Taylor & Francis
Taylor & Francis Group

Journal of Computational and Graphical Statistics

ISSN: 1061-8600 (Print) 1537-2715 (Online) Journal homepage: https://www.tandfonline.com/loi/ucgs20

A Matrix-Free Likelihood Method for Exploratory Factor Analysis of High-Dimensional Gaussian Data

Fan Dai, Somak Dutta & Ranjan Maitra

To cite this article: Fan Dai, Somak Dutta & Ranjan Maitra (2020) A Matrix-Free Likelihood Method for Exploratory Factor Analysis of High-Dimensional Gaussian Data, Journal of Computational and Graphical Statistics, 29:3, 675-680, DOI: 10.1080/10618600.2019.1704296

To link to this article: https://doi.org/10.1080/10618600.2019.1704296

JOURNAL OF COMPUTATIONAL AND GRAPHICAL STATISTICS
2020, VOL. 29, NO. 3, 675–680
https://doi.org/10.1080/10618600.2019.1704296

Taylor & Francis
Taylor & Francis Group

SHORT TECHNICAL NOTE

A Matrix-Free Likelihood Method for Exploratory Factor Analysis of High-Dimensional Gaussian Data

Fan Dai, Somak Dutta, and Ranjan Maitra

Department of Statistics, Iowa State University, Ames, IA

ABSTRACT

This technical note proposes a novel profile likelihood method for estimating the covariance parameters in exploratory factor analysis of high-dimensional Gaussian datasets with fewer observations than number of variables. An implicitly restarted Lanczos algorithm and a limited-memory quasi-Newton method are implemented to develop a matrix-free framework for likelihood maximization. Simulation results show that our method is substantially faster than the expectation-maximization solution without sacrificing accuracy. Our method is applied to fit factor models on data from suicide attempters, suicide ideators, and a control group. Supplementary materials for this article are available online.

ARTICLE HISTORY
Received July 2019
Revised November 2019

KEYWORDS
EM algorithm; fMRI; Lanczos algorithm; L-BFGS-B; Profile likelihood

1. Introduction

Factor analysis (Anderson 2003) is a multivariate statistical technique that characterizes dependence among variables using a small number of latent factors. Suppose that we have a sample $\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_n$ from the p-variate Gaussian distribution $\mathcal{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$. We assume that $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}^\top + \boldsymbol{\Psi}$, where $\boldsymbol{\Lambda}$ is a $p \times q$ matrix of rank q that describes the amount of variance shared among the p coordinates and $\boldsymbol{\Psi}$ is a diagonal matrix with positive diagonal entries representing the unique variance specific to each coordinate. Factor analysis of Gaussian data for $p < n$ was first formalized by Lawley (1940) with efficient maximum likelihood (ML) estimation methods proposed by Jöreskog (1967), Lawley and Maxwell (1962), Mardia, Kent, and Bibby (2006), Anderson (2003), and others. These methods however do not apply to datasets with $p > n$ that occur in applications such as the processing of microarray data (Sundberg and Feldmann 2016), sequencing data (Leek and Storey 2007; Leek 2014; Buettner et al. 2017), analyzing the transcription factor activity profiles of gene regulatory networks using massive gene expression datasets (Pournara and Wernisch 2007), portfolio analysis in stock returns (Ng, Yau, and Chan 2014), and others (Trendafilov and Unkel 2011). Necessary and sufficient conditions for the existence of MLE when $p > n$ have been obtained by Robertson and Symons (2007). In such cases, the available computer memory may be inadequate to store the sample covariance matrix $\mathbf{S}$ or to make multiple copies of the dataset needed during the computation.

The expectation-maximization (EM) approach of Rubin and Thayer (1982) can be applied to datasets with $p > n$ but is computationally slow. So, here we develop a profile likelihood method for high-dimensional Gaussian data. Our method allows us to compute the gradient of the profile likelihood function at negligible additional computational cost and to check first-order optimality, guaranteeing high accuracy. We develop a fast sophisticated computational framework called factor analysis of data (FAD) to compute ML estimates of $\boldsymbol{\Lambda}$ and $\boldsymbol{\Psi}$. Our framework is implemented in an R (R Core Team 2019) package called fad.

The remainder of this article is organized as follows. Section 2 describes the factor model for Gaussian data and an ML solution using the EM algorithm, and then proposes the profile likelihood and FAD. The performance of FAD relative to EM is evaluated in Section 3. Section 4 applies our methodology on a functional magnetic resonance imaging (fMRI) dataset related to suicidal behavior (Just et al. 2017). Section 5 concludes with some discussion. An online supplement, with sections, tables, and figures referenced here with the prefix "S," is available.

