Matrix Completion and Factor Model Analysis of Missing Data

Jizhou Liu

May 22, 2021

1 Introduction

Missing data is a prevalent problem in multivariate time series data and large dimensional panel data. A variety of reasons can lead to missing data, such as contamination and attrition in data collecting processes. In financial data, introduction of new stock return series in the case of IPO introduces missing observations before a company is listed. In both financial and macroeconomic data, missing data comes from the fact that different time series variables might be collected in different frequencies. For examples, in stock market data, companies with large trading volume usually are traded more frequently. In tech companies, users have different levels of exposure and activities when using the product, let alone many of them are not willing to disclosure or provide all the information. In the case of Netflix challenge, or more generally recommendation systems, data is missing by nature. Because most users would not have watched all the movies on Netflix, bought all products on Amazon and etc.

In order to describe mechanism of missing data, let $\Omega \in \{0,1\}^{N\times T}$ be the indicator matrix where $\Omega_{it} = 0$ means that the observation is missing otherwise it is not. The mechanism is defined as the conditional probability of Ω given the data $X \in \mathbb{R}^{N\times T}$, which is denoted by $\phi(\Omega \mid X)$. When the conditional distribution is identical for all possible values of X, we say that the data is missing completely at random (MCAR). Some of the popular MCAR mechanisms include:

1. *Missing at random:* $P(\Omega_{it} = 1) = p$ for all i and t. In this case all units and times are equally likely to be observed.

$$X = \begin{pmatrix} \omega_{11} & x_{12} & x_{13} & x_{14} & * \\ x_{21} & * & x_{23} & x_{24} & x_{25} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} \\ x_{51} & x_{52} & * & x_{54} & x_{55} \\ x_{61} & x_{62} & x_{63} & * & x_{65} \\ * & x_{72} & x_{73} & x_{74} & x_{75} \\ x_{81} & x_{82} & x_{83} & x_{84} & x_{85} \end{pmatrix}$$

2. *Block-wise Missing:* $P(\Omega_{it} = 1) = 1$ for $i \geq a$ and $t \geq q$.

$$X = \begin{pmatrix} \omega_{11} & x_{12} & x_{13} & x_{14} & x_{15} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} \\ x_{61} & x_{62} & * & * & * \\ x_{71} & x_{72} & * & * & * \\ x_{81} & x_{82} & * & * & * \end{pmatrix}$$

3. *Staggered treatment adoption:* $P(\Omega_{it} = 1) = 1$ for $i \geq a$ and $t \geq q_i$.

$$X = \begin{pmatrix} \omega_{11} & x_{12} & x_{13} & x_{14} & x_{15} \\ x_{21} & x_{22} & x_{23} & x_{24} & x_{25} \\ x_{31} & x_{32} & x_{33} & x_{34} & x_{35} \\ x_{41} & x_{42} & x_{43} & x_{44} & x_{45} \\ x_{51} & x_{52} & x_{53} & x_{54} & x_{55} \\ x_{61} & x_{62} & x_{63} & * & * \\ x_{71} & x_{72} & * & * & * \\ x_{81} & * & * & * & * \end{pmatrix}$$

The above three missing patterns are very popular among all types of missing data. The data set that has contamination or attrition during survey shows the same pattern as *Missing at random:*. In the context of program evaluation, we can view the outcome of untreated units as missing in terms of the potential outcome of being untreated. In this case, *Block-wise Missing* and *Staggered treatment adoption* are prevalent. When the data have mixture of two frequencies, the data can be re-arranged by moving all the missing entries to the lower-right of the matrix, which would make the data look like *Block-wise Missing.*

The more complicated missing mechanism, where the data are not missing at random (NMAR), introduces selection bias to the data set. Hence, it is more difficult to deal with. For instance, in stock return data where there exists mixtures of frequencies, it is likely that firms with better performance are traded more frequently. In the famous Netflix challenge, the data is an extremely large and sparse matrix whose entries are user ratings. The data entries are missing when the user chooses not to submit his ratings or haven't watch the movie. Possibly, users choose to watch and rate movies that are more interesting to them. In the extreme case, users might not give ratings to those movies that they would give low ratings, i.e. $P(\Omega_{it} = 0 \mid X_{it}) = 1$ if $X_{it} \leq 1$. If the data are NMAR, the observed data set is not representative for the population it comes from.

A common starting point of imputing missing values is to assume a low rank structure for the underlying data matrix. We can write the data matrix as the sum of a low rank matrix and observation errors.

$$\tilde{X} = M + \epsilon \tag{1}$$

and

$$X_{it} = \begin{cases} \tilde{X}_{it} & \text{if } \Omega_{it} = 1 \\ \text{NA} & \text{if } \Omega_{it} = 0 \end{cases}$$

where $\text{rank}(M) \ll \min\{N, T\}$. In the context of multivariate time series and panel data, people also tend to assume a factor model structure for the data matrix.

$$\tilde{X} = F\Lambda^T + \epsilon \tag{2}$$

where $F \in \mathbb{R}^{N\times r}$, $\Lambda \in \mathbb{R}^{r\times T}$, and $r \ll \min\{N, T\}$. When assuming a low rank matrix, we can use the existing observations to extract the underlying low rank matrix which serves as a good estimate to missing entries in data.

