The MM algorithm for sparse logistic PCA using the tight bound

A supplementary note to

"Sparse Logistic Principal Components Analysis for Binary Data"

Seokho Lee, Jianhua Z. Huang, Jianhua Hu

Abstract

We develop the MM algorithm for sparse logistic PCA using the tight majorizing bound. Comparison of the developed algorithm with the MM algorithm using the uniform bound in terms of computing time is also presented.

1 MM algorithm with the tight bound

In this section, we develop the MM algorithm using the tight majorizing bound (4.1) in the paper, which is

$$-\log\pi(x) \le -\log\pi(y) - \{1-\pi(y)\}(x-y) + \tfrac{2\pi(y)-1}{4y}(x-y)^2. \tag{1}$$

By substituting $q_{ij}\theta_{ij}$ and $q_{ij}\theta_{ij}^{(m)}$ into x and y, the right hand side of (1) becomes quadratic of the form $w_{ij}^{(m)}(\theta_{ij}-x_{ij}^{(m)})^2$ as with the uniform bound, but with different weight $w_{ij}^{(m)}$ and working variable $x_{ij}^{(m)}$ as

$$w_{ij}^{(m)} = \{2\pi(\theta_{ij}^{(m)})-1\}/4\theta_{ij}^{(m)} \quad \text{and} \quad x_{ij}^{(m)} = \theta_{ij}^{(m)}/\{2\pi(q_{ij}\theta_{ij}^{(m)})-1\}. \tag{2}$$

When $\theta_{ij}^{(m)}$'s are available, we update $\theta_{ij}^{(m+1)}$ by minimizing the sum of squares

$$\sum_{i=1}^{n}\sum_{j=1}^{d} w_{ij}^{(m)}(\theta_{ij}-x_{ij}^{(m)})^2$$

over θ_{ij}, or equivalently, over μ_j, $\mathbf{a}_i$ and $\mathbf{b}_j$. Then we obtain $w_{ij}^{(m+1)}$ and $x_{ij}^{(m+1)}$ based on $\theta_{ij}^{(m+1)}$ and use them for the next iteration step.

For the tight bound case, $x_{ij}^{(m)}$ and $w_{ij}^{(m)}$ are not well defined when $\theta_{ij}^{(m)}=0$ and will be replaced by the limit of the corresponding quantities when $\theta_{ij}^{(m)}\to 0$. To be specific, applying $\lim_{\theta\to 0}\{2\pi(\theta)-1\}/\theta = 1/2$, we define

$$\begin{aligned} x_{ij}^{(m)} &= \lim_{\theta_{ij}^{(m)}\to 0} \frac{\theta_{ij}^{(m)}}{2\pi(q_{ij}\theta_{ij}^{(m)})-1} = \frac{2}{q_{ij}}, \\ w_{ij}^{(m)} &= \lim_{\theta_{ij}^{(m)}\to 0} \frac{2\pi(\theta_{ij}^{(m)})-1}{4\theta_{ij}^{(m)}} = \frac{1}{8} \end{aligned} \tag{3}$$

when $\theta_{ij}^{(m)} = 0$. Now we use the same quadratic objective function defined in (4.9) of the paper, which is

$$\begin{aligned} & g(\boldsymbol{\mu}, \mathbf{A}, \mathbf{B} | \boldsymbol{\mu}^{(m)}, \mathbf{A}^{(m)}, \mathbf{B}^{(m)}) \\ & = \sum_{i=1}^{n} \sum_{j=1}^{d} \Big[w_{ij}^{(m)} \{ x_{ij}^{(m)} - (\mu_j + \mathbf{a}_i^T \mathbf{b}_j) \}^2 + \mathbf{b}_j^T \mathbf{D}_{\boldsymbol{\lambda},j}^{(m)} \mathbf{b}_j \Big], \end{aligned} \tag{4}$$

but with the general weights and working variables giving in (3). Theorem 4.1 also holds for such choice of weights and working variables (see the proof given in Appendix A.1 of the paper). We, again, alternate the minimization of (4) with respect to $\boldsymbol{\mu}$, $\mathbf{A}$ and $\mathbf{B}$. The three weighted least squares problems have closed-form solutions given as follows:

