Matrix Factor Analysis: From Least Squares to Iterative Projection

Abstract

In this article, we study large-dimensional matrix factor models and estimate the factor loading matrices and factor score matrix by minimizing square loss function. Interestingly, the resultant estimators coincide with the Projected Estimators (PE) in Yu et al. (2022), which was proposed from the perspective of simultaneous reduction of the dimensionality and the magnitudes of the idiosyncratic error matrix. In other word, we provide a least-square interpretation of the PE for matrix factor model, which parallels to the least-square interpretation of the PCA for the vector factor model. We derive the convergence rates of the theoretical minimizers under sub-Gaussian tails. Considering the robustness to the heavy tails of the idiosyncratic errors, we extend the least squares to minimizing the Huber loss function, which leads to a weighted iterative projection approach to compute and learn the parameters. We also derive the convergence rates of the theoretical minimizers of the Huber loss function under bounded $(2+\epsilon)$th moment of the idiosyncratic errors. We conduct extensive numerical studies to investigate the empirical performance of the proposed Huber estimators relative to the state-of-the-art ones. The Huber estimators perform robustly and much better than existing ones when the data are heavy-tailed, and as a result can be used as a safe replacement in practice. An application to a Fama-French financial portfolio dataset demonstrates the empirical advantage of the Huber estimator.

Keywords: Huber loss; Least squares; Matrix factor model; Projection estimation.

1 Introduction

Large-dimensional factor model has been a powerful tool of summarizing information from large datasets and draws growing attention in the era of "big-data" when more and more records of variables are available. The last two decades have seen many studies on large-dimensional approximate vector factor models, since the seminal work by Bai & Ng (2002) and Stock & Watson (2002), see for example, the representative works by Bai (2003),Onatski (2009), Ahn & Horenstein (2013), Fan et al. (2013), and Trapani (2018), Aït-Sahalia & Xiu (2017), Aït-Sahalia et al. (2020). These works all require the fourth moments (or even higher moments) of factors and idiosyncratic errors, and there are some works on relaxing the restrictive moment conditions, see the endeavors by Yu et al. (2019), Chen et al. (2021) and He et al. (2022).

In the last few years, large-dimensional matrix factor models have drawn much attention in view of the fact that observations are usually well structured to be an array, such as in macroeconomics and finance, see Chen & Fan (2021) for further examples of matrix observations. The seminal work is the one by Wang et al. (2019), who proposed the following formulation for matrix time series observations $\{\mathbf{X}_t, 1 \le t \le T\}$:

$$\underbrace{\mathbf{X}_t}_{p_1\times p_2} = \underbrace{\mathbf{R}_0}_{p_1\times k_1} \times \underbrace{\mathbf{F}_{0t}}_{k_1\times k_2} \times \underbrace{\mathbf{C}_0^\top}_{k_2\times p_2} + \underbrace{\mathbf{E}_t}_{p_1\times p_2}, \tag{1.1}$$

where $\mathbf{R}_0$ is the row factor loading matrix exploiting the variations of $\mathbf{X}_t$ across the rows, $\mathbf{C}_0$ is the $p_2 \times k_2$ column factor loading matrix reflecting the differences across the columns of $\mathbf{X}_t$, $\mathbf{F}_{0t}$ is the common factor matrix for all cells in $\mathbf{X}_t$ and $\mathbf{E}_t$ is the idiosyncratic component. Wang et al. (2019) proposed estimators of the factor loading matrices and numbers of the row and column factors based on an eigen-analysis of the auto-cross-covariance matrix. Chen & Fan (2021) proposed an α-PCA method for inference of (1.1), which conducts eigen-

analysis of a weighted average of the sample mean and the column (row) sample covariance matrix; Yu et al. (2022) proposed a projected estimation method which further improved the estimation efficiency of the factor loading matrices and the numbers of factors. He et al. (2021*b*) proposed a strong rule to determine whether there is a factor structure of matrix time series and a sequential procedure to determine the numbers of factors. Extensions and applications of the matrix factor model include the dynamic transport network in the international trade flows by Chen & Chen (2020), the constrained matrix factor model by Chen, Tsay & Chen (2020), the threshold matrix factor model in Liu & Chen (2019) and the online change point model by He et al. (2021*a*). Gao et al. (2021) proposed an interesting two-way factor model and provided solid theory. Jing et al. (2021) introduced mixture multilayer stochastic block model and proposed a tensor-based algorithm (TWIST) to reveal both global/local memberships of nodes, and memberships of layers for worldwide trading networks. There exist some recent works in the broader context of tensor factor model, see for example, Han et al. (2022), Chen, Xia, Cai & Fan (2020), Han et al. (2020), Lam (2021), Chen, Han, Li, Xiao, Yang & Yu (2022), Chang et al. (2021), Chen & Lam (2022), Chen, Yang & Zhang (2022). In general, the projection-based method would lead to more accurate estimation of the loading spaces, but would be computationally a bit demanding (Yu et al. 2022, Han et al. 2020).

A natural competitor of (1.1) is the group (vector) factor model (see e.g. Ando & Bai 2016), however, in such a model only one cross-section exists, which contains variables of the same nature well-grouped with known or unknown group membership. The model organizes the common factors into groups, and characterizes the interrelations within and between groups by such group factors. In contrast, the data $\mathbf{X}_t$ in matrix factor model (1.1) are genuinely matrix-valued, with two cross-sectional dimensions of different nature. The common components $\mathbf{R}_0\mathbf{F}_{0t}\mathbf{C}_0^\top$ in (1.1) reflect the interplay between the two different cross-sections. In the context of recommending systems illustrated in He et al. (2021*b*),

ratings are high whenever the purchasers' consumption preferences (rows in $\mathbf{R}_0$) match the underlying characteristics of items displayed online (rows in $\mathbf{C}_0$), thus the matrix factor model naturally captures the interactive effect between the row and column cross sections. In this context, matrix factor model (1.1) takes the intrinsic matrix nature of the data into account and shows advantage over models based on vectorising $\mathbf{X}_t$, where the presence of groups arises from artificially stacking the columns (rows) of matrix-valued data. Moreover, the matrix factor model in (1.1) has a more parsimonious factor structure, thus being statistically and computationally more efficient (Chen & Fan 2021, He et al. 2021*b*, Yu et al. 2022).

In the current work, we find the equivalence between the least square approach and the PE method by Yu et al. (2022). In other word, we provide the least squares interpretation of the PE for matrix factor model, which parallels to the least squares interpretation of traditional PCA for vector factor models. This finding provides another rationale for the PE method, which was initially proposed for reducing the magnitudes of the idiosyncratic error components and thereby increasing the signal-to-noise ratio. Motivated by the least squares formulation, we further propose a robust method for estimating large-dimensional matrix factor models, by substituting the least squares loss function with the Huber Loss function. The resultant estimators of factor loading matrices can be simply viewed as the eigenvectors of weighted sample covariance matrices of the projected data, which are easily obtained by an iterative algorithm. As far as we know, this is the first methodology work, with weighted iterative projection algorithms, on robust analysis of matrix factor models. As an illustration, we check the sensitivity of the α-PCA method by Chen & Fan (2021) and the Projected Estimation (PE) method (or least squares minimization method) in Yu et al. (2022) to the heavy-tailedness of the idiosyncratic errors with a synthetic dataset. We generate the idiosyncratic errors from matrix-variate normal, matrix-variate t_3 distributions that will be described in detail in Section 5. Figure 1 depicts the boxplots of

the estimation errors of the factor loading matrices over 1000 replications. It is clearly seen that the α-PCA and PE methods lead to much bigger biases and higher dispersions as the distribution tails become heavier. The proposed Robust Matrix Factor Analysis (RMFA) method still works quite satisfactorily when the idiosyncratic errors are from matrix-variate t_3 distribution.

To do factor analysis, the first step is to determine the number of factors. As for the Matrix Factor Model (MFM), both the row and column factor numbers should be determined in advance. Wang et al. (2019) proposed an estimation method based on ratios of consecutive eigenvalues of auto-covariance matrices; Chen & Fan (2021) proposed an α-PCA based eigenvalue-ratio method and Yu et al. (2022) further proposed a projection-based iterative eigenvalue-ratio method. All the methods above borrowed idea from Ahn & Horenstein (2013) and, to the best of our knowledge, He et al. (2021*b*) is the only work that determines the factor numbers of MFM from the perspective of sequential hypothesis testing and the authors also provide a strong rule to determine whether there is a row/column factor structure in the matrix time series. However, none methods mentioned above take the well-known heavy-tailedness of the data into account (see also Figure 2 in the real data section). In the current work, we also present a robust iterative eigenvalue-ratio method to estimate the numbers of factors following the Huber (Huber 1964) loss formulation.

The contributions of the present work lie in the following aspects. Firstly, we formulate the estimation of MFM from the least squares point of view and find that minimizing the square loss function under the identifiability conditions naturally leads to the iterative projection method invented in Yu et al. (2022) and thus enjoys the nice properties of the PE method. Secondly, we further propose a robust estimation method for MFM by taking the Huber loss in place of the least squares loss. We also propose an iterative weighted projection approach to solve the corresponding optimization problem, which in each iteration is doable by simple PCA solution. Thirdly, we propose an iterative algorithm for estimating

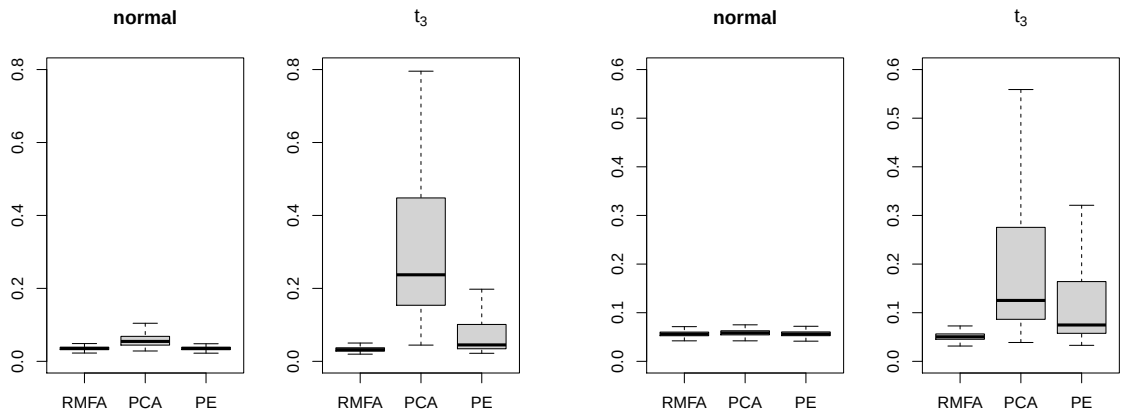

Figure 1: Boxplot of the distance between the estimated loading space and the true loading space by RMFA, PCA and PE methods under different distributions (matrix normal and matrix t_3). $p_1 = 20, p_2 = T = 50$. The left two plots depict the distances between the estimated loading space $\widehat{\mathbf{R}}$ and true loading space $\mathbf{R}_0$, and the right two plots depict the distances between the estimated loading space $\widehat{\mathbf{C}}$ and true loading space $\mathbf{C}_0$.

the row/column factor number robustly. Lastly, we derive the convergence rates of the theoretical minimizers, first time for the matrix factor model. To the best of our knowledge, this is the first work that derives the convergence rates of the theoretical minimizers of Huber loss function under bounded $(2+\epsilon)$th moment condition of the idiosyncratic errors for the matrix factor model, with the theoretical tool of self-normalized large/moderate deviations (Shao 1997, Jing et al. 2008).

The rest of the article goes as follows. In Section 2, we first formulate the estimation of factor loading matrices and factor score matrix by minimizing the square loss function and provide solutions to the optimization problem, from which we can see its equivalence to the projected estimation method. In Section 3, we investigate the theoretical minimizers of the least squares under mild conditions. In Section 4, we provide robust estimators by considering the Huber loss function and present detailed algorithm to obtain the minimizers. We also propose robust estimators for the pair of factor numbers. In Section 5, we conduct thorough numerical studies to illustrate the advantages of the RMFA method and the robust iterative eigenvalue-ratio method over the state-of-the-art methods. In Section 6,

we analyze a financial dataset to illustrate the empirical usefulness of the proposed methods. We discuss possible future research direction and conclude the article in Section 7. The proofs of the main theorems and additional details are collected in the supplementary materials.

