Scientific Research Publishing

Journal of Data Analysis and Information Processing, 2024, 12, 115-137
https://www.scirp.org/journal/jdaip
ISSN Online: 2327-7203
ISSN Print: 2327-7211

# Non-Linear Matrix Completion

**Fengrui Zhang, Randy C. Paffenroth, David Worth**

Worcester Polytechnic Institute, Worcester, MA, USA
Email: zhangfengrui95@gmail.com

**How to cite this paper:** Zhang, F.R., Paffenroth, R.C. and Worth, D. (2024) Non-Linear Matrix Completion. *Journal of Data Analysis and Information Processing*, **12**, 115-137.
https://doi.org/10.4236/jdaip.2024.121007

**Received:** January 23, 2024
**Accepted:** February 26, 2024
**Published:** February 29, 2024

## Abstract

Current methods for predicting missing values in datasets often rely on simplistic approaches such as taking median value of attributes, limiting their applicability. Real-world observations can be diverse, taking stock price as example, ranging from prices post-IPO to values before a company's collapse, or instances where certain data points are missing due to stock suspension. In this paper, we propose a novel approach using Nonlinear Matrix Completion (NIMC) and Deep Matrix Completion (DIMC) to predict associations, and conduct experiment on financial data between dates and stocks. Our method leverages various types of stock observations to capture latent factors explaining the observed date-stock associations. Notably, our approach is nonlinear, making it suitable for datasets with nonlinear structures, such as the Russell 3000. Unlike traditional methods that may suffer from information loss, NIMC and DIMC maintain nearly complete information, especially in high-dimensional parameters. We compared our approach with state-of-the-art linear methods, including Inductive Matrix Completion, Nonlinear Inductive Matrix Completion, and Deep Inductive Matrix Completion. Our findings show that the nonlinear matrix completion method is particularly effective for handling nonlinear structured data, as exemplified by the Russell 3000. Additionally, we validate the information loss of the three methods across different dimensionalities.

## Keywords

Matrix Completion, Data Pipeline, Machine Learning

## 1. Introduction

Data imputation, a crucial step in data preprocessing, involves replacing missing values with substitutes. Traditional methods often resort to discarding columns with significant gaps or replacing missing values with column means, resulting in substantial information loss.

The significance of data imputation gained prominence with the Netflix Prize in 2013, where Netflix sought a solution to the challenge of processing missing values in its dataset to construct a robust recommendation system. This global initiative, offering a $100 million reward, underscored the importance of addressing missing data issues in diverse datasets, such as Netflix's user ratings and movie items.

Furthermore, the field of biology [1] faced a high demand for predicting missing values, leading to the development of innovative methods. An example is the Inductive Matrix Completion approach proposed by the University of Texas at Austin, which reconstructs the original dataset P through the product of parameters W and factors H. This inductive method proves effective for datasets with largely empty fields and new items with limited previous observations, although it primarily addresses linear data structures.

Building on the concept of inductive matrix completion, we introduce a nonlinear version called Nonlinear Inductive Matrix Completion (NIMC). Additionally, drawing inspiration from neural network structures, we present a deep version known as Deep Inductive Matrix Completion (DIMC). This project aims to investigate the performance of these three advanced data imputation methods with optimal parameters, evaluating them in both synthetic and real-world datasets, including the Russell 3000.

Our approach involves three key steps. First, we formulate the problem and construct the three matrix completion methods, detailing the mathematical aspects in the optimization section. Next, we assess the performance and determine optimal parameters on synthetic data. Finally, we compare the methods' performance on the real-world dataset Russell 3000, analyzing information loss across different dimensionalities.

In conclusion, while Inductive Matrix Completion IMC, NIMC, and DIMC effectively handle linear low-rank synthetic data, IMC struggles with nonlinear data structures. NIMC and DIMC outperform IMC in handling nonlinear data structures, albeit with a higher information loss in low-dimensional settings. IMC demonstrates versatility in managing both low and high-dimensional data but lacks sensitivity to parameter dimensions. Real-world experiments on Russell 3000 highlight NIMC's superior performance in handling partial linear and partial nonlinear data structures. Cross-missing value data proves challenging for all three methods, indicating the need for increased observations to enhance performance.

The analysis emphasizes the importance of understanding the dataset structure when choosing an imputation method [2], especially for inductive matrix completion approaches. This prompts a potential future research avenue: data structure identification.

## 2. Methodology

### 2.1. Inductive Matrix Completion Method [3]

The inductive method reconstructs original dataset $P$ by parameter $W \times H$. Once the reconstructed data and training data close enough, the reconstructed data also can automatically close to the missing value by gradient descent. First of all, the missing value matrix $P \in R^{m\times n}$. Given a sample of observed entries Ω from a true underlying matrix $M \in R^{m\times n}$, the goal is to estimate missing entries under additional assumptions on the structure of the matrix. The most common assumption is that the matrix is low-rank, *i.e.*, $M = W \times H^{\mathrm{T}}$, as shown in **Figure 1**.

We could solve the following optimization problem:

$$\min_{W\in R^{m\times k}, H\in R^{m\times k}} \sum \left(P - W^{\mathrm{T}} H\right)^2 + \frac{1}{2}\lambda\left(\|W\|_F^2 + \|H\|_F^2\right),$$

where $\lambda$ is a regularization parameter, $W$ and $H$ denote the parameters to reconstruct. We want to learn factors $W \in R^{m\times k}$ and $H \in R^{n\times k}$ such that the estimated values are close to the observed entries, and the $WH^{\mathrm{T}}$ is low rank. This problem can be represented as following figure:

## 2.2. Nonlinear Inductive Matrix Completion

Inspired by linear version IMC, we come up with nonlinear version called Nonlinear Inductive Matrix Completion. The main idea is adding nonlinear function into each parameter. Consider Inductive Matrix Completion method with nonlinear activation function with following optimization problem [4]:

$$\min_{W\in R^{m\times k}, H\in R^{m\times k}} \sum \left(P - \phi\left(W^{\mathrm{T}}\right)\phi(H)\right)^2 + \frac{1}{2}\lambda\left(\|W\|_F^2 + \|H\|_F^2\right),$$

Here function $\phi$ can be any nonlinear function such as sigmoid function, ReLU function, Tanh function. When function $\phi$ output is input itself, Nonlinear IMC will automatically become a linear version inductive matrix completion.

## 2.3. Deep Inductive Matrix Completion

Inspired by one layer neural network [5], we introduce multiple layers structure, we come up with deep version called Deep Inductive Matrix Completion. The main idea is adding multiple parameters layers. As Shown in **Figure 2**.

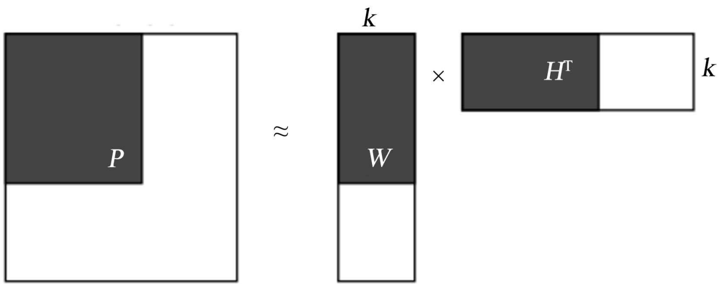


**Figure 1.** IMC method reconstructs original dataset $P$ by parameter $W$ times $H$. Once the reconstructed data and training data close enough, the reconstructed data can automatically close to the missing value. First of all, the missing value matrix $P \in R^{m\times n}$. Given a sample of observed entries Ω from a true underlying matrix $M \in R^{m\times n}$, the goal is to estimate missing entries under additional assumptions on the structure of the matrix.

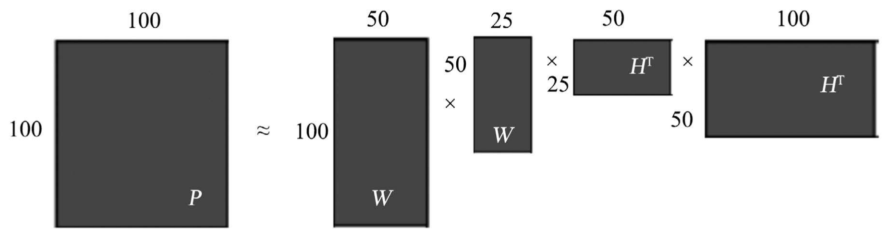


**Figure 2.** DIMC method shows multiple layers structure as an extension of NIMC. The figure is only an instance for 2 layers deep inductive matrix completion. Similarly, we can derives a deeper optimization problem such as 3 layers deep inductive matrix completion as following.