$$\begin{aligned} \hat{\mu}_j &= \arg\min_{\mu_j} \sum_{i=1}^{n} w_{ij} (x_{ij}^{\dagger} - \mu_j)^2 = \frac{\sum_{i=1}^{n} w_{ij} x_{ij}^{\dagger}}{\sum_{i=1}^{n} w_{ij}}, \\ \hat{\mathbf{a}}_i &= \arg\min_{\mathbf{a}_i} (\mathbf{x}_i^* - \mathbf{B}\mathbf{a}_i)^T \mathbf{W}_i (\mathbf{x}_i^* - \mathbf{B}\mathbf{a}_i) \\ &= (\mathbf{B}^T \mathbf{W}_i \mathbf{B})^{-1} \mathbf{B}^T \mathbf{W}_i \mathbf{x}_i^*, \\ \hat{\mathbf{b}}_j &= \arg\min_{\mathbf{b}_j} (\tilde{\mathbf{x}}_j^* - \mathbf{A}\mathbf{b}_j)^T \widetilde{\mathbf{W}}_j (\tilde{\mathbf{x}}_j^* - \mathbf{A}\mathbf{b}_j) + n \mathbf{b}_j^T \mathbf{D}_{\boldsymbol{\lambda},j} \mathbf{b}_j \\ &= (\mathbf{A}^T \widetilde{\mathbf{W}}_j \mathbf{A} + n \mathbf{D}_{\boldsymbol{\lambda},j})^{-1} \mathbf{A}^T \widetilde{\mathbf{W}}_j \tilde{\mathbf{x}}_j^*, \end{aligned} \tag{5}$$

where $\mathbf{W}_i = \text{diag}(\mathbf{w}_i)$ with $\mathbf{w}_i = (w_{i1}, \ldots, w_{id})^T$, $\widetilde{\mathbf{W}}_j = \text{diag}(\tilde{\mathbf{w}}_j)$ with $\tilde{\mathbf{w}}_j = (w_{1j}, \ldots, w_{nj})^T$ and other notations are the same as those previously defined. The MM algorithm will alternate solutions in (5) until convergence. When the constant weighting scheme $w_{ij} = 1/8$ is used, the solutions in (5) will reduce to the solutions described in Section 4 for the uniform bound case. The details are summarized in **Algorithm 2**. As we claimed before, Theorem 4.1 also holds in the tight bound case to guarantee the convergence of the algorithm.

Our experience with Algorithm 1 and Algorithm 2 is that use of the tight bound usually leads to less number of iterations of the algorithm but longer computing time because of the complexity involved in computing the bound.

2 Comparison of two algorithms

The main difference of the two algorithms is that different majorization bounds are used for the negative log likelihood function. To focus on this difference, we conducted logistic PCA without regularization using the two algorithms and compared them in terms of computing time and number of iterations needed for convergence. We generated 100 simulated binary datasets of size $(n, d) = (20, 50)$, $(20, 100)$, $(20, 200)$, $(50, 20)$, $(100, 20)$ and $(200, 20)$ using a rank-2 model with the same specification of $\boldsymbol{\mu}$, $\mathbf{A}$ and $\mathbf{B}$ as in Section 6 of the paper. We computed the mean and standard