The rest of the paper is structured as follows. In the next section, we present numerous methods proposed by different communities including econometrics, statistics and computer science. We go through their assumptions and algorithms, and then compare their advantages and disadvantages. In the simulation section, we construct multiple models to verify the arguments of pros and cons made in the second section. Finally, we apply representative methods introduced in the previous section to the problem of forecasting macroeconomic variables with real data.

2 Methods

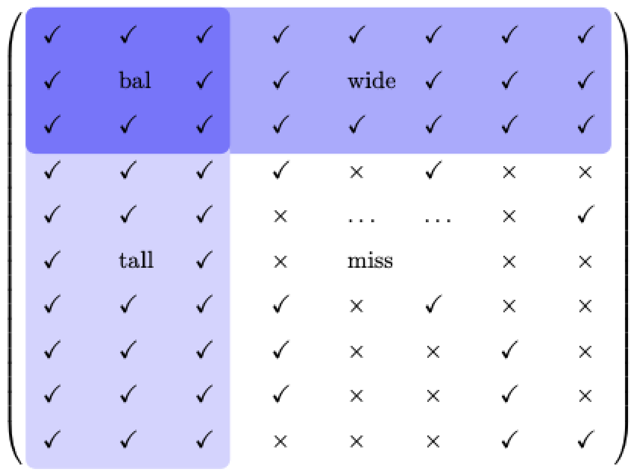
There are generally two solutions to the problem of missing data: Amputation and Imputation. In most empirical studies, amputation is the default method. Typical approaches are either removing the rows of data that contains missing values or the variables with missing values. Imputation is more sophisticated, and fills in missing values by model estimates.

Numerous methods are proposed to deal with missing data by researchers from different communities: Econometrics, Statistics and Machine Learning. Even though most of methods rely on the assumption of low rank matrices, they have distinct differences in theoretical results. Computer scientists focus more on the accuracy algorithm shows in experiments, while statistician and econometrician analyze the finite or large sample properties of the method. This difference leads to the development of different types of methods. In this paper, we are more interested in comparing and analyzing methods from econometrics literature. That said, we also implement some of the benchmark algorithms from the Machine Learning community and compare them with econometric methods. In the following subsetions, we first go through several recent work from econometrics community, which are all based on the framework of factor models.

2.1 The Tall-Wide Estimator

Bai and Ng (2021) proposes an imputation procedure that takes advantage of strong factor structure of the observation matrices. By re-arranging the rows and columns with missing entries, all missing entries can be moved to the lower-right block of the data matrix. Then, factor structure

Figure 1: The Tall-Wide Block Representation of Data



of the entire data matrix can be recovered using the data from the tall and wide blocks only. Their idea is to estimate the factors and loadings by imposing factor structure intead of missing data pattern. Let $T_o \times N_o$ denote the dimensions of the upper-left block BAL, and $T_m = T - T_o$, $N_m = N - N_o$. Thus, the missing block is a $T_m \times N_m$ matrix. Their estimator is based on the idea that extracts factors and loadings from TALL $(\tilde{F}_{\text{tall}}, \tilde{\Lambda}_{\text{tall}})$, and from WIDE $(\tilde{F}_{\text{wide}}, \tilde{\Lambda}_{\text{wide}})$ by asymptotic principle components (APC), then combing $\tilde{F}_{\text{tall}}$ and $\tilde{\Lambda}_{\text{wide}}$ to form the entire factor structure. The detailed steps of algorithm is given as follows.

1. Pre-process data: demean, standardization and etc.

2. From the TALL block of X, obtain $(\tilde{F}_{\text{tall}}, \tilde{\Lambda}_{\text{tall}})$ by APC where $\tilde{F}_{\text{tall}}$ is $T \times r$ and $\tilde{\Lambda}_{\text{tall}}$ is $N_0 \times r$.
3. From the WIDE block of X, obtain $(\tilde{F}_{\text{wide}}, \tilde{\Lambda}_{\text{wide}})$ by APC where $\tilde{\Lambda}_{\text{wide}}$ is $N \times r$ and $\tilde{F}_{\text{wide}}$ is $T_0 \times r$.
4. Regress $\tilde{\Lambda}_{\text{tall}}$ on the $N_0 \times r$ submatrix of $\tilde{\Lambda}_{\text{wide}}$ and get $\tilde{H}_{\text{miss}}$.
5. Output $\tilde{X}_{it} = \begin{cases} X_{it} & (i,t) \text{ observed} \\ \tilde{C}_{it} = \tilde{F}_{\text{tal}} \tilde{H}_{\text{miss}} \, \tilde{\Lambda}'_{\text{wide}} & (i,t) \text{ missing} \end{cases}$

In additional to conventional factor model assumptions, the method imposes order conditions on T_o, N_o that $T_o, N_o, T, N \to \infty$.

$$\frac{\sqrt{N}}{\min\{N_o, T_o\}} \to 0 \text{ and } \frac{\sqrt{T}}{\min\{N_o, T_o\}} \to 0$$