2. Methodology

2.1. Background and Preliminaries

Let $\mathbf{Y}$ be the $n \times p$ data matrix with $\mathbf{Y}_i$ as its ith row. Then, in the setup of Section 1, the ML method profiles out $\boldsymbol{\mu}$ using the sample mean vector and then maximizes the log-likelihood,

$$\ell(\boldsymbol{\Lambda}, \boldsymbol{\Psi}) = -\frac{n}{2}\{p\log(2\pi) + \log\det\boldsymbol{\Sigma} + \mathrm{Tr}\,\boldsymbol{\Sigma}^{-1}\mathbf{S}\}, \quad (1)$$

where $\bar{\mathbf{Y}} = \mathbf{Y}^\top\mathbf{1}/n$, $\mathbf{S} = (\mathbf{Y} - \mathbf{1}\bar{\mathbf{Y}}^\top)^\top(\mathbf{Y} - \mathbf{1}\bar{\mathbf{Y}}^\top)/n$, where $\mathbf{1}$ is the vector of 1's. The matrix $\mathbf{S}$ is almost surely singular and has rank n when $p > n$. The factor model (1) is not identifiable because the matrices $\boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}\mathbf{Q}$ give rise to the same likelihood for any orthogonal matrix $\mathbf{Q}$. So, additional constraints (see Anderson 2003; Mardia, Kent, and Bibby 2006, for more details) are imposed.

CONTACT Somak Dutta somakd@iastate.edu Department of Statistics, Iowa State University, Ames, IA.
Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/r/JCGS.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JCGS.

2.1.1. EM Algorithms for Parameter Estimation

The EM algorithm (Rubin and Thayer 1982; McLachlan and Krishnan 1996) exploits the structure of the factor covariance matrix by assuming q-variate standard normal latent factors and writing the factor model as $\mathbf{Y}_i = \boldsymbol{\mu} + \boldsymbol{\Lambda}\mathbf{Z}_i + \boldsymbol{\epsilon}_i$ where $\boldsymbol{\epsilon}_i$'s are iid $\mathcal{N}_p(\mathbf{0}, \boldsymbol{\Psi})$ and $\mathbf{Z}_i$'s are independent of $\boldsymbol{\epsilon}_i$'s. The EM algorithm is easily developed, with analytical expressions for both the expectation (E-step) and maximization (M-step) steps that can be speedily computed (see Section S1.1).

Although EM algorithms are guaranteed to increase the likelihood at each iteration and converge to a local maximum, they are well-known for their slow convergence. Further, these iterations run in a $(pq + p)$-dimensional space that can be slow for very large p. Accelerated variants (Liu and Rubin 2002; Varadhan and Roland 2008) show superior performance in low-dimensional problems but come with additional computational overhead that dominates the gain in rate of convergence in high dimensions. EM algorithms also compromise on numerical accuracy by not checking for first-order optimality to enhance speed. So, we next develop a fast and accurate method for exploratory factor analysis (EFA) that is applicable in high dimensions.

2.2. Profile Likelihood for Parameter Estimation

We start with the common and computationally useful identifiability restriction on $\boldsymbol{\Lambda}$ that constrains $\boldsymbol{\Gamma} = \boldsymbol{\Lambda}^\top\boldsymbol{\Psi}^{-1}\boldsymbol{\Lambda}$ to be diagonal with decreasing diagonal entries. This scale-invariant constraint is completely determined except for changes in sign in the columns of $\boldsymbol{\Lambda}$. Under this constraint, $\boldsymbol{\Lambda}$ can be profiled out for a given $\boldsymbol{\Psi}$ as described in the following

Lemma 1. Suppose that $\boldsymbol{\Psi}$ is a given p.d. diagonal matrix. Suppose that the q largest singular values of $\mathbf{W} = n^{-1/2}(\mathbf{Y} - \mathbf{1}\bar{\mathbf{Y}}^\top)\boldsymbol{\Psi}^{-1/2}$ are $\sqrt{\theta_1} \geq \sqrt{\theta_2} \geq \cdots \geq \sqrt{\theta_q}$ and the corresponding p-dimensional right-singular vectors are the columns of $\mathbf{V}_q$. Then the function $\boldsymbol{\Lambda} \mapsto \ell(\boldsymbol{\Lambda}, \boldsymbol{\Psi})$ is maximized at $\hat{\boldsymbol{\Lambda}} = \boldsymbol{\Psi}^{1/2}\mathbf{V}_q\boldsymbol{\Delta}$, where $\boldsymbol{\Delta}$ is a $q \times q$ diagonal matrix with ith diagonal entry as $[\max(\theta_i - 1, 0)]^{1/2}$. The profile log-likelihood equals,

$$\ell_p(\boldsymbol{\Psi}) = c - \frac{n}{2}\left\{\log\det\boldsymbol{\Psi} + \operatorname{Tr}\boldsymbol{\Psi}^{-1}\mathbf{S} + \sum_{i=1}^{q}(\log\theta_i - \theta_i + 1)\right\}, \tag{2}$$

where c is a constant that depends only on $\mathbf{Y}$, n, p and q but not on $\boldsymbol{\Psi}$. Furthermore, the gradient of $\ell_p(\boldsymbol{\Psi})$ is given by: $\nabla\ell_p(\boldsymbol{\Psi}) = -\frac{n}{2}\operatorname{diag}(\hat{\boldsymbol{\Lambda}}\hat{\boldsymbol{\Lambda}}^\top + \boldsymbol{\Psi} - \mathbf{S})$.