Algorithm 2 *Sparse Logistic PCA Algorithm II*

```
1. Initialize with $\boldsymbol{\mu}^{(1)} = (\mu_1^{(1)}, \dots, \mu_d^{(1)})^T$, $\mathbf{A}^{(1)} = (\mathbf{a}_1^{(1)}, \dots, \mathbf{a}_n^{(1)})^T$ and $\mathbf{B}^{(1)} = (\mathbf{b}_1^{(1)}, \dots, \mathbf{b}_d^{(1)})^T$. Set $m = 1$.
2. Compute $x_{ij}^{(m)}$ and $w_{ij}^{(m)}$ using (2). Set $\mathbf{X}^{(m)} = (x_{ij}^{(m)})$.
3. Set $x_{ij}^{(m)\dagger} = x_{ij}^{(m)} - \mathbf{a}_i^{(m)T}\mathbf{b}_j^{(m)}$. Update $\boldsymbol{\mu}$ using $\boldsymbol{\mu}^{(m+1)} = (\mu_1^{(m+1)}, \dots, \mu_d^{(m+1)})^T$ using
       $\mu_j^{(m+1)} = \frac{\sum_{i=1}^n w_{ij}^{(m)} x_{ij}^{(m)\dagger}}{\sum_{i=1}^n w_{ij}^{(m)}}, \qquad j = 1, \cdots, d.$
4. Set $\mathbf{X}^{(m+1)*} = (x_{ij}^{(m+1)*}) = \mathbf{X}^{(m)} - \mathbf{1}_n \otimes \boldsymbol{\mu}^{(m+1)T}$.
5. Denote the $i$th row vector of $\mathbf{X}^{(m+1)*}$ as $\mathbf{x}_i^{(m+1)*}$. Set $\mathbf{W}_i^{(m)} = \mathrm{diag}(\mathbf{w}_i^{(m)})$ with $\mathbf{w}_i^{(m)} = (w_{i1}^{(m)}, \dots, w_{id}^{(m)})^T$. Update $\mathbf{A}$ by $\mathbf{A}^{(m+1)} = (\mathbf{a}_1^{(m+1)}, \dots, \mathbf{a}_n^{(m+1)})^T$ using
       $\mathbf{a}_i^{(m+1)} = \big(\mathbf{B}^{(m)T}\mathbf{W}_i^{(m)}\mathbf{B}^{(m)}\big)^{-1}\mathbf{B}^{(m)T}\mathbf{W}_i^{(m)}\mathbf{x}_i^{(m+1)*}, \qquad i = 1, \cdots, n.$
   Compute the QR decomposition $\mathbf{A}^{(m+1)} = \mathbf{QR}$ and replace $\mathbf{A}^{(m+1)}$ by $\mathbf{Q}$.
6. Denote the $j$th column vector of $\mathbf{X}^{(m+1)*}$ as $\tilde{\mathbf{x}}_j^{(m+1)*}$. Set $\widetilde{\mathbf{W}}_j^{(m)} = \mathrm{diag}(\tilde{\mathbf{w}}_j^{(m)})$ with $\tilde{\mathbf{w}}_j^{(m)} = (w_{1j}^{(m)}, \dots, w_{nj}^{(m)})^T$. Compute $\mathbf{D}_{\boldsymbol{\lambda},j}^{(m)}$ as in (4). Update $\mathbf{B}$ by $\mathbf{B}^{(m+1)} = (\mathbf{b}_1^{(m+1)}, \dots, \mathbf{b}_d^{(m+1)})^T$ using
       $\mathbf{b}_j^{(m+1)} = \big(\mathbf{A}^{(m+1)T}\widetilde{\mathbf{W}}_j^{(m)}\mathbf{A}^{(m+1)} + n\mathbf{D}_{\boldsymbol{\lambda},j}^{(m)}\big)^{-1}\mathbf{A}^{(m+1)T}\widetilde{\mathbf{W}}_j^{(m)}\tilde{\mathbf{x}}_j^{(m+1)*}, \quad j = 1, \cdots, d.$
7. Repeat steps 2 through 6 until convergence.
```

deviation of the computing time (in seconds) and the number of iterations from 100 simulated datasets. The results are depicted Figure 1 where in the top panels we fix the dimension $d = 20$ and vary the sample size n and in the bottom panels we fix the sample size $n = 20$ and vary the dimension d. It is not surprising that the computing time and the number of iterations needed for convergence increase with the sample size and the dimension. It is also very clear that using the tight bound (Algorithm 2) requires smaller numbers of iterations but using the uniform bound (Algorithm 1) has a big advantage in computing time.