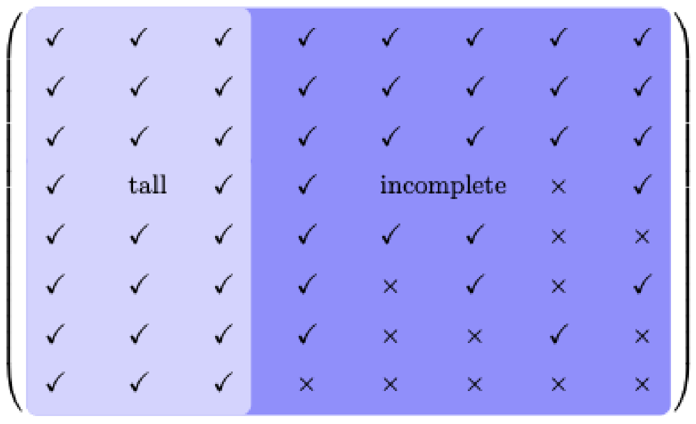
The assumption allows N_o, T_o grows in a slower rate than N, T, but requires it grows faster than $\sqrt{N}, \sqrt{T}$. Then, it imposes a strong factor assumption on sub-block matrix, which requires the sub-sample moment matrices for the factor and factor loadings to be positive definite so that the factors from the subsamples are identifiable. Finally, the method requires block stationarity, i.e. subsample limits same as full sample. Given the assumptions, Bai and Ng (2021) provides asymptotic normality results for each element of the factor estimator.

2.2 The Tall-Project Estimator

In the previous section, we learn that the TP algorithm requires pre-processing and organizing data into blocks, and then utilize the factor estimates from TALL and WIDE blocks to recover the factor structure for entire data matrix. However, the algorithm doesn't use all the observed data as the observed entries in MISSING block are not used at all.

Therefore, following Bai and Ng (2021), Cahan, Bai and Ng (2021) proposes a projection based method that takes advantage of all the observed data. One of the distinct feature of TW is that delivers consistent and asymptotically normal estimates without iteration. First of all, we need to sort and put fully observed columns in TALL block, but unlike TP arranging the rows is not needed. Then, the data matrix becomes the following. The goal is similar to TW, which is to

Figure 2: The Tall-Incomplete Representation of the Data



recover the factors from tall block. The difference is that, we estimate factor loadings for each unit by regressions with all observed data in INCOMPLETE. Exact steps are given by:

1. Estimate the $T \times r$ matrix $\tilde{F}_{\text{tall}}$ from the tall block;
2. For $i \in [1, N]$, regress the $T_{o_i} \times 1$ vector X_{o_i} on the $T_{o_i} \times r$ matrix of factor estimates $\tilde{F}_{\text{tall}i}$ to obtain $\tilde{\Lambda}_i$;
3. Let $\tilde{C}_{it} = \tilde{F}'_{\text{tall},t}\tilde{\lambda}_i$ and output $\tilde{X}_{it} = \begin{cases} X_{it} & (i,t) \text{ observed} \\ \widetilde{C}_{it} & (i,t) \text{ missing} \end{cases}$

The X_{o_i} stands for the vector of observed entries of i-th unit, and T_{o_i} is the corresponding subvector. In additional to the order assumption in TW, TP imposes an extra assumption on order to ensure boundedness that $N_t \geq N_o$ for all t, and there exists an M such that $0 < M < \infty$, for all N,

$$\frac{N - N_t}{N_o} \leq M, \quad \frac{N_t}{N} \to \tau_t \in (0,1).$$

2.3 Weighted PCA by Xiong and Pelger (2020)

The two methods discussed above both rely more or less on block-wise structure. Their estimations are based on the idea of using observations of sub-blocks to extract information about the entire matrix. Xiong and Pelger (2020) proposes a weighted PCA method that accepts more flexible missing patterns. For example, if the missing patter is given as follows, then the previous methods wouldn't work well as the tall block becomes too narrow.

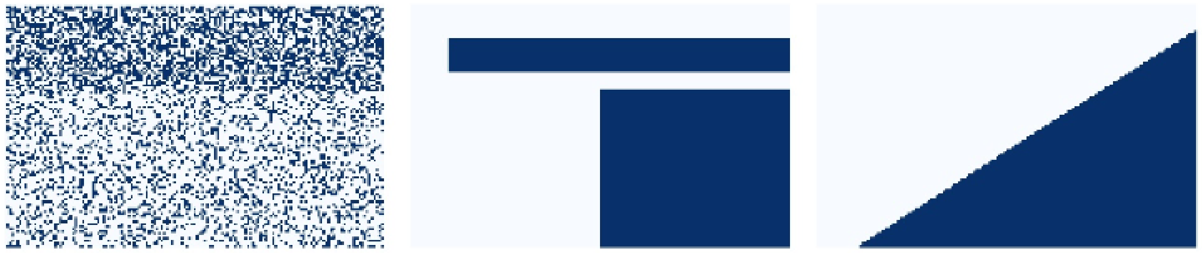
$$X = \left(\begin{array}{c|cccc} \omega_{11} & x_{12} & x_{13} & x_{14} & * \\ x_{21} & x_{22} & x_{23} & * & x_{25} \\ x_{31} & x_{32} & * & x_{34} & x_{35} \\ x_{41} & x_{42} & * & x_{44} & x_{45} \\ x_{51} & x_{52} & x_{53} & * & x_{55} \\ x_{61} & x_{62} & * & * & * \\ x_{71} & * & x_{73} & x_{74} & x_{75} \\ x_{81} & * & x_{83} & x_{84} & x_{85} \end{array}\right)$$

In Xiong and Pelger (2020), very general missing patterns are allowed. They assume independence between missing patterns and factors or observational errors, but allow dependence between missing patterns and factor loadings. Since the factor model is "symmetric" in N and T, we can switch factors and factor loadings, so that missing patterns can depend on factor loadings but not factors. Their framework include not but limited to the following three popular missing patterns.