Consider Nonlinear Inductive Matrix Completion method involved in deep factor learning process. The optimization problem can be defined as following [5]:

$$\min_{W\in R^{m\times n},H\in R^{n\times k}} \sum\left(P-\phi\left(W\phi\left(H\phi\left(H^{\mathrm{T}}\phi\left(W^{\mathrm{T}}\right)\right)\right)\right)\right)^2+\frac{1}{2}\lambda\left(\|W\|_F^2+\|H\|_F^2\right),$$

Here $W\in R^{m\times n}$ and $H\in R^{n\times k}$, we have $k<m$, which can be clearly show as following picture:

$$\min_{A\in R^{m\times n},B\in R^{n\times k},C\in R^{k\times l}} \sum\left(P-\phi\left(A\phi\left(B\phi\left(C\cdots\phi\left(C^{\mathrm{T}}\phi\left(B^{\mathrm{T}}\phi\left(A^{\mathrm{T}}\right)\right)\right)\right)\right)\right)\right)^2$$
$$+\frac{1}{2}\lambda\left(\|A\|_F^2+\|B\|_F^2+\|C\|_F^2+\cdots\right),$$

Here $m>n>k>l$. Noticed that deep IMC with only one layer will automatically become a linear version IMC.

## 2.4. Optimization

The optimization process is to solve the above function with loss function. Our target is the original matrix $P$ and the reconstructed matrix as close as possible. In all above objective function, we choose gradient descent method to solve it. Gradient descent is a first-order iterative optimization algorithm for finding the local minimum of a function. Here we will give mathematical detail about gradient descent to solve inductive matrix completion. The process can divide to 3 steps: firstly compute gradient for each parameters. Secondly update parameters in each iteration. Last but not least, repeat iteration until we find minimial of loss function.

For example we want to optimize linear version inductive matrix completion method. The first thing is to randomly initialize parameters $W$ and $H$. Then fix parameters $W$ to train another parameter $H$, and then fix parameter $H$ to train parameter $W$ until we have minimal loss. The loss function can be written as following [5]:

$$J(\theta)=\sum\left(P-W^{\mathrm{T}}H\right)^2+\frac{1}{2}\lambda\left(\|W\|_F^2+\|H\|_F^2\right),$$

Here $\theta$ is parameter $W$ and $H$.

Secondly, we can compute gradient to find the fastest direction from initialized point to the local minimal respect to $W_i$ and $H_j$, we have:

$$\frac{\partial J(\theta)}{\partial W} = 2\times\left(P - WH^{H}\right)\times(-H) + \lambda\|W\|_F$$

Similarly,

$$\frac{\partial J(\theta)}{\partial H} = 2\times\left(P - WH^{\mathrm{T}}\right)\times(-W) + \lambda\|H\|_F$$

Now, we can update parameters in each iteration as following:

$$W = W - \alpha\frac{\partial J(\theta)}{\partial W}$$

$$H = H - \alpha\frac{\partial J(\theta)}{\partial H}$$

Here $W \times H^{\mathrm{T}}$ is our reconstructed matrix with full observation.

## 2.5. Algorithm

```
1: Initialize all parameters
2: Split training data and test data
3: for $i \in (1,\cdots,N)$ do
4: Set loss function
5: Calculate difference between original matrix and reconstructed matrix
6: Count the number of correct and total number of train/test value
7: Calculate training accuracy and test accuracy
8: end for
9: Gradient descent respect to loss function
10: Update parameters
```

## 2.6. Evaluation

In this paper, we focus on test accuracy since our target is to put an appropriate value in the missing position. However, we also need to define training accuracy since training accuracy shows the proportion of correct predicting value in total number of observation. We can track the change of training accuracy to know the progress of parameters training process. The training accuracy is defined as:

$$\text{training accuracy} = \frac{\text{number of correct value in non-missing part}}{\text{total number of non-missing part}}$$

Similarly, we can define test accuracy as the proportion of correct predicting value in total number of missing value. The following is formula:

$$\text{test accuracy} = \frac{\text{number of correct value in missing part}}{\text{total number of missing part}}$$

We regard $\varepsilon = 10^{-2}$ in this test, which means when the difference between original matrix and reconstructed matrix element is less than $10^{-2}$, we think the method predicts correct value.

## 3. Data

In this paper, we will test performance of three methods on fake linear data, fake nonlinear data and real data Russell 3000. We will test fake data first to see how the parameters change affects the accuracy of model. Then we apply an appropriate method to Russell 3000 data to test method performance on real world dataset.

### 3.1. Test Linear Data Generation

Our target is to investigate this section will show how to generate fake linear data in detail. Firstly, let's only consider a small size matrix. We generate the 100 × 100 original matrix $P$ by $W \times H^{\mathrm{T}}$ for IMC and NIMC, here $W$ and $H$ are 100 × $k$ vectors randomly generated within range 0 to 1. Notice that dimension $k$ must be small number since the normal assumption is matrix $P$ must be low rank so that the problem gradient descend method can solve. As shown in **Figure 3**.

Now we want to generate linear deep style structure data. Inspired by the neural network structure, we try to set linear data structure with multiple layers. Similarly, we generate the 100 × 100 original matrix $P$ by $W \times H^{\mathrm{T}}$ but adding one more layer. The result is we set original matrix $P$ by $A \times B \times B^{\mathrm{T}} \times A^{\mathrm{T}}$, here $A$ and $B$ are 100 × $m$ and $m$ × $k$ vectors, which randomly generate within range 0 to 1. Noticed that dimension $k$ must be small number since matrix $P$ must be low rank so that the problem gradient descend method can solve. The generation detail can be shown in **Figure 4**.

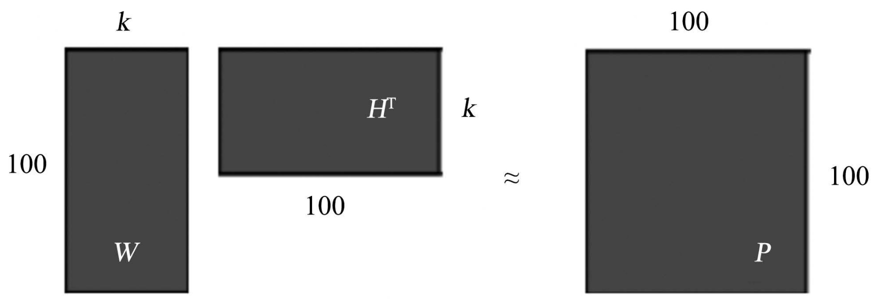


**Figure 3.** Linear Data Generation: we generate the 100 × 100 original matrix $P$ by $W \times H^{\mathrm{T}}$ for IMC and NIMC, here $W$ and $H$ are 100 × $k$ vectors randomly generated within range 0 to 1.

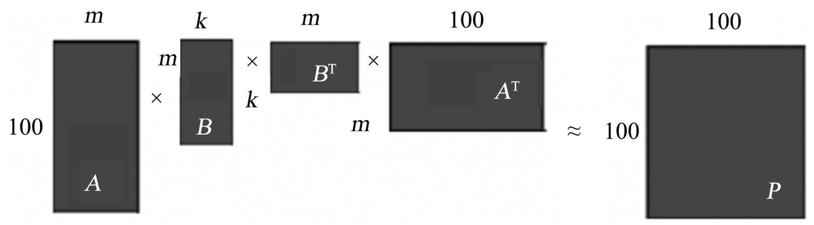


**Figure 4.** Linear Deep Data Generation: we set linear data structure with multiple layers. Similarly, we generate the 100 × 100 original matrix $P$ by $W \times H^{\mathrm{T}}$ but adding one more layer. The result is we set original matrix $P$ by $A \times B \times B^{\mathrm{T}} \times A^{\mathrm{T}}$, here $A$ and $B$ are 100 × $m$ and $m$ × $k$ vectors.

For above dataset, we randomly take out 30% value as missing value from $P$, hence, each test will use totally different original dataset.

## 3.2. Test Nonlinear Data Generation

Based on the linear data generation, we want to add nonlinear function to generate nonlinear data. In this way, we want to figure out how the nonlinear IMC method handle nonlinear data structure and how deep IMC handle deep style data structure. The generation of nonlinear data is much similar with linear data generation; however, the most important difference is to add different nonlinear function into each parameters column. The $\phi$ can be different nonlinear function such as sigmiod, RuLU, cos, sin or any nonlinear function. As shown in **Figure 5**.

## 3.3. Real Data Russell 3000 Generation

After testing on fake data, we want to know how our method handle real data. Here we choose Russell 3000 as our research target. As shown in **Figure 6**.

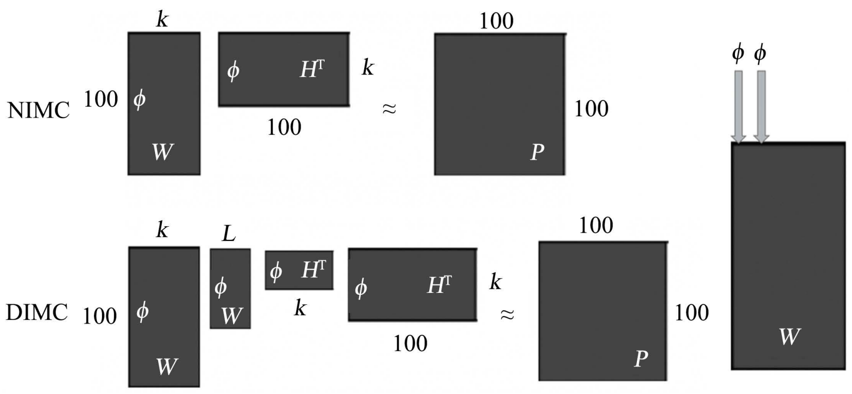


**Figure 5.** Nonlinear Data Generation: we add nonlinear function in each column to generate nonlinear data. Nonlinear function can be different nonlinear function such as sigmiod, RuLU, cos, sin or any nonlinear function.

