Applied Mathematics, Modeling and Computer Simulation
C.-H. Chen et al. (Eds.)

doi:10.3233/ATDE221061

Estimating Hidden Parameters from a Dataset with Large Amount of Missing Data

Yuzhong HU[1]

Yanqi Laike Beijing Institute of Mathematical Sciences and Applications, China

Abstract. Dealing with missing data is a fundamental problem in data science. In some rare cases, the large amount of missing data can be a big problem for any data analysing tasks. In this paper, we propose a method based on maximum likelihood estimation and expectation maximization to get estimation on the hidden parameter with large amount of missing data in the dataset. We perform the numerical experiments to validate the feasibility and stability of our method. We test the algorithm for different patterns of missing data, different amount of samples and different noise levels. The result indicates that the algorithm is effective in handling large amount of missing data if enough samples are provided.

Keywords. Missing Data Imputation, Maximum Likelihood Estimation, Expectation Maximization.

1. Introduction

Missing data are frequently observed in real world problems. Sometimes, there can be large proportion of missing data among few intact ones. For example, in the field of automobile safety, there is a large quantity of data involving all kinds of accidents, but most of them miss part of data because of various reasons. It is crucial to retrieve the information from the missing data to construct a better model rather than just drop them. Therefore, the method of handling missing data is crucial in this situation.

So far, there are plenty of methods dealing with missing data problems. The easy and common way is to drop the missing data. However, there is research indicating the drawback of such method.[1] Such deletion may cause substantial bias through the process. Therefore, many previous researches try to find an optimal way to fill the absent values. Such methods include simple mean imputation, hot-deck imputation. [2] There are also imputation methods based on statistical learning, such as K-nearest neighbor imputation, clustering-based imputation and regression-based imputation. [3][4][5] If we resort to the parameter estimation methods for imputation, expectation maximization is a simple end effective way to deal with missing values.[6] Multiple imputation (MI) is another powerful tool used by many data scientists. Unlike single statistical model, MI tries to utilize multiple models to get the missing data estimation by analyzing and combining different estimations. [7] Recently there is a growing body of literatures that explores the imputation methods based on neural networks and deep

[1] Yuzhong HU, Yanqi Laike Beijing Institute of Mathematical Sciences and Applications, China. Email: huyuzhong@bimsa.cn.

learning. Common models including multi-layer perception, deep autoencoders and convolutional neural networks are used for missing data imputation. [8][9][10] Despite all the efforts, handling large chunk of missing data still remains a major challenge in data science.

In this paper, we propose a method of handling large amount of missing data based on maximum likelihood estimation in order to give an accurate estimation on the hidden parameters. We also perform numerical experiments to test our method. This paper is organized as follows: Section 2 gives mathematical definitions to our problem, followed by the approach we take to solve the problem in Section 3. The numerical experiments are conducted and the results are analyzed in Section 4, and the conclusion is drawn in the last section.

2. Mathematical Definition and Data Generation

2.1. Mathematical Definition

Suppose we have a dataset $\mathcal{D}$ with N samples, where each sample containing M parameters. Missing values appear randomly in these samples. For each sample, we know the evaluation function $f(x;\theta)$, where θ is certain unknown parameter.

For example, in this paper, we choose certain θ to be the true parameter and we define the evaluation function to be the Euclidean distance between the sample and θ, i.e.

$$f(x^i;\theta) = \sqrt{\sum_j \left(x_j^i - \theta_j\right)^2}$$

With N samples, we have N scores given by $s_i = f(x^i;\theta)$. Using the information given by these scores, our goal is to give an accurate estimation of the hidden parameter θ.

More specifically, table 1 indicates the meaning of the symbols used in this paper. Figure 1 gives an illustrated description of our problem.

Table 1. The description of the symbols used in this paper

Symbols	Description
$\boldsymbol{\mathcal{D}}$	Dataset
N	Number of samples
M	Number of parameters of each sample
$\boldsymbol{x^i}$	The i-th sample in $\mathcal{D}$
$\boldsymbol{x_j^i}$	The j-th parameter of x^i
$\boldsymbol{\hat{x}^i}$	The estimated value for sample x^i
$\boldsymbol{\hat{x}_j^i}$	The estimated value for missing x_j^i
$\boldsymbol{m_i}$	The number of missing values in x^i
$\boldsymbol{\theta = (\theta_1, \theta_2, \dots, \theta_M)}$	Hidden parameter
$\boldsymbol{\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_M)}$	Hidden parameter estimation
$\boldsymbol{f(x;\theta)}$	The evaluation function of any sample x with respect to certain parameter θ
$\boldsymbol{s_i = f(x^i;\theta)}$	The evaluation score of each sample
$\boldsymbol{N_m}$	Number of samples with exactly m missing values

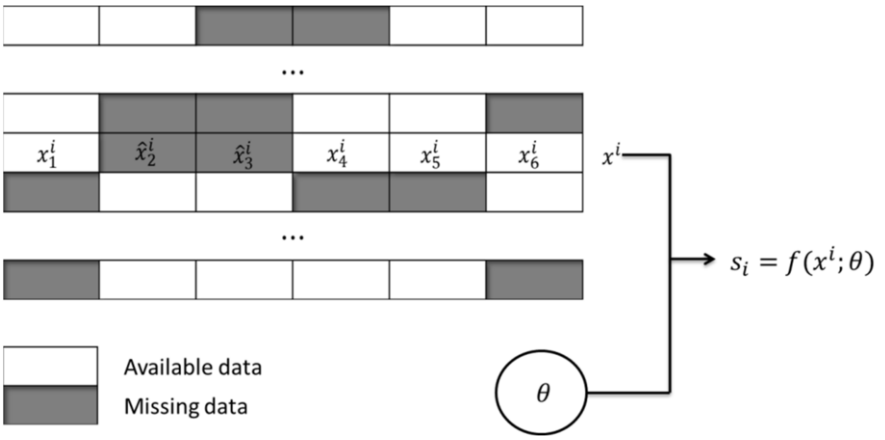

Figure 1. The description of the problem

2.2. Data Generation

The dataset $\mathcal{D}$ is generated randomly, with each parameter of the sample being independent. In this paper, we fix the number of parameters M to be 10 in each sample. Hence, we get a randomly generated dataset

$$\mathcal{D} = \{x^i = (x_1^i, x_2^i, \dots, x_{10}^i) | i = 1,2, \dots, N\}$$

As for the missing value part, we generate them based on two different rules.

Rule A is generating them just by randomization. For each parameter in a sample x_j^i, there is a certain possibility that it is absent. The proportion of the missing values depends on p, the probability of a missing value occurs. We tried different $p \in \{0.2,0.3,0.4,0.5\}$ in this paper. When N equals 1000, the relation between m and N_m are displayed in figure 2.

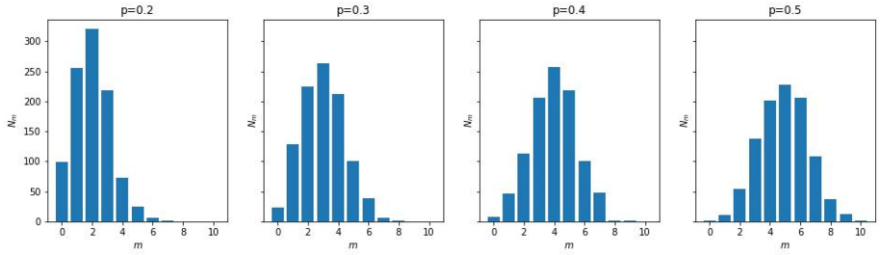

Figure 2. When N=1000 and we adopt rule A for generating missing values, the relation between number of missing values m and corresponding number of samples N_m for $p \in \{0.2,0.3,0.4,0.5\}$

Rule B is generating the missing values by certain rules of missing data. We assume the dataset contains much more 'sparse' samples than 'dense' ones. Therefore, we adopt the exponential relation between the number of missing values m and number of samples with exactly m absent values N_m. We set α the parameter controlling the exponential relation. Here α equal to zero means the N_m's are the same for different m's, and the larger the α, the larger N_M, meaning more sparse samples in the dataset. More specifically, let $\frac{N_m}{N_{m-1}} = e^{\alpha}, m = 2,3, \dots, 9$, and we can calculate the N_m's based

on this relation. In this paper, we choose $\alpha \in \{0,0.1,0.2,0.3\}$. When N is equal to 1000, the relation between m and N_m are displayed in figure 3.