Before ending this section, we introduce the notations used throughout the paper. For any vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_p)^\top \in \mathbb{R}^p$, let $\|\boldsymbol{\mu}\|_2 = (\sum_{i=1}^p \mu_i^2)^{1/2}$, $\|\boldsymbol{\mu}\|_\infty = \max_i |\mu_i|$. For a real number a, denote $[a]$ as the largest integer smaller than or equal to a, let $sgn(a) = 1$ if $a \geq 0$ and $sgn(a) = -1$ if $a < 0$. For a square matrix $\mathbf{A}$ whose jth diagonal element is denoted as A_{jj}, define $sgn(\mathbf{A})$ as a diagonal matrix whose jth diagonal element is equal to $sgn(A_{jj})$. Let $I(\cdot)$ be the indicator function and $\text{diag}(a_1, \ldots, a_p)$ be a $p \times p$ diagonal matrix, whose diagonal entries are $a_1 \ldots, a_p$. For a matrix $\mathbf{A}$, let A_{ij} (or $\text{A}_{i,j}$) be the (i,j)-th entry of $\mathbf{A}$, $\mathbf{A}^\top$ the transpose of $\mathbf{A}$, $\text{Tr}(\mathbf{A})$ the trace of $\mathbf{A}$, $\text{rank}(\mathbf{A})$ the rank of $\mathbf{A}$ and $\text{diag}(\mathbf{A})$ a vector composed of the diagonal elements of $\mathbf{A}$. Denote $\lambda_j(\mathbf{A})$ as the j-th largest eigenvalue of a nonnegative definitive matrix $\mathbf{A}$, and let $\|\mathbf{A}\|$ be the spectral norm of matrix $\mathbf{A}$ and $\|\mathbf{A}\|_F$ be the Frobenius norm of $\mathbf{A}$. For two series of random variables, X_n and Y_n, $X_n \asymp Y_n$ means $X_n = O_p(Y_n)$ and $Y_n = O_p(X_n)$. For two random variables (vectors) $\boldsymbol{X}$ and $\boldsymbol{Y}$, $\boldsymbol{X} \overset{d}{=} \boldsymbol{Y}$ means the distributions of $\boldsymbol{X}$ and $\boldsymbol{Y}$ are identical. The constants c, C_1, C_2 in different lines can be nonidentical.

2 Least Squares and Projected Estimation

In this section, we establish the equivalence between minimizing the square loss function under the identifiability conditions and the projection estimation procedure. We show that both angles coincide in the same iterative algorithm. We assume the matrix factor model (1.1) and let $\mathbf{S}_t = \mathbf{R}_0 \mathbf{F}_{0t} \mathbf{C}_0^\top$ be the common component matrix. The loading matrices $\mathbf{R}_0$ and $\mathbf{C}_0$ in model (1.1) are not separately identifiable. Without loss of generality,

for identifiability issue, we assume that $\mathbf{R}_0^\top \mathbf{R}_0/p_1 = \mathbf{I}_{k_1}$ and $\mathbf{C}_0^\top \mathbf{C}_0/p_2 = \mathbf{I}_{k_2}$, see also Chen & Fan (2021), Yu et al. (2022).

From (1.1), it is a natural idea to estimate $\mathbf{R}_0$ and $\mathbf{C}_0$ by minimizing the square loss under the identifiability condition:

$$\begin{aligned} \min_{\{\mathbf{R},\mathbf{C},\mathbf{F}_t\}} L_1(\mathbf{R},\mathbf{C},\mathbf{F}_t) &= \frac{1}{T}\textstyle\sum_{t=1}^T \|\mathbf{X}_t - \mathbf{R}\mathbf{F}_t\mathbf{C}^\top\|_F^2, \\ \text{s.t.} \frac{1}{p_1}\mathbf{R}^\top\mathbf{R} = \mathbf{I}_{k_1}, &\frac{1}{p_2}\mathbf{C}^\top\mathbf{C} = \mathbf{I}_{k_2}. \end{aligned} \tag{2.1}$$

The right hand side of (2.1) can be simplified as:

$$\frac{1}{T}\sum_{t=1}^T \|\mathbf{X}_t - \mathbf{R}\mathbf{F}_t\mathbf{C}^\top\|_F^2 = \frac{1}{T}\sum_{t=1}^T \left[\mathrm{Tr}(\mathbf{X_t}^\top\mathbf{X_t}) - 2\mathrm{Tr}(\mathbf{X_t}^\top\mathbf{R}\mathbf{F_t}\mathbf{C}^\top) + p_1p_2\mathrm{Tr}(\mathbf{F_t}^\top\mathbf{F_t})\right].$$

The optimization is non-convex in $\{\mathbf{R},\mathbf{C},\mathbf{F}_t\}$, but given the others, the loss function is convex over the remaining parameter. For instance, given $(\mathbf{R},\mathbf{F}_t)$, $L_1(\mathbf{R},\mathbf{C},\mathbf{F}_t)$ is convex over $\mathbf{C}$. Then we first assume that $(\mathbf{R},\mathbf{C})$ are given and solve the optimization problem on $\mathbf{F_t}$. For each t, taking $\partial L_1(\mathbf{R},\mathbf{C})/\partial\mathbf{F_t} = 0$, we obtain

$$\mathbf{F_t} = \frac{1}{p_1p_2}\mathbf{R}^\top\mathbf{X_t}\mathbf{C}.$$

Thus by substituting $\mathbf{F_t} = \mathbf{R}^\top\mathbf{X_t}\mathbf{C}/(p_1p_2)$ in the loss function $L_1(\mathbf{R},\mathbf{C},\mathbf{F}_t)$, we further have

$$\begin{aligned} \min_{\{\mathbf{R},\mathbf{C}\}} L_1(\mathbf{R},\mathbf{C}) &= \frac{1}{T}\textstyle\sum_{t=1}^T \left[\mathrm{Tr}(\mathbf{X_t}^\top\mathbf{X_t}) - \frac{1}{p_1p_2}\mathrm{Tr}(\mathbf{X_t}^\top\mathbf{R}\mathbf{R}^\top\mathbf{X_t}\mathbf{C}\mathbf{C}^\top)\right], \\ \text{s.t.} \frac{1}{p_1}\mathbf{R}^\top\mathbf{R} = \mathbf{I}_{k_1}, &\frac{1}{p_2}\mathbf{C}^\top\mathbf{C} = \mathbf{I}_{k_2}. \end{aligned} \tag{2.2}$$

The Lagrangian function is as follows:

$$\min_{\{\mathbf{R},\mathbf{C}\}} \mathcal{L}_1 = L_1(\mathbf{R},\mathbf{C}) + \mathrm{Tr}\left[\mathbf{\Theta}(\frac{1}{p_1}\mathbf{R}^\top\mathbf{R} - \mathbf{I}_{k_1})\right] + \mathrm{Tr}\left[\mathbf{\Lambda}(\frac{1}{p_2}\mathbf{C}^\top\mathbf{C} - \mathbf{I}_{k_2})\right],$$

where the Lagrangian multipliers $\mathbf{\Theta}$ and $\mathbf{\Lambda}$ are symmetric matrices. According to the KKT condition, let

$$\frac{\partial \mathcal{L}_1}{\partial \mathbf{R}} = -\frac{1}{T}\sum_{t=1}^{T}\frac{2}{p_1 p_2}\mathbf{X}_t\mathbf{C}\mathbf{C}^\top\mathbf{X}_t^\top\mathbf{R} + \frac{2}{p_1}\mathbf{R}\mathbf{\Theta} = 0,$$

$$\frac{\partial \mathcal{L}_1}{\partial \mathbf{C}} = -\frac{1}{T}\sum_{t=1}^{T}\frac{2}{p_1 p_2}\mathbf{X}_t^\top\mathbf{R}\mathbf{R}^\top\mathbf{X}_t\mathbf{C} + \frac{2}{p_2}\mathbf{C}\mathbf{\Lambda} = 0,$$

respectively, it holds that

$$\begin{cases} \left(\frac{1}{Tp_2}\sum_{t=1}^{T}\mathbf{X}_t\mathbf{C}\mathbf{C}^\top\mathbf{X}_t^\top\right)\mathbf{R} = \mathbf{R}\mathbf{\Theta}, \\ \left(\frac{1}{Tp_1}\sum_{t=1}^{T}\mathbf{X}_t^\top\mathbf{R}\mathbf{R}^\top\mathbf{X}_t\right)\mathbf{C} = \mathbf{C}\mathbf{\Lambda}, \end{cases} \quad \text{or} \quad \begin{cases} \mathbf{M}_c\mathbf{R} = \mathbf{R}\mathbf{\Theta}, \\ \mathbf{M}_r\mathbf{C} = \mathbf{C}\mathbf{\Lambda}, \end{cases} \tag{2.3}$$

where

$$\mathbf{M}_c = \frac{1}{Tp_2}\sum_{t=1}^{T}\mathbf{X}_t\mathbf{C}\mathbf{C}^\top\mathbf{X}_t^\top, \quad \mathbf{M}_r = \frac{1}{Tp_1}\sum_{t=1}^{T}\mathbf{X}_t^\top\mathbf{R}\mathbf{R}^\top\mathbf{X}_t.$$

We denote the first k_1 eigenvectors of $\mathbf{M}_c$ as $\{\boldsymbol{r}(1),\ldots,\boldsymbol{r}(k_1)\}$ and the corresponding eigenvalues as $\{\theta_1,\ldots,\theta_{k_1}\}$. Similarly, we denote the first k_2 eigenvectors of $\mathbf{M}_r$ as $\{\boldsymbol{c}(1),\ldots,\boldsymbol{c}(k_2)\}$ and the corresponding eigenvalues as $\{\lambda_1,\ldots,\lambda_{k_2}\}$. From (2.3), $\mathbf{R} = \sqrt{p_1}(\boldsymbol{r}(1),\ldots,\boldsymbol{r}(k_1))$, $\mathbf{C} = \sqrt{p_2}(\boldsymbol{c}(1),\ldots,\boldsymbol{c}(k_2))$, $\mathbf{\Theta} = \mathrm{diag}(\theta_1,\ldots,\theta_{k_1})$, $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1,\ldots,\lambda_{k_2})$ satisfy the KKT condition. However, $\mathbf{M}_c$ relies on the unknown column factor loading $\mathbf{C}$ while $\mathbf{M}_r$ relies on the unknown row factor loading $\mathbf{R}$, which motivates us to consider an iterative procedure to get the estimators, where in each iteration a simple PCA manipulation is enough. This turns out to be the same as our projected estimation procedure in Yu et al. (2022). We summarized the algorithm in Algorithm 1, where the initial estimators $\widehat{\mathbf{R}}^{(0)}$ and $\widehat{\mathbf{C}}^{(0)}$ could be selected as the α-PCA estimators.

In the following, let's briefly review the Projection Estimation (PE) method in Yu et al. (2022). First assume $\mathbf{C}$ is known and satisfies the orthogonal condition $\mathbf{C}^\top\mathbf{C}/p_2 = \mathbf{I}_{k_2}$. In

Algorithm 1 Least squares (Projection Estimation) for estimating matrix factor spaces

Input: Data matrices $\{\mathbf{X}_t\}_{t\le T}$, the pair of row and column factor numbers k_1 and k_2

Output: Factor loading matrices $\widetilde{\mathbf{R}}$ and $\widetilde{\mathbf{C}}$

1: obtain the initial estimators $\widehat{\mathbf{R}}^{(0)}$ and $\widehat{\mathbf{C}}^{(0)}$ by α-PCA with $\alpha = 0$;

2: project the data matrices to lower dimensions by defining: $\widehat{\mathbf{Y}}_t = \mathbf{X}_t\widehat{\mathbf{C}}^{(0)}$ and $\widehat{\mathbf{Z}}_t = \mathbf{X}_t^\top\widehat{\mathbf{R}}^{(0)}$;

3: given $\widehat{\mathbf{Y}}_t$ and $\widehat{\mathbf{Z}}_t$, define $\widehat{\mathbf{M}}_r = (Tp_1)^{-1}\sum_{t=1}^T \widehat{\mathbf{Y}}_t\widehat{\mathbf{Y}}_t^\top$ and $\widehat{\mathbf{M}}_c = (Tp_2)^{-1}\sum_{t=1}^T \widehat{\mathbf{Z}}_t\widehat{\mathbf{Z}}_t^\top$, and obtain the the leading k_1 eigenvectors of $\widehat{\mathbf{M}}_c$, denote as $\{\widehat{\boldsymbol{r}}(1),\ldots,\widehat{\boldsymbol{r}}(k_1)\}$ and the the leading k_2 eigenvectors of $\widehat{\mathbf{M}}_r$, denoted as $\{\widehat{\boldsymbol{c}}(1),\ldots,\widehat{\boldsymbol{c}}(k_2)\}$; Then update $\widehat{\mathbf{R}}$ and $\widehat{\mathbf{C}}$ as $\widehat{\mathbf{R}}^{(1)} = \sqrt{p_1}(\widehat{\boldsymbol{r}}(1),\ldots,\widehat{\boldsymbol{r}}(k_1))$ and $\widehat{\mathbf{C}}^{(1)} = \sqrt{p_2}(\widehat{\boldsymbol{c}}(1),\ldots,\widehat{\boldsymbol{c}}(k_2))$.

4: repeat step 2 and 3 until convergence and output the estimators from the last step and denote them as $\widetilde{\mathbf{R}}$ and $\widetilde{\mathbf{C}}$.

