Truncated Matrix Completion - An Empirical Study

Rishhabh Naik, Nisarg Trivedi, Davoud Ataee Tarzanagh, Laura Balzano
Electrical and Computer Engineering
University of Michigan
Ann Arbor, MI, USA
{rishhabh, nisargtr, tarzanaq, girasole}@umich.edu

***Abstract*—Low-rank Matrix Completion (LRMC) describes the problem where we wish to recover missing entries of partially observed low-rank matrix. Most existing matrix completion work deals with sampling procedures that are independent of the underlying data values. While this assumption allows the derivation of nice theoretical guarantees, it seldom holds in real-world applications. In this paper, we consider various settings where the sampling mask is dependent on the underlying data values, motivated by applications in sensing, sequential decision-making, and recommender systems. Through a series of experiments, we study and compare the performance of various LRMC algorithms that were originally successful for data-independent sampling patterns.**

***Index Terms*—Truncated Matrix Completion, Low Rank Matrices, Missing Not at Random.**

I. Introduction

Consider the Matrix Completion (MC) problem, where the goal is to estimate an unknown $m \times n$ matrix $\mathbf{X}^*$ given only few of its entries, possibly corrupted by noise. MC has found applications in diverse fields, including recommender systems [16], environmental sensing [15], sequential decision-making [10], computer vision [13], and bioinformatics [31], to name a few. In recommender systems (RS), for example, the matrix $\mathbf{X}^*$ corresponds to ratings of items (columns) by users (rows). MC in this case corresponds to predicting the ratings of all users on all items based on a few observed ratings.

For the MC problem to be well-posed, the following assumptions are used extensively: i) $\mathbf{X}^*$ is inherently of low-rank $r \ll \min(m, n)$; ii) $\mathbf{X}^*$ satisfies incoherence conditions [8]; and iii) the sample of observed entries is random and independent from the matrix $\mathbf{X}^*$. Let $\Omega \subseteq [m] \times [n]$ be the subset of observed indices, and $\mathbf{X}$ the matrix with observed entries in Ω and zero in its complement Ω^c. Let $\|\mathbf{A}\|^2_{\mathrm{F}(\Omega)} := \sum_{(i,j)\in\Omega} a^2_{ij}$ for any matrix $\mathbf{A} = (a_{ij})$. Then, low-rank MC (LRMC) problem is

$$\min_{\mathbf{Z}} \|\mathbf{Z} - \mathbf{X}\|_{\mathrm{F}(\Omega)} \quad \text{subject to} \quad \operatorname{rank}(\mathbf{Z}) \leq r. \tag{1}$$

In many real datasets, the low-rank assumption is at least approximately realistic. Further, low-rank approximations often yield matrices that generalize well to the unobserved entries. Incoherence is another sensible assumption; it essentially requires that key information for a good low-rank approximation can be captured with only a small sample of the matrix entries. The assumption on the independence of the sampling mask Ω, however, does not often hold in real world applications [1]. For instance, in a movie-ratings matrix for RS, users are more likely to watch/rate movies that they will like. The probability of observing ratings for those movies would be higher than the other ones, thus violating independence from $\mathbf{X}^*$. For another example, consider a collection of environmental sensors that monitor chemical levels in water. Those sensors are typically most accurate in a certain calibrated range, and outside that range the sensors will return a truncated value [21]. These data can be treated as missing, and so in this case the missing entries are deterministically dependent on the values of $\mathbf{X}^*$.

Most methods for LRMC can be assigned to one of the two classes: One class consists of algorithms that optimize over all possible $m \times n$ matrices while encouraging low-rank [8, 19], whereas the second class consists of methods that explicitly enforce the rank r constraint in (1) [5, 14]. Several methods in the first class replace the rank constraint by a low-rank inducing penalty [8, 19, 29]. Most MC methods were not designed with matrix-dependent sampling patterns in mind. The purpose of this paper is to investigate their performance in this setting.

Our Contributions. In this paper, we study the MC problem with data-dependent observations, i.e., when Ω depends on $\mathbf{X}^*$. Specifically, we give numerical results with three synthetic sampling patterns motivated by real-world applications that violate the standard assumption of data-independent sampling. We show that some state-of-the-art MC algorithms consistently recover $\mathbf{X}^*$ under dependent sampling while others do not.

Notation. Vectors and matrices are denoted by bold letters, i.e., $\mathbf{x}$ and $\mathbf{X}$, with their elements indexed as x_i and x_{ij}, respectively. Operations such as $\mathbf{x} > 0$, $\mathbf{X} \geq 0$ etc. are applied element-wise. The singular values are assumed to be arranged in non-increasing order, i.e., $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$. For any matrix $\mathbf{X}$, we define $\|\mathbf{X}\|_* := \sum_i \sigma_i^2$, $\|\mathbf{X}\|_{\mathrm{F}} := \sqrt{\sum_{ij} |x_{ij}|^2}$, and $\|\mathbf{X}\|_{\mathrm{F}(\Omega)} = \sqrt{\sum_{(i,j)\in\Omega} x^2_{ij}}$. The sampling operator $\mathcal{P}_\Omega$ extracts the entries of a matrix according to Ω, such that $\mathcal{P}_\Omega(\mathbf{X})$ is a vector with entries x_{ij} for $(i, j) \in \Omega$.