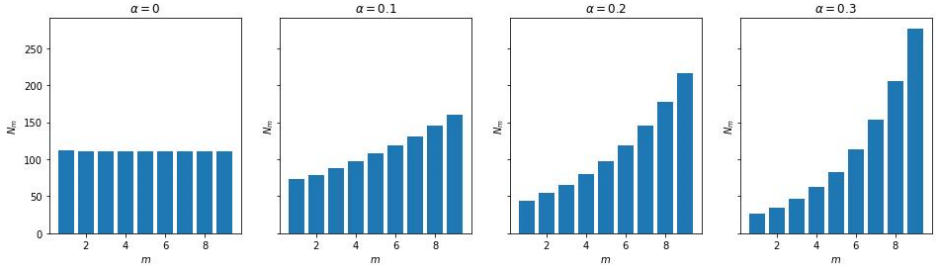

Figure 3. When N=1000 and we adopt rule B for generating missing values, the relation between number of missing values m and corresponding number of samples N_m for $\alpha \in \{0,0.1,0.2,0.3\}$

3. Methods

We use the statistical tools from maximum likelihood estimation to build the basis of our method.

3.1. Maximum Likelihood Estimation

Maximum Likelihood Estimation is a method commonly used in many machine learning algorithms. It involves treating the problem as an optimization or search problem, where we seek a set of parameters that results in the best fit for the joint probability of the data sample. [1]

In our problem setting, θ is a hidden parameter to be learned. The values of θ is closely related to those x^i's. Therefore, we can write:

$$L(\theta) = \prod_i P(x^i|\theta) = \prod_{i,j} P(\hat{x}_j^i|\theta).$$

The second equality here comes from the assumption that these missing values are independent from each other.

Now the problem becomes how to get an estimation of θ to maximize $L(\theta)$.

3.2. Expectation Maximization

The Expectation-Maximization algorithm (EM) is an approach for maximum likelihood estimation in the presence of latent variables. [6]

The EM algorithm is an iterative approach that cycle between two modes. The first mode attempts to estimate the missing or latent variables, called the estimation-step or E-step. The second mode attempts to optimize the parameters of the model to best explain the data, called the maximization-step or M-step.

E-Step: Estimate the missing variables in the dataset.

M-Step: Maximize the parameters of the model in the presence of the data.

There are several properties that can be proved about EM algorithm. First, in the process of learning, the incomplete data likelihood function is non-decreasing after each EM step. Second, the convergence rate of the EM algorithm is quite fast with

respect to the likelihood function. Least but not last, the EM algorithm provides a condition to automatically satisfy probabilistic constraints of mixture models.[12]

With these properties, we can safely adopt the EM algorithm in the maximum likelihood estimation with relatively low time cost.

3.3. Method Towards Our Problem

The basic idea is to utilize EM algorithm in this problem. We adopt some variant of the original EM algorithm to make it suitable to our problem. The details are described in the following algorithm. The algorithm can be briefly viewed as in figure 4.

Algorithm.

1) Pre-process the missing data $\hat{x}_j^i$ by simple imputation methods like mean imputation. Then we get an initial guess $\hat{\theta}$ of θ based on the pre-processed data.
2) For the filled $\hat{x}^i$, we use optimization methods to update $\hat{\theta}$, such that
$$\hat{\theta} = \text{argmin} \sum_i w_i \left(f\left(\hat{x}^i; \hat{\theta}\right) - s_i\right)^2,$$
where w_i denotes the weights of the i-th sample. Since there exists exponential relation between the number of missing values and the information it contained, we define $w_i = e^{-m_i}$, where m_i denotes the number of missing values in x^i.
This can be viewed as the M-step in the EM algorithm, since it fits in the most probable estimation $\hat{\theta}$ according to the current guess of the missing data.
3) For the updated $\hat{\theta}$, we again compute the probability distribution of $\hat{x}^i$ for each missing value, to maximize the likelihood.
$$\hat{x}^i = \text{argmax}\, P\left(\hat{x}^i \middle| \hat{\theta}\right), i = 1,2, \ldots, N$$
This can be viewed as the E-step in the EM algorithm, since it gives the estimate of the missing value based on the current parameter estimation.
4) Repeat step 2) and 3) until $\hat{\theta}$ converges.