Proof. See Section S1.2. □

The profile log-likelihood $\ell_p(\boldsymbol{\Psi})$ in (2) depends on $\mathbf{Y}$ only through the q largest singular values of $\mathbf{W}$. So, to compute $\ell_p(\boldsymbol{\Psi})$ and $\nabla\ell_p(\boldsymbol{\Psi})$ we need to only compute the q largest singular values of $\mathbf{W}$ and the right singular vectors. For $q \ll \min(n, p)$, as is usually the case, these largest singular values and singular vectors can be computed very fast using Lanczos algorithm.

Further constraints on $\boldsymbol{\Psi}$ (e.g., $\boldsymbol{\Psi} = \sigma^2\mathbf{I}_p$, $\sigma^2 > 0$) can be easily incorporated. Also, $\nabla\ell_p(\boldsymbol{\Psi})$ is available in closed form that enables us to check first-order optimality and ensure high accuracy.

Finally, $\ell_p(\boldsymbol{\Psi})$ is expressed in terms of $\mathbf{S}$. However, ML estimators are scale-equivariant, so we can estimate $\boldsymbol{\Lambda}$ and $\boldsymbol{\Psi}$ using the correlation matrix and scale back to $\mathbf{S}$. A particular advantage of using the sample correlation matrix is that $\ell_p(\boldsymbol{\Psi})$ needs to be optimized over a fixed bounded rectangle $(0, 1)^p$ that does not depend on the data and is conceivably numerically robust.

2.3. Matrix-Free Computations

2.3.1. A Lanczos Algorithm for Calculating Partial Singular Values and Vectors

To compute the profile likelihood and its gradient, we need the q largest singular values and right singular vectors of $\mathbf{W}$. We use the Lanczos algorithm (Baglama and Reichel 2005; Dutta and Mondal 2014) with reorthogonalization and implicit restart. Suppose that $m = \max\{2q + 1, 20\}$ and that $\mathbf{f}_1 \in \mathbb{R}^n$ is any random vector with $\|\mathbf{f}_1\| = 1$. Let

- $\mathbf{g}_1 = \mathbf{W}^\top\mathbf{f}_1, \alpha_1 = \|\mathbf{g}_1\|$, and
- set $\mathbf{F}_1 = \mathbf{f}_1$ and $\mathbf{G}_1 = \mathbf{g}_1$.

Next, for $j = 1, \ldots, m$

- let $\mathbf{r}_j = \mathbf{W}\mathbf{g}_j - \alpha_j\mathbf{f}_j$,
- reorthogonalize $\mathbf{r}_j = \mathbf{r}_j - \mathbf{F}_j\mathbf{F}_j^\top\mathbf{r}_j$ and
- set $\beta_j = \|\mathbf{r}_j\|$.
- If $j < m$,
  - update $\mathbf{f}_{j+1} = \mathbf{r}_j/\beta_j$,
  - $\mathbf{F}_{j+1} = [\mathbf{F}_j, \mathbf{f}_{j+1}]$,
  - $\mathbf{g}_{j+1} = \mathbf{W}^\top\mathbf{f}_{j+1} - \beta_j\mathbf{g}_j$,
  - reorthogonalize $\mathbf{g}_{j+1} = \mathbf{g}_{j+1} - \mathbf{G}_j\mathbf{G}_j^\top\mathbf{g}_{j+1}$,
  - $\alpha_{j+1} = \|\mathbf{g}_{j+1}\|$,
  - $\mathbf{g}_{j+1} = \mathbf{g}_{j+1}/\alpha_{j+1}$, and
  - set $\mathbf{G}_{j+1} = [\mathbf{G}_j, \mathbf{g}_{j+1}]$.

Next, consider the $m \times m$ bidiagonal matrix $\mathbf{B}_m$ with diagonal entries $\alpha_1, \alpha_2, \ldots, \alpha_m$ with $(j, j+1)$th entry β_j for $j = 1, 2, \ldots, m-1$ and all other entries as 0. Now suppose that $h_1 \geq h_2 \geq \cdots \geq h_m$ are the singular values of $\mathbf{B}_m$ and that $\tilde{\mathbf{u}}_j$'s and $\tilde{\mathbf{v}}_j$'s are the corresponding right and left singular vectors, which can be computed via a Sturm sequencing algorithm (Wilkinson 1958). Also, let $\mathbf{u}_j = \mathbf{F}_m\tilde{\mathbf{u}}_j$ and $\mathbf{v}_j = \mathbf{G}_m\tilde{\mathbf{v}}_j$ $(1 \leq j \leq m)$. Then it can be shown that $\forall j$,

$$\mathbf{W}^\top\mathbf{u}_j = h_j\mathbf{v}_j \text{ and } \mathbf{W}\mathbf{v}_j = h_j\mathbf{u}_j + \tilde{v}_{j,m}\mathbf{r}_m,$$

where $\tilde{v}_{j,m}$ is the last entry of $\tilde{\mathbf{v}}_j$. Because $\|\mathbf{r}_m\| = \beta_m$ and h_1 is approximately the largest singular value of $\mathbf{W}$, the algorithm stops if $\beta_m|\tilde{v}_{j,m}| \leq h_1\delta$ holds for $j = 1, 2, \ldots, q$, where δ is some prespecified tolerance level, and $h_1, h_2, \ldots, h_q$ and $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_q$ are accurate approximations of the q largest singular values and corresponding right singular vectors of $\mathbf{W}$.