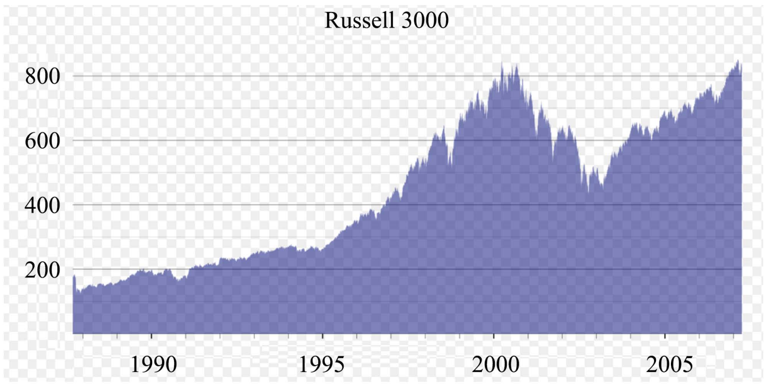


**Figure 6.** Russell 3000 Index: market-capitalization-weighted equity index maintained by the FTSE Russell that provides exposure to the entire U.S. stock market. Similar as S&P 500, the index tracks the performance of the 3000 largest U.S.-traded stocks which represent about 98% of all U.S incorporated equity securities.

The first time I met Russell 3000 is at Professor Randy Paffenroth group meeting. One of his PhD students Nitish research on finding financial market dimensions [6]. The main idea is to find out how many dimensions can represent the whole market data, which is a hard topic and enough to devote several years' research.

As for my perspective, Russell 3000 is a dataset with largely empty. The reason why the Russell 3000 is largely empty is the stocks have not completed IPO process at the date or suddenly disappear in the financial market due to the company collapse, or company under suspend for a period of time. The target we choose this dataset is to find out the how our method handle real world dataset. Also, Nitish find out Russell 3000 has nonlinear data structure during his research. However, this is a guess or conclusion we have from experiment. Unfortunately, we do not have formal theory to prove it now. Besides, we can try IMC, NIMC and DIMC on Russell 3000, which is an alternative way to verify data structure for Russell 3000. The reason why we can think in this way is all three method only can solve the problem that has same data structure with method itself. Hence, we can verify Russell 3000 data structure by testing which method works.

Since we never know the what is the correct value in missing position since they does not exist in real world market, we randomly choose 2000 stocks from Russell 3000 with 2000 valid date. In this way, we have a 2000 × 2000 matrix from Russell 3000 with all observation. The next thing is pick missing value. We have two kinds of missing value position to model real world market since the real world market exists the two situation. The first one models the situation suspend happen in the market. We define Random as randomly taking out of 30% value from dataset as missing value. The second one models the situation company have not done IPO process yet or suddenly disappear in stock market. We define Cross as taking out of large number value as missing value while left only a few observations. The two situations are shown in **Figure 7**.

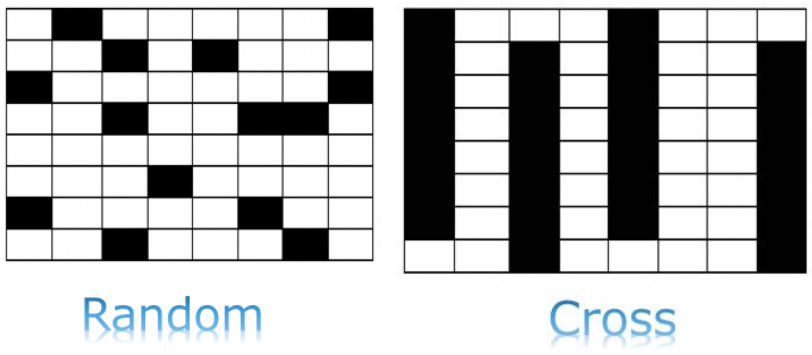


**Figure 7.** Missing Value Pick Method: the left one model the situation suspend happen in the market. We define Random as randomly taking out of 30% value from dataset as missing value. The right one model the situation company have not done IPO process yet or suddenly disappear in stock market. We define Cross as taking out of large number value as missing value while left only a few observations.

## 4. Test Data Experiment

### 4.1. Experiment 1: IMC with Linear Low Rank Fake Data

In experiment 1, we want to know the IMC method performance with linear low rank fake data on different dimensions. Also, this can be clearly shown in the relation between parameters dimension and test accuracy. Here we test IMC by setting parameters in dimensions 1, 5 and 20, as shown in **Figure 8**.

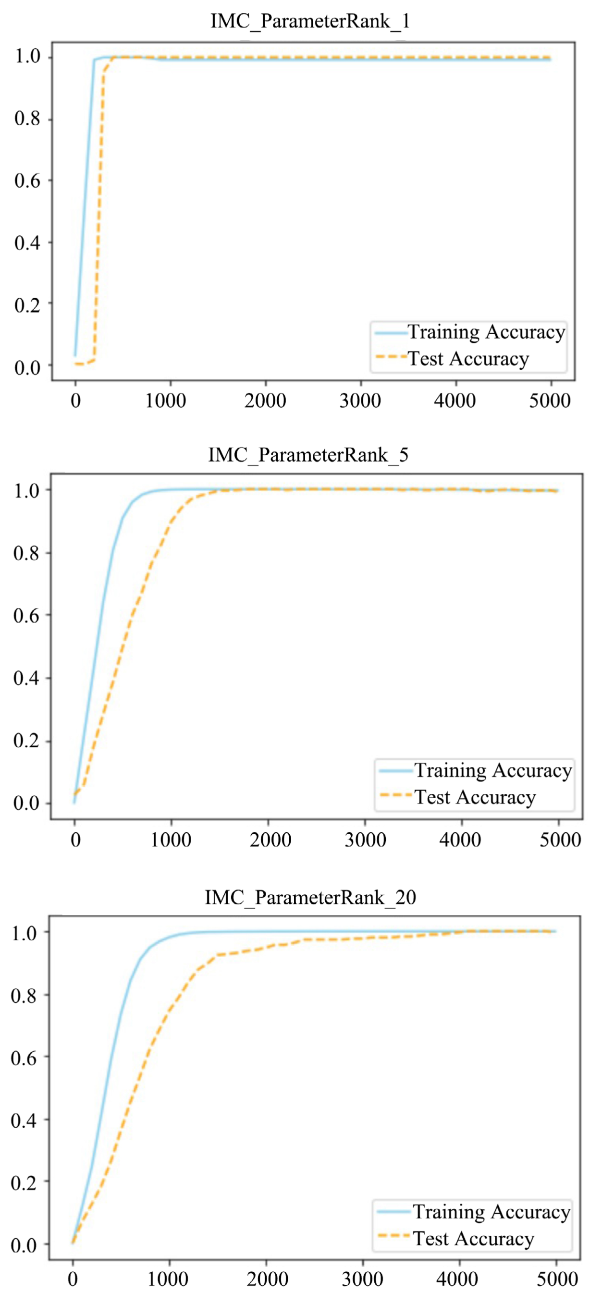


**Figure 8.** IMC with rank 1, 5, 20: IMC shows rank 1 parameters finished training first. Rank 5 and rank 20 are obviously slower than rank 1. However, IMC can have few information loss in all dimensions given linear low rank dataset, which means we can use IMC to do data imputation as long as our target dataset is linear low rank.

### 4.2. Experiment 2: Nonliear IMC with Linear Low Rank Fake Data in Different Parameter Rank

In experiment 2, we do the similar thing again but using NIMC in linear low rank data. We want to see how nonlinear method handle linear low rank data. Here we test NIMC by setting parameters in dimensions 1 and 5, as shown in Figure 9.

### 4.3. Experiment 3: Deep IMC with Linear Deep Fake Data

In experiment 3, we do the similar thing again but using DIMC in linear low rank data. We want to see how the Deep IMC method handle linear low rank data. Here we test DIMC by setting parameters in dimension $100 \times 10 \times 5$. We show the performance of deep version IMC as Figure 10.