Their strategy is to first use all observations available to estimate the covariance matrix, and then run a weighted PCA. The steps are given as follows.

1. Estimate each entries of the covariance matrix by using the time periods when both units are observed.
$$\tilde{\Sigma}_{ij} = \frac{1}{|\mathcal{Q}_{ij}|} \sum_{t \in \mathcal{Q}_{ij}} Y_{it} Y_{jt}$$
where $\mathcal{Q}_{ij} = \{t : W_{it} = 1 \text{ and } W_{jt} = 1\}$.

Figure 3: Examples of patterns for missing observations


(a) Randomly missing (b) Simultaneous treatment adoption (c) Staggered treatment adoption

2. Estimate the loadings $\tilde{\Lambda}$ as $\sqrt{N}$ times the eigenvectors of the r largest eigenvalues of the sample covariance matrix, that is

$$\frac{1}{N}\tilde{\Sigma}\tilde{\Lambda} = \tilde{\Lambda}\tilde{V}$$

where $\tilde{V}$ is a diagonal matrix.

3. For every time period t, we regress Y_t on $\tilde{\Lambda}$ to estimate the factors:

$$\tilde{F}_t = \left(\sum_{i=1}^{N} W_{it}\tilde{\Lambda}_i\tilde{\Lambda}_i^T\right)^{-1}\left(\sum_{i=1}^{N} W_{it}\tilde{\Lambda}_i Y_{it}\right)$$

Their framework also accepts information about the missing patterns. If given some features S_i, $P(W_{it} = 1 \mid S_i)$ is known or well-estimated, they propose a less efficient but more robust propensity weighted method.

$$\tilde{F}_t^S = \left(\sum_{i=1}^{N} \frac{W_{it}}{P(W_{it} = 1 \mid S_i)}\tilde{\Lambda}_i\tilde{\Lambda}_i^T\right)^{-1}\left(\sum_{i=1}^{N} \frac{W_{it}}{P(W_{it} = 1 \mid S_i)}\tilde{\Lambda}_i Y_{it}\right)$$

Though being able to handle more flexible missing patterns, Pelger and Xiong (2020) requires computing Eigenvalue Decomposition (EVD) of covariance matrix, which is discouraged in conventional PCA algorithm. Due to numerical issues, we are always supposed to apply SVD to the original data matrix to extract factor and factor loadings.

2.4 Nuclear Normal Matrix Completion

In the statistics community, various methods were proposed to complete matrices with missing entries. The idea is still to fit a low rank matrix to the data, but it allows the latent matrix to have more general structures. There are generally many ways to impose low rank structure on latent matrices. Most popular approaches rely on matrix norms for regularization. One of the popular choice of matrix norm is Nuclear Norm, which is defined by:

$$\|M\|_* = \sum_{i=1}^{N} \sigma_i(M)$$

where $\sigma_i(M)$ is the i-th singular values of matrix X. A similar structure is assumed for observed data matrix in Equation (1). The estimator is given as the solution to the following optimization problem.

$$\hat{M} = \arg\min \frac{1}{\sum_{i,t} \Omega_{it}} \sum_{i,t} \Omega_{it}(Y_{it} - M_{it})^2 + \lambda \|M\|_* \tag{3}$$

This approach was recently adopted by econometrics community as well for estimating causal effect in panel data (see Athey and etc (2021)). With reasonable regularity conditions, they provide a upper bound for root-mean-squared-error (RMSE) of $\hat{M}$. In order to address the computational complexity incurred by a direct application of convex optimization algorithm, they propose a shrinkage algorithm. The original work by Cai, Candes and Shen (2010) proposed the Singular Value Thresholding algorithm for approximately solving the nuclear norm minimization problem, which is the tightest convex relaxation of the NP-hard rank minimization problem given below.

$$\begin{array}{ll} \text{minimize} & \|\boldsymbol{M}\|_* \\ \text{subject to} & \mathcal{P}_\Omega(\boldsymbol{M}) = \mathcal{P}_\Omega(\boldsymbol{Y}) \end{array} \tag{4}$$

with optimization variable $M \in \mathbb{R}^{N\times T}$. Fix $\tau > 0$ and a sequence $\{\delta_k\}_{k\geq 1}$ of scalar step sizes. Then starting with $\boldsymbol{Y}^0 = \mathbf{0} \in \mathbb{R}^{N\times T}$, the algorithm inductively defines

$$\left\{ \begin{array}{l} \boldsymbol{M}^k = \operatorname{shrink}\left(\boldsymbol{X}^{k-1}, \tau\right) \\ \boldsymbol{X}^k = \boldsymbol{X}^{k-1} + \delta_k \mathcal{P}_\Omega\left(\boldsymbol{Y} - \boldsymbol{M}^k\right) \end{array} \right.$$

until a stopping criterion is reached.

2.5 Collaborative Filtering and Latent Factor Models

Matrix completion or imputing missing observations is an essential problem in recommendation systems. In the famous Netflix challenge, many algorithms have been developed to learn the unobserved user ratings on movies. Two of the most basic and popular method are collaborative filtering and latent factor models. The latter has similar ideas with the previous methods, while the former method is more based on logic instead of statistical modeling.

In collaborative filtering, the goal is to find similar users or movies by computing some similarity metrics, and then to estimate the ratings by using the ratings from those users or movies. Popular similarity measures include Jaccard, Cosine measure and Pearson correlation coefficient.

$$\operatorname{sim}(\mathbf{x}, \mathbf{y}) = \cos(r_{\mathbf{x}}, r_{\mathbf{y}}) = \frac{r_{\mathbf{x}} \cdot r_{\mathbf{y}}}{\|r_{\mathbf{x}}\| \cdot \|r_{\mathbf{y}}\|} \text{ and } \operatorname{sim}(x, y) = \frac{\sum_{s\in S_{xy}} (r_{xs} - \bar{r}_x)(r_{ys} - \bar{r}_y)}{\sqrt{\sum_{s\in S_{xy}} (r_{xs} - \overline{r_x})^2} \sqrt{\sum_{s\in S_{xy}} (r_{ys} - \overline{r_y})^2}}$$

The common practice of collaborative filtering is:

1. Define similarity s_{ij} of items i and j (could be either row or columns in the data).
2. Select k nearest neighbors $\mathcal{N}_i$ that are observed.
3. Compute the basline estimate $\hat{b}_{it} = \hat{\mu} + \hat{b}_t + \hat{b}_i$ where

$$\hat{\mu} = \frac{1}{\sum_{i,t} \Omega_{it}} \sum_{i,t} \Omega_{it} Y_{it}, \quad \hat{b}_t = \frac{1}{\sum_i \Omega_{it}} \sum_i \Omega_{it} Y_{it}, \quad \hat{b}_i = \frac{1}{\sum_t \Omega_{it}} \sum_t \Omega_{it} Y_{it}$$

4. Estimate the missing entries $\hat{Y}_{it}$ as the weighted average:

$$\hat{Y}_{it} = \hat{b}_{it} + \frac{\sum_{j \in \mathcal{N}_i} s_{ij} \cdot (Y_{jt} - \hat{b}_{jt})}{\sum_{j \in \mathcal{N}_i} s_{ij}}$$

The latent factor model approach assumes a low rank structure of the data matrix. One of the popular model assumes the latent matrix is a product of two matrices P, Q. We optimize

$$\min_{P,Q} \frac{1}{\sum_{i,t} \Omega_{it}} \sum_{i,t} \Omega_{it}(Y_{ij} - Q_i P_t)^2 + \lambda_1 \sum_t \|P_t\|^2 + \lambda_2 \sum_i \|Q_i\|^2.$$

No analytical solutions are available for the problem. The modern practice is Stochastic Gradient Descend (SGD), see Algorithm 1. In most cases, theoretical results are not available for algorithms proposed in the literature of recommendation system.

Algorithm 1: Stochastic Gradient Descent

```
Initialization P and Q (using SVD, pretend missing ratings are 0);
while $i <$ MAX_EPOCH do
    for $i$=1:N, $t$=t:N do
        $\epsilon_{it} = 2(Y_{it} - Q_i P_t)$;
        $Q_i \leftarrow Q_i + \mu_1(\epsilon_{it} P_t - 2\lambda_2 Q_i)$;
        $P_i \leftarrow P_i + \mu_2(\epsilon_{it} Q_t - 2\lambda_1 P_i)$;
    end
    $i \leftarrow i + 1$;
end
```

3 Simulation

In this section, we conduct multiple simulation studies to analyze the performance of each methods. We want to verify some claims made by the authors, and also the idea whether methods with stronger theoretical results tend to perform worse empirically.

3.1 Model Setup

In most of the approaches mentioned in the previous section, latent factor models are assumed. Thus, we keep our simulation model consistent with this assumption. However, we want to test all methods under different the missing patterns and dimensions in order to find out the best methods in each use cases. First, start with a factor model.

$$\tilde{X} = F\Lambda^T + \epsilon$$

where $\tilde{X}$ is $N \times T$, F is $N \times r$, and λ is $T \times r$. Standard Gaussian distribution is assumed for all components, i.e. $F_t \sim N(0, I_r)$, $\Lambda_i \sim N(0, I_r)$ and $\epsilon_{it} \in N(0, 1)$. For simplicity, we assume $F_t, \Lambda_i, \epsilon_{it}$ are independent for all i, t. Then, we consider the following missing patterns.

1. *Lower-Right Block Only:* Given N_o, T_o, draw $\Omega_m \in \{0, 1\}^{(N-N_o)\times(T-T_o)}$ by i.i.d Bernoulli distribution with probability p, and set the rest of Ω to be 1, i.e. non-missing. In this case, only the lower-right block have missing entries with probability p.

2. *Right Block Only:* Given N_o, draw $\Omega_m \in \{0,1\}^{N\times(T-T_o)}$ by i.i.d Bernoulli distribution with probability p, and set the rest of Ω to be 1.

3. *Completely random:* Draw $\Omega_m \in \{0,1\}^{N\times T}$ by i.i.d Bernoulli distribution with probability p.

For each missing pattern models, we general problems with different dimensions and different number of missing values.

3.2 Results

The criterion used to determine the performance of each algorithms is out-of-sample Mean Squared Error (MSE).

$$\text{MSE} = \frac{1}{\sum_{i,t} 1 - \Omega_{it}} \sum_{i,t} (1 - \Omega_{it})(Y_{it} - \hat{Y}_{it})^2$$

First, start with *Lower-Right Block Only.* We choose the dimension of the entire data matrix as $N = 200, T = 100$, and the number of factors to be $r = 3$. Then, we pick 5 different combinations of N_o, T_o: $(20, 10)$, $(50, 25)$, $(100, 50)$, $(100, 75)$, $(150, 75)$.