Convergence of the reorthogonalized Lanczos algorithm often suffers from numerical instability that slows down convergence. To resolve this instability, Baglama and Reichel (2005) proposed restarting the Lanczos algorithm, but instead of starting from scratch, they initialized with the first q singular

vectors. To that end, let $\mathbf{f}_{m+1} = \mathbf{r}_m/\beta_m$ and reset $\mathbf{F}_{q+1} = [\mathbf{u}_1, \ldots, \mathbf{u}_\mathbf{q}, \mathbf{f}_{m+1}]$. Then for $j = 1, 2, \ldots, q$,

- let $\rho_j = \beta_m \tilde{v}_{j,m}$, and reset:
- $\mathbf{r}_q = \mathbf{W}^\top \mathbf{f}_{m+1} - \sum_{j=1}^{q} \rho_j \mathbf{v}_j$,
- $\alpha_{q+1} = \|\mathbf{r}_q\|$,
- $\mathbf{g}_{q+1} = \mathbf{r}_q/\alpha_{q+1}$, and
- $\mathbf{G}_{q+1} = [\mathbf{v}_1, \ldots, \mathbf{v}_q, \mathbf{g}_{q+1}]$.
- Compute $\gamma = \mathbf{f}_{m+1}^\top \mathbf{W}\mathbf{g}_{q+1}$ and
- set $\mathbf{r}_{q+1} = \mathbf{W}\mathbf{g}_{q+1} - \gamma \mathbf{f}_{m+1}$.

Next for $j = 1, 2, \ldots, m - q - 1$, compute

- $\beta_{q+j} = \|\mathbf{r}_{q+j}\|$,
- $\mathbf{f}_{q+j+1} = \mathbf{r}_{q+j}/\beta_{q+j}$, $\mathbf{F}_{q+j+1} = [\mathbf{F}_{q+j},\ \mathbf{f}_{q+j+1}]$,
- $\mathbf{g}_{q+j+1} = (\mathbf{I} - \mathbf{G}_{q+j}\mathbf{G}_{q+j}^\top)\mathbf{W}^\top \mathbf{f}_{q+j+1}$,
- $\alpha_{q+j+1} = \|\mathbf{g}_{q+j+1}\|$,
- $\mathbf{g}_{q+j+1} = \mathbf{g}_{q+j+1}/\alpha_{q+j+1}$, and
- $\mathbf{r}_{q+j+1} = (\mathbf{I} - \mathbf{F}_{q+j+1}\mathbf{F}_{q+j+1}^\top)\mathbf{W}\mathbf{g}_{q+j+1}$.

This yields a matrix $\mathbf{B}_m$ with entries $b_{j,j} = h_j$ and $b_{j,q} = \rho_j$ for $j = 1, 2, \ldots, q$, and $b_{i,i} = \alpha_i$ for $q+1 \le i \le m$ and $b_{i,i+1} = \beta_i$ for $q + 1 \le i \le m - 1$, and all other entries 0. The matrix $\mathbf{B}_m$ is not bidiagonal but is still small-dimensioned matrix whose singular value decomposition can be calculated very fast. Convergence of the Lanczos algorithm can be checked as before.

The only way that $\mathbf{W}$ enters this algorithm is through matrix-vector products of the forms $\mathbf{W}\mathbf{g}$ and $\mathbf{W}^\top\mathbf{f}$, both of which can be computed without explicitly storing $\mathbf{W}$. Overall, this algorithm yields the q largest singular values and vectors in $O(qnp)$ computational cost using only $O(qp)$ additional memory, resulting in substantial gains over the traditional methods (Lawley and Maxwell 1962; Jöreskog 1967). These traditional methods require a full eigenvalue decomposition of $\mathbf{W}^\top\mathbf{W}$ that is of $O(p^3)$ computational complexity and requires $O(p^2)$ memory storage space.

Having described a scalable algorithm for computing $\ell_p(\mathbf{\Psi})$ and $\nabla\ell_p(\mathbf{\Psi})$, we detail a numerical algorithm for computing the ML estimators.

2.3.2. Numerical Optimization of the Profile Log-Likelihood

On the correlation scale, ψ_{ii}'s lie between 0 and 1. Under this box constraint, the factanal function in R and factoran function in MATLAB® employ the limited-memory Broyden–Fletcher–Goldfarb–Shanno quasi-Newton algorithm (Byrd et al. 1995) with box-constraints (L-BFGS-B) to obtain the ML estimator of $\mathbf{\Psi}$. However, in high dimensions, the advantages of the L-BFGS-B algorithm are particularly prominent. Because Newton methods require the search direction $-\mathbf{H}(\mathbf{\Psi})^{-1}\nabla\ell_p(\mathbf{\Psi})$, where $\mathbf{H}(\mathbf{\Psi})$ is the $p \times p$ Hessian matrix of $\ell_p(\mathbf{\Psi})$, they are computationally prohibitive in high dimensions in terms of storage and numerical complexity. The quasi-Newton BFGS replaces the computation of the exact search direction by an iterative approximation using the already computed values of $\ell_p(\mathbf{\Psi})$ and $\nabla\ell_p(\mathbf{\Psi})$. The limited-memory implementation, moreover, uses only the last few (typically <10) values of $\ell_p(\mathbf{\Psi})$ and $\nabla\ell_p(\mathbf{\Psi})$ instead of using all the past values. Overall, L-BFGS-B reduces the storage cost from $O(p^2)$ to $O(np)$ and the computational complexity from $O(p^3)$ to $O(np)$. Interested readers are referred to Byrd et al. (1995) and Zhu et al. (1994) for more details on the L-BFGS-B algorithm.

The L-BFGS-B algorithm requires both ℓ_p and $\nabla\ell_p$ to be computed at each iteration. Because $\nabla\ell_p$ is available as a byproduct while computing ℓ_p (see Sections 2.2 and 2.3.1), we modify the implementation to jointly compute both quantities with a single call to the Lanczos algorithm at each L-BFGS-B iteration. In comparison to R's default implementation (factanal) that separately calls ℓ_p and $\nabla\ell_p$ in its optimization routines, this tweak halves the computation time.

3. Performance Evaluations

3.1. Experimental Setup

The performance of FAD was compared to EM using 100 simulated datasets with true $q = 3$ or 5 and $(n, p) \in \{(100, 1000), (225, 3375), (400, 8000)\}$. For each setting, we simulated $\psi_{ii} \sim$ iid $\mathcal{U}(0.2, 0.8)$ and $\lambda_{ij} \sim$ iid $\mathcal{N}(0, 1)$ and set $\boldsymbol{\mu} = \mathbf{0}$. We also evaluated performance with $(n, p, q) \in \{(160, 24{,}547, 2), (180, 24{,}547, 2), (340, 24{,}547, 4)\}$ to match the settings of our application in Section 4: the true $\mathbf{\Psi}, \mathbf{\Lambda}, \boldsymbol{\mu}$ were set to be the ML estimates from that dataset.

For the EM algorithm, $\mathbf{\Lambda}$ was initialized as the first q principal components (PCs) of the scaled data matrix computed via the the Lanczos algorithm while $\mathbf{\Psi}$ was started at $\mathbf{I}_p - \text{diag}(\mathbf{\Lambda}\mathbf{\Lambda}^\top)$. FAD requires only $\mathbf{\Psi}$ to be initialized, which was done in the same way as the EM. We stopped FAD when the relative increase in $\ell_p(\mathbf{\Psi})$ was below $100\epsilon_0$ and $\|\nabla\ell_p\|_\infty < \sqrt{\epsilon}_0$ where ϵ_0 is the machine tolerance, which in our case was approximately 2.2×10^{-16}. The EM algorithm was terminated if the relative change in $\ell(\mathbf{\Lambda}, \mathbf{\Psi})$ was less than 10^{-6} and $\|\nabla\ell_p\|_\infty < \sqrt{\epsilon}_0$, or if the number of iterations reached 5000. Therefore, FAD and EM had comparable stopping criteria. For each simulated dataset, we fit models with $k = 1, 2, \ldots, 2q$ factors and chose the number of factors by the Bayesian information criterion (BIC): $-2\hat{\ell}_k + pk \log n$ (Schwarz 1978), where $\hat{\ell}_k$ is the maximum log-likelihood value with k factors. All experiments were done using R (R Core Team 2019) on a workstation with Intel E5-2640 v3 CPU clocked @2.60 GHz and 64GB RAM.

3.2. Results

Because BIC always correctly picked q, we evaluated model fit for each method in terms of $\ell(\hat{\mathbf{\Lambda}}, \hat{\mathbf{\Psi}})$, $d_{\hat{\mathbf{R}}} = \|\hat{\mathbf{R}} - \mathbf{R}\|_F/\|\mathbf{R}\|_F$ and $d_{\hat{\mathbf{\Gamma}}} = \|\hat{\mathbf{\Gamma}} - \mathbf{\Gamma}\|_F/\|\mathbf{\Gamma}\|_F$ where $\hat{\Gamma} = \hat{\mathbf{\Lambda}}^\top \hat{\mathbf{\Psi}}^{-1} \hat{\mathbf{\Lambda}}$ and $\mathbf{R}$ and $\hat{\mathbf{R}}$ are the correlation matrices corresponding to $\mathbf{\Sigma}$ and $\hat{\mathbf{\Sigma}} = \hat{\mathbf{\Lambda}}\hat{\mathbf{\Lambda}}^\top + \hat{\mathbf{\Psi}}$.