II. Related Work

Related work falls into three categories: MC with data missing completely at random (MCAR), MC with data missing at random (MAR), and MC with data missing not at random (MNAR).

MC with data missing completely at random (MCAR). The classical formulation and study of the LRMC problem focuses on data-independent sampling patterns with MCAR assumption [8, 5, 32, 19, 17, 12]. In particular, they assume that each entry is revealed at random, independent of everything else, with *uniform* probability. Under this assumption, the theoretical analysis of LRMC methods including finite-sample consistency for both convex [8] and non-convex formulations [14] has been provided. However, MCAR is likely unrealistic due to the presence of *unobserved factors* that determine both the entries of the underlying matrix and the missingness pattern in the observed matrix [1].

MC with data missing at random (MAR). In this case, entries are revealed at random, independent of the underlying matrix and conditioned on observed covariates, with *non-uniform* probability [18]. For example, in RS, covariates are demographic and social network information of users, tags and content information of items. When the probabilities are non-uniform, the MC objective is biased [23] which makes LRMC more challenging. To overcome this issue, some recent works advocate de-biasing the objective by, for example, propensity estimation techniques [20, 23]. The work in [9] proves LRMC methods succeed with sampling dependent on the leverage of the rows/columns of the matrix.

MC with data missing not at random (MNAR). In this case, the missingness pattern of the matrix can be *dependent* on the underlying values in that matrix, and observing the outcome of one entry can alter the probability of observing another. To address these challenges, there has been exciting recent progress on MC methods including [6, 25, 30] under limited version of MNAR. In particular, they assume the probability of observing each entry is a nice function solely of latent factors.

A related literature that shares similarities with MNAR is MC with *deterministic* sampling pattern [2, 3, 22, 24]. In particular, [2, 3] consider very restricted sparsity patterns that are not particularly suitable for important applications of MC that arise in RS or sequential decision-making. [22] gave deterministic sampling conditions for unique completability and [24] provided an algebraically verifiable sufficient condition for the local uniqueness of minimum rank MC solutions. Despite sharing some similarities with MNAR, their missing pattern still does not depend on the matrix values. Our work is closely related to [1] where the authors proposed a causal model with provable guarantees to analyze MC with MNAR data where the probability that an entry of the matrix is missing can *depend* on the underlying values in the matrix itself and depend on which other entries are missing. Motivated in part by [1], we aim to investigate the efficacy of the state-of-the-art MC algorithms for recovering $\mathbf{X}^*$ under various data-independent missing patterns.

III. Sampling Schemes for Empirical Evaluation of Truncated Matrix Completion

In this section, we describe MC problems where matrix entries are observed with a data-dependent distribution.

A. ReLU-based Sampling (ReLU-S)

In this problem, only non-negative entries of a matrix are observed. The motivation for this mask setting is two-fold: first, understanding recoverability of a ReLU-thresholded matrix can provide insights about how deep neural networks are able to learn despite the potential loss of information through multiple ReLU layers [27]. Second, ReLU is just a special case of more generalized thresholding. Indeed, many real-world applications involve data clipped outside of certain range, e.g., voltage clipping devices [21]. Understanding recovery of ReLU-thresholded matrices will help us in this more generic and useful case of matrices thresholded from both sides. Given $\mathbf{X}^*$, let $\mathbf{X} \in \mathbb{R}^{m\times n}$ be such that $\mathbf{X} = \text{ReLU}(\mathbf{X}^*)$. Therefore the observation mask Ω is

$$(i,j) \in \Omega \quad \text{if} \quad x^*_{ij} \geq 0. \tag{2}$$

B. Group-Specific Sampling (GS-S)

Arguably, the most well-known application of MC is RS, ubiquitous in modern online platforms. The motivation behind GS-S setting is that it seeks to mimic the structure of the data in RS. In particular, we aim to design a mask setting that reflects the self-selection bias phenomena, where most users tend to provide ratings if they particularly liked or disliked an item. However, they are much less inclined to provide a rating or even try out an item that they are lukewarm about.

The data matrix was generated using $\mathbf{X}^* = \mathbf{U}\mathbf{V}^\top$ and was re-scaled to have all entries between 1 and 5. This is similar to RS where users' ratings are in $[1, 5]$. In GS-S setting, the mask Ω depends on the entries of $\mathbf{X}^*$ with probabilities

$$p_{ij} = \begin{cases} 0.8, & \text{if } x^*_{ij} \in [1,2] \cup [4,5] \\ 0.2, & \text{if } x^*_{ij} \in (2,4). \end{cases} \tag{3}$$

This equation defines the mask Ω with smaller probability for moderate rating or entries. Note that one can use a different set of probabilities or clustering of $\mathbf{X}$ in (3) for specific applications.

C. Mean-Centric Truncated Sampling (MCT-S)

Environmental sensing is a field where low-rank models are appropriate [4, 26], but missing data issues can be so prevalent to make it too challenging to estimate low-rank models. The motivation behind the MCT-S setting is that it seeks to mimic the data structure of measurements from sensor nodes. For example, assume that the sensor measurements are most accurate only within a specific calibrated range. We can attempt to recover entries outside that range.