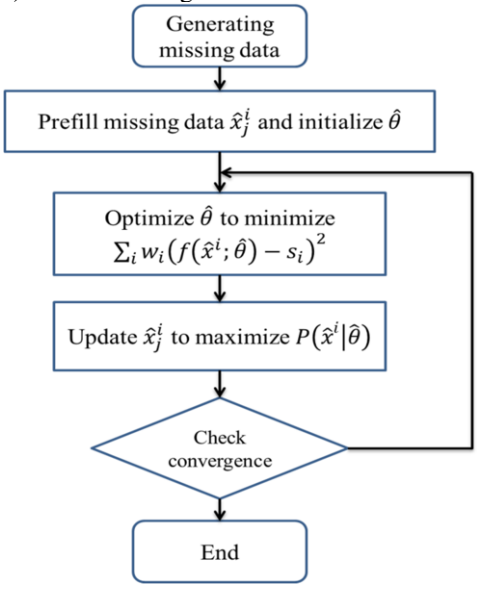

Figure 4. Algorithm in this paper.

4. Numerical Experiments

4.1. Experiments Setting

We conduct the numerical experiments with different values of N, the data generation rules and noise levels.

The implementation of Algorithm 1 in the experiments is straightforward. However, there are several details need to be clarified.

In step 2) of the algorithm, the optimal value of $\hat{\theta}$ is obtained by using optimization method. Since the objective loss function $g(\hat{\theta}) = \sum_i w_i \left(f(\hat{x}^i; \hat{\theta}) - s_i\right)^2$ is convex with respect to $\hat{\theta}$ in our setting, common convex optimization techniques are used to accelerate its convergence.

In step 3), the search for the most suitable missing values $\hat{x}_j^i$ is realized using Monte-Carlo method. We generate multiple possible values of $\hat{x}_j^i$ and compute their probabilities. Then we choose the value with the maximum probability. The probability $P(\hat{x}^i|\hat{\theta})$ is computed by counting the number of complete data points in the neighbourhood of $\hat{x}^i$. Namely,

$$P(\hat{x}^i|\hat{\theta}) \approx \frac{|\mathcal{D} \cap B(\hat{x}^i, \epsilon)|}{N},$$

where ϵ is some small positive number and $B(\hat{x}^i, \epsilon)$ represents the ball neighbourhood of $\hat{x}^i$ with radius ϵ. For the choice of ϵ, we conduct several experiments with exactly same settings but different ϵ. The result is shown as in figure 5.

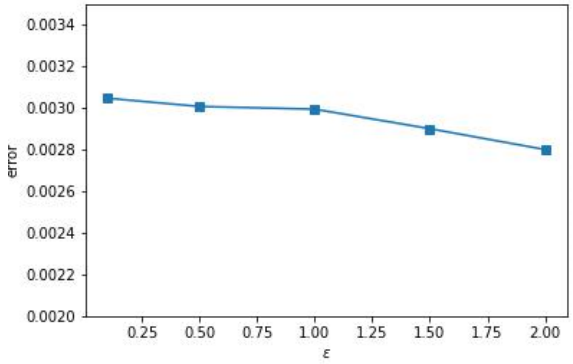

Figure 5. Error of θ with different ϵ

The difference here is neglectable. Hence in this paper we set $\epsilon = 1$ for all the computations. In our experiments, it turns out that this strategy is quite effective.

4.2. Experiment Results

We experiment our algorithm for different settings of N and data generation rules. Set the real parameter $\theta = (\theta_1, \theta_2, \ldots, \theta_{10})$, where $\theta_i = i - 1, i = 1,2, \ldots,10$. Set the initial value $\hat{\theta}$ randomly, then do the optimization step from then on. Since the likelihood function increases for each optimization step, it is guaranteed to reach convergence of $\hat{\theta}$ after certain steps of optimization. To ensure the convergence, we set the number of

optimization steps to 100. In this setting, we find that all the experiments recorded do converge. For example, in one of the experiments, the training curve is depicted in figure 6. The training loss $g(\hat{\theta})$ and relative error of $\hat{\theta}$ both decrease as the number of iteration increases. The optimization loss converges to zero after certain iterations, while the relative error converges to some nonzero value.

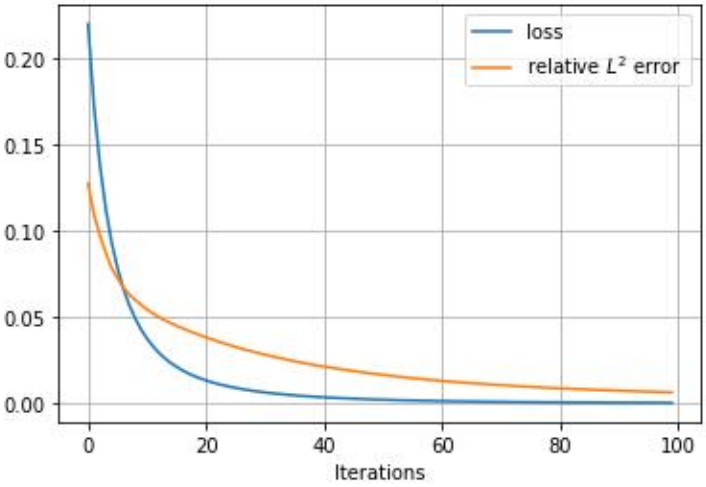

Figure 6. A typical optimization curve for the algorithm

4.3. Results Obtained From Generation Rule A

We generate the data and missing values based on rule A we mentioned in 2.2. We do the numerical experiments for different settings of N and p. In each simulation, we record the final relative L^2 error after convergence of $\hat{\theta}$.

$$E = \frac{\|\hat{\theta} - \theta\|_2}{\|\theta\|_2} = \sqrt{\frac{\sum_i (\hat{\theta}_i - \theta_i)^2}{\sum_i \theta_i^2}}$$

To mitigate the uncertainty caused by randomness, for each circumstance, we repeat the experiment for 10 times and compute the mean value of the results. The results are shown in Table 2.

Table 2. Relative L2 error for different N and p in generation rule A

N	$p = 0.2$	$p = 0.3$	$p = 0.4$	$p = 0.5$
100	0.033001	0.084779	0.12625	0.170639
200	0.022362	0.042355	0.081252	0.105003
500	0.005187	0.017401	0.033747	0.120283
1000	0.006314	0.014972	0.031985	0.071583
2000	0.005802	0.012728	0.022353	0.045894
3000	0.001292	0.005141	0.017484	0.036037
4000	0.000979	0.004929	0.013566	0.040897
5000	0.001160	0.005343	0.012331	0.032995
10000	0.000831	0.004699	0.010588	0.028801

The relation between the relative L^2 error and different choices of N and p are shown in the figure 7. As we can see, for all cases of p, the relative error decreases as N grows. With larger missing value proportion p, the relative error gets larger. These observations agree with the intuition. Moreover, the decreasing trend of error curve appears linear with respect to N in the logarithmic scale.

In addition, we can roughly infer the minimum number of samples needed in the dataset if we want to achieve certain accuracy of the estimation. For example, we choose 3% as our target error, then we can draw the graph showing how the increasing missing proportions affect the number of samples needed. As is shown in figure 8, the relation between p and N appears exponential. This is in accordance with the observation that the information contained in the sample is of exponential relation with the number of missing values.