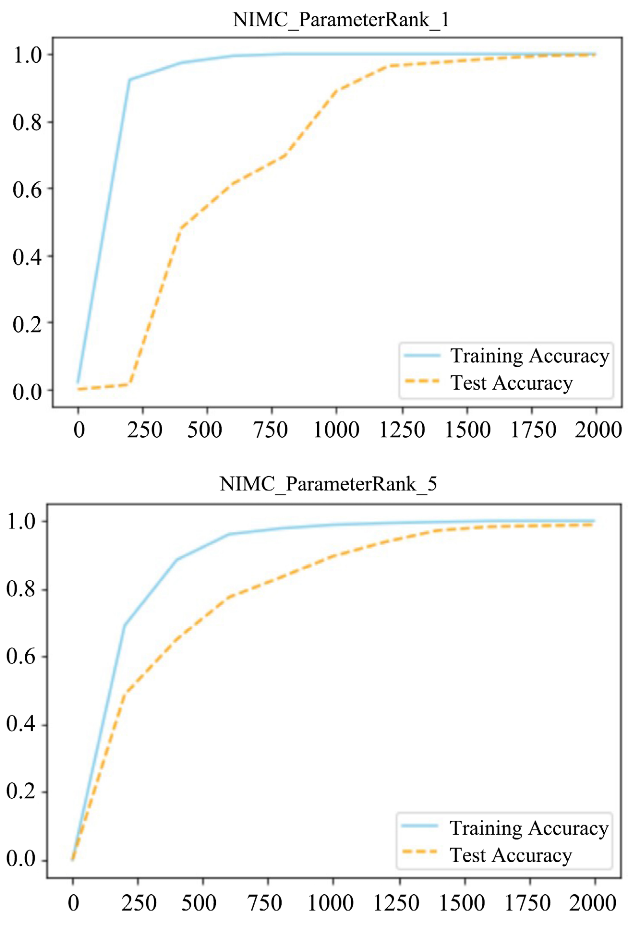


Figure 9. NIMC with rank 1, 5: NIMC shows rank 5 parameters have finished training first. Rank 1 obviously slower than rank 5. However, NIMC has larger information loss in low dimension with a few number of iteration compared to IMC. We may guess NIMC needs higher dimension to matrix completion. The reason why NIMC need higher dimension is that our fake data is generated linearly; hence, NIMC must solve this problem in nonlinear way while IMC can solve this problem in linear way. The conclusion is the NIMC needs problem in high parameters dimension to avoid information loss if we use NIMC solve linear data, however, absolutely IMC is more suitable to solve linear problem.

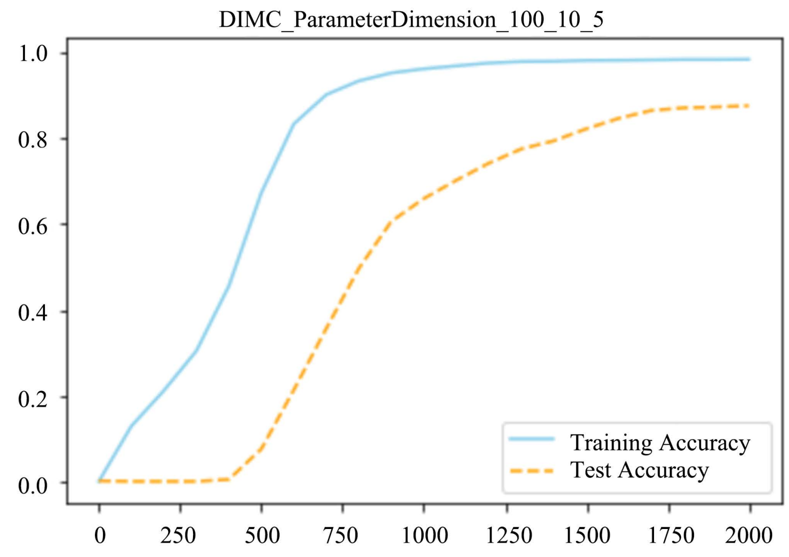


**Figure 10.** NIMC with rank 100_10_5: Deep version has the most information loss under the parameters low dimension 5, meaning DIMC does not have superiority compared to Nonlinear and linear version when we want to solve linear data problem. The reason is the same. The data we test is generating linearly and not accommodate with nonlinear model and deep model. Hence, we should use IMC to solve problem when we consider linear low rank data instead of NIMC or DIMC methods.

### 4.4. Experiment 4: IMC/NIMC/DIMC with Nonlinear Fake Data

From experiments 1, 2, 3, we have IMC as the best method to solve linear low rank data instead of applying NIMC and DIMC. Now we want to test how the IMC, NIMC and DIMC handle nonlinear low rank fake data. We set the same parameters dimension by 5 in all three tests. The test result can be tracked as shown in Figure 11.

## 5. Real Data Experiment with Russell 3000

After testing on fake data, we want to know how our method handles real data. Here we choose Russell 3000 as our research target, noticed that Russell 3000 is a dataset with nonlinear data structure. From our experiment, we can guess the NIMC is more suitable to solve dataset with nonlinear data structure. In the following test, we will investigate the performance of three methods on Russell 3000.

### 5.1. Experiment 5: IMC with Russell 3000

First we want to test how IMC handle Russell 3000 with random and cross missing value position. The test has two kinds of missing value position, which models two situation happens in real world. As shown in Figure 12.

### 5.2. Experiment 6: NIMC Russell 3000

This experiment we want to test NIMC performance on Russell 3000. Similar as the experiment 5, we test two missing value position on NIMC. Here is test with NIMC in parameters dimension 5. As shown in Figure 13.

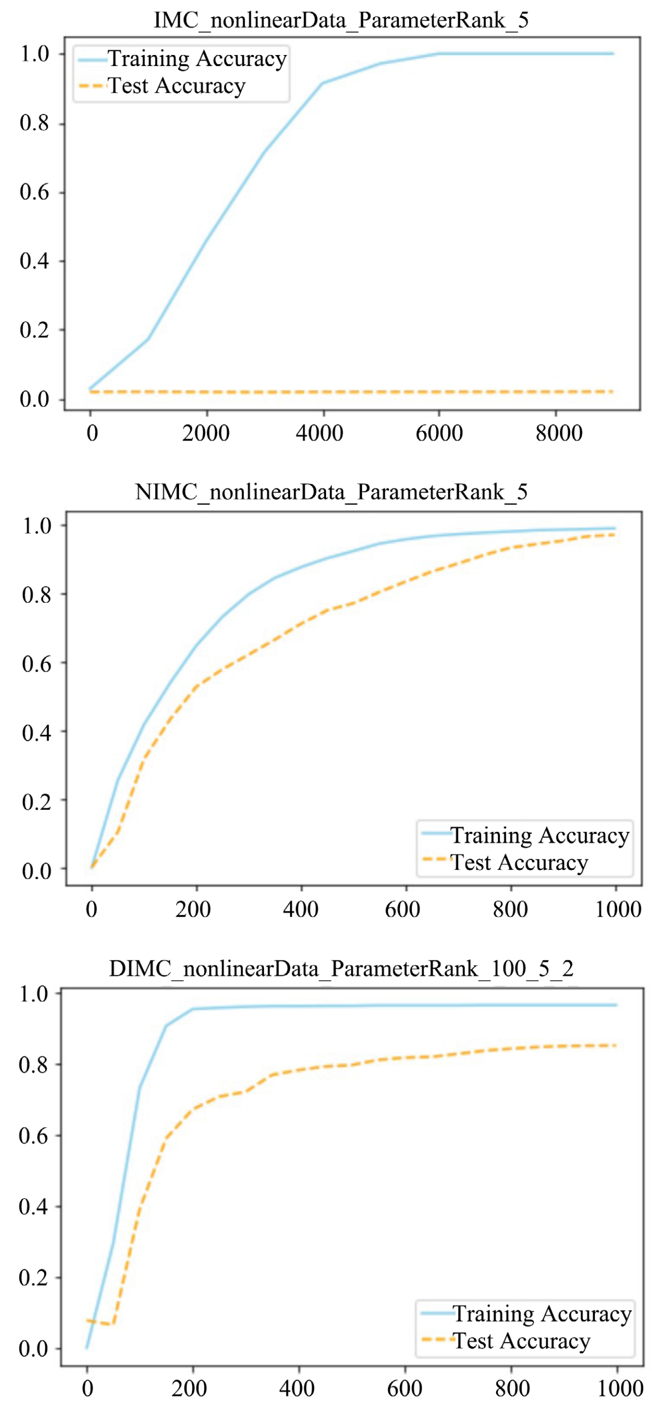
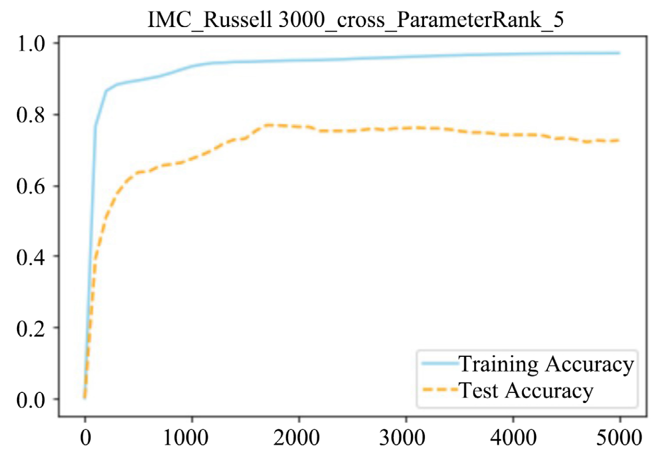


**Figure 11.** IMC/NIMC/DIMC with Nonlinear Fake Data: IMC can not solve nonlinear structured data while NIMC and DIMC can solve this problem well. More specifically, NIMC has better performance than DIMC. IMC has the most information loss under the nonlinear low rank data, meaning IMC has zero ability to handle nonlinear date, and hence, does not have superiority compared to Nonlinear and Deep version when we want to solve nonlinear data problem. The reason is IMC only can solve problem in linear way, therefore, when the IMC meet nonlinear structured data, it failed. The data we test is generate nonlinearly and accommodates with nonlinear model. Hence, we should use NIMC to solve problem when we consider nonlinear low rank data instead of NIMC or DIMC methods.



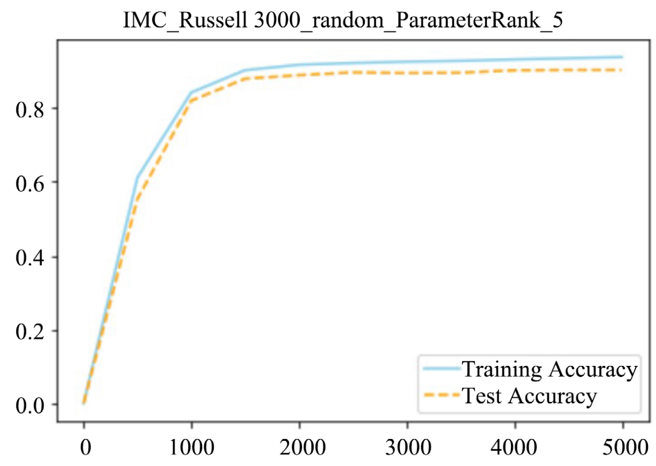


**Figure 12.** IMC with random/cross nan-value position in Russell 3000: IMC cannot handle Russell 3000 well since both missing value position have large information loss. IMC can only reach around 70% test accuracy on the cross missing data and 85% on the random missing data, meaning the imputation data only has 70% correct if we apply IMC to solve Russell 3000 dataset. Again, the reason why IMC cannot handle Russell 3000 well is Russell 3000 is naturally generated nonlinearly, and a linear version method IMC can only solve problem in linear way, hence, IMC is not good enough to handle Russell 3000.

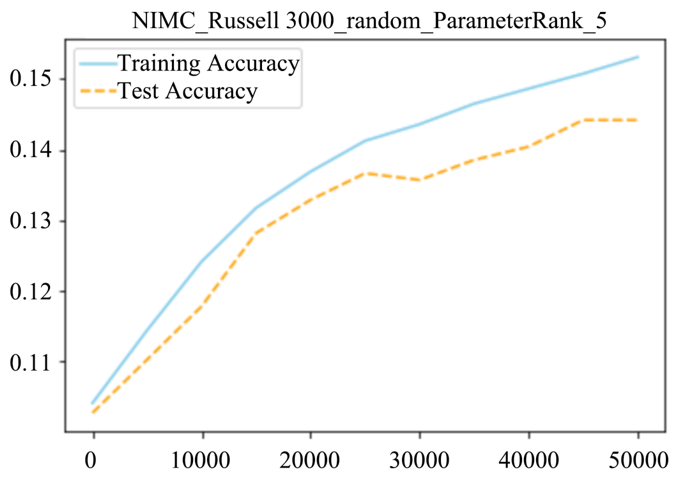


**Figure 13.** NIMC with Russell 3000.

From **Figure 13**, NIMC with Russell 3000 on parameters dimension 5 has huge information loss, which only has 14% correct missing value predicting. From the experiment from fake data, we know nonlinear method usually needs high dimension parameters to reconstruct matrix, hence, we try to increase parameters dimension by 80 as shown in **Figure 14**.

## 5.3. Experiment 7: DIMC with Random/Cross Nan-Value Position in Russell 3000

After we test Russell 3000 with NIMC, we know nonlinear method can handle it well in random missing value position instead of cross missing value position. Moreover, we want to know DIMC performance on the two kinds of missing value dataset. This test shows the performance that DIMC handle Russell 3000 with random and cross missing value position, as shown in **Figure 15**.

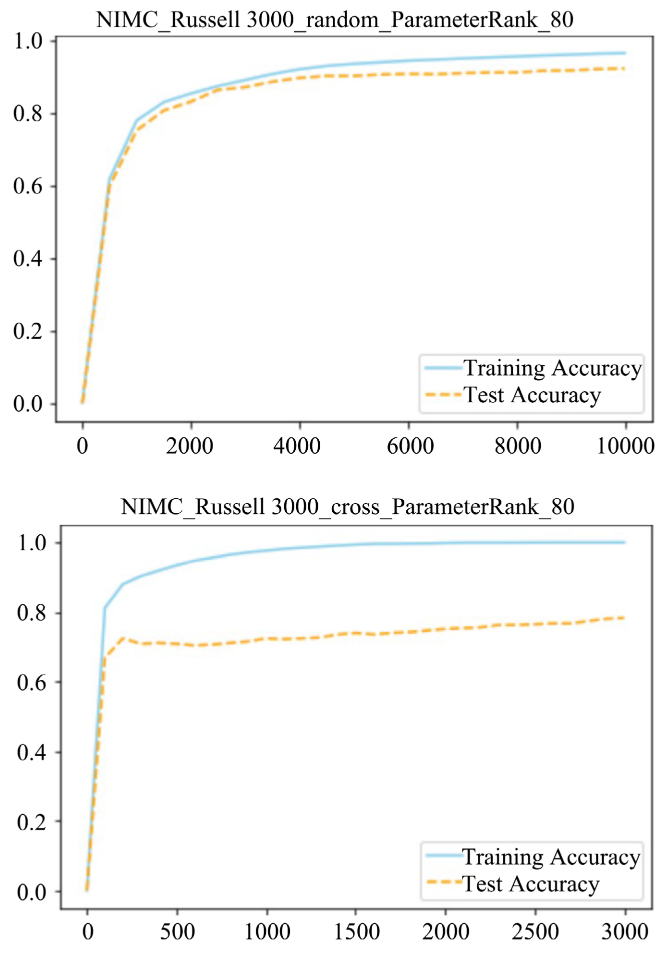
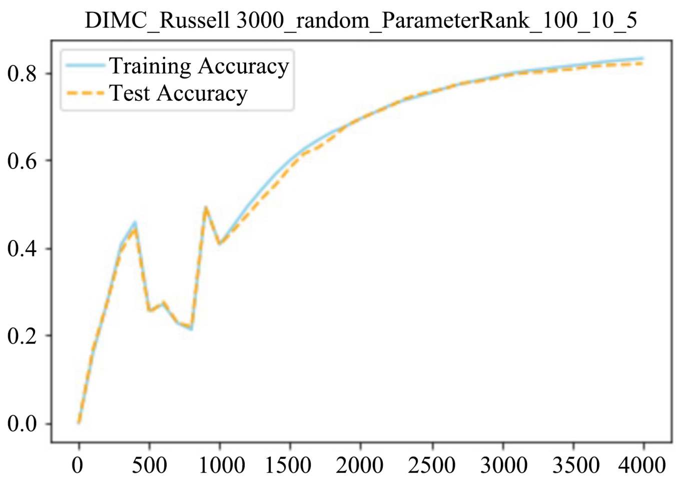


**Figure 14.** NIMC with random/cross nan-value position in Russell 3000: NIMC with random missing value position can reach about 95% test accuracy while with cross missing value position only reach about 80% test accuracy, meaning NIMC can handle random missing value well on parameters dimension 80 but not good enough to handle cross missing value position even on high parameters dimension. The conclusion is NIMC is more suitable to process missing value in Russell 3000 than IMC, especially in high parameters dimensions.



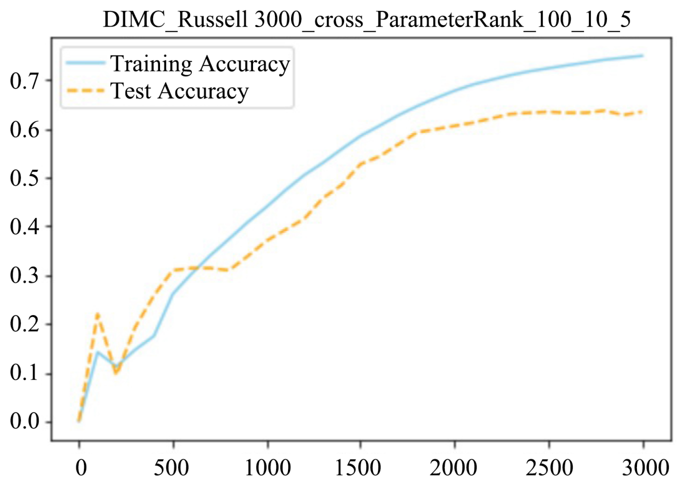


**Figure 15.** DIMC with random/cross nan-value position in Russell 3000: DIMC handle both of random and cross missing value data well but not good enough compared to NIMC. The test accuracy only reach around 80%. The reason is Russell 3000 has nonlinear data structure instead of deep style data structure, and hence, we should choose NIMC to solve Russell 3000 instead of DIMC.

## 5.4. Information Loss in Different Parameters Dimension

The last experiment we want to summarize information loss in the three methods on Russell 3000. We do experiment with three method but different parameter dimensions on 5, 10, 20, 50, 100, trying to find out how the dimension change affect test accuracy in real data Russell 3000. We do each test for 5 times and record error in Figure 16.