Yu et al. (2022), we projected the data matrix to a lower dimensional space by setting

$$\mathbf{Y}_t = \frac{1}{p_2}\mathbf{X}_t\mathbf{C} = \frac{1}{p_2}\mathbf{R}\mathbf{F}_t\mathbf{C}^\top\mathbf{C} + \frac{1}{p_2}\mathbf{E}_t\mathbf{C} := \mathbf{R}\mathbf{F}_t + \widetilde{\mathbf{E}}_t. \tag{2.4}$$

After projection, $\mathbf{Y}_t$ lies in a much lower column space than $\mathbf{X}_t$, and $\mathbf{F}_t$ and $\widetilde{\mathbf{E}}_t$ are deemed as factors and idiosyncratic errors for $\mathbf{Y}_t$, respectively. For the ith row of $\widetilde{\mathbf{E}}_t$, denoted as $\widetilde{\boldsymbol{e}}_{t,i\cdot}$, $\mathbb{E}\|\widetilde{\boldsymbol{e}}_{t,i\cdot}\|^2 \le cp_2^{-1}$ as long as the original errors $\{e_{t,ij}\}_{j=1}^{p_2}$ are weakly dependent column-wise. Thus $\mathbf{Y}_t$ can be viewed as satisfying a nearly noise-free factor model when p_2 is large. In other word, projecting the observation matrix onto the column factor space would not only simplify factor analysis for matrix sequences to that of a lower-dimensional tensor, but also reduce the magnitudes of the idiosyncratic error components, thereby increasing the signal-to-noise ratio. The next step of the PE method is to construct the row sample covariance matrix with $\{\mathbf{Y}_t\}$, which is exactly $\mathbf{M}_c$ up to a constant. Once $\mathbf{M}_c$ is constructed, the following steps of the PE method are exactly the same as stated in Algorithm 1. Surprisingly, we find that two totally different angles, least squares and projection approach coincide in matrix factor analysis.

3 Theoretical results

Though not computationally reachable, we establish the convergence rate of the theoretical minimizers of (2.1) under sub-Gaussian tails of idiosyncratic errors, instead of the two-step iteration estimators in Algorithm 1 by Yu et al. (2022). For optimization problem (2.1), let $\theta = (\mathbf{R}, \mathbf{F}_1, \cdots, \mathbf{F}_T, \mathbf{C})$ and $\theta_0 = (\mathbf{R}_0, \mathbf{F}_{01}, \cdots, \mathbf{F}_{0T}, \mathbf{C}_0)$ be the true parameters, where $\mathbf{R} = (\mathbf{r}_1, \cdots, \mathbf{r}_{p_1})^\top, \mathbf{C} = (\mathbf{c}_1, \cdots, \mathbf{c}_{p_2})^\top$. In this section, we present the asymptotic properties of the theoretical minimizers $\widehat{\theta}$, defined as

$$\widehat{\theta} = (\widehat{\mathbf{R}}, \widehat{\mathbf{F}}_1, \cdots, \widehat{\mathbf{F}}_T, \widehat{\mathbf{C}}) = \arg\min_{\theta \in \Theta} \frac{1}{T} \sum_{t=1}^{T} \|\mathbf{X}_t - \mathbf{R}\mathbf{F}_t\mathbf{C}^\top\|_F^2,$$

where

$$\begin{aligned} \Theta \;=\; & \Big\{\theta : \mathbf{R} \in \mathcal{A} \subset \mathbb{R}^{p_1 \times k_1}, \mathbf{C} \in \mathcal{B} \subset \mathbb{R}^{p_2 \times k_2}, \mathbf{F}_t \in \mathcal{F} \subset \mathbb{R}^{k_1 \times k_2} \text{ for all } t, \\ & \frac{1}{p_1}\mathbf{R}^\top\mathbf{R} = \mathbf{I}_{k_1}, \frac{1}{p_2}\mathbf{C}^\top\mathbf{C} = \mathbf{I}_{k_2}\Big\}. \end{aligned}$$

To obtain the theoretical properties of $\widehat{\theta}$, we assume that the following three assumptions hold.

Assumption 1: $\mathcal{A}, \mathcal{B}$ and $\mathcal{F}$ are compact sets and $\theta_0 \in \Theta$. $\|\mathbf{R}_0\|_F/\sqrt{p_1}$ and $\|\mathbf{C}_0\|_F/\sqrt{p_2}$ are bounded. The factor matrix satisfies

$$\frac{1}{T}\sum_{t=1}^{T} \mathbf{F}_{0t}\mathbf{F}_{0t}^\top \to \mathbf{\Sigma}_1 \text{ and } \frac{1}{T}\sum_{t=1}^{T} \mathbf{F}_{0t}^\top\mathbf{F}_{0t} \to \mathbf{\Sigma}_2, \quad \text{as} \quad T \to \infty,$$

where $\mathbf{\Sigma}_i, i = 1, 2$ is a $k_i \times k_i$ positive diagonal matrix with bounded distinctive eigenvalues.

Assumption 2: Given $\{\mathbf{F}_{0t}, t = 1, \cdots, T\}, \{e_{ij,t}\}$ are independent across i, j and t.

Assumption 3: For some $K > 0$, $E(|e_{ij,t}|^p | \{\mathbf{F}_{0t}\}) \le K^p p^{p/2}$ for all $p > 1$, and $E(e_{ij,t} | \{\mathbf{F}_{0t}\}) = 0$.

Assumption 1 is standard in large factor models, $\mathbf{\Sigma}_1, \mathbf{\Sigma}_2$ are assumed to be diagonal matrices with distinctive diagonal elements for further identifiability issue, and we refer, for example, to Bai & Ng (2013), Chen & Fan (2021) and He et al. (2021*b*). Assumption 2 assumes that the error terms are *i.i.d* conditional on the matrix factors, but they may not be *i.i.d* unconditionally. Although this condition seems to be restrictive, it is an exchange for simplicity, see for example Chen et al. (2021). Assumption 3 assumes that the idiosyncratic errors have sub-Gaussian tails given $\{\mathbf{F}_{0t}\}$, i.e., there are positive constants C, ν such that for every $t > 0$, $P(|e_{ij,t}| > t|\{\mathbf{F}_{0t}\}) \leq C\exp(-\nu t^2)$. The sub-Gaussian condition is for simplicity of theoretical proof, which is common in high-dimensional statistical inference, see for example Wainwright (2019).

The following theorem presents the asymptotic property of $\widehat{\mathbf{R}}, \widehat{\mathbf{C}}$ and $\widehat{\mathbf{F}}_t, t = 1, \cdots, T$ in terms of both Frobenius norm and spectral norm.

Theorem 3.1. Let $\widehat{\mathbf{S}}_1 = sgn\Big(\frac{1}{T}\sum_{t=1}^{T}(\widehat{\mathbf{F}}_t\mathbf{F}_{0t}^\top)\Big)$ and $\widehat{\mathbf{S}}_2 = sgn\Big(\frac{1}{T}\sum_{t=1}^{T}(\widehat{\mathbf{F}}_t^\top\mathbf{F}_{0t})\Big)$. Then, under Assumptions 1-3,

$$\frac{1}{p_1}\|\widehat{\mathbf{R}} - \mathbf{R}_0\widehat{\mathbf{S}}_1\|_F^2 + \frac{1}{p_2}\|\widehat{\mathbf{C}} - \mathbf{C}_0\widehat{\mathbf{S}}_2\|_F^2 + \frac{1}{T^2}\|\sum_{t=1}^{T}(\widehat{\mathbf{F}}_t - \widehat{\mathbf{S}}_1\mathbf{F}_{0t}\widehat{\mathbf{S}}_2)\|_F^2 = O_p\left(\frac{1}{L}\right),$$

where $L = \min\{p_1p_2, Tp_2, Tp_1\}$. In particular,

$$\frac{1}{p_1}\|\widehat{\mathbf{R}} - \mathbf{R}_0\widehat{\mathbf{S}}_1\|^2 + \frac{1}{p_2}\|\widehat{\mathbf{C}} - \mathbf{C}_0\widehat{\mathbf{S}}_2\|^2 + \frac{1}{T^2}\|\sum_{t=1}^{T}(\widehat{\mathbf{F}}_t - \widehat{\mathbf{S}}_1\mathbf{F}_{0t}\widehat{\mathbf{S}}_2)\|^2 = O_p\left(\frac{1}{L}\right).$$

In Theorem 3.1, the sign matrices $\widehat{\mathbf{S}}_1, \widehat{\mathbf{S}}_2$ appear due to the intrinsic sign indeterminacy of factors and loadings, i.e., the factor structure remains the same when a factor and its loading are both multiplied by -1. Theorem 3.1 shows that the convergence rate of the least square estimator of θ is no slower than that of the α-PCA estimator in Chen & Fan (2021). In particular, when $p_1 \ll Tp_2$, the theoretical minimizer converges at rate $O_p((Tp_1)^{-1} +$

$(p_1p_2)^{-1}$), much faster than $O_p(p_1^{-1})$ of the α-PCA estimator. In terms of the summed errors of $\widehat{\theta}$, i.e.,

$$\frac{1}{p_1}\|\widehat{\mathbf{R}} - \mathbf{R}_0\widehat{\mathbf{S}}_1\|_F^2 + \frac{1}{p_2}\|\widehat{\mathbf{C}} - \mathbf{C}_0\widehat{\mathbf{S}}_2\|_F^2 + \frac{1}{T^2}\|\sum_{t=1}^{T}(\widehat{\mathbf{F}}_t - \widehat{\mathbf{S}}_1\mathbf{F}_{0t}\widehat{\mathbf{S}}_2)\|_F^2,$$

the least square estimate in Theorem 3.1 (also in Theorem 4.1) has the same convergence rate as the PE estimate given in Yu et al. (2022): both are $O_p(L^{-1})$. However, the convergence rate of the least square estimator derived in Theorem 3.1 is generally slower than that of the PE estimator in Yu et al. (2022), except that one of $((Tp_1)^{-1}, (Tp_2)^{-1}, (p_1p_2)^{-1})$ dominates the others. The reason is that $(\mathbf{R}_0, \mathbf{C}_0)$ are estimated jointly with $\{\mathbf{F}_{0t}, t = 1, \ldots, T\}$ as the theoretical minimizers of the empirical square loss in the current work, while the two-step PE approach estimates $\mathbf{R}_0$ or $\mathbf{C}_0$ without knowing $\{\mathbf{F}_{0t}\}$ first. For example, the PE estimate of $\mathbf{R}_0$ converges at rate $O_p((Tp_2)^{-1} + (Tp_1)^{-2} + (p_1p_2)^{-2})$. When $(Tp_2)^{-1}$ dominates $(Tp_1)^{-1}$ and $(p_1p_2)^{-1}$, the PE estimate converges at the same rate as that derived in Theorem 3.1. The reason is that the two-step estimate of $\mathbf{R}_0$ of the PE method depends on $\mathbf{C}_0$ in only one step and does not rely on $\{\mathbf{F}_{0t}, t = 1, \ldots, T\}$, and thus can be estimated individually, while the least square minimizes the empirical loss function jointly in θ.

4 Robust Extension with the Huber Loss

To account for the possible heavy-tailedness of the data distribution, we extend the above work by replacing the square loss by the Huber loss, and provide an efficient algorithm to solve the optimization problem. It turns out that the algorithm is simply a weighted version of Algorithm 1. We also provide an iterative algorithm to determine the numbers of the column/row factors.