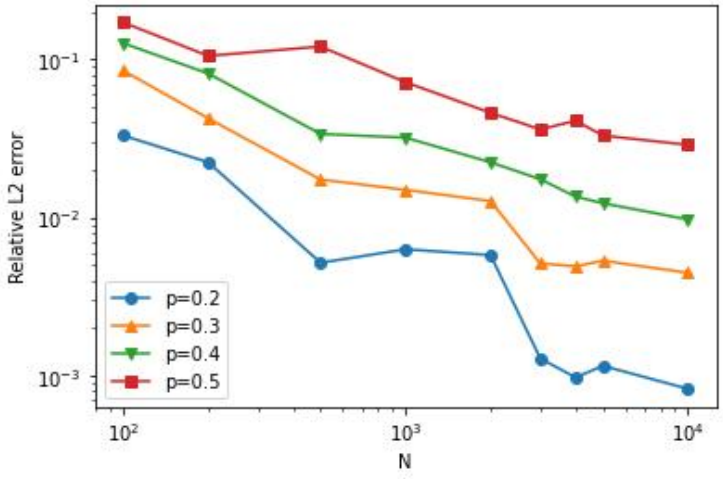

Figure 7. The trend of relative L2 error corresponding to N and p

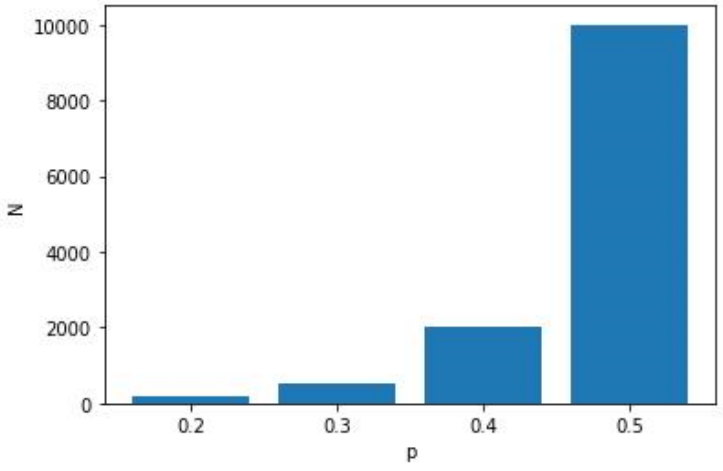

Figure 8. The rough number of samples needed to reach 3% accuracy for different p

4.3.1 Results Obtained From Generation Rule B

Similarly, we generate the data and missing values based on rule B we mentioned in 2.2. We do the numerical experiments for different settings of N and α. Each setting the simulation repeats 10 times to prevent uncertainty. In each simulation, we record the final relative L^2 error after convergence of $\hat{\theta}$. The experiment results are shown in table 3.

Table 3. Relative L2 error for different N and α in generation rule B

N	$\alpha = 0$	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$
100	0.170347	0.165375	0.226244	0.225555
200	0.214389	0.099733	0.2097	0.179663
500	0.021707	0.075418	0.066721	0.159187
1000	0.025434	0.046212	0.05293	0.066281
2000	0.021199	0.03759	0.050619	0.063558
3000	0.019379	0.037268	0.046212	0.058138
4000	0.017889	0.034286	0.043231	0.053666
5000	0.016063	0.028757	0.038555	0.048034
10000	0.011643	0.0167	0.028324	0.032796

We visualize the relation between the error and N with different α's in figure 9. It is obvious that the error decreases as N increases. This trend of error corresponding to N is similar to the results of previous experiments, which is in accordance with the intuition. The increase of α will cause more missing values, hence result in the increasing error. For all the α, the relation between the error and N also appears linear in the logarithmic scale.

Moreover, we compute the minimum number of samples needed in the dataset if we want to achieve an estimation of less than 3% error. As is in shown in figure 10, with larger α, we need exponentially larger N to compensate the missing information in order to obtain the same accuracy.