Table 1: Out-of-sample MSE for different size of missing blocks

(N_o, T_o)	TP	TW	Weighted PCA	Matrix Completion	Latent Factor Model
(20, 10)	5.940	1.480	1.117	3.866	4.733
(50, 25)	3.416	1.146	1.088	3.901	4.573
(100, 50)	2.953	1.092	1.072	3.797	4.839
(100, 75)	2.840	1.079	1.077	3.938	4.927
(150, 75)	1.567	1.087	1.063	3.909	4.805

From Table 1, we found that Weighted PCA has the best performance even in the *Lower-Right Block Only* setting, where TP and TW are proposed. Among the five methods, Matrix Completion and Latent Factor Model unsurprisingly perform the worst as they don't leverage the underlying factor structures or missing patterns. By reading the MSE of TP, we verify its asymptotic properties as the Tall blocks gets wider, the accuracy of TP becomes better. Similar trend happens for TW as well, but TW shows better convergence as it uses all the data in the lower-right block. In addition, we explore the effect of different likelihood of missing entries by generating Ω with different probability p. Table 2 shows that as long as enough observations for fitting the algorithms, chance of getting missing entries $1 - p$ doesn't make too much difference in all five methods. One potential reason could be that all methods work well enough with $\frac{3}{4}$ of observed entries, so by varying the percentage of missing values in the $\frac{1}{4}$ lower-right block doesn't make too much difference.

Table 2: Out-of-sample MSE for different probability of missingness p

p	TP	TW	Weighted PCA	Matrix Completion	Latent Factor Model
0.1	3.05695	1.11487	1.06928	4.17165	4.60743
0.3	3.53925	1.10695	1.06725	4.10571	4.61513
0.5	2.91883	1.09852	1.06643	3.97384	4.66605
0.7	3.05724	1.09975	1.08974	4.08565	4.42117
0.9	3.97864	1.10545	1.09419	3.94499	4.53348

Then, we want to see how the number of latent factors affect the results. We run simulations with different number of factors for data generating process $r = 3, 5, 10, 20, 30$. The model has

$N = 200, T = 100, N_o = 100, T_o = 50$. Table 6 shows the results. When r is large, it is similar to say that the number of observations available to estimate the factor structure is relatively small. Thus, we find that TW and Weighted PCA are less susceptible to data insufficiency. Interestingly, when r becomes large enough, the edge of Weighted PCA over TW diminishes. This is consistent with the arguments made by Pelger and Xiong (2020) that TW is a most efficient method under the block-wise missing patterns, while their method has close performance but accepts general missing patterns.

We then investigate the setting where TP no longer works, i.e. Figure 2. The result in Table 5 shows that TW algorithm performs the best when there is enough data in TALL block, meaning that T_o is large. At the same time, Weighted PCA is still the overall best method due to efficient way of using the data and leveraging the factor structure.

Table 3: Out-of-sample MSE for Completely Random Missing Data

N, T	Weighted PCA	Matrix Completion	Latent Factor Model
(50, 20)	3.40582	4.19273	7.19311
(100, 50)	1.36613	4.12333	6.44007
(200, 100)	1.1675	3.94935	5.5129
p			
0.7	1.36054	3.97542	6.06873
0.8	1.73699	3.87792	6.46254
0.9	8.35783	4.08056	7.05677

Finally, we look into the third missing pattern, *Completely random*. In this setup, both TP and TW don't work, so we can only compare the other three methods. The first three rows in Table 3 verifies the previous simulation results that when number of observations increases, all methods perform better. Weighted PCA benefit more from the increase of matrix dimensions than the two machine learning approach, and also outperforms the other two approaches. In the next three rows, we explore how sparsity affects the performance of each methods. We find that when p is large enough, meaning that there are many missing entries in the data, the two machine learning methods perform better than Weighted PCA. Apparently, when dealing with missing data, it is very unlikely that 70 percent or more entries are missing. However, in the problem of recommendation system, sparsity could be even higher than 90%.

To sum up, the Weighted PCA is overall best-performing imputation method under the factor model based panel data setting in terms of the out-of-sample MSE. That said, in the Tall-Incomplete setting, TW performs marginally better than Weighted PCA, an argument agreed upon by both Pelger and Xiong (2020) and Bai and Ng (2021). However, when the data matrix has high sparsity, Matrix Completion becomes the best performing algorithm. Therefore, we recommend using Weighted PCA for missing data imputation, and Matrix Completion or other Machine Learning method for recommendation systems, i.e. high sparsity.

4 Empirical analysis

In this section, we compare the usefulness of the four methods, not including TP, by considering factor-augmented regressions to forecast macroeconomic variables. The reason why we leave out TP is that it requires re-arranging the order of time series dimension, which doesn't come along with a forecasting problem. The forecasting procedure starts with estimating the latent factor, followed by predictive regression using estimated factors. When there exists missing values in the panel data, the usual practice is amputation, which deletes rows and columns with missing values. The PC estimation of factors uses balanced panel after amputation. Inevitably, we lose some useful information by doing so. All of the imputation methods are supposed to use data more efficiently, and the performance of the prediction task should reflect how well the imputation method works.