3.2.1. CPU Time

Figure 1 presents the relative speed of FAD to EM. Our compute times include the common initialization times. Specifically, for datasets of size $(n, p) \in \{(100, 1000), (225, 3375), (400, 8000)\}$, FAD was 10–70 times faster than EM, with maximum speedup at true q. However, EM did not converge within 5000 iterations in any of the overfitted models. In contrast, FAD always converged but it took longer than in other cases so the speedup is underestimated because of the censoring with EM. Also, the

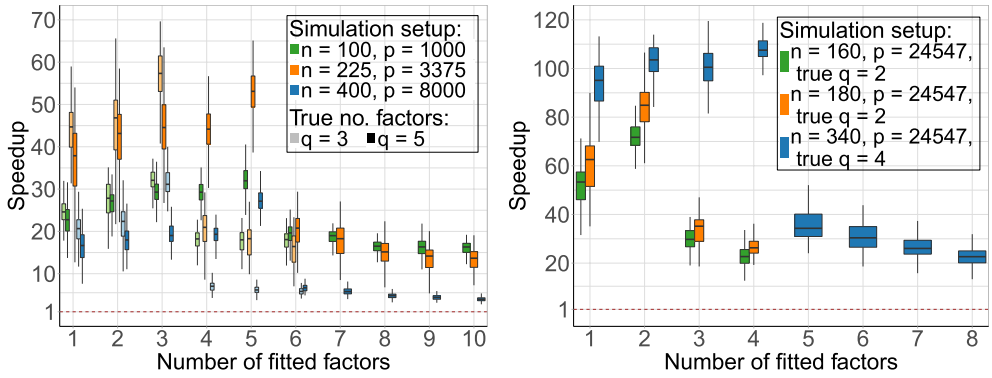

Figure 1. Relative speed of FAD to EM on (left) randomly simulated and (right) data-driven cases. Lighter ones correspond to true $q = 3$ and the darker ones correspond to true $q = 5$.

Table 1. CPU times (rounded to the nearest seconds) for FAD and EM applied to the suicide ideation study dataset.

	q	1	2	3	4	5	6	7	8	9	10
Attempters	FAD	3	3	4	5	5	6	6	7	9	9
	EM	146	173	207	198	229	236	228	250	239	254
Ideators	FAD	4	4	5	6	6	6	6	9	9	10
	EM	118	197	207	200	222	244	241	226	258	258
Controls	FAD	5	5	8	7	8	8	9	10	12	13
	EM	300	451	456	407	426	461	483	438	566	519

speedup is more pronounced (see Section S2.1) in the data-driven simulations where p is much larger.

3.2.2. Parameter Estimation and Model Fit

Under the best fitted models, FAD and EM yield identical values of $\ell_p(\hat{\mathbf{\Lambda}}, \hat{\mathbf{\Psi}})$, $\hat{\mathbf{\Psi}}$, $\hat{\Gamma}$, and $\hat{\mathbf{\Lambda}}\hat{\mathbf{\Lambda}}^\top$. Thus, the relative errors (see Figure S2) in estimating these parameters are also identical.

3.3. Additional Experiments in High-Noise Scenarios

We conclude this section by evaluating performance in situations where ostensibly, weak factors are hardly distinguished from high noise by SVD methods and where EM may be preferable (Owen and Wang 2015). We applied FAD and EM to the simulation setup of Owen and Wang (2015): Here, the uniquenesses were sampled from three inverse Gamma distributions with unit means and variances of 0, 1 and 10, and $(n, p) \in \{(200, 1000), (100, 5000)\}$. Figure S3 shows that our algorithm was substantially faster while having similar accuracy as EM.

4. Suicide Ideation Study

We applied EFA to data from Just et al. (2017) on an fMRI study conducted while 20 words connoting negative affects were shown to 9 suicide attempters, 8 suicide non-attempter ideators and 17 non-ideator control subjects. For each subject-word combination, Just et al. (2017) provided voxel-wise percent changes in activation relative to the baseline in $50 \times 61 \times 23$ image volumes. Restricting attention to the 24,547 in-brain voxels yields datasets for the attempters, ideators and controls of sizes $(n, p) \in \{(180, 24{,}547), (160, 24{,}547), (340, 24{,}547)\}$. We assumed each dataset to be a random sample from the multinormal distribution. Our interest was in determining if the variation in the percent relative change in activation for each subject type can be explained by a few latent factors and whether there are differences in these factors between the three groups of subjects.