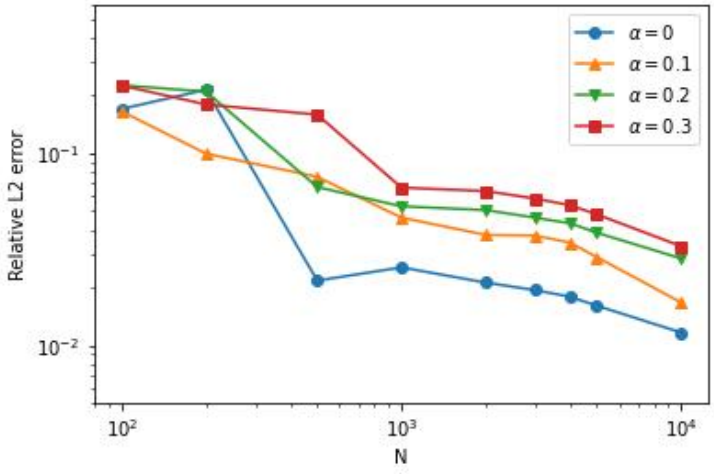

Figure 9. The trend of relative error corresponding to N and α

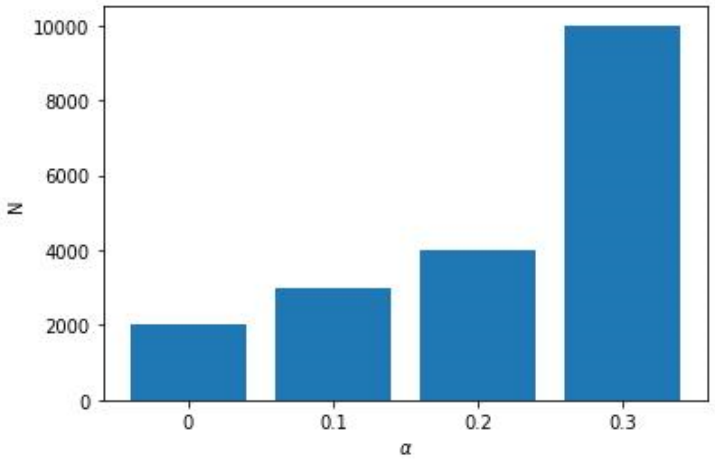

Figure 10. the rough number of samples needed to reach 3% accuracy for different α

4.3.2 Experiments With Noised Data

We add different levels of noise to the s_i to test the stability of our algorithm. Set the number of samples to be 1000, 5000 and 10000 respectively, and the noises range to be one of $\{0\%, 2\%, 5\%\}$. Then we run the algorithm on both missing value generation rules.

First, we test the noisy samples on generation rule A. As shown in table 4 and figure 11, the noise has a significant impact when less missing data are present, while the noise does not play an important role for the case of large amounts of missing data.

Table 4. Experiment results of noisy data using generation rule A.

N	Noise	$p = 0.2$	$p = 0.3$	$p = 0.4$	$p = 0.5$
1000	0	0.001071	0.007503	0.027267	0.085742
	2%	0.007016	0.017146	0.034906	0.087117
	5%	0.015657	0.032796	0.040249	0.087952
5000	0	0.00116	0.005343	0.012331	0.032995
	2%	0.004981	0.007454	0.017889	0.03656
	5%	0.013152	0.011371	0.019444	0.040623
10000	0	0.00083	0.004487	0.009737	0.028801
	2%	0.00394	0.005745	0.010023	0.023973
	5%	0.009036	0.015262	0.014419	0.024048

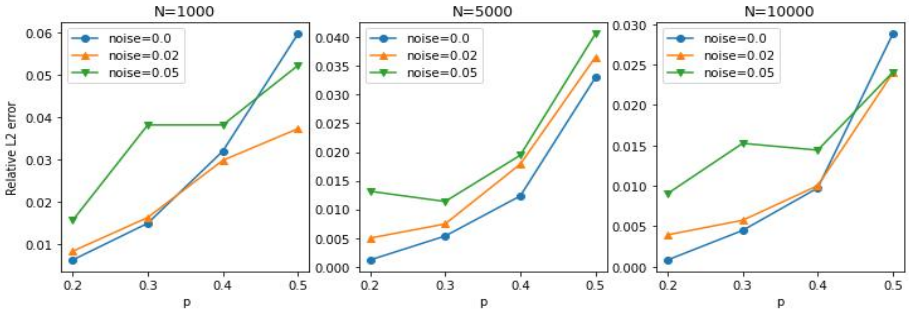

Figure 11. From left to right: relative l2 errors for the experiments with sample size equal to 1000, 5000 and 10000 respectively. For each choice of sample size, noise level equal to 0, 2%, 5% is tested for different p.

Then we go through the same process for the generation rule B. Table 5 and figure 12 present the experiment results using different noise levels and α. As can be seen from the figure, we have similar findings compared to rule A. When the data is relatively complete, the increase of noise will greatly raise the error. When the data is sparse, the impact of the noise is not significant.

Table 5. Experiment results of noisy data using generation rule B

N	Noise	$\alpha = 0$	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$
1000	0	0.013800	0.026393	0.045067	0.087179
	2%	0.040442	0.029586	0.045406	0.096519
	5%	0.040895	0.044721	0.056588	0.093349
5000	0	0.00116	0.005343	0.012331	0.032995
	2%	0.004981	0.007454	0.017889	0.03656
	5%	0.013152	0.011371	0.019444	0.040623
10000	0	0.00083	0.004487	0.009737	0.028801
	2%	0.00394	0.005745	0.010023	0.023973
	5%	0.009036	0.015262	0.014419	0.024048

