% samples for training and the rest for test—for 10 times and ran all the methods on each partition. In each run, we used 10-fold cross validation on the training data to tune the free parameters. Table 3 lists the average prediction accuracy and standard errors on the original datasets. As we can see, in all datasets, except for *Tied* in regression, and *classic* and *k1b* in classification, our algorithms, FLAS* or FLAS, obtain smaller root mean square errors or classification error rates. We also examined the average training time of all the methods and it turns out that our approach spent less or comparable time than the others. For example, the running time in seconds on *gse5680* and *reviews* are {lasso:2.03, elastic net:2.26, capped L_1:15.3, ARD:3.52, EP-L: 2.53, FLAS*:2.1, FLAS:0.3}, and {lasso:0.32, elastic net:0.29, capped L_1:2.3, ARD:26.7, EP-L:10.2, FLAS*:1.37, FLAS:0.10} respectively. Table 4 shows the prediction accuracy on the datasets with reduced dimensionality; that is for the comparison with intractable EP and VB algorithms. It turns out that our methods performed better or comparable than the intractable EP and VB; however, our methods have the scalability advantage in high dimensional problems.

Table 2: The size of the real large benchmark data.

(a) Regression

datasets	GSE5680	10k corpus	House-census	tied	Yearprediction	dlbcl
N	120	3308	22784	750	463715	240
p	31041	150358	138	999	90	752

(b) Classification

datasets	classic	hitech	k1b	reviews	sports	ng3sim	ohscal	la12
N	7094	2301	2340	4069	8580	2998	11162	6279
p	41681	10080	21819	18483	14870	15810	11465	31472

[1] `www.shi-zhong.com/software/docdata.zip` [2] `archive.ics.uci.edu/ml/datasets.html`

[3] `www.cs.toronto.edu/~delve/data/census-house/desc.html`

[4] `www.causality.inf.ethz.ch/repository.php`

Table 3: Root mean square error on regression datasets (the first 6 rows) and classification error rates (%) on large binary classification datasets (the last 8 rows). The results are averaged over 10 runs. Note that EP-L is designed for classification task only and thus does not have results on the regression datasets.

dataset	lasso	elast net	capped L_1	ARD	EP-L	FLAS*	FLAS
gse5680	0.107 ± 0.003	0.107 ± 0.003	0.107± 0.003	0.136 ± 0.005	NA	0.122 ± 0.002	**0.089± 0.002**
10k corpus	0.382 ± 0.002	0.382 ± 0.002	0.382 ± 0.002	0.382 ± 0.384	NA	0.383 ± 0.003	**0.372 ± 0.003**
tied	0.656 ± 0.013	0.627 ± 0.014	0.656 ± 0.013	**0.532 ± 0.017**	NA	0.719 ± 0.012	0.656 ± 0.013
House	1.576 ± 0.011	1.578 ± 0.017	1.587 ± 0.012	0.435 ±0.0006	NA	0.561 ± 0.015	**0.425 ±0.002**
Year	0.296 ± 0.009	0.293 ± 0.007	0.307 ± 0.004	0.306 ± 0.006	NA	0.248 ± 0.0005	**0.234 ± 0.0001**
dlbcl	1.76 ± 0.026	1.75 ± 0.027	1.75 ± 0.028	2.38 ± 0.063	NA	**1.60 ± 0.047**	**1.60± 0.047**
classic	6.69 ± 0.002	5.94 ± 0.002	**4.14 ± 0.002**	18.2 ± 0.002	8.94 ± 0.002	5.76 ± 0.002	4.20 ± 0.001
hitech	23.2 ± 0.005	21.4 ± 0.004	21.3 ± 0.003	28.5 ± 0.019	25.2 ± 0.001	**19.4 ± 0.003**	19.9 ± 0.003
k1b	5.44 ± 0.005	4.91 ± 0.004	**4.42 ± 0.004**	23.0 ± 0.013	7.94 ± 0.004	5.03 ± 0.005	4.74 ± 0.005
reviews	7.68 ± 0.003	6.47 ± 0.002	6.09 ± 0.001	35.4 ± 0.05	8.28 ± 0.002	5.93 ± 0.002	**5.54 ± 0.001**
sports	3.72 ± 0.001	3.15 ± 0.0008	3.25 ± 0.0009	24.1 ± 0.032	10.9 ± 0.008	2.78 ±0.001	**2.77 ± 0.007**
ng3sim	19.3 ± 0.005	16.2 ± 0.003	15.4 ± 0.003	21.3 ± 0.006	14.5 ± 0.002	13.7 ± 0.003	**13.6 ± 0.002**
ohscal	13.8 ± 0.001	13.7 ± 0.001	13.8 ± 0.001	37.3 ± 0.02	13.7 ±0.002	**11.9 ± 0.001**	13.1 ± 0.001
la12	13.6 ± 0.002	12.5 ± 0.002	12.2 ± 0.002	30.1 ± 0.025	13.2 ± 0.002	11.1 ± 0.002	**11.1 ± 0.001**

Table 4: Root mean square error on regression datasets (the first 3 rows) and classification error rates (%) on binary classification datasets (the last 4 rows) after dimension reduction. The results are averaged over 10 runs. FLAS is applied to reduce the data dimensions before the test.

dataset	EP	VB	FLAS*	FLAS
gse5680	**0.191 ± 0.008**	0.238 ± 0.009	0.195±0.008	0.197±0.008
10k corpus	0.382 ± 0.002	0.382 ± 0.002	**0.381 ± 0.002**	**0.381± 0.002**
tied	**0.5787± 0.013**	0.7983 ± 0.011	0.5868 ± 0.013	0.5877±0.013
hitech	24.31 ± 0.046	22.48 ± 0.006	19.64 ± 0.004	**19.38 ± 0.004**
k1b	9.37 ± 0.003	9.34 ± 0.002	**5.4 ± 0.006**	5.91 ± 0.006
reviews	10.2 ± 0.0004	10.1 ± 0.004	6.39 ± 0.002	**6.16 ± 0.002**
ng3sim	21.3 ± 0.006	24.37 ± 0.007	**14.2 ± 0.004**	14.65 ± 0.005