4.1 Algorithm

Standard statistical procedures that are based on the method of least squares often behave poorly in the presence of heavy-tailed data. The observed data are often heavy-tailed in areas such as finance and macroeconomics. To deal with heavy-tailed data, a natural idea is to replace the least squares loss function with the Huber loss function, i.e., we consider the following optimization problem:

$$\begin{aligned} &\min_{\{\mathbf{R},\mathbf{C},\mathbf{F}_t\}} L_2(\mathbf{R},\mathbf{C},\mathbf{F}_t) = \frac{1}{T}\sum_{t=1}^{T} H_\tau\Big(\sqrt{\|\mathbf{X}_t - \mathbf{R}\mathbf{F}_t\mathbf{C}^\top\|_F^2}\Big), \\ &\text{s.t.} \frac{1}{p_1}\mathbf{R}^\top\mathbf{R} = \mathbf{I}_{k_1}, \frac{1}{p_2}\mathbf{C}^\top\mathbf{C} = \mathbf{I}_{k_2}. \end{aligned} \tag{4.1}$$

where the Huber loss $H_\tau(x)$ is defined as

$$H_\tau(x) = \begin{cases} x^2, & |x| \le \tau\ , \\ 2\tau|x| - \tau^2, & |x| > \tau. \end{cases}$$

By similar arguments as for the least squares case (see details in the supplementary material), we have

$$\begin{aligned} &\mathbf{R}\mathbf{\Theta} = \mathbf{M}_c^w\mathbf{R} \ \text{ and } \ \mathbf{C}\mathbf{\Lambda} = \mathbf{M}_r^w\mathbf{C}, \text{ where} \\ &\mathbf{M}_c^w = \frac{1}{Tp_2}\sum_{t=1}^{T} w_t\mathbf{X}_t\mathbf{C}\mathbf{C}^\top\mathbf{X}_t^\top, \ \ \mathbf{M}_r^w = \frac{1}{Tp_1}\sum_{t=1}^{T} w_t\mathbf{X}_t^\top\mathbf{R}\mathbf{R}^\top\mathbf{X}_t, \end{aligned} \tag{4.2}$$

where $\mathbf{\Theta}$ and $\mathbf{\Lambda}$ are diagonal Lagrangian multipliers matrices and the weights are

$$w_t = \begin{cases} 1, & \sqrt{\|\mathbf{X}_t - \mathbf{R}\mathbf{F}_t\mathbf{C}^T\|_F^2} \le \tau, \\ \tau\dfrac{1}{\sqrt{\mathrm{Tr}(\mathbf{X}_t^\top\mathbf{X}_t) - \dfrac{1}{p_1p_2}\mathrm{Tr}(\mathbf{X}_t^\top\mathbf{R}\mathbf{R}^\top\mathbf{X}_t\mathbf{C}\mathbf{C}^\top)}}, & \sqrt{\|\mathbf{X}_t - \mathbf{R}\mathbf{F}_t\mathbf{C}^T\|_F^2} > \tau. \end{cases}$$

By (4.2), we see that the $\mathbf{M}_c^w$ and $\mathbf{M}_r^w$ are weighted versions of the $\mathbf{M}_c$ and $\mathbf{M}_r$, respectively. We denote the first k_1 eigenvectors of $\mathbf{M}_c^w$ as $\{\boldsymbol{r}^w(1), \ldots, \boldsymbol{r}^w(k_1)\}$ and the corresponding eigenvalues as $\{\theta_1^w, \ldots, \theta_{k_1}^w\}$. Similarly, we denote the first k_2 eigenvectors of $\mathbf{M}_r^w$ as $\{\boldsymbol{c}^w(1), \ldots, \boldsymbol{c}^w(k_2)\}$ and the corresponding eigenvalues as $\{\lambda_1^w, \ldots, \lambda_{k_2}^w\}$. From (4.2), we clearly see that $\mathbf{R}^w = \sqrt{p_1}(\boldsymbol{r}^w(1), \ldots, \boldsymbol{r}^w(k_1))$, $\mathbf{C}^w = \sqrt{p_2}(\boldsymbol{c}^w(1), \ldots, \boldsymbol{c}^w(k_2))$, $\mathbf{\Theta}^w = \text{diag}(\theta_1^w, \ldots, \theta_{k_1}^w)$, $\mathbf{\Lambda}^w = \text{diag}(\lambda_1^w, \ldots, \lambda_{k_2}^w)$ satisfy the KKT condition. Both $\mathbf{M}_c^w$ and $\mathbf{M}_r^w$ rely on the unknown row/column factor loadings (see the weights w_t).

We propose an iterative procedure to get the estimators, which turns out to be slightly different from the iterative procedure in the last section, as we need to update $\mathbf{R}^w$ and $\mathbf{C}^w$ simultaneously to update the weights w_t. We summarized the algorithm in Algorithm 2 and the initial estimators $\widehat{\mathbf{R}}$ and $\widehat{\mathbf{C}}$ can also be chosen as the estimators by α-PCA. Once the factor loading matrices are estimated, the factor matrix $\mathbf{F}_t$ can be estimated by $\widetilde{\mathbf{F}}_t^w = \widetilde{\mathbf{R}}^{w\top}\mathbf{X}_t\widetilde{\mathbf{C}}^w/(p_1p_2)$ and thus the common component matrix $\mathbf{S}$ can be estimated by $\widetilde{\mathbf{S}}^w = \widetilde{\mathbf{R}}^w\widetilde{\mathbf{F}}_t^w\widetilde{\mathbf{C}}^{w\top}$. As in each step, we need to update the weights, it is hard to derive the asymptotic theory for the estimators in Algorithm 2 and we demonstrate its superiority by extensive numerical experiments. We develop the asymptotic theory for the theoretical minimizers in the following and leave the asymptotic theory of the estimators from Algorithm 2 to our future work.

Algorithm 2 Robust Matrix Factor Analysis

Input: Data matrices $\{\mathbf{X_t}\}_{t\leq T}$, the row factor number k_1, the column factor number k_2
Output: Factor loading matrices $\widetilde{\mathbf{R}}^w$ and $\widetilde{\mathbf{C}}^w$

1: Obtain the initial estimators $\widehat{\mathbf{R}}^{(0)}$ and $\widehat{\mathbf{C}}^{(0)}$ by α-PCA;
2: Compute the weights $\{w_t\}, t = 1, \ldots, T$;
3: Using $\{w_t\}$ and $\widehat{\mathbf{R}}^{(0)}$ and $\widehat{\mathbf{C}}^{(0)}$ to compute $\mathbf{M}_c^w$ and its corresponding first k_1 eigenvectors $\{\boldsymbol{r}^w(1), \ldots, \boldsymbol{r}^w(k_1)\}$. Update $\widehat{\mathbf{R}}^{(1)}$ as $\sqrt{p_1}(\boldsymbol{r}^w(1), \ldots, \boldsymbol{r}^w(k_1))$.
4: Using $\{w_t\}$ and $\widehat{\mathbf{R}}^{(0)}$ and $\widehat{\mathbf{C}}^{(0)}$ to compute $\mathbf{M}_r^w$ and its corresponding first k_2 eigenvectors $\{\boldsymbol{c}^w(1), \ldots, \boldsymbol{c}^w(k_2)\}$. Update $\widehat{\mathbf{C}}^{(1)}$ as $\sqrt{p_2}(\boldsymbol{c}^w(1), \ldots, \boldsymbol{c}^w(k_2))$.
5: Repeat steps 2-4 until convergence and output the estimators from the last step and denoted as $\widetilde{\mathbf{R}}^w$ and $\widetilde{\mathbf{C}}^w$.

In (2.4), when there exist outliers in observations $\{\mathbf{Y}_t\}$, we naturally would consider a weighted sample covariance matrix to decrease the impact of outliers, and $\mathbf{M}_c^w$ would be an ideal choice. This turns out to be the estimators by minimizing the Huber loss function, which is exactly a weighted version of the projection technique. The weighted projection technique not only reduces the magnitudes of the idiosyncratic error components and thus increases the signal-to-noise ratio, but also neglects the impact of outliers by putting very small weights on them. As far as we know, this is the first time that the weighted projection technique is proposed in matrix/tensor-valued data analysis.

In the following, we derive the convergence rates of the theoretical minimizers of the Huber loss function in (4.1), denoted by $(\widehat{\mathbf{R}}_h, \widehat{\mathbf{F}}_{th}, \widehat{\mathbf{C}}_h)$. To this end, we introduce the following Assumption 3′, parallel to the Assumption 3 in Section 3.

Assumption 3′: The distribution functions of $e_{ij,t}|\{\mathbf{F}_{0t}\}$ have a common support covering an open neighborhood of the origin, $T\log T/(p_1p_2) = o(1)$, $E(e_{ij,t}|\{\mathbf{F}_{0t}\}) = 0$ and $E((e_{ij,t})^{2+\epsilon}|\{\mathbf{F}_{0t}\}) \le C$ for some constant C and any arbitrarily small number $\epsilon > 0$.

Assumption 3′ relaxes the sub-Gaussian tail constraint of the idiosyncratic errors and only requires the boundedness of the $(2+\epsilon)$th moment, though with a mild scaling condition on (p_1, p_2, T). Similar to Theorem 3.1, we obtain the following theorem which establishes the same convergence rates either under Assumption 3 or Assumption 3′.

Theorem 4.1. Let $\widehat{\mathbf{S}}_{1h} = \mathrm{sgn}\Big(\frac{1}{T}\sum_{t=1}^{T}(\widehat{\mathbf{F}}_{th}\mathbf{F}_{0t}^\top)\Big)$ and $\widehat{\mathbf{S}}_{2h} = \mathrm{sgn}\Big(\frac{1}{T}\sum_{t=1}^{T}(\widehat{\mathbf{F}}_{th}^\top\mathbf{F}_{0t})\Big)$. Further let $\tau = \tau'\sqrt{p_1p_2}$ for some constant $\tau' > 0$, then either under Assumptions 1,2,3, or under Assumptions 1, 2, 3′, we have

$$\frac{1}{p_1}\|\widehat{\mathbf{R}}_h - \mathbf{R}_0\widehat{\mathbf{S}}_{1h}\|_F^2 + \frac{1}{p_2}\|\widehat{\mathbf{C}}_h - \mathbf{C}_0\widehat{\mathbf{S}}_{2h}\|_F^2 + \frac{1}{T^2}\|\sum_{t=1}^{T}(\widehat{\mathbf{F}}_{th} - \widehat{\mathbf{S}}_{1h}\mathbf{F}_{0t}\widehat{\mathbf{S}}_{2h})\|_F^2 = O_p\left(\frac{1}{L}\right),$$

where $L = \min\{p_1p_2, Tp_2, Tp_1\}$.

The convergence rates in Theorem 4.1 are still correct for the spectral norm which is upper bounded by the Frobenius norm. The constant τ' in the theorem serves as a tuning parameter that controls the fraction of data to be winsorized. As in Huang & Ding (2008), we suggest to set τ' so that a half of the data matrices $\{\mathbf{X}_t, t = 1, \ldots, T\}$ are winsorized, which is justified by extensive simulation studies later.

4.2 Determining the Pair of Factor Numbers

It's well known that accurate estimation of the numbers of factors is of great importance to do matrix factor analysis (Yu et al. 2022). We borrow the eigenvalue-ratio idea from Ahn & Horenstein (2013). In detail, k_1 and k_2 are estimated by

$$\widehat{k}_1 = \arg\max_{j \le k_{\max}} \frac{\lambda_j(\mathbf{M}_c^w)}{\lambda_{j+1}(\mathbf{M}_c^w)}, \quad \widehat{k}_2 = \arg\max_{j \le k_{\max}} \frac{\lambda_j(\mathbf{M}_r^w)}{\lambda_{j+1}(\mathbf{M}_r^w)} \tag{4.3}$$

where $k_{\max}$ is a predetermined value larger than k_1 and k_2.

Algorithm 3 Robust iterative algorithm to estimate the numbers of factors

Input: Data $\mathbf{X_t}$, maximum number k_{max}, maximum iteration steps m
Output: Numbers of row and column factors $\widehat{k}_1$ and $\widehat{k}_2$

1: initialization: $\widehat{k}_1^{(0)} = k_{max}, \widehat{k}_2^{(0)} = k_{max}$;
2: estimate $\mathbf{R}$ and $\mathbf{C}$ by α-PCA and denote the estimators as $\widehat{\mathbf{R}}^{(0)}$ and $\widehat{\mathbf{C}}^{(0)}$, respectively.
3: for $t = 1, 2, \cdots, m$, given $\widehat{k}_2^{(t-1)}$, calculate $\mathbf{M}_c^{w(t)}$ using $\widehat{\mathbf{R}}^{(t-1)}$ and $\widehat{\mathbf{C}}^{(t-1)}$, then obtain $\widehat{k}_1^{(t)}$ by (4.3) ;
4: given $\widehat{k}_1^{(t)}$, update $\widehat{\mathbf{R}}^{(t)}$ by (4.2), and calculate $\mathbf{M}_r^{w(t)}$ using $\widehat{\mathbf{R}}^{(t)}$ and $\widehat{\mathbf{C}}^{(t-1)}$, then obtain $\widehat{k}_2^{(t)}$ by (4.3);
5: given $\widehat{k}_2^{(t)}$, update $\widehat{\mathbf{C}}^{(t)}$ by (4.2).
6: repeat steps 3-5 until $\widehat{k}_1^{(t)} = \widehat{k}_1^{(t-1)}$ and $\widehat{k}_2^{(t)} = \widehat{k}_2^{(t-1)}$ or reach the maximum iteration steps.

If the common factors are sufficiently strong, the leading k_1 eigenvalues of $\mathbf{M}_c^w$ ($\mathbf{M}_r^w$) are well separated from the others, and the eigenvalue ratios in equation (4.3) will be asymptotically maximized exactly at $j = k_1$ ($j = k_2$). To avoid vanishing denominators,

we can add an asymptotically negligible term to the denominator of equation (4.3). The remaining problem is that to calculate $\mathbf{M}_c^w$ or $\mathbf{M}_r^w$, both $\mathbf{R}$ and $\mathbf{C}$ must be predetermined, which further indicates that k_1, k_2 must be given in advance. However, both k_1 and k_2 are unknown empirically. To circumvent the problem, we propose an iterative algorithm to determine the pair of factor numbers, see Algorithm 3. Thus our method is an **R**obust **it**erative **E**igenvalue-**R**atio (Rit-ER) based procedure.

