SUPPLEMENTARY MATERIAL

Efficient Computation of the Projection Matrix

Consider the computation of the ICPCA projection vectors $\mathbf{w}_k$, given by Equation (3), which we repeat here for convenience

$$\mathbf{w}_k = \left(\prod_{t=1}^{k-1} \mathbf{A}_t\right) \mathbf{v}_k = \left(\prod_{t=1}^{k-1} (\mathbf{I} - \mathbf{V}_t \mathbf{B}_t)\right) \mathbf{v}_k. \qquad (6)$$

Recall that $\mathbf{V}_t$ has all columns equal to $\mathbf{v}_t$ (where $\mathbf{v}_t$ is computed at step 2 of iteration t) and $\mathbf{B}_t = \text{diag}\,(\mathbf{b}_t)$ where $\mathbf{b}_t$ contains the coefficients $b_1, \ldots, b_D$ from iteration t (computed at step 3).

Consider now the first multiplication we need to compute in Equation (6). This can be rewritten as

$$\begin{aligned}
\mathbf{A}_{k-1}\mathbf{v}_k &= (\mathbf{I} - \mathbf{V}_{k-1}\mathbf{B}_{k-1})\mathbf{v}_k \\
&= \mathbf{v}_k - \mathbf{V}_{k-1}\mathbf{B}_{k-1}\mathbf{v}_k \\
&= \mathbf{v}_k - \mathbf{v}_{k-1}\mathbf{1}^\mathsf{T}\mathbf{B}_{k-1}\mathbf{v}_k \\
&= \mathbf{v}_k - \mathbf{v}_{k-1}\mathbf{b}_{k-1}^\mathsf{T}\mathbf{v}_k \qquad | \quad c_{k-1} := \mathbf{b}_{k-1}^\mathsf{T}\mathbf{v}_k \\
&= \mathbf{v}_k - c_{k-1}\mathbf{v}_{k-1}.
\end{aligned}$$

Thus in order to compute the multiplication by matrix $\mathbf{A}_{k-1}$, all we need to do is to take an inner product between two vectors and then subtract two vectors from each other which is very efficient. To compute the full product (6) we simply perform this operation in a loop, so that we first initialize $\mathbf{v}' = \mathbf{v}_k$ and repeat for $t = k-1, k-2, \ldots, 1$ the operation $\mathbf{v}' = \mathbf{A}_t\mathbf{v}'$. After the last multiplication the resulting vector will give us $\mathbf{w}_k$.

Datasets

The datasets we used for the comparisons are summarized in Table 1. All of them are classification problems and most datasets are available at http://featureselection.asu.edu/datasets.php. Although we are mostly interested in the "small n, large D" realm such as the microarray studies, we also wanted to consider how the different methods perform in other high-dimensional problems, such as in text classification where the features are typically word counts (Dexter, Basehock, PCMac). Two of the datasets (Arcene, Dexter) are taken from the NIPS 2003 feature selection challenge (http://clopinet.com/isabelle/Projects/NIPS2003/). These datasets are real problems but contain additional distractor features (probes) that have no predictive power.

Table 1: Summary of the real world classification datasets used for the experiments; dataset type, number of classes, dataset size n and number of features D. Type 'Gene' refers to gene expression data and 'Text' to text classification (features are word counts).

Dataset	Type	Classes	n	D
Ovarian	Gene	2	54	1536
Colon	Gene	2	62	2000
Prostate	Gene	2	102	5966
Leukemia	Gene	2	72	7129
Glioma	Gene	2	85	22283
Glioma-4c	Gene	4	50	4434
Lung	Gene	2	187	19993
Lung-5c	Gene	5	203	3312
Arcene	Other	2	200	10000
Dexter	Text	2	600	20000
Basehock	Text	2	1993	4862
PCMac	Text	2	1943	3289