From Figure 16, the result shows the relation between accuracy loss and dimensionality change. We have NIMC and DIMC have large information loss in low dimension, especially DIMC suddenly decreases when dimension decreases by 5, however, IMC has less information loss on all dimensions, meaning NIMC and DIMC require high dimension to reconstruct dataset. However, we still need NIMC and DIMC to solve complex structured data since we know IMC only can solve linear structured data from fake data experiments.

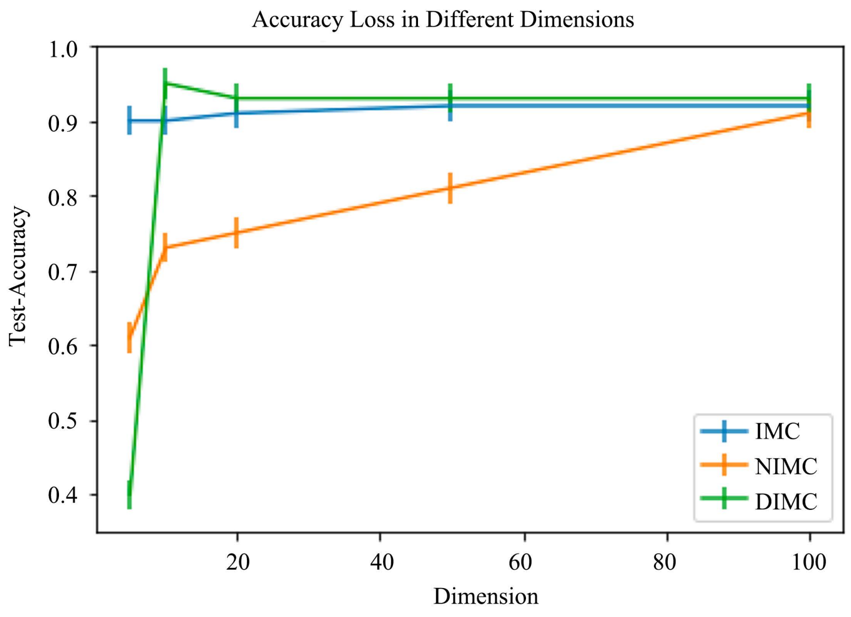


**Figure 16.** Accuracy loss in different dimensions.

## 6. Real-World Imputation Example—UL Safety Index

The UL Safety Index is a country level safety measurement of 17 indicators across 187 countries. This equates to a total of 3179 possible data points. The dataset is characterized by good data coverage, 93.1 percent. Data coverage per driver is very good or excellent for most countries. However, the 2018 UL Safety Index has 291 missing data points. In particular, a few specific countries and indicators are influenced by missing data: Micronesia, Taiwan Province of China, Consumer Protections, and Labor Rights. Here data imputation is desirable since the UL Safety Index is calculated using the geometric mean and any driver with a value of zero will result in an overall UL Safety Index of zero. Without data imputation, the UL Data Science team had to exclude these countries from some of the statistical measures.

In this section, we will present the detail about how to impute missing values within the UL Safety Index with nonlinear IMC. The process can be divided into 5 steps. The first step is clean data, choose appropriate indicators for imputation. The second step is data normalization, make sure all data used for training is normal distributed with 0 mean and 1 standard deviation. The third step is applying nonlinear IMC with Tanh activation function to impute data. The forth step is un-normalize data. We regard the imputed data that exceeds $2\sigma$ (standard deviation) is an abnormal point, and hence, we adjust these data by normalizing to the normal distribution. The last step is repeat step 3rd for 100 times, recording 100 times value then calculates mean value and standard deviation for each missing data points. Then pick half standard deviation above or below mean value as the appropriate boundary to calculate UL Safety Index range since half standard deviation includes most of possible output situations. Therefore, the prediction of UL Safety Index is a range instead of a fixed value.

## 6.1. Indicator

In this case, we apply indicators as following: transport_injury, falls, drowning, fires_heat_hot_substances, poisonings, exposure_to_mechanical_forces, foreign_body, education, network_readiness, other_unintentional_injuries, expo-sure_to_forces_of_nature__diunintentional_injuries, ul_standards, govern-ment_effectiveness,, consume_protection_survey, ul_labor_rights_index, road_safety, population, population_under_15, population_over_65, dgp, un_region, who_region, iso_membership, iec_membership, un_sub_region, un_development_status. We have total 29 indicators to impute data. Moreover, we add 2018 UL Safety Index into model to control prediction since any information we provided will be benefit for result. The reason is machine learning algorithm would increase accuracy once we provide more information and this data is integral to the imputation process, as it provides the "pattern" that the method leverages. Hence, totally we have 29 indicators to impute data while 291 missing data points at this moment. Lastly, the method only can feed numerical value instead of category words, hence, we set factorize indicators to numercial value as following: un_region, un sub region, un development status, who region, iso membership, iec_membership.

The above process can be found in code "Get Origial Data" and "Get Categroy Data by Int" part.

## 6.2. Normalization: Original Data

For all 29 indicators, we want to normalize data by each indicators' mean and standard deviation as following formula:

$$X' = \frac{X - \mu}{\sigma}$$

Here, $X'$ is normalized data, $X$ is data point, $\mu$ is mean value of each indicators, $\sigma$ is standard deviation of each indicators. Now, we have clean data to impute, which can be found as "df clean.csv" after run code provided.

## 6.3. Imputation: NIMC with Tanh Funtion

For the data imputation part, we applied non-linear IMC with Tanh activation function since Tanh give us best training accuracy after try sigmoid function, ReLU function, sin function, cos function or even linear version IMC. Here we set parameters $W$ and $H$ randomly generated by 178 $\times$ 50 and 50 $\times$ 30 matrix, learning rate alpha is 10. Do 7000 iteration for each training. The result show 7000 iteration is enough to reach 98% training accuracy. The code can automatically output loss function and training accuracy on each iteration, then plot visualization graph. This section can be found in code "Data Imputation" part.

## 6.4. Adjust Abnormal Points

For the data we impute, most of them are "reasonable", but a few of them are "unreasonable". For example, for the ul_labor right, one value will excess 77, which is impossible considering the maximum value of the initial setting of ul labor right indicator is 77. Hence, we regard the data imputed which exceeds $2\sigma$ is abnormal point, and then we adjust these data by normalizing to the normal distribution by applying formula in 2.2. Totally we have about 1220 abnormal data point in each experiment. The value can be print from code "adjust normalize" part.

### 6.5. Repeat above Process to Calculate Impute Range

The last step is repeat above process to output a range for imputed data. After checking output by applying randomly input, we realized our model is stable since the imputed data follows normal distributed with $N(0, 1)$.

By the result, we repeat step 3rd data imputation for 100 times, recording 100 times value then calculate mean value and standard deviation for each missing data points. Then pick half standard deviation above or below mean value as the appropriate boundary to calculate UL Safety Index. Therefore, the prediction of UL Safety Index is a range instead of a fixed value.

## 7. Real-World Data Imputation Result

### 7.1. Distribution: Consume Protection Indicator

The data distribution shows most of data points follow the trend that high Consumer Protection score consistently with high Safety Index as shown in Figure 17.

Only one or two data point seems "unreasonable". For example, the lowest red point with high Safety Index but a very low Consumer Protection score prediction. Indeed, we have to admit any prediction will surely have "mismatch" point. However, according to Stability analysis above, we can calculate prediction range instead of a fix value, which is close to the truth and acceptable for us.

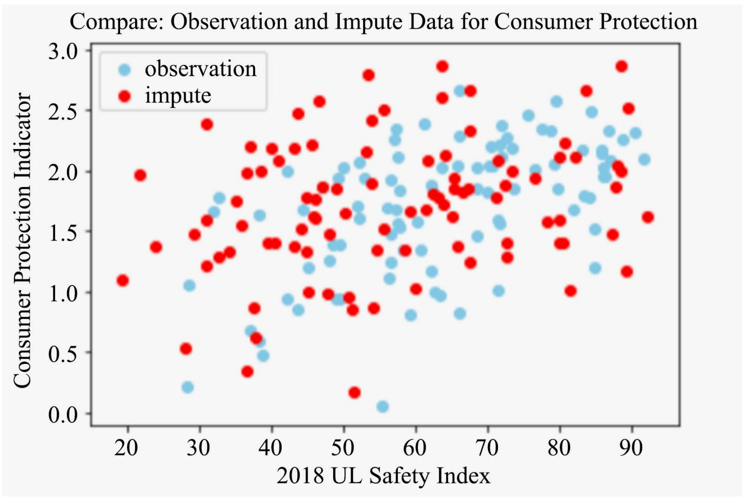


Figure 17. Compare distribution of observation and impute date for consumer protection indicator. The y-axis represents value of consumer protection of 187 countries, x-axis is 2018 UL Safety Index.

### 7.2. Distribution: Labor Right Indicator

The data distribution shows most of data points follow the trend that high Labor Right score consistently with high Safety Index as shown in Figure 18.

Only one or two data points seem "unreasonable". For example, the upper left red point with low Safety Index but a very high Labor Right score prediction, or the lower right red point, has a high Safety Index but a very low Labor Right score. Also, we have to admit any prediction will surely have a "mismatch" point. However, according to the Stability analysis above, we can calculate the prediction range instead of a fixed value, which is close to the truth and acceptable for us.