5 Simulation Study

5.1 Data Generation

In this section, we introduce the data generating mechanism of the synthetic dataset in order to verify the performance of the proposed **R**obust-**M**atrix-**F**actor-**A**nalysis (RMFA) method in Algorithm 2.

We set $k_1 = k_2 = 3$, draw the entries of $\mathbf{R}_0$ and $\mathbf{C}_0$ independently from the uniform distribution $\mathcal{U}(-1, 1)$, and let

$$\text{Vec}(\mathbf{F}_{0t}) = \phi \times \text{Vec}(\mathbf{F}_{0,t-1}) + \sqrt{1-\phi^2} \times \epsilon_t, \epsilon_t \overset{i.i.d}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I}_{k_1 \times k_2}),$$

$$\text{Vec}(\mathbf{E}_t) = \psi \times \text{Vec}(\mathbf{E}_{t-1}) + \sqrt{1-\psi^2} \times \text{Vec}(\mathbf{U}_t),$$

where ϕ and ψ control the temporal and cross-sectional correlations, and $\mathbf{U}_t$ is generated either from the matrix normal distribution or matrix t-distribution respectively. In detail, when $\mathbf{U}_t$ is from a matrix normal distribution $\mathcal{MN}(\mathbf{0}, \mathbf{U}_E, \mathbf{V}_E)$, then $\text{Vec}(\mathbf{U}_t) \overset{i.i.d}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{V}_E \otimes \mathbf{U}_E)$. When $\mathbf{U}_t$ is from a matrix t-distribution $t_{p_1,p_2}(\nu, \mathbf{M}, \mathbf{U}_E, \mathbf{V}_E)$, $\mathbf{U}_t$ has probability density function

$$f(\mathbf{U}_t; \nu, \mathbf{M}, \mathbf{U}_E, \mathbf{V}_E) = K \times \left| \mathbf{I}_{p_1} \mathbf{U}_E^{-1} (\mathbf{U}_t - \mathbf{M}) \mathbf{V}_E^{-1} (\mathbf{U}_t - \mathbf{M})^\top \right|^{-\frac{\nu + p_1 + p_2 - 1}{2}},$$

where K is the regularization parameter. In our simulation study, we set $\mathbf{M} = 0$ and let $\mathbf{U}_E$ and $\mathbf{V}_E$ be matrices with ones on the diagonal, and the off-diagonal entries are $1/p_1$ and $1/p_2$, respectively. For matrix t-distribution, we resort to the **R** package "MixMatrix" to generate random samples. The pair of factor numbers is given except in Section 5.4. We also investigate in the supplement the performance of various methods in reconstructing the common components with the factor numbers estimated by Algorithm 3, and we find no big difference with the case where k_1 and k_2 are known. All the simulation results reported hereafter are based on 1000 replications. As in Huang & Ding (2008), the parameter τ is set as the median of $\left\{\|\mathbf{X}_t - \widehat{\mathbf{R}}\widehat{\mathbf{F}}_t\widehat{\mathbf{C}}^\top\|_F, t = 1\ldots, T\right\}$, where $\widehat{\mathbf{R}}$ and $\widehat{\mathbf{C}}$ are the initial estimators by the α-PCA method with $\alpha = 0$ and $\widehat{\mathbf{F}}_t = \widehat{\mathbf{R}}^\top\mathbf{X_t}\widehat{\mathbf{C}}/(p_1p_2)$.

In the supplement, we also present the computation time of various methods in terms of estimating the loading spaces and the factor numbers. The computational burden of our RMFA method is sort of in the middle in all procedures given below.

5.2 Estimation Error for the Loading Spaces

In this section, we compare the RMFA method in Algorithm 2, the α-PCA method by Chen & Fan (2021), the PE method by Yu et al. (2022) in Algorithm 1, the Time series Outer-Product Unfolding Procedure (TOPUP), the Time series Inner-Product Unfolding Procedure (TIPUP) both by Chen, Yang & Zhang (2022), and the corresponding iterative version of TOPUP/TIPUP (denoted by iTOPUP/iTIPUP) by Han et al. (2020), in terms of estimating the factor loading spaces. We consider the following two settings:

- **Setting A**: $p_1 = 20, T = p_2 \in \{20, 50, 100, 150, 200\}, \phi = \psi = 0.1$.
- **Setting B**: $p_2 = 20, T = p_1 \in \{20, 50, 100, 150, 200\}, \phi = \psi = 0.1$.

We first introduce a metric between two factor spaces as the factor loading matrices $\mathbf{R}$ and $\mathbf{C}$ are not identifiable. For two column-wise orthogonal matrices $(\boldsymbol{Q}_1)_{p\times q_1}$ and $(\boldsymbol{Q}_2)_{p\times q_2}$,

Table 1: Averaged estimation errors and standard errors of $\mathcal{D}(\widehat{\mathbf{R}}, \mathbf{R})$ and $\mathcal{D}(\widehat{\mathbf{C}}, \mathbf{C})$ for Settings A and B under Matrix Normal distribution and Matrix t_3 distribution over 1000 replications. "RMFA": proposed robust matrix factor analysis method. "α-PCA": α-PCA with $\alpha = 0$. "PE": projection estimation method. "TOPUP": Time series Outer-Product Unfolding Procedure. "TIPUP": Time series Inner-Product Unfolding Procedure. "iTOPUP": iterative procedure based on the TOPUP. "iTIPUP": iterative procedure based on the TIPUP.

Evaluation	T	p_1	p_2	RMFA	α-PCA	PE	TOPUP	TIPUP	iTOPUP	iTIPUP
Matrix Normal Distribution										
$\mathcal{D}(\widehat{\mathbf{R}}, \mathbf{R})$	20	20	20	0.0915(0.0160)	0.1117(0.0314)	0.0919(0.0164)	0.1488(0.0398)	0.4649(0.1000)	0.1727(0.0390)	0.4318(0.1020)
	50		50	0.0356(0.0052)	0.0594(0.0233)	0.0356(0.0052)	0.0814(0.0275)	0.3845(0.1219)	0.0974(0.0180)	0.3442(0.1115)
	100		100	0.0176(0.0024)	0.0467(0.0193)	0.0176(0.0023)	0.0605(0.0207)	0.3133(0.1583)	0.0658(0.0122)	0.2505(0.1074)
	150		150	0.0117(0.0016)	0.0432(0.0196)	0.0116(0.0016)	0.0527(0.0206)	0.2367(0.1632)	0.0516(0.0091)	0.1858(0.0995)
	200		200	0.0088(0.0012)	0.0421(0.0201)	0.0088(0.0012)	0.0489(0.0209)	0.2023(0.1664)	0.0435(0.0080)	0.1491(0.0952)
$\mathcal{D}(\widehat{\mathbf{C}}, \mathbf{C})$	20	20	20	0.0930(0.0162)	0.1140(0.0313)	0.0935(0.0167)	0.1501(0.0401)	0.4652(0.1016)	0.1731(0.0375)	0.4351(0.1055)
	50		50	0.0567(0.0060)	0.0590(0.0066)	0.0569(0.0061)	0.0956(0.0093)	0.4131(0.0891)	0.1409(0.0173)	0.4073(0.0925)
	100		100	0.0398(0.0032)	0.0404(0.0034)	0.0399(0.0033)	0.0823(0.0055)	0.3689(0.0820)	0.1301(0.0129)	0.3682(0.0827)
	150		150	0.0323(0.0025)	0.0325(0.0025)	0.0324(0.0025)	0.0772(0.0045)	0.3362(0.0815)	0.1240(0.0114)	0.3367(0.0820)
	200		200	0.0281(0.0021)	0.0282(0.0021)	0.0282(0.0022)	0.0750(0.0038)	0.3060(0.0789)	0.1204(0.0111)	0.3071(0.0792)
$\mathcal{D}(\widehat{\mathbf{R}}, \mathbf{R})$	20	20	20	0.0915(0.0160)	0.1117(0.0314)	0.0919(0.0164)	0.1488(0.0398)	0.4649(0.1000)	0.1727(0.0390)	0.4318(0.1020)
	50	50		0.0567(0.0061)	0.0591(0.0069)	0.0569(0.0063)	0.0955(0.0095)	0.4162(0.0937)	0.1411(0.0176)	0.4114(0.0948)
	100	100		0.0399(0.0034)	0.0405(0.0036)	0.0400(0.0035)	0.0823(0.0056)	0.3698(0.0821)	0.1306(0.0135)	0.3698(0.0847)
	150	150		0.0323(0.0024)	0.0325(0.0025)	0.0324(0.0025)	0.0771(0.0045)	0.3353(0.0829)	0.1239(0.0116)	0.3345(0.0815)
	200	200		0.0280(0.0020)	0.0281(0.0020)	0.0281(0.0021)	0.0752(0.0041)	0.3068(0.0781)	0.1208(0.0114)	0.3081(0.0796)
$\mathcal{D}(\widehat{\mathbf{C}}, \mathbf{C})$	20	20	20	0.0930(0.0162)	0.1140(0.0313)	0.0935(0.0167)	0.1501(0.0401)	0.4652(0.1016)	0.1731(0.0375)	0.4351(0.1055)
	50	50		0.0356(0.0053)	0.0589(0.0225)	0.0356(0.0053)	0.0802(0.0241)	0.3854(0.1225)	0.0970(0.0178)	0.3378(0.1100)
	100	100		0.0177(0.0025)	0.0479(0.0216)	0.0177(0.0025)	0.0606(0.0222)	0.3148(0.1588)	0.0658(0.0123)	0.2551(0.1105)
	150	150		0.0118(0.0017)	0.0437(0.0246)	0.0118(0.0017)	0.0537(0.0277)	0.2462(0.1711)	0.0516(0.0099)	0.1900(0.1042)
	200	200		0.0088(0.0012)	0.0428(0.0204)	0.0088(0.0012)	0.0487(0.0212)	0.1891(0.1555)	0.0435(0.0084)	0.1443(0.0933)
Matrix t_3 Distribution										
$\mathcal{D}(\widehat{\mathbf{R}}, \mathbf{R})$	20	20	20	0.1454(0.1416)	0.3752(0.1713)	0.2830(0.2004)	0.4109(0.1694)	0.5368(0.1118)	0.3340(0.2049)	0.5004(0.1282)
	50		50	0.0433(0.0738)	0.2959(0.1716)	0.1437(0.1943)	0.3285(0.1768)	0.4860(0.1383)	0.1956(0.1941)	0.4044(0.1416)
	100		100	0.0171(0.0315)	0.2657(0.1620)	0.0830(0.1630)	0.2921(0.1736)	0.4451(0.1649)	0.1397(0.1817)	0.3307(0.1554)
	150		150	0.0111(0.0179)	0.2488(0.1575)	0.0684(0.1571)	0.2728(0.1700)	0.4128(0.1740)	0.1137(0.1672)	0.2727(0.1612)
	200		200	0.0088(0.0226)	0.2382(0.1513)	0.0581(0.1475)	0.2621(0.1662)	0.3754(0.1799)	0.1006(0.1625)	0.2300(0.1655)
$\mathcal{D}(\widehat{\mathbf{C}}, \mathbf{C})$	20	20	20	0.1439(0.1407)	0.3709(0.1733)	0.2783(0.2008)	0.4040(0.1667)	0.5327(0.1158)	0.3302(0.2026)	0.5002(0.1278)
	50		50	0.0613(0.0732)	0.2154(0.1853)	0.1740(0.1941)	0.2696(0.1911)	0.4876(0.1291)	0.2348(0.1880)	0.4581(0.1305)
	100		100	0.0362(0.0347)	0.1387(0.1651)	0.1197(0.1720)	0.2025(0.1831)	0.4464(0.1368)	0.1989(0.1755)	0.4161(0.1293)
	150		150	0.0288(0.0166)	0.1125(0.1588)	0.0986(0.1621)	0.1749(0.1719)	0.4056(0.1374)	0.1812(0.1588)	0.3867(0.1279)
	200		200	0.0253(0.0242)	0.0992(0.1535)	0.0892(0.1581)	0.1608(0.1657)	0.3713(0.1413)	0.1726(0.1538)	0.3538(0.1308)
$\mathcal{D}(\widehat{\mathbf{R}}, \mathbf{R})$	20	20	20	0.1454(0.1416)	0.3752(0.1713)	0.2830(0.2004)	0.4109(0.1694)	0.5368(0.1118)	0.3340(0.2049)	0.5004(0.1282)
	50	50		0.0598(0.0660)	0.2143(0.1837)	0.1734(0.1925)	0.2618(0.1896)	0.4902(0.1254)	0.2324(0.1876)	0.4577(0.1266)
	100	100		0.0359(0.0237)	0.1362(0.1578)	0.1142(0.1621)	0.1995(0.1760)	0.4467(0.1315)	0.2014(0.1695)	0.4225(0.1266)
	150	150		0.0293(0.0231)	0.1120(0.1539)	0.0972(0.1572)	0.1784(0.1756)	0.4109(0.1430)	0.1887(0.1666)	0.3858(0.1320)
	200	200		0.0247(0.0176)	0.0923(0.1447)	0.0815(0.1473)	0.1625(0.1724)	0.3697(0.1418)	0.1760(0.1570)	0.3524(0.1302)
$\mathcal{D}(\widehat{\mathbf{C}}, \mathbf{C})$	20	20	20	0.1439(0.1407)	0.3709(0.1733)	0.2783(0.2008)	0.4040(0.1667)	0.5327(0.1158)	0.3302(0.2026)	0.5002(0.1278)
	50	50		0.0418(0.0669)	0.2992(0.1742)	0.1442(0.1922)	0.3258(0.1773)	0.4872(0.1329)	0.1933(0.1913)	0.4072(0.1388)
	100	50		0.0168(0.0231)	0.2644(0.1584)	0.0802(0.1578)	0.2908(0.1678)	0.4473(0.1608)	0.1424(0.1774)	0.3307(0.1530)
	150	150		0.0115(0.0252)	0.2572(0.1571)	0.0652(0.1518)	0.2803(0.1687)	0.4068(0.1752)	0.1211(0.1768)	0.2704(0.1635)
	200	200		0.0082(0.0153)	0.2459(0.1559)	0.0521(0.1408)	0.2685(0.1703)	0.3865(0.1813)	0.1055(0.1674)	0.2254(0.1618)

we define

$$\mathcal{D}(\boldsymbol{Q}_1, \boldsymbol{Q}_2) = \left(1 - \frac{1}{\max(q_1, q_2)} \mathrm{Tr}\Big(\boldsymbol{Q}_1 \boldsymbol{Q}_1^\top \boldsymbol{Q}_2 \boldsymbol{Q}_2^\top\Big)\right)^{1/2}.$$