Predictive Models and Priors

In Section 4.3, for the binary classification problems we used standard logistic regression model

$$p(y_i = 1 \,|\, \boldsymbol{\beta}) = \frac{1}{1 + \exp(-\boldsymbol{\beta}^\mathsf{T}\mathbf{x}_i)},$$

where $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_D)$ denotes the model parameters including the intercept β_0 (the notation assumes the first element of the predictor vector $\mathbf{x}$ is a constant $x_0 = 1$). For the intercept we used a diffuse prior $\beta_0 \sim \text{N}\big(0, 10^2\big)$ and for the regression coefficients $j = 1, \ldots, D$ the regularized horseshoe (Piironen and Vehtari, 2017c)

$$\begin{aligned}
\beta_j \,|\, \lambda_j, \tau, c &\sim \text{N}\Big(0, \tau^2\tilde{\lambda}_j^2\Big), \quad \tilde{\lambda}_j^2 = \frac{c^2\lambda_j^2}{c^2 + \tau^2\lambda_j^2}, \\
\lambda_j &\sim \text{C}^+(0, 1), \\
\tau &\sim \text{C}^+\big(0, \tau_0^2\big), \\
c^2 &\sim \text{Inv-Gamma}\big(\nu/2, \nu s^2/2\big).
\end{aligned}$$

This prior will shrink the coefficients of the irrelevant features heavily towards zero and softly regularize those that are far from zero. Following the recommendations of the aforementioned paper, we chose $\tau_0 = \frac{p_0}{D - p_0}\frac{2}{\sqrt{n}}$ with $p_0 = 1$ as our prior guess for the number of relevant features, and $\nu = 4$ and $s = 5$ as the parameters for the hyperprior on the regularizer c^2.

In the multiclass problems with H classes we used the multinomial softmax regression

$$p(y_i = \ell \,|\, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_H) = \frac{\exp(\boldsymbol{\beta}_\ell^\mathsf{T}\,\mathbf{x}_i)}{\sum_{h=1}^H \exp(\boldsymbol{\beta}_h^\mathsf{T}\,\mathbf{x}_i)}.$$

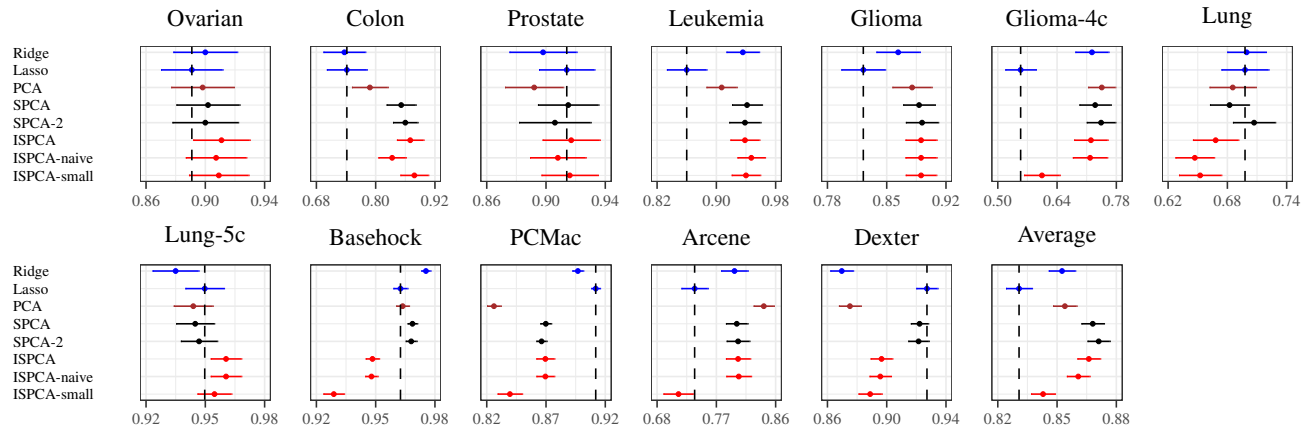

Figure 5: Classification accuracies on test data for the different methods on different datasets (larger is better). Horizontal bars denote the 95% intervals. The dashed vertical line denotes the performance estimate for the Lasso. The last plot denotes the average over all the datasets.