### 7.3. Imputation Visualization: Comparison with Original Ranking, Traditional Method and NIMC Imputation

See Figure 19.

## 8. Real World Data Experiment Comparison

When comparing the original data to the traditional imputation method and NIMC imputation method, the impact of discarding the missing values is illustrated as shown in Figure 20 and Figure 21.

In the first figure, the traditional method and NIMC imputation method nearly overlap and have few changes since these countries have good data coverage, *i.e.* few missing values. For example, Norway, Finland, New Zealand, China, and Jordan almost all overlap. However, in the second figure, some countries have a significant change. According to the result, the original calculation, where missing values are not imputed, may overestimate or underestimate a country's safety situation. For example, when applying the traditional and NIMC

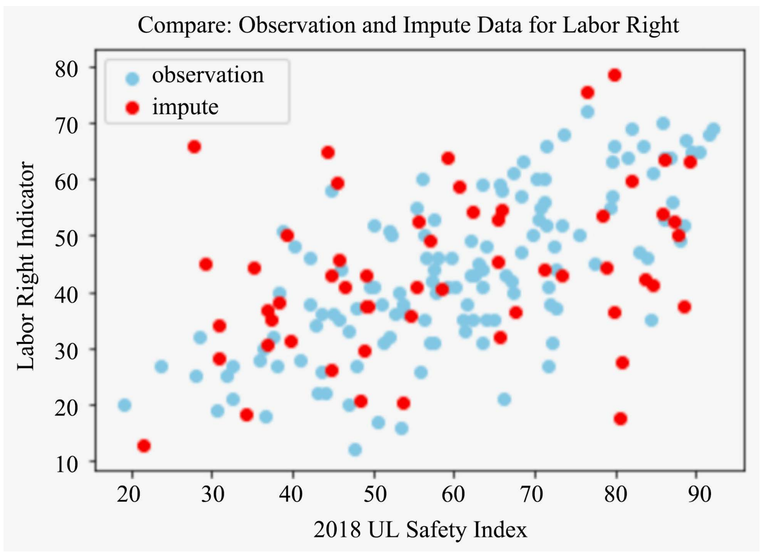
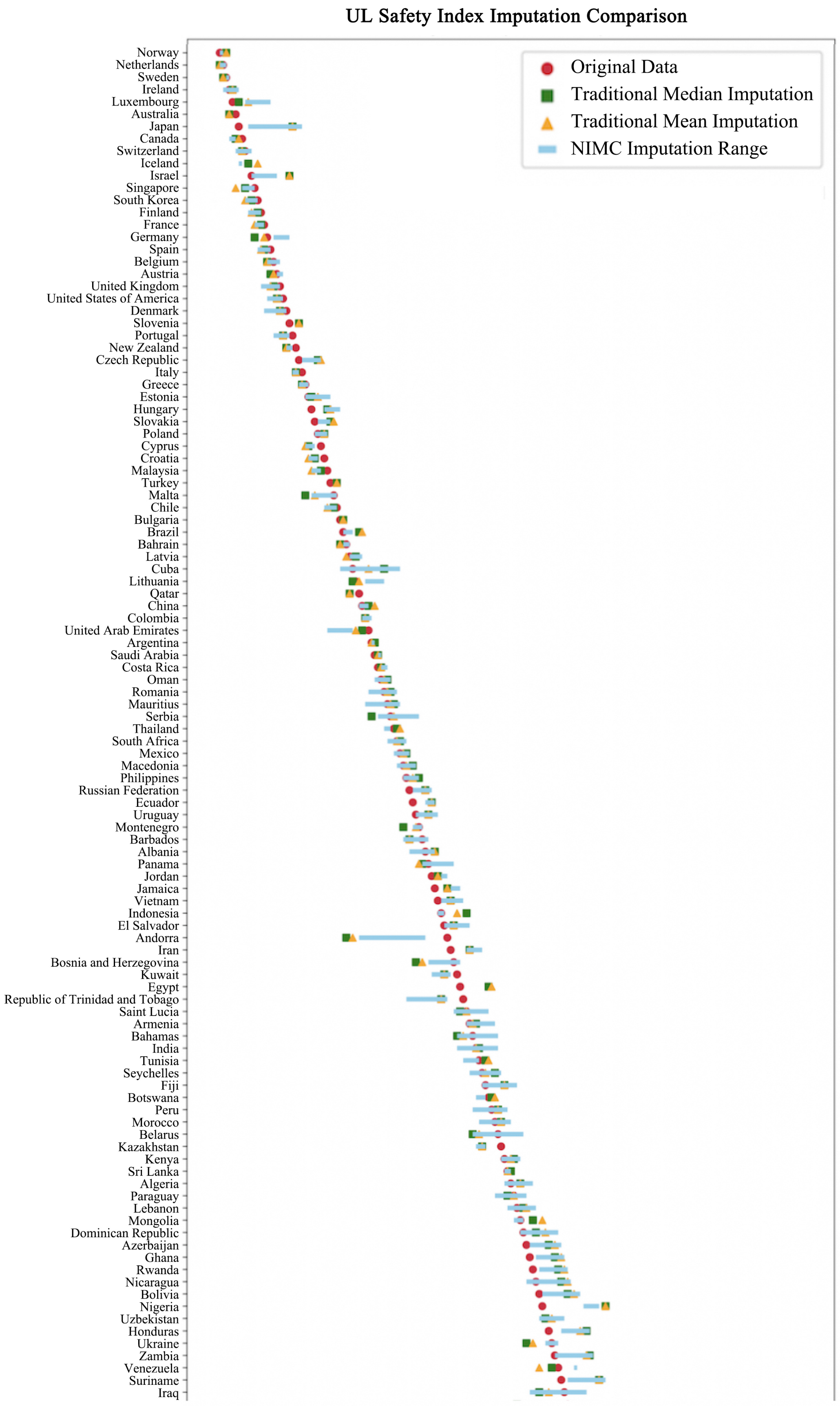


Figure 18. Compare distribution of observation and impute date for Labor Right indicator. The y-axis represents value of Labor Right of 187 countries, x-axis is 2018 UL Safety Index.


UL Safety Index Imputation Comparison
Original Data
Traditional Median Imputation
Traditional Mean Imputation
NIMC Imputation Range
Norway
Netherlands
Sweden
Ireland
Luxembourg
Australia
Japan
Canada
Switzerland
Iceland
Israel
Singapore
South Korea
Finland
France
Germany
Spain
Belgium
Austria
United Kingdom
United States of America
Denmark
Slovenia
Portugal
New Zealand
Czech Republic
Italy
Greece
Estonia
Hungary
Slovakia
Poland
Cyprus
Croatia
Malaysia
Turkey
Malta
Chile
Bulgaria
Brazil
Bahrain
Latvia
Cuba
Lithuania
Qatar
China
Colombia
United Arab Emirates
Argentina
Saudi Arabia
Costa Rica
Oman
Romania
Mauritius
Serbia
Thailand
South Africa
Mexico
Macedonia
Philippines
Russian Federation
Ecuador
Uruguay
Montenegro
Barbados
Albania
Panama
Jordan
Jamaica
Vietnam
Indonesia
El Salvador
Andorra
Iran
Bosnia and Herzegovina
Kuwait
Egypt
Republic of Trinidad and Tobago
Saint Lucia
Armenia
Bahamas
India
Tunisia
Seychelles
Fiji
Botswana
Peru
Morocco
Belarus
Kazakhstan
Kenya
Sri Lanka
Algeria
Paraguay
Lebanon
Mongolia
Dominican Republic
Azerbaijan
Ghana
Rwanda
Nicaragua
Bolivia
Nigeria
Uzbekistan
Honduras
Ukraine
Zambia
Venezuela
Suriname
Iraq