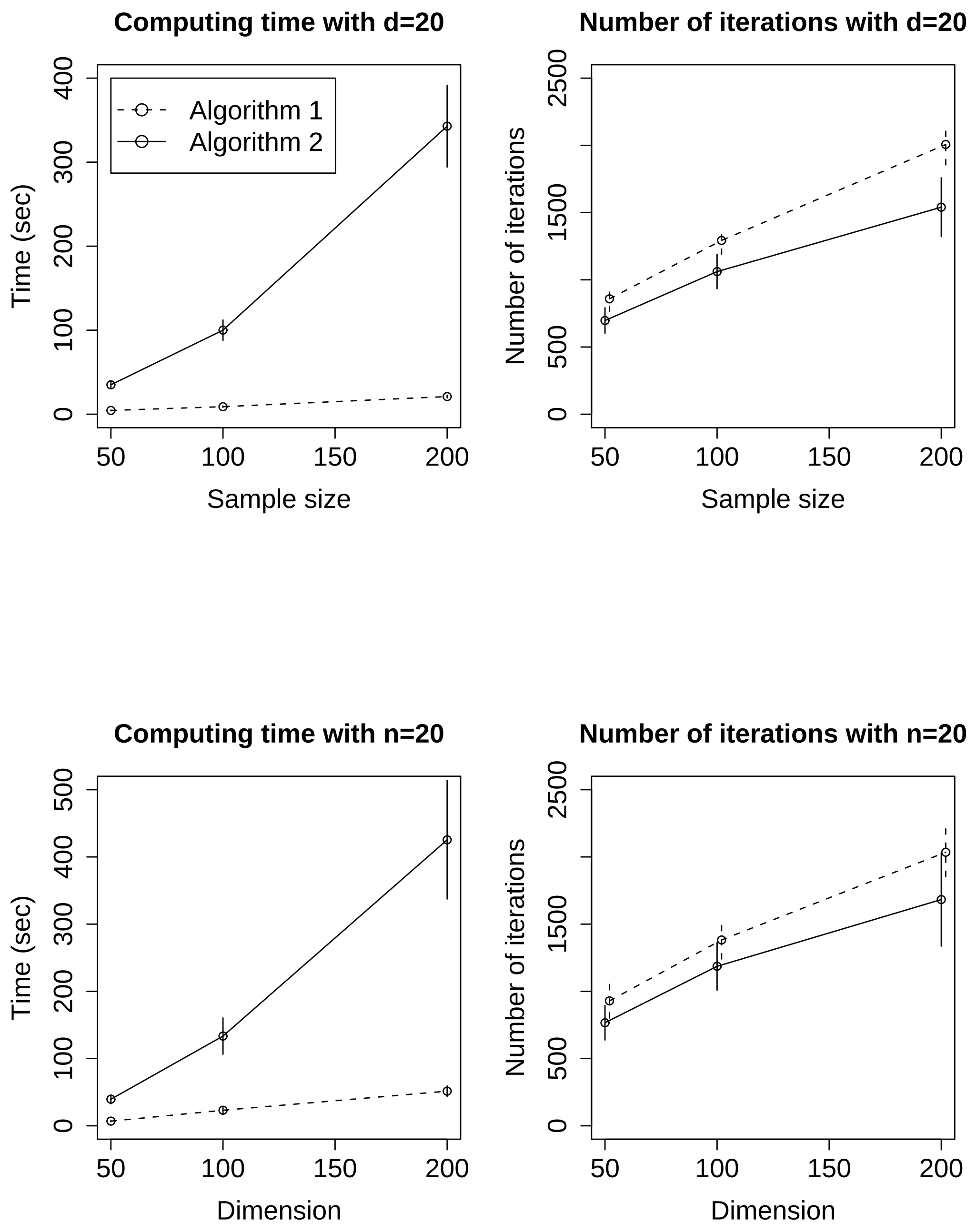

Figure 1: Comparison of two algorithms in terms of computing time and number of iterations. The means are shown as circles and the vertical bars stand for +/− one standard deviation from the mean.