For each dataset, we performed EFA with $q = 0, 1, 2, \ldots, 10$ factors and using both FAD and EM. Table 1 demonstrates the computational benefits of using FAD over EM. We also used BIC to decide on the optimal q (q_o) and obtained 2-factor models for both suicide attempters and ideators, and a 4-factor model for the control subjects. Figure 2 provides voxel-wise displays of the q_o factor loadings, obtained using the quartimax criterion (Costello and Osborne 2005), for each type of subject. All the factor loadings are nonnegligible only in voxels around the ventral attention network (VAN) which represents one of two sensory orienting systems that reorient attention toward notable stimuli and is closely related to involuntary actions (Vossel, Geng, and Fink 2014). However, there are differences between the factor loadings in each group and also among them. For instance, for the suicide attempters, each factor is a contrast between different areas of the VAN, but the contrasts themselves differ between the two factors. The first factor for the ideators is a weighted mean of the voxels while the second factor is a contrast of the values at the VAN voxels. For the controls, the first three factors are different contrasts of the values at different voxels while the fourth factor is more or less a mean of the values at these voxels. Further, the factor loadings in the control group are more attenuated than for either the suicide attempters or ideators. While a detailed analysis of our results is outside the purview of this article, we note that EFA has provided us with distinct factor loadings that potentially explains the variation in suicide attempters, non-attempter ideators, and controls. However, our analysis assumed that the image volumes are

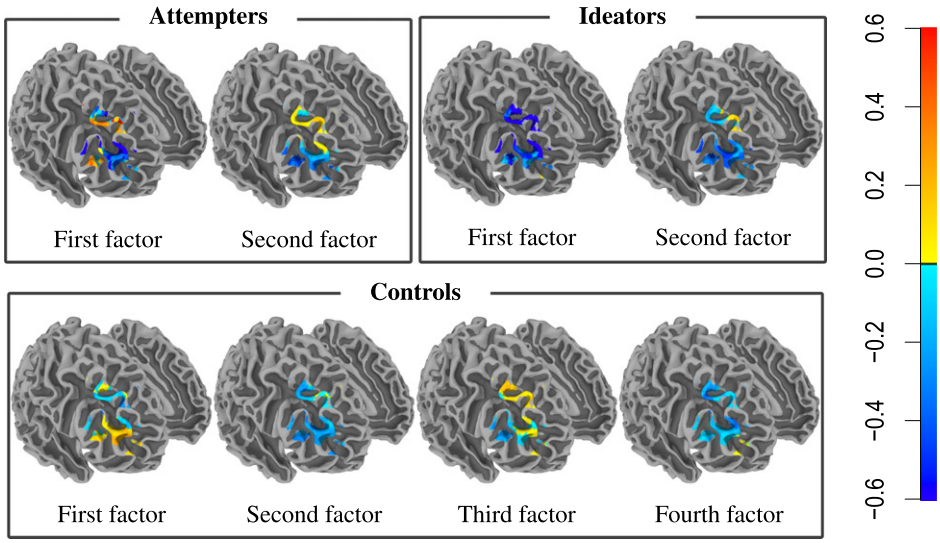

Figure 2. Loading values of fitted factors for suicide attempters, ideators, and controls.

independent and Gaussian: further approaches relaxing these assumptions may be appropriate.

5. Discussion

In this article, we propose a new ML-based EFA method called FAD using a sophisticated computational framework that achieves both high accuracy in parameter estimation and fast convergence via matrix-free algorithms. We implement a Lanczos method for computing partial singular values and vectors and a limited-memory quasi-Newton method for ML estimation. This implementation alleviates the computational limitations of current state-of-the-art algorithms and is capable of EFA for datasets with $p \gg n$. In our experiments, FAD always converged but EM struggled with overfitted models. Although not demonstrated in this article, FAD is also well-suited for distributed computing systems because it only uses the data matrix for computing matrix-vector products. FAD paves the way to develop fast methods for mixtures of factor analyzers and factor models for non-Gaussian data in high-dimensional clustering and classification problems.

Supplementary Materials

Supplement: Provides additional details on the algorithms and performance evaluations on the methods discussed in this article. Documented codes for reproducing the results are also included.

fad: An R package implementing the algorithm in this article available at CRAN.

Acknowledgments

We thank the editor and the associate editor for their helpful comments that improved the presentation of the article. An earlier version of this article won, the first author, a Student Poster Competition award at the 2nd Midwest Statistical Machine Learning Colloquium and a 2020 Student Paper Competition award from the ASA Section on Statistical Computing.

Funding

This research was supported in part by the United States Department of Agriculture (USDA) National Institute of Food and Agriculture (NIFA) Hatch project IOW03617. The research of the third author was also supported in part by the National Institute of Biomedical Imaging and Bioengineering (NIBIB) of the National Institutes of Health (NIH) under grant R21EB016212. The content of this article is however solely the responsibility of the authors and does not represent the official views of the NIBIB, the NIH, the NIFA, or the USDA.