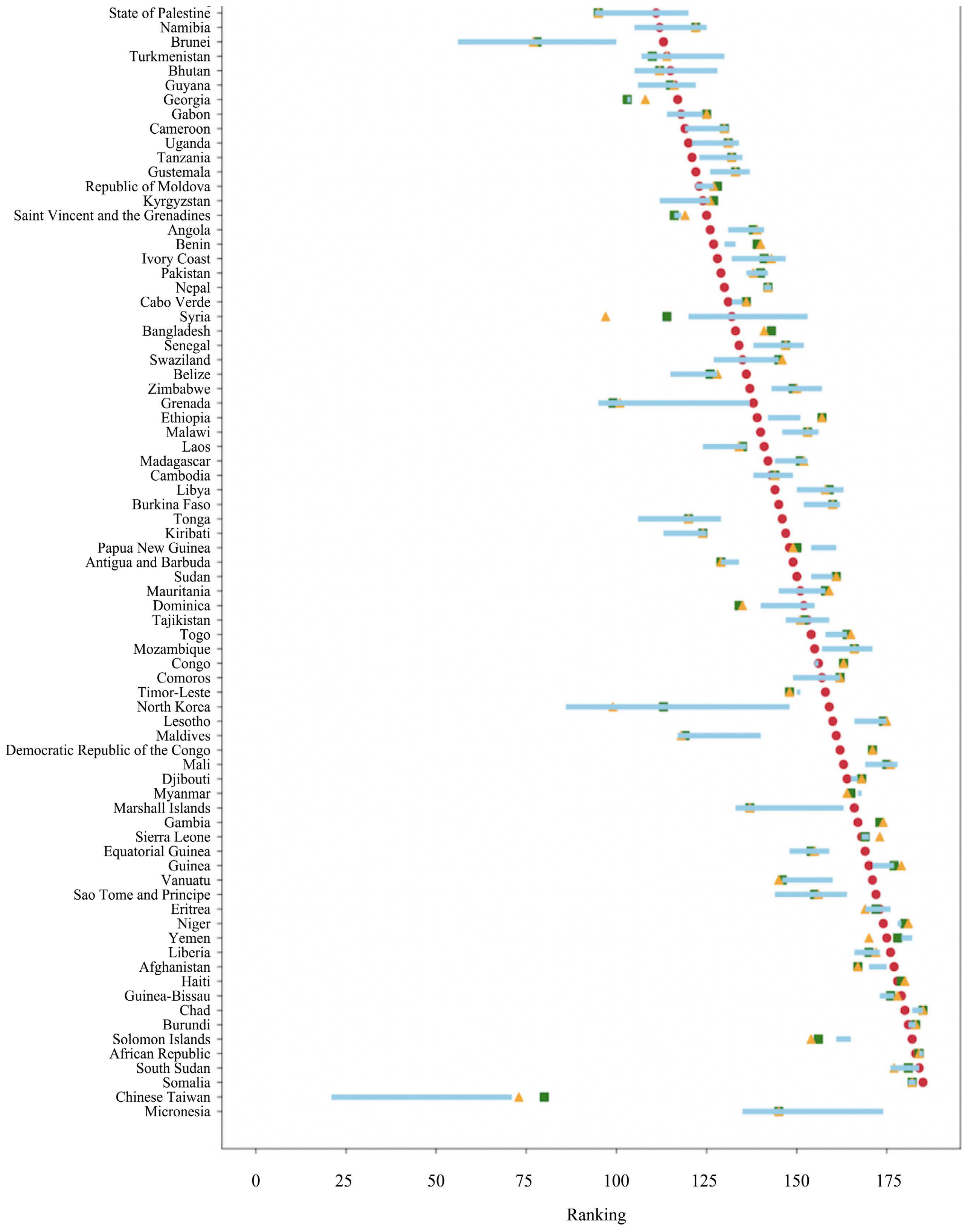


**Figure 19.** Comparison with the original result, traditional imputation method, and NIMC imputation method. The original data directly discard missing value indicators. The traditional method uses column mean and median value to replace missing data points.

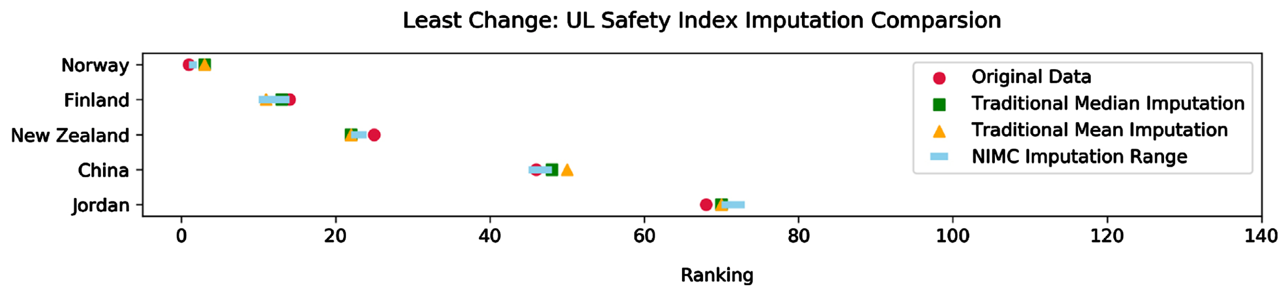


Figure 20. Comparison with the original result, traditional imputation method, and NIMC imputation method. The original data directly discard missing value indicators. The traditional method uses column mean and median value to replace missing data points.

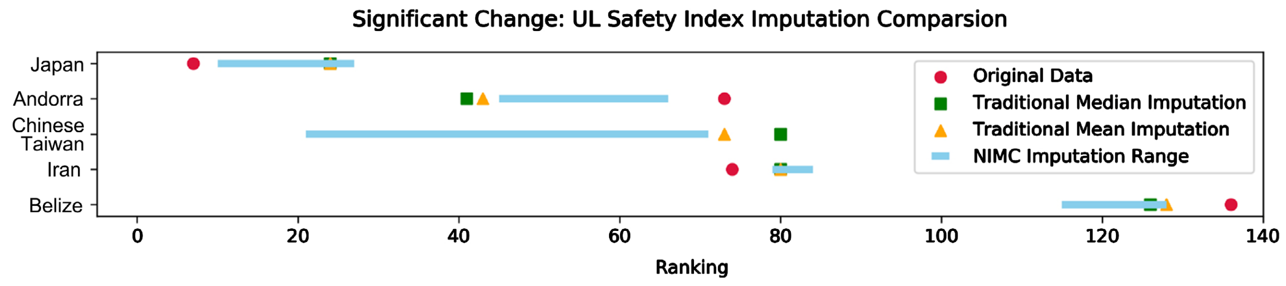


Figure 21. Comparison with the original result, traditional imputation method, and NIMC imputation method. The original data directly discard missing value indicators. The traditional method uses column mean and median value to replace missing data points.

methods to Japan and Iran, the results indicate they may have worse safety environments than our initial estimation. Similarly, Andorra, Belize may have better safety environment than our initial estimation. More telling, however, is a safety evaluation prediction is available for Taiwan Province of China despite having only a few original data points. While the resulting UL Safety Index using imputation is a modeled prediction and difficult to validate in the real world, it provides insights and direction to discuss the safety situation.

## 9. Conclusions

In conclusion, our experimentation with synthetic data demonstrates that Inductive Matrix Completion (IMC), Nonlinear Inductive Matrix Completion (NIMC), and Deep Inductive Matrix Completion (DIMC) excel in handling linear low-rank synthetic data. Notably, IMC exhibits proficiency in managing both low and high-dimensional datasets. However, it struggles with nonlinear data structures, a limitation effectively addressed by NIMC and DIMC. The intrinsic nature of nonlinearly generated data, with bends and folds in space, presents challenges for linear methods like IMC. In contrast, the non-linear and deep learning approaches of NIMC and DIMC navigate these complexities using gradient descent, showcasing their superiority in capturing intricate structures.

Our investigation with real-world data, specifically the Russell 3000 dataset, further validates the effectiveness of these methods. Given the dataset's partial linear and partial nonlinear structure, both IMC and NIMC prove capable, with

NIMC demonstrating superior performance. However, challenges arise when handling Cross missing value data, where limited observations hinder effective reconstruction. Increasing observations could enhance performance in such scenarios. DIMC, while proficient, does not exhibit significant superiority over IMC and NIMC in Russell 3000, unless tested on more intricate datasets with deep structures.

Examining information loss reveals that NIMC and DIMC experience substantial losses in low dimensions, particularly DIMC, which exhibits a sudden decrease when dimensions are reduced by 5. In contrast, IMC demonstrates lower information loss across all dimensions, indicating its versatility in handling datasets of varying complexities. Despite this, the necessity of NIMC and DIMC persists for addressing intricate data structures, acknowledging IMC's limitation in handling only linear structures, as evidenced by our synthetic data experiments.

In summary, our analysis emphasizes the importance of understanding the data structure when employing inductive matrix completion methods for matrix completion or data imputation tasks. Recognizing the dataset's structure becomes the primary step in building effective and tailored solutions. This prompts consideration for future research: the identification of data structures and their implications in enhancing matrix completion methodologies.

## Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

## References

[1] Weidman, L., Schenker, N. and Treiman, D.J. (1993) Analyses of Public Use Decennial Census Data with Multiply Imputed Industry and Occupation Codes. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, **42**, 545-556. https://doi.org/10.2307/2986331

[2] Cai, J.-F., Candès, E.J. and Shen, Z.W. (2010) Singular Value Thresholding Algorithm for Matrix Completion. *SIAM Journal on Optimization*, **20**, 1956-1982. https://doi.org/10.1137/080738970

[3] Dhillon, I.S. and Natarajan, N. (2014) Inductive Matrix Completion for Predicting Gene-Disease Associations. *Bioinformatics*, **30**, i60-i68. https://doi.org/10.1093/bioinformatics/btu269

[4] Paffenroth, R.C. (2019) DS502 Statistics Method in Data Science Lecture Notes. Worcester Polytechnic Institute, Worcester, MA.

[5] Lee, K. (2019) CS539 Machine Learning Lecture Notes. Worcester Polytechnic Institute. Worcester, MA.

[6] Paffenroth, R., Bahadur, N. and Gajamannage, K. (2018) A Study of Russell 3000 Dimensionality Using Non-Linear Dimensionality Reduction Techniques. *Bioinformatics*.