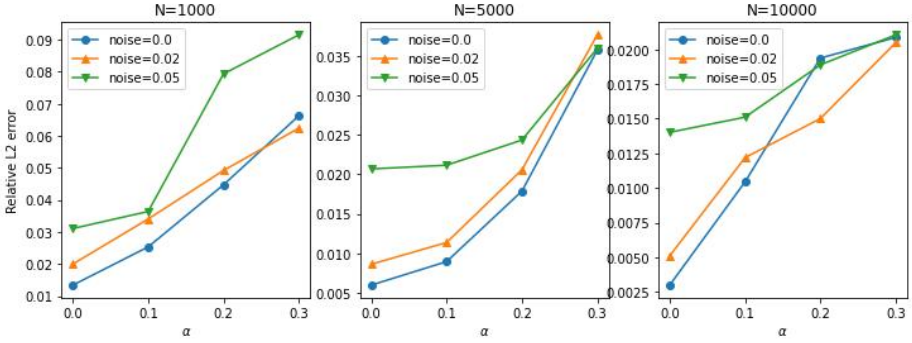

Figure 12. From left to right: relative l2 errors for the experiments with sample size equal to 1000, 5000 and 10000 respectively. For each choice of sample size, noise level equal to 0, 2%, 5% is tested for different α.

5. Conclusion and Discussion

This paper proposes an approach to deal with large chunk of missing data to achieve a relatively accurate estimation on the hidden parameters. This approach is based on the theory of maximum likelihood estimation and the EM algorithm. Through the numerical experiments of different settings of the problem, it leads to the conclusion that the algorithm we proposed can achieve a relatively accurate estimation of the hidden parameter even if more than half values are missing from the samples, as long as enough samples are provided. With more missing data, we need exponentially more samples to achieve certain accuracy. And the noise of the sample is a huge factor when the data is relatively complete, while it is not significant when the missing values are prevalent.

Further study is needed to find the exact relations among the distribution of the missing values, the number of samples and the relative error of the estimation. Besides, since our research makes the simple assumption that all the samples and parameters are independent and the values are missing completed at random, it is natural to consider the case when the parameters have certain correlations or the data does not miss at random. In that case, more information can be gained from the missing data because we can infer the relation between the parameters using more machine learning tools.

References

[1] Little R., Rubin D. Statistical analysis with missing data, 2nd ed. [M]. Hoboken, New Jersey: John Wiley & Sons, Inc.; 2002.
[2] Rubin D. Multiple imputation for nonresponse in surveys [M]. Hoboken, New Jersey: John Wiley & Sons, Inc.; 1987.
[3] Tutz G., Ramzan S. Improved Methods for the Imputation of Missing Data by Nearest Neighbour Methods [J]. Computational Statistics & Data Analysis, 2015, 90: 84-99.
[4] Hathaway R. and Bezdek J. Fuzzy C-means Clustering of Incomplete Data [J]. IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics, 2001, 31(5): 735-744.
[5] Raghunathan E., Lepkowski M., Van Hoewyk J., et al. A multivariate technique for multiple imputing missing values using a sequence of regression models [J]. Survery Methodology, 2001, 27(1): 85-96.
[6] Dempster P., Laird N. M., Rubin D. B. Maximum likelihood from incomplete data via the EM algorithm [J]. Journal of the royal statistical society. Series B (methodological), 1977: 1-38.
[7] Murray J. Multiple Imputation: A Review of Practical and Theoretical Findings [J]. Statistical Science, 2018, 33(2), 142–159.
[8] Smieja M., Struski L., Tabor J. et al. Processing of missing data by neural networks [C]. 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.
[9] Cloudhury S., Pal N. Imputation of missing data with neural networks for classification [J]. Knowledge-Based Systems, volume 182, 2019.
[10] Jerez J., Molina I., et al. Missing data imputation using statistical and machine learning methods in a real breast cancer problem [J]. Artificial Intelligence in Medicine, 50(2010), 105-115.
[11] Hastie T., Tibshirani R., Friedman J. Elements of statistical learning –Data Mining, Inference, and Prediction, 2nd ed. [M]. Springer-Verlag, 2009.
[12] Sammaknejad N., Zhao Y., Huang B. A review of the Expectation Maximization algorithm in data-driven process identification [J]. Journal of Process Control 73(2019), 123-136.