References

Anderson, T. W. (2003), *An Introduction to Multivariate Statistical Analysis*, Wiley Series in Probability and Statistics, Hoboken, NJ: Wiley. [675]

Baglama, J., and Reichel, L. (2005), "Augmented Implicitly Restarted Lanczos Bidiagonalization Methods," *SIAM Journal on Scientific Computing*, 27, 19–42. [676]

Buettner, F., Pratanwanich, N., McCarthy, D. J., Marioni, J. C., and Stegle, O. (2017), "f-scLVM: Scalable and Versatile Factor Analysis for Single-Cell RNA-Seq," *Genome Biology*, 18, 212. [675]

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995), "A Limited Memory Algorithm for Bound Constrained Optimization," *SIAM Journal on Scientific Computing*, 16, 1190–1208. [677]

Costello, A. B., and Osborne, J. (2005), "Best Practices in Exploratory Factor Analysis: Four Recommendations for Getting the Most From Your Analysis," *Practical Assessment, Research & Evaluation*, 10, 1–9. [678]

Dutta, S., and Mondal, D. (2014), "An *h*-Likelihood Method for Spatial Mixed Linear Model Based on Intrinsic Autoregressions," *Journal of the Royal Statistical Society*, Series B, 77, 699–726. [676]

Jöreskog, K. G. (1967), "Some Contributions to Maximum Likelihood Factor Analysis," *Psychometrika*, 32, 443–482. [675,677]

Just, M. A., Pan, L. A., Cherkassky, V. L., McMakin, D. L., Cha, C. B., Nock, M. K., and Brent, D. P. (2017), "Machine Learning of Neural Representations of Suicide and Emotion Concepts Identifies Suicidal Youth," *Nature Human Behaviour*, 1, 911–919. [675,678]

Lawley, D. N. (1940), "The Estimation of Factor Loadings by the Method of Maximum Likelihood," *Proceedings of the Royal Society of Edinburgh*, 60, 64–82. [675]

Lawley, D. N., and Maxwell, A. E. (1962), "Factor Analysis as a Statistical Method," *Journal of the Royal Statistical Society*, Series D, 12, 209–229. [675,677]

Leek, J. T. (2014), "svaseq: Removing Batch Effects and Other Unwanted Noise From Sequencing Data," *Nucleic Acids Research*, 42, e161. [675]

Leek, J. T., and Storey, J. D. (2007), "Capturing Heterogeneity in Gene Expression Studies by Surrogate Variable Analysis," *PLoS Genetics*, 3, e161. [675]

Liu, C., and Rubin, D. B. (2002), "Maximum Likelihood Estimation of Factor Analysis Using the ECME Algorithm With Complete and Incomplete Data," *Statistica Sinica*, 8, 729–747. [676]

Mardia, K. V., Kent, J. T., and Bibby, J. M. (2006), *Multivariate Analysis*, Amsterdam: Elsevier. [675]

McLachlan, G., and Krishnan, T. (1996), *The EM Algorithm and Extensions*, Wiley Series in Probability and Statistics, Hoboken, NJ: Wiley. [676]

Ng, C. T., Yau, C. Y., and Chan, N. H. (2014), "Likelihood Inferences for High-Dimensional Factor Analysis of Time Series With Applications in Finance," *Journal of Computational and Graphical Statistics*, 24, 866–884. [675]

Owen, A. B., and Wang, J. (2015), "Bi-Cross-Validation for Factor Analysis," *Statistical Science*, 31, 119–139. [678]

Pournara, I., and Wernisch, L. (2007), "Factor Analysis for Gene Regulatory Networks and Transcription Factor Activity Profiles," *BMC Bioinformatics*, 8, 61. [675]

R Core Team (2019), *R: A Language and Environment for Statistical Computing*, Vienna, Austria: R Foundation for Statistical Computing, available at *https://www.R-project.org/*. [675,677]

Robertson, D., and Symons, J. (2007), "Maximum Likelihood Factor Analysis With Rank-Deficient Sample Covariance Matrices," *Journal of Multivariate Analysis*, 98, 813–828. [675]

Rubin, D. B., and Thayer, D. T. (1982), "EM Algorithms for ML Factor Analysis," *Psychometrika*, 47, 69–76. [675,676]

Schwarz, G. E. (1978), "Estimating the Dimension of a Model," *The Annals of Statistics*, 6, 461–464. [677]

Sundberg, R., and Feldmann, U. (2016), "Exploratory Factor Analysis—Parameter Estimation and Scores Prediction With High-Dimensional Data," *Journal of Multivariate Analysis*, 148, 49–59. [675]

Trendafilov, N. T., and Unkel, S. (2011), "Exploratory Factor Analysis of Data Matrices With More Variables Than Observations," *Journal of Computational and Graphical Statistics*, 20, 874–891. [675]

Varadhan, R., and Roland, C. (2008), "Simple and Globally Convergent Methods for Accelerating the Convergence of Any EM Algorithm," *Scandinavian Journal of Statistics*, 35, 335–353. [676]

Vossel, S., Geng, J. J., and Fink, G. R. (2014), "Dorsal and Ventral Attention Systems: Distinct Neural Circuits But Collaborative Roles," *The Neuroscientist*, 20, 150–159. [678]

Wilkinson, J. H. (1958), "The Calculation of the Eigenvectors of Codiagonal Matrices," *Computer Journal*, 1, 90–96. [676]

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. (1994), "Algorithm 778: L-BFGS-B: Fortran Subroutines for Large-Scale Bound-Constrained Optimization," *ACM Transactions on Mathematical Software*, 23, 550–560. [677]