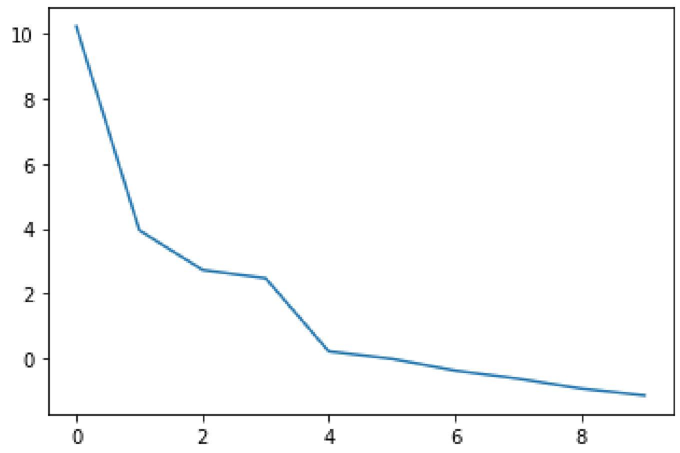
We use a panel dataset FRED-MD, a publicly accessible large macroeconomic databases designed for the empirical analysis of "big data". McCraken and Ng (2015) offer detailed descriptions of the database and conducted some preliminary empirical analysis to showcase the usefulness of the dataset, including extracting factors for macroeconomic prediction. In this paper, we used the most updated monthly series "current.csv" from their website. [1] And we processed each macroeconomic variable using the method suggested by the author, i.e. the transform code given in the second row of the data, with the help of their published MATLAB code. After processing, the unbalanced panel consists of 128 monthly macroeconomic variables from March 1959 to March 2020, a total number 733 observations. Following the lead of McCraken and Ng (2015), we check for outliers in each variable where an outlier is defined as an observation that deviates from the observed sample median by more than 10 times interquantile range. Then outliers are removed and treated as missing observations. In the end, there are 1073 total number of missing values. The amputation method deletes 38 variables with more than one missing entries, and leaves 90 variables for factor estimation.

In the empircal study, we consider the forecasts of IP Index (INDPRO), Unemployment Rate (UNRATE) and Consumer Price Index (CPIAUCSL), all of which have full observations in the entire sample. The factor-augmented autoregression (FA-AR) models are given as follows.

$$y_{t+1} = \phi_0 + \phi_1(L)\hat{F}_t + \phi_2(L)y_t + \varepsilon_{t+1}$$

where y_t is one of the three macro-variables (i.e., INDPRO, UNRATE and CPIUCSL), $\hat{F}_t$ is the estimated vector of factors, ϕ_0 is the intercept term, L is the lag operator, $\phi_1(L)$ and $\phi_2(L)$ are finite-order polynomials of the lag operators. We select $L = 4$ for all of the three approaches, and three out-of-sample splits. The number of factors are determined by visualization, i.e. by reading the plot of explained variance ratio. Figure 4 shows that the first principle component explains most

Figure 4: Logarithm of explained variance with each eigenvalues



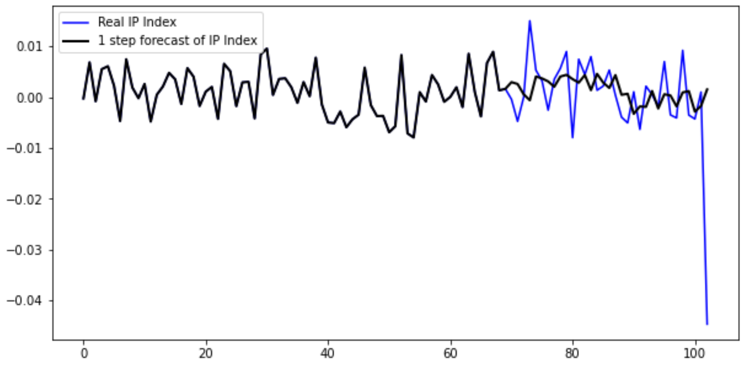
of variations. In fact, the explained variance ratio is more than 90 percent. Since the explained variance drops significant after the third eigenvalue, we choose $r = 3$ for our forecast problem.

[1] https://research.stlouisfed.org/econ/mccracken/fred-databases/

When obtaining the plot, we both run PCA on the data matrix after amputation and EVD on the estimated covariance matrix from Weighted PCA method. We find that both approach yields the same results.

In our main results, we compare three missing data imputation methods discussed in the previous section: TP, Weighted PCA and Nuclear Norm Matrix Completion. Both TW and Latent factor models are omitted here. First of all, we believe that TW is not suitable for macroeconomic forecast problem since it requires re-arranging the time observations across time. In terms of Latent Factor Model, we remove it from the empirical section because it shares similar properties with Matrix Completion, but is more computationally expensive. Besides, it has more tuning parameters. Among the three method, Nuclear Norm Matrix Completion is the slowest and requires an additional PCA step to estimate the factors, as it estimates the latent matrix directly without assuming factor structures. Both TP and Weighted PCA estimate the factor structure first, and then output the imputedm matrix. Since authors of both approaches claim that the estimated

Figure 5: Real IP vs Forecast with Amputation



factor is consistent and enjoys good asymptotic properties, we didn't re-estimate the factors based on the imputed matrices. Here, we only present the results of 1-step out-of-sample forecast. Figure 5 shows the benchmark forecast results of Industrial Production Index. We can see that at the end of the series, the IP index dropped significantly. It is due to the COVID-19 pandemic. Apart from that, most of the forecasts are good.