By the definition of $\mathcal{D}(\boldsymbol{Q}_1, \boldsymbol{Q}_2)$, we can easily see that its value lies in the interval $[0, 1]$, which measures the distance between the column spaces spanned by $\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$. The column spaces spanned by $\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$ are the same when $\mathcal{D}(\boldsymbol{Q}_1, \boldsymbol{Q}_2) = 0$, and are orthogonal

when $\mathcal{D}(\boldsymbol{Q}_1, \boldsymbol{Q}_2) = 1$. The Gram-Schmidt orthonormal transformation can be used when $\boldsymbol{Q}_1$ and $\boldsymbol{Q}_2$ are not column-orthogonal matrices.

Table 1 shows the averaged estimation errors with standard errors in parentheses under Settings A and B for matrix normal distribution and matrix-variate t_3 distribution. Yu et al. (2022)'s simulation study showed that for the α-PCA method, the performances for $\alpha \in \{-1, 0, 1\}$ are comparable, thus we only report the simulation results for the α-PCA with $\alpha = 0$. The TOPUP/TIPUP by Chen, Yang & Zhang (2022) and iTOPUP/iTIPUP by Han et al. (2020) all performed worse than the other three methods, as these methods all rely on the strong persistency of the factors. All methods benefit from large dimensions, and when p_1 is small, RMFA and PE methods always show advantage over α-PCA in terms of estimating $\mathbf{R}$. When p_2 is small, RMFA and PE methods always show advantage over α-PCA in terms of estimating $\mathbf{C}$, which is consistent with the findings by Yu et al. (2022). What we want to emphasize is that the RMFA and PE method perform comparably under the normal idiosyncratic error case, which is also clearly seen from Figure 1 in the Introduction section. When the idiosyncratic errors are from heavy-tailed matrix t_3 distribution, the RMFA method is superior over the α-PCA and PE methods in all scenarios. The estimation errors by the PE and α-PCA methods are at least twice of those by the RMFA method, which indicates that the weights of the sample covariance matrix of the projected data involved in RMFA method play an important role when outliers exist. The simulation results for matrix t_4 distribution are put in Table 1 in the supplement and similar conclusions are drawn as for matrix t_3 distribution. As a result, the RMFA can be used as a safe replacement of the α-PCA and PE methods.

5.3 Estimation Error for Common Components

In this section, we compare the performances of the RMFA method with those of the α-PCA and PE methods in terms of estimating the common component matrices. We evaluate the

performance of different methods by the Mean Squared Error, i.e.,

$$\text{MSE} = \frac{1}{Tp_1p_2}\sum_{t=1}^{T}\|\widehat{\mathbf{S}}_t - \mathbf{S}_t\|_F^2,$$

where $\widehat{\mathbf{S}}_t$ refers to an arbitrary estimate and $\mathbf{S}_t$ is the true common component matrix at time point t.

Table 2 shows the averaged MSEs with standard errors in parentheses under Settings A and B for matrix normal distribution and matrix-variate t_3 and t_4 distributions. From Table 2, we see that the RMFA and PE methods perform comparably under the normal case, and both perform better than the α-PCA method. This attributes to the projection technique of both the RMFA and PE methods. In contrast, under the heavy-tailed t_3 and t_4 cases, the RMFA performs much better than both PE and α-PCA methods. The TOPUP/TIPUP and iTOPUP/iTIPUP all performed worse than the RMFA, PE and α-PCA methods.

5.4 Estimating the Numbers of Factors

In this subsection, we compare the empirical performances of the proposed Rit-ER method with those of the α-PCA based ER method (α-PCA-ER) by Chen & Fan (2021), the IterER method by Yu et al. (2022), the information-criterion methods iTOP-IC/iTIP-IC based on iTOPUP/iTIPUP by Han et al. (2022), the eigenvalue-ratio methods iTOP-ER/iTIP-ER based on iTOPUP/iTIPUP by Han et al. (2022) and the **T**otal mode-k **Cor**relation **Th**resholding method (TCorTh) by Lam (2021) in terms of estimating the numbers of factors.

Table 3 presents the frequencies of exact estimation and underestimation over 1000 replications under Setting A and Setting B. We set $k_{\max} = 10$ for IterER, α-PCA-ER, Rit-ER, iTOP-IC/iTIP-IC, iTOP-ER/iTIP-ER. Under the normal case, the IterER and Rit-ER perform comparably, but both perform better than all other methods. The information-

Table 2: Mean squared error and its standard deviation of the common component under Settings A and B over 1000 replications. "RMFA": proposed robust matrix factor analysis method. "α-PCA": α-PCA with $\alpha = 0$. "PE": projection estimation method. "TOPUP": Time series Outer-Product Unfolding Procedure. "TIPUP": Time series Inner-Product Unfolding Procedure. "iTOPUP": iterative procedure based on the TOPUP. "iTIPUP": iterative procedure based on the TIPUP.

Distribution	T	RMFA	α-PCA	PE	TOPUP	TIPUP	iTOPUP	iTIPUP
				Setting A: $p_1 = 20$, $T = p_2$				
normal	20	0.0136(0.0021)	0.0188(0.0042)	0.0137(0.0022)	0.0333(0.0085)	0.3346(0.1026)	0.0475(0.0123)	0.2931(0.1047)
	50	0.0040(0.0004)	0.0058(0.0014)	0.0040(0.0004)	0.0135(0.0027)	0.2774(0.1049)	0.0261(0.0049)	0.2508(0.0977)
	100	0.0018(0.0001)	0.0032(0.0010)	0.0018(0.0002)	0.0094(0.0018)	0.2250(0.1141)	0.0199(0.0033)	0.1887(0.0814)
	150	0.0011(0.0001)	0.0025(0.0009)	0.0011(0.0001)	0.0080(0.0015)	0.1774(0.1146)	0.0172(0.0027)	0.1491(0.0764)
	200	0.0008(0.0001)	0.0022(0.0009)	0.0008(0.0001)	0.0074(0.0014)	0.1472(0.1096)	0.0157(0.0025)	0.1200(0.0675)
t_4	20	0.0102(0.0250)	0.0519(0.0631)	0.0270(0.0589)	0.0835(0.0835)	0.3104(0.1251)	0.0582(0.0930)	0.2664(0.1209)
	50	0.0019(0.0004)	0.0219(0.0499)	0.0101(0.0444)	0.0363(0.0629)	0.2619(0.1279)	0.0275(0.0650)	0.2162(0.1121)
	100	0.0008(0.0001)	0.0098(0.0264)	0.0028(0.0229)	0.0201(0.0420)	0.2145(0.1349)	0.0163(0.0429)	0.1603(0.1014)
	150	0.0005(0.0001)	0.0083(0.0273)	0.0023(0.0257)	0.0161(0.0373)	0.1667(0.1275)	0.0134(0.0380)	0.1195(0.0844)
	200	0.0004(0.0001)	0.0070(0.0228)	0.0018(0.0216)	0.0151(0.0364)	0.1333(0.1212)	0.0127(0.0413)	0.0936(0.0781)
t_3	20	0.0408(0.0743)	0.1762(0.1242)	0.1228(0.1354)	0.2197(0.1426)	0.4219(0.1344)	0.1815(0.1849)	0.3736(0.1490)
	50	0.0100(0.0500)	0.1181(0.1371)	0.0745(0.1403)	0.1591(0.1654)	0.3909(0.1520)	0.1137(0.1858)	0.3204(0.1520)
	100	0.0030(0.0389)	0.0890(0.1232)	0.0499(0.1269)	0.1268(0.1607)	0.3526(0.1675)	0.0902(0.1836)	0.2672(0.1580)
	150	0.0013(0.0124)	0.0766(0.1152)	0.0420(0.1177)	0.1081(0.1474)	0.3052(0.1655)	0.0729(0.1604)	0.2221(0.1520)
	200	0.0014(0.0179)	0.0708(0.1127)	0.0381(0.1136)	0.1002(0.1446)	0.2658(0.1649)	0.0672(0.1574)	0.1880(0.1524)
				Setting B: $p_2 = 20$, $T = p_1$				
normal	20	0.0136(0.0021)	0.0188(0.0042)	0.0137(0.0022)	0.0333(0.0085)	0.3346(0.1026)	0.0475(0.0123)	0.2931(0.1047)
	50	0.0040(0.0004)	0.0058(0.0014)	0.0040(0.0004)	0.0133(0.0026)	0.2807(0.1075)	0.0260(0.0049)	0.2490(0.0955)
	100	0.0018(0.0002)	0.0033(0.0011)	0.0018(0.0002)	0.0093(0.0018)	0.2247(0.1144)	0.0198(0.0033)	0.1906(0.0848)
	150	0.0011(0.0001)	0.0025(0.0013)	0.0011(0.0001)	0.0081(0.0020)	0.1826(0.1195)	0.0172(0.0027)	0.1497(0.0784)
	200	0.0008(0.0001)	0.0022(0.0009)	0.0008(0.0001)	0.0074(0.0015)	0.1413(0.1052)	0.0158(0.0027)	0.1200(0.0698)
t_4	20	0.0102(0.0250)	0.0519(0.0631)	0.0270(0.0589)	0.0835(0.0835)	0.3104(0.1251)	0.0582(0.0930)	0.2664(0.1209)
	50	0.0020(0.0005)	0.0187(0.0394)	0.0077(0.0344)	0.0342(0.0546)	0.2561(0.1251)	0.0260(0.0589)	0.2155(0.1140)
	100	0.0008(0.0001)	0.0112(0.0328)	0.0041(0.0300)	0.0208(0.0428)	0.2107(0.1348)	0.0160(0.0403)	0.1616(0.1005)
	150	0.0005(0.0001)	0.0092(0.0299)	0.0029(0.0269)	0.0180(0.0453)	0.1736(0.1353)	0.0161(0.0482)	0.1256(0.0944)
	200	0.0004(0.0001)	0.0072(0.0241)	0.0017(0.0216)	0.0149(0.0377)	0.1401(0.1203)	0.0129(0.0377)	0.0968(0.0751)
t_3	20	0.0408(0.0743)	0.1762(0.1242)	0.1228(0.1354)	0.2197(0.1426)	0.4219(0.1344)	0.1815(0.1849)	0.3736(0.1490)
	50	0.0087(0.0418)	0.1160(0.1291)	0.0723(0.1341)	0.1520(0.1564)	0.3916(0.1440)	0.1097(0.1774)	0.3198(0.1501)
	100	0.0021(0.0163)	0.0843(0.1121)	0.0447(0.1136)	0.1195(0.1428)	0.3457(0.1557)	0.0875(0.1647)	0.2656(0.1488)
	150	0.0017(0.0178)	0.0776(0.1137)	0.0392(0.1139)	0.1117(0.1482)	0.3015(0.1655)	0.0778(0.1630)	0.2185(0.1531)
	200	0.0010(0.0120)	0.0707(0.1129)	0.0349(0.1149)	0.1043(0.1513)	0.2653(0.1661)	0.0710(0.1609)	0.1828(0.1531)

criterion methods iTOP-IC/iTIP-IC perform the worst in all cases. As the tails of the idiosyncratic errors become heavier (from normal to t_4, t_3), all estimates deteriorate, especially for the α-PCA-ER and TCorTh methods. The proposed Rit-ER method is the most robust and always performs the best for heavy-tailed data. As the sample size T grows, the proportion of exact estimation by Rit-ER raises towards one.

Table 3: The frequencies of exact estimation and underestimation of the numbers of factors under Settings A and B over 1000 replications. "Rit-ER": the proposed robust iterative eigenvalue-ration based method. "α-PCA-ER": α-PCA based eigenvalue-ratio method with $\alpha = 0$. "IterER": iterative eigenvalue-ration based method. "iTOP-IC"/"iTIP-IC": information-criterion method based on iTOPUP/ iTIPUP. "iTOP-ER"/"iTIP-ER": eigenvalue-ratio method based on iTOPUP/iTIPUP. "TCorTh": Total mode-k Correlation Thresholding method.

Distribution	T	Rit-ER	α-PCA-ER	IterER	iTOP-IC	iTIP-IC	iTOP-ER	iTIP-ER	TCorTh
Setting A: $p_1 = 20$, $T = p_2$									
normal	20	0.992(0.008)	0.647(0.353)	0.992(0.008)	0.000(1.000)	0.000(1.000)	0.758(0.242)	0.093(0.907)	0.672(0.328)
	50	1.000(0.000)	0.891(0.109)	1.000(0.000)	0.000(1.000)	0.000(1.000)	0.973(0.027)	0.196(0.804)	0.982(0.018)
	100	1.000(0.000)	0.880(0.120)	1.000(0.000)	0.000(1.000)	0.000(1.000)	0.994(0.006)	0.371(0.629)	0.997(0.003)
	150	1.000(0.000)	0.912(0.088)	1.000(0.000)	0.000(1.000)	0.000(1.000)	0.998(0.002)	0.525(0.475)	0.999(0.001)
	200	1.000(0.000)	0.901(0.099)	1.000(0.000)	0.000(1.000)	0.000(1.000)	0.996(0.004)	0.648(0.352)	0.999(0.001)
t_4	20	0.774(0.226)	0.310(0.690)	0.720(0.280)	0.000(1.000)	0.000(1.000)	0.565(0.435)	0.112(0.888)	0.765(0.235)
	50	0.937(0.063)	0.622(0.378)	0.864(0.136)	0.000(1.000)	0.000(1.000)	0.795(0.205)	0.224(0.776)	0.414(0.586)
	100	0.986(0.014)	0.665(0.335)	0.933(0.067)	0.000(1.000)	0.000(1.000)	0.888(0.112)	0.402(0.598)	0.148(0.852)
	150	0.992(0.008)	0.655(0.345)	0.955(0.045)	0.000(1.000)	0.000(1.000)	0.879(0.121)	0.581(0.419)	0.065(0.935)
	200	0.991(0.009)	0.651(0.349)	0.953(0.047)	0.000(1.000)	0.000(1.000)	0.879(0.121)	0.685(0.315)	0.034(0.966)
t_3	20	0.360(0.640)	0.097(0.903)	0.307(0.693)	0.001(0.999)	0.000(1.000)	0.263(0.737)	0.050(0.950)	0.479(0.521)
	50	0.704(0.296)	0.249(0.751)	0.579(0.421)	0.001(0.999)	0.000(1.000)	0.496(0.504)	0.155(0.845)	0.022(0.978)
	100	0.810(0.190)	0.264(0.736)	0.643(0.357)	0.000(1.000)	0.000(1.000)	0.555(0.445)	0.266(0.734)	0.000(1.000)
	150	0.855(0.145)	0.276(0.724)	0.691(0.309)	0.000(1.000)	0.000(1.000)	0.611(0.389)	0.391(0.609)	0.000(1.000)
	200	0.873(0.127)	0.260(0.740)	0.695(0.305)	0.000(1.000)	0.000(1.000)	0.609(0.391)	0.470(0.530)	0.000(1.000)
Setting B: $p_2 = 20$, $T = p_1$									
normal	20	0.992(0.008)	0.647(0.353)	0.992(0.008)	0.000(1.000)	0.000(1.000)	0.758(0.242)	0.093(0.907)	0.672(0.328)
	50	1.000(0.000)	0.895(0.105)	1.000(0.000)	0.000(1.000)	0.000(1.000)	0.988(0.012)	0.187(0.813)	0.978(0.022)
	100	1.000(0.000)	0.907(0.093)	1.000(0.000)	0.000(1.000)	0.000(1.000)	1.000(0.000)	0.343(0.657)	0.995(0.005)
	150	1.000(0.000)	0.909(0.091)	1.000(0.000)	0.000(1.000)	0.000(1.000)	0.998(0.002)	0.534(0.466)	0.999(0.001)
	200	1.000(0.000)	0.903(0.097)	1.000(0.000)	0.000(1.000)	0.000(1.000)	0.999(0.001)	0.646(0.354)	0.997(0.003)
t_4	20	0.774(0.226)	0.310(0.690)	0.720(0.280)	0.000(1.000)	0.000(1.000)	0.565(0.435)	0.112(0.888)	0.765(0.235)
	50	0.926(0.074)	0.643(0.357)	0.871(0.129)	0.000(1.000)	0.000(1.000)	0.826(0.174)	0.248(0.752)	0.403(0.597)
	100	0.950(0.050)	0.672(0.328)	0.913(0.087)	0.000(1.000)	0.000(1.000)	0.884(0.116)	0.416(0.584)	0.171(0.829)
	150	0.963(0.037)	0.652(0.348)	0.928(0.072)	0.000(1.000)	0.000(1.000)	0.908(0.092)	0.532(0.468)	0.049(0.951)
	200	0.968(0.032)	0.650(0.350)	0.937(0.063)	0.000(1.000)	0.000(1.000)	0.927(0.073)	0.655(0.345)	0.030(0.970)
t_3	20	0.360(0.640)	0.097(0.903)	0.307(0.693)	0.001(0.999)	0.000(1.000)	0.263(0.737)	0.050(0.950)	0.479(0.521)
	50	0.624(0.376)	0.223(0.777)	0.551(0.449)	0.000(1.000)	0.000(1.000)	0.515(0.485)	0.131(0.869)	0.025(0.975)
	100	0.716(0.284)	0.277(0.723)	0.631(0.369)	0.000(1.000)	0.000(1.000)	0.569(0.431)	0.272(0.728)	0.000(1.000)
	150	0.723(0.277)	0.270(0.730)	0.640(0.360)	0.001(0.999)	0.000(1.000)	0.584(0.416)	0.376(0.624)	0.000(1.000)
	200	0.775(0.225)	0.268(0.732)	0.703(0.297)	0.000(1.000)	0.000(1.000)	0.642(0.358)	0.487(0.513)	0.000(1.000)

6 Real data example

In this section, we illustrate the empirical performance of our proposed methods by analyzing a financial portfolio dataset, which was studied by Wang et al. (2019) and Yu et al. (2022). The financial portfolio dataset is composed of monthly returns of 100 portfolios, well structured into a 10×10 matrix at each time point, with rows corresponding to 10 levels of market capital size (denoted as S1-S10) and columns corresponding to 10 levels of book-to-equity ratio (denoted as BE1-BE10). The dataset collects monthly returns from January

1964 to December 2019 covering totally 672 months. The details are available at the website http://mba.tuck.dartmouth.edu/pages /faculty/ken.french/data_library.html.

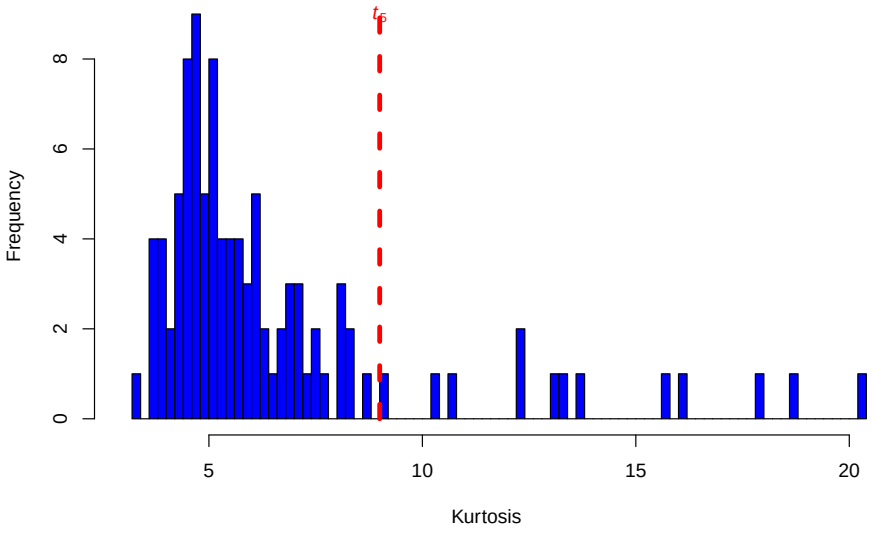

Figure 2: Histogram of the sample kurtosis for 100 portfolios and the red dashed line is the theoretical kurtosis of t_5 distribution.

Following the same preprocessing strategy by Wang et al. (2019) and Yu et al. (2022), we first subtract monthly market excess returns from the original return series and then standardize each of the series. We impute the missing values with the factor-model-based method by Xiong & Pelger (2022). The result of the augmented Dickey-Fuller test indicates the stationarity of the return series. The histogram of the sample kurtosis for 100 portfolios in Figure 2 demonstrates that the data are possibly heavy-tailed. The strong rule in He et al. (2021*b*) is implemented to diagnose the matrix factor structure in this dataset. For the parameters involved in the test, we set $\alpha = 0.01, M = 100$, $S \in \{200, 300, 400\}$, $f_1(S) = 1-\alpha-\sqrt{2\ln S/S}$, $f_2(S) = 1-\alpha-S^{-1/3}$, $f_3(S) = 1-\alpha-S^{-1/4}$, $f_4(S) = 1-\alpha-S^{-1/5}$, $f_5(S) = (1-\alpha)/2$. The results for all test specifications show that there is overwhelming evidence in favour of a matrix structure in the dataset.

The IterER method suggests that $(k_1, k_2) = (2, 1)$, the Rit-ER suggests that $(k_1, k_2) =$

Table 4: Loading matrices for Fama–French data set, after varimax rotation and scaling by 30. "RMFA": the robust matrix factor analysis method, "PE": the projected estimation method by Yu et al. (2022), α-PCA: the method in Chen & Fan (2021) with $\alpha = 0$, "ACCE": the approach proposed by Wang et al. (2019), "TOPUP" and "TIPUP": the methods proposed by Chen, Yang & Zhang (2022), "iTOPUP" and "iTIPUP" : the methods proposed by Han et al. (2020).

Size											
Method	Factor	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
RMFA	1	-16	-15	-13	-11	-8	-6	-3	0	4	6
	2	-6	-2	2	5	8	10	12	14	15	10
PE	1	-16	-15	-12	-10	-8	-5	-3	-1	4	7
	2	-6	-1	3	5	8	10	12	13	15	11
α-PCA	1	-14	-14	-13	-11	-9	-7	-4	-2	3	7
	2	-4	-2	1	3	6	9	12	13	16	14
ACCE	1	-12	-14	-12	-13	-10	-6	-3	-1	4	9
	2	-1	-1	-1	2	5	10	11	18	15	11
TOPUP	1	-12	-14	-12	-13	-10	-6	-3	-1	4	9
	2	-1	-1	-1	2	5	10	11	18	15	11
TIPUP	1	0	0	0	-4	-8	-11	-10	-18	-13	-11
	2	-13	-14	-13	-11	-7	-5	-4	1	5	11
iTOPUP	1	1	-1	1	-2	-6	-11	-10	-19	-13	-11
	2	-13	-13	-12	-13	-10	-6	-3	0	4	10
iTIPUP	1	1	0	1	-4	-7	-11	-11	-17	-13	-10
	2	-14	-14	-13	-11	-8	-6	-2	0	5	9
Book-to-Equity											
Method	Factor	BE1	BE2	BE3	BE4	BE5	BE6	BE7	BE8	BE9	BE10
RMFA	1	6	1	-3	-6	-9	-11	-12	-13	-12	-11
	2	19	17	12	9	5	3	0	-1	-1	0
PE	1	6	1	-4	-7	-10	-11	-12	-12	-12	-10
	2	20	17	11	8	4	2	0	-1	-1	0
α-PCA	1	6	2	-4	-7	-10	-11	-12	-13	-12	-11
	2	19	18	12	8	4	2	0	-1	-1	-1
ACCE	1	6	-1	-4	-8	-8	-9	-10	-13	-15	-12
	2	21	15	11	6	5	2	1	-2	-3	1
TOPUP	1	6	-1	-4	-8	-8	-9	-10	-13	-15	-12
	2	-21	-15	-11	-6	-5	-2	-1	2	3	-1
TIPUP	1	-18	-14	-13	-10	-7	-6	-4	0	2	-1
	2	-8	0	0	5	8	4	5	12	16	17
iTOPUP	1	6	0	-4	-8	-7	-9	-9	-13	-15	-13
	2	-21	-16	-11	-7	-6	-3	-1	2	4	-3
iTIPUP	1	5	0	-1	-5	-9	-7	-7	-13	-15	-17
	2	-17	-17	-12	-10	-7	-3	-3	3	4	0

$(1, 2)$, the α-PCA-ER, iTOP-IC and iTIP-IC all suggest $(k_1, k_2) = (1, 1)$, while iTOP-ER, iTIP-ER and TCorTh all suggest $(k_1, k_2) = (2, 2)$.

The estimated row and column loading matrices after varimax rotation and scaling are reported in Table 4. From the table, we observe two different loading patterns across the rows (sizes). For the RMFA, PE, α-PCA, ACCE, and TOPUP, the small size portfolios

Table 5: Rolling validation for the Fama–French portfolios. $12n$ is the sample size of the training set. $k_1 = k_2 = k$ is the number of factors. $\overline{MSE}$, $\bar{\rho}$, $\bar{v}$ are the mean pricing error, mean unexplained proportion of total variances and mean variation of the estimated loading space. "RMFA", "PE", "ACCE", "α-PCA", "TOPUP", "TIPUP", "iTOPUP" and "iTIPUP" are the same as in Table 4.

n	k	RMFA	PE	α-PCA	ACCE	TOPUP	TIPUP	iTOPUP	iTIPUP
						$\overline{MSE}$			
5	1	**0.860**	0.870	0.862	0.885	0.885	0.960	0.943	0.943
10	1	**0.849**	0.855	0.860	0.880	0.880	0.950	0.937	0.943
15	1	**0.850**	0.853	0.860	0.884	0.884	0.933	0.928	0.933
5	2	**0.594**	0.596	0.601	0.667	0.667	0.704	0.706	0.703
10	2	**0.600**	0.601	0.611	0.655	0.655	0.716	0.671	0.698
15	2	**0.602**	0.603	0.612	0.637	0.637	0.695	0.655	0.686
5	3	**0.518**	0.522	0.529	0.564	0.564	0.632	0.584	0.615
10	3	**0.516**	0.519	0.526	0.573	0.573	0.616	0.583	0.600
15	3	**0.515**	0.517	0.522	0.560	0.560	0.612	0.577	0.591
						$\bar{\rho}$			
5	1	**0.793**	0.802	0.796	0.828	0.828	0.943	0.933	0.935
10	1	**0.776**	0.784	0.791	0.815	0.815	0.921	0.914	0.923
15	1	**0.778**	0.782	0.792	0.812	0.812	0.890	0.907	0.914
5	2	**0.618**	0.625	0.628	0.673	0.673	0.740	0.719	0.733
10	2	**0.623**	0.628	0.636	0.668	0.668	0.738	0.687	0.728
15	2	**0.622**	0.626	0.630	0.652	0.652	0.714	0.677	0.711
5	3	**0.545**	0.550	0.556	0.590	0.590	0.664	0.610	0.637
10	3	**0.544**	0.548	0.555	0.594	0.594	0.645	0.604	0.624
15	3	**0.541**	0.545	0.544	0.582	0.582	0.637	0.601	0.614
						$\bar{v}$			
5	1	**0.144**	0.176	0.241	0.303	0.303	0.622	0.602	0.585
10	1	**0.062**	0.085	0.203	0.165	0.165	0.354	0.385	0.389
15	1	**0.045**	0.064	0.234	0.152	0.152	0.291	0.347	0.345
5	2	**0.163**	0.239	0.350	0.460	0.460	0.579	0.620	0.635
10	2	**0.080**	0.092	0.261	0.257	0.257	0.427	0.419	0.429
15	2	**0.050**	0.057	0.173	0.189	0.189	0.302	0.304	0.303
5	3	**0.241**	0.286	0.432	0.497	0.497	0.628	0.556	0.599
10	3	**0.110**	0.114	0.353	0.303	0.303	0.402	0.344	0.390
15	3	**0.072**	0.084	0.308	0.299	0.299	0.384	0.330	0.353

load heavily on the first factor while large size portfolios on the second factor, but for the TIPUP, iTOPUP and iTIPUP, the pattern reverses. We also find two distinct patterns across the columns (Book-to-Equity).

To further compare these methods, we use a rolling-validation procedure as in Wang et al. (2019). For each year t from 1996 to 2019, we repeatedly use n (bandwidth) years before t to fit the matrix-variate factor model. The fitted loadings are then used to estimate the monthly factors and idiosyncratic errors in the current year t. Let $\mathbf{Y}_t^i$ and $\widehat{\mathbf{Y}}_t^i$ be the observed and estimated price matrix of month i in year t, and $\bar{\mathbf{Y}}_t$ be the mean price matrix.

Define

$$\text{MSE}_t = \frac{1}{12 \times 10 \times 10} \sum_{i=1}^{12} \|\widehat{\mathbf{Y}}_t^i - \mathbf{Y}_t^i\|_F^2, \quad \rho_t = \frac{\sum_{i=1}^{12} \|\widehat{\mathbf{Y}}_t^i - \mathbf{Y}_t^i\|_F^2}{\sum_{i=1}^{12} \|\mathbf{Y}_t^i - \bar{\mathbf{Y}}_t\|_F^2},$$

as the mean squared pricing error and unexplained proportion of total variances, respectively. In the rolling-validation procedure, the variation of loading space is measured by $v_t := \mathcal{D}(\widehat{\mathbf{C}}_t \otimes \widehat{\mathbf{R}}_t, \widehat{\mathbf{C}}_{t-1} \otimes \widehat{\mathbf{R}}_{t-1})$.

We tried diverse combinations of n and numbers of factors ($k_1 = k_2 = k$). We report the means of MSE_t, ρ_t and v_t in Table 5. The RMFA method has the lowest pricing errors across the board.

7 Discussion

The current work studies the large-dimensional matrix factor model from the least squares and Huber Loss points of view. The KKT condition of minimizing the residual sum of squares under the identifiability condition naturally motivates one to adopt the iterative projection estimation algorithm by Yu et al. (2022). For the Huber loss, the corresponding KKT condition motivates us to do PCA on weighted sample covariance matrix of the projected data. We investigate the properties of theoretical minimizers of the squared loss and the Huber loss under the identifiability condition. We also propose robust estimators of the pair of factor numbers. The limitation of the current work lies in that we do not provide a theoretical guarantee of the estimators in the solution path of the iterative algorithm, which involves both statistical and computational accuracy. We leave it as a future research direction.

References

Ahn, S. C. & Horenstein, A. R. (2013), 'Eigenvalue ratio test for the number of factors', *Econometrica* **81**(3), 1203–1227.

Aït-Sahalia, Y., Kalnina, I. & Xiu, D. (2020), 'High-frequency factor models and regressions', *Journal of Econometrics* **216**, 86–105.

Aït-Sahalia, Y. & Xiu, D. (2017), 'Using principal component analysis to estimate a high dimensional factor model with high frequency data', *Journal of Econometrics* **201**, 388–399.

Ando, T. & Bai, J. (2016), 'Panel data models with grouped factor structure under unknown group membership', *Journal of Applied Econometrics* **31**(1), 163–191.

Bai, J. (2003), 'Inferential theory for factor models of large dimensions', *Econometrica* **71**(1), 135–171.

Bai, J. & Ng, S. (2002), 'Determining the number of factors in approximate factor models', *Econometrica* **70**(1), 191–221.

Bai, J. & Ng, S. (2013), 'Principal components estimation and identification of static factors', *Journal of econometrics* **176**(1), 18–29.

Chang, J., He, J., Yang, L. & Yao, Q. (2021), 'Modelling matrix time series via a tensor cp-decomposition', *arXiv e-prints: 2112.15423* .

Chen, E. Y. & Chen, R. (2020), 'Modeling dynamic transport network with matrix factor models: with an application to international trade flow.', *arXiv e-prints:1901.00769* .

Chen, E. Y. & Fan, J. (2021), 'Statistical inference for high-dimensional matrix-variate factor models', *Journal of the American Statistical Association (just-accepetd)* pp. 1–44.

Chen, E. Y., Tsay, R. S. & Chen, R. (2020), 'Constrained factor models for high-dimensional matrix-variate time series', *Journal of the American Statistical Association* **115**(530), 775–793.

Chen, E. Y., Xia, D., Cai, C. & Fan, J. (2020), 'Semiparametric tensor factor analysis by iteratively projected SVD', *arXiv e-prints: 2007.02404* .

Chen, L., Dolado, J. J. & Gonzalo, J. (2021), 'Quantile factor models', *Econometrica* **89**(2), 875–910.

Chen, R., Han, Y., Li, Z., Xiao, H., Yang, D. & Yu, R. (2022), 'Analysis of tensor time series: tensorts', *Journal of Statistical Software, in press* .

Chen, R., Yang, D. & Zhang, C.-H. (2022), 'Factor models for high-dimensional tensor time series', *Journal of the American Statistical Association* **117**(537), 94–116.

Chen, W. & Lam, C. (2022), 'Rank and factor loadings estimation in time series tensor factor model by pre-averaging', *arXiv e-prints: 2208.04012* .

Fan, J., Liao, Y. & Mincheva, M. (2013), 'Large covariance estimation by thresholding principal orthogonal complements', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **75**(4), 603–680.

Gao, Z., Yuan, C., Jing, B.-Y., Wei, H. & Guo, J. (2021), 'A two-way factor model for high-dimensional matrix data', *arXiv e-prints:2103.07920* .

Han, Y., Chen, R., Yang, D. & Zhang, C. (2020), 'Tensor factor model estimation by iterative projection', *arXiv e-prints: 2006.02611* .

Han, Y., Chen, R. & Zhang, C.-H. (2022), 'Rank determination in tensor factor model', *Electronic Journal of Statistics* **16**(1), 1726–1803.

He, Y., Kong, X. B., Trapani, L. & Yu, L. (2021*a*), 'Online change-point detection for matrix-valued time series with latent two-way factor structure', *arXiv e-prints:2112.13479* .

He, Y., Kong, X., Trapani, L. & Yu, L. (2021*b*), 'One-way or two-way factor model for matrix sequences?', *arXiv: arXiv:2110.01008* .

He, Y., Kong, X., Yu, L. & Zhang, X. (2022), 'Large-dimensional factor analysis without moment constraints', *Journal of Business & Economic Statistics* **40**, 302–312.

Huang, H. & Ding, C. (2008), Robust tensor factorization using R_1 norm, *in* '2008 IEEE Conference on Computer Vision and Pattern Recognition', IEEE Computer Society, pp. 1–8.

Huber, P. J. (1964), 'Robust estimation of a location parameter', *The Annals of Mathematical Statistics* **35**(1), 73–101.

Jing, B.-Y., Li, T., Lyu, Z. & Xia, D. (2021), 'Community detection on mixture multi-layer networks via regularised tensor decomposition', *The Annals of Statistics* **49**(6), 3181–3205.

Jing, B.-Y., Shao, Q.-M. & Zhou, W. (2008), 'Towards a universal self-normalized moderate deviation', *Transactions of the American Mathematical Society* **360**(8), 4263–4285.

Lam, C. (2021), Rank determination for time series tensor factor model using correlation thresholding, Technical report, Working paper LSE.

Liu, X. & Chen, E. (2019), 'Helping effects against curse of dimensionality in threshold factor models for matrix time series', *arXiv:1904.07383* .

Onatski, A. (2009), 'Testing hypotheses about the number of factors in large factor models', *Econometrica* **77**, 1447–1479.

Shao, Q.-M. (1997), 'Self-normalized large deviations', *The Annals of Probability* **25**(1), 285–328.

Stock, J. H. & Watson, M. W. (2002), 'Forecasting using principal components from a large number of predictors', *Journal of the American statistical association* **97**(460), 1167–1179.

Trapani, L. (2018), 'A randomised sequential procedure to determine the number of factors', *Journal of the American Statistical Association* **113**, 1341–1349.

Wainwright, M. (2019), 'High-dimensional statistics: A non-asymptotic viewpoint', *Cambridge University Press, Cambridge, UK* .

Wang, D., Liu, X. & Chen, R. (2019), 'Factor models for matrix-valued high-dimensional time series', *Journal of Econometrics* **208**(1), 231–248.

Xiong, R. & Pelger, M. (2022), 'Large dimensional latent factor modeling with missing observations and applications to causal inference', *Journal of Econometrics, in press* .

Yu, L., He, Y., Kong, X. & Zhang, X. (2022), 'Projected estimation for large-dimensional matrix factor models', *Journal of Econometrics* **229**, 201–217.

Yu, L., He, Y. & Zhang, X. (2019), 'Robust factor number specification for large-dimensional elliptical factor model', *Journal of Multivariate analysis* **174**, 104543.