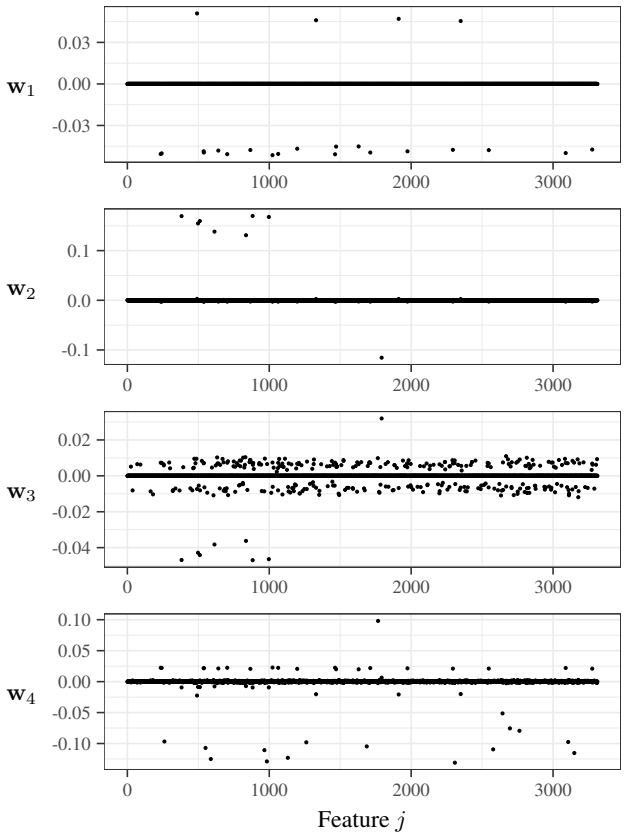

Figure 6: The first four ISPCs (columns of the projection matrix $\mathbf{W}$) for the Lung-5c cancer data ($n = 203, D = 3312$). The nonzero values indicate the genes that are characteristic for separating the corresponding class from the other classes (see Figure 3).

We used the same prior as in the binary case, so that each of the HD regression coefficients was given its own local scale parameter λ_j with one global scale τ. This allows the regression coefficient for some feature to be far from zero for some class h but be close to zero for the other classes, encoding the information that a feature can be relevant for separating one class from the others but irrelevant for separating the other classes from one another.

All the Bayesian models were fitted using Stan (Stan Development Team, 2017), running 4 chains, 2000 samples each, first halves discarded as warm-up. Ridge and Lasso solutions were computed with the default settings of the R-package `glmnet` (Friedman et al., 2010).

Extra Results

Figure 5 shows the classification accuracies for the different models considered in Section 4.3 and Table 2 typical computation times for some of the datasets.

Figure 6 shows the first ISPCs for Lung-5c dataset considered for data visualization in Section 4.2.

Table 2: Average computation time (in seconds) over five repeated runs for a representative set of datasets. For PCA, SPCA and ISPCA, the time contains both the dimension reduction and model fitting (the number in the parenthesis indicating the relative amount of time spent in the dimension reduction), and for Lasso the cross-validation of the regularization parameter.

Dataset	Classes	n	D	Computation time			
				PCA	SPCA	ISPCA	Lasso
Leukemia	2	72	7129	9.6 (2%)	8.3 (21%)	8.4 (24%)	1.0
Glioma	2	85	22283	14.6 (5%)	16.6 (33%)	14.5 (28%)	2.7
Lung-5c	5	203	3312	81.0 (1%)	82.2 (12%)	89.0 (19%)	5.2
PCMac	2	1943	3289	511.4 (2%)	303.3 (4%)	565 (22%)	18.9