On basis of that, we apply all fours methods to forecast the three macroeconomic variables under three different windows of out-of-sample period, see Table 4. The first result is that TP performs almost identical to amputation, which is not surprising at all. According to the estimation procedure of TP, factor estimation only uses the data in TALL block, which is basically the data matrix after amputation, i.e. removing the variables or columns with missing values. Surprisingly, Weighted PCA, the overall best-performing algorithm in simulation, performs the worst in most cases. We believe that the potential cause lies in the fact of using EVD instead of SVD for factor estimation. As mentioned before, EVD based factor estimation leads to bad numerical performance, as the computation of covariance matrix and the following EVD step both suffer from numerical issues. The all-time best performing algorithm now becomes Matrix Completion. Apparently, the reason is that Matrix Completion utilizes more observations than TP and Amputation, and suffers no numerical issues.

Table 4: Auto-regression Forecast with different Macro Variables and Methods

IP Index				
	RMSE	Ratio		
	Amputation	Tall-Projection	Weighted PCA	Matrix Completion
July 2017-Mar 2020	9.131e-5	1	1.026	0.969
May 2013-Mar 2020	4.969e-5	1	1.103	0.992
Mar 2009-Mar 2020	4.080e-5	1	1.061	0.980
Unemployment Rate				
	RMSE	Ratio		
	Amputation	Tall-Projection	Weighted PCA	Matrix Completion
July 2017-Mar 2020	0.04553	1	1.0123	0.951
May 2013-Mar 2020	0.03044	1	1.027	0.972
Mar 2009-Mar 2020	0.02889	1	0.978	0.947
CPI				
	RMSE	Ratio		
	Amputation	Tall-Projection	Weighted PCA	Matrix Completion
July 2017-Mar 2020	3.613e-6	1	1.010	0.876
May 2013-Mar 2020	3.667e-6	1	0.998	0.928
Mar 2009-Mar 2020	4.702e-6	1	1.012	0.931

5 Conclusion

This paper compares five missing data imputation methods: TW, TP, Weigthed PCA, Matrix Completion and Latent Factor Model. We discuss their assumptions and estimation procedures, while laying out advantages and disadvantages of some methods. Through various simulation studies, we conclude that Weighted PCA is the best-performing algorithm in terms of out-of-sample MSE in missing data imputation. However, in the empirical section, we find that Weighted PCA is no longer the best when being used to estimating factors. We suspect the cause is its use of EVD instead of SVD for factor estimation. Standard PCA on the imputed data with Matrix Completion stands out in the forecasting problem.

References

[1] Jushan Bai, Serena Ng 2021. *Factor-Based Imputation of Missing Values and Covariances in Panel Data of Large Dimensions.* arXiv:1910.06677v4 [econ.EM] 4 Mar 2021.

[2] Ercument Cahan, Jushan Bai, Serena Ng 2021. *Factor-Based Imputation of Missing Values and Covariances in Panel Data of Large Dimensions.* arXiv:2103.03045v1 [econ.EM] 4 Mar 2021

[3] Xiong, Ruoxuan and Pelger, Markus, *Large Dimensional Latent Factor Modeling with Missing Observations and Applications to Causal Inference* (November 8, 2020). Available at SSRN: https://ssrn.com/abstract=3465357 or http://dx.doi.org/10.2139/ssrn.3465357

[4] Sainan Jin, Ke Miao, Liangjun Su (2021) *On factor models with random missing: EM estimation, inference, and cross validation.* Journal of Econometrics 222 (2021) 745–777

[5] Susan Athey, Mohsen Bayati, Nikolay Doudchenko, Guido Imbens, Khashayar Khosravi *Matrix Completion Methods for Causal Panel Data Models.* arXiv:1710.10251v4 [math.ST] 23 Feb 2021

[6] Emmanuel Candès and Benjamin Recht. 2012. *Exact matrix completion via convex optimization.* Commun. ACM 55, 6 (June 2012), 111-119.

[7] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. 2010. *A Singular Value Thresholding Algorithm for Matrix Completion.* SIAM J. on Optimization 20, 4 (March 2010), 1956-1982. DOI=http://dx.doi.org/10.1137/080738970

[8] KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data miningJune 2009 Pages 447–456https://doi.org/10.1145/1557019.1557072

Appendix

Table 5: Out-of-sample MSE for different T_o where $N_o = 0$

T_o	TW	Weighted PCA	Matrix Completion	Latent Factor Model
10	1.46163	1.16088	4.10211	5.24816
25	1.19855	1.1573	4.17485	4.85445
50	1.12186	1.14804	4.19006	4.5926
75	1.07938	1.11302	3.73653	4.92703
90	1.07029	1.10844	3.63128	4.42292

Table 6: Out-of-sample MSE for different number of factors r

r	TP	TW	Weighted PCA	Matrix Completion	Latent Factor Model
3	2.76400	1.09345	1.07155	3.92781	4.56066
5	7.72883	1.18991	1.15037	6.01531	4.21347
10	14.51624	1.37185	1.36280	11.40516	5.37027
20	37.13129	2.03497	2.21119	20.79150	12.41589
30	56.67683	3.19361	3.94989	31.30469	41.17405