We aim to recover entries of matrix $\mathbf{X}^*$ with values on either side of the matrix mean $(mn)^{-1}\sum_{i=1}^m\sum_{j=1}^n x^*_{ij}$, by only observing the entries lying within a certain range. In our numerical experiments, we find an α so that we observe $\approx 50\%$ of the total entries, i.e.,

$$(i,j) \in \Omega \quad \text{if} \quad |x^*_{ij}| \leq \alpha \implies |\Omega| \approx \frac{mn}{2}\,. \tag{4}$$

D. Uniformly at Random Sampling (UAR-S)

Uniform sampling is the most common model of missingness assumed in the MC literature [8, 14, 19, 12]. Although UAR-S is likely unrealistic outside experimental settings, this regime remains a popular abstraction in machine learning and statistics to study MC problems. In the UAR-S scheme, the sampling is independent of the underlying distribution of the data and does not take any information from the matrix for the same.

IV. Experiments

This section gives simulation results demonstrating the performance of several state-of-the-art algorithms on mask settings described in Section III. The following convex and non-convex MC methods are compared through experimental results:

CVX: Here, we directly solve the convex relaxation of [8], i.e., minimizing the nuclear norm subject to the linear constraints produced by the observed matrix entries. The CVX environment is a general modelling system for solving disciplined convex problems [11]. We used the SDPT3 solver [28], a primal-dual interior-point algorithm that uses the path-following paradigm to solve semi-definite programming problems.

FPCA: The Fixed Point Continuation with Approximate SVD algorithm [19] solves

$$\min_{\mathbf{X} \in \mathbb{R}^{m \times n}} \mu ||\mathbf{X}||_* + \frac{1}{2} ||\mathbf{Y} - \mathbf{X}||_F^2 \tag{5}$$

using a fixed point iterative scheme through operator splitting. Here, $\mu > 0$ is a regularization parameter. The default implementation of FPCA [19] has a demo with UAR-S sampling.

NNLS: Nuclear Norm Least Squares uses an accelerated proximal gradient algorithm to solve the composite problem (5). The default implementation of NNLS [29] has a demo with UAR-S sampling.

R2RILS: Rank $2r$ Iterative Least Squares by [5] is a factorization based method that estimates the row and column subspaces of $\mathbf{X}^*$, simultaneously. More specifically, given the current estimate $(\mathbf{U}_t, \mathbf{V}_t)$, R2RILS first solves

$$\min_{\mathbf{A} \in \mathbb{R}^{m \times n},\ \mathbf{B} \in \mathbb{R}^{m \times n}} ||\mathbf{U}_t \mathbf{B}^\top + \mathbf{A} \mathbf{V}_t^\top - \mathbf{X}||_{\mathrm{F}(\Omega)} \ , \tag{6}$$

and then uses the minimizers to obtain the new row and column subspace estimates $(\mathbf{U}_{t+1}, \mathbf{V}_{t+1})$. In the default implementation of R2RILS, Ω was generated using a $\mathrm{Binom}(m.n, p)$. Here, $p = \rho \cdot r(m+n-r) \ / \ (mn)$ and $\rho = |\Omega| \ / \ r(m+n-r)$ is oversampling ratio.

GNMR: Gauss-Newton Algorithm for Matrix Recovery [32] uses the Gauss-Newton linearization to solve the nonconvex MC problem. More specifically, given the current iterate $(\mathbf{U}_t, \mathbf{V}_t)$, the new iterate $(\mathbf{U}_{t+1}, \mathbf{V}_{t+1})$ is generated by solving

$$\min_{\mathbf{U} \in \mathbb{R}^{m \times r}, \mathbf{V} \in \mathbb{R}^{n \times r}} \|\mathcal{P}_\Omega(\mathbf{U}_t \mathbf{V}^\top + \mathbf{U} \mathbf{V}_t^\top - \mathbf{U}_t \mathbf{V}_t^\top) - b\|^2. \tag{7}$$

Here, b is the vector of observed entries. In the original implementation of GNMR [32], the mask Ω was repeatedly sampled uniformly without replacement until the condition for unique recovery of the matrix [22] was satisfied.

During our initial empirical tests we observed that SNN (Synthetic Nearest Neighbors introduced in [1]) did not converge for our mask settings. We conjecture this is because we had 50% missing entries; the SNN algorithm seemed to work well in cases where a smaller fraction of entries were missing.

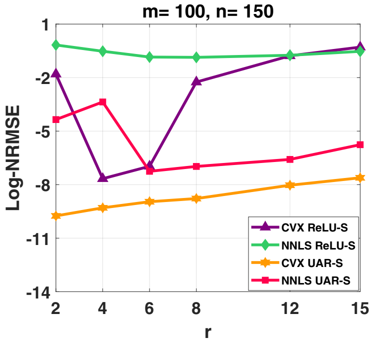

Fig. 1: Performance of CVX and NNLS algorithms under ReLU-S and UAR-S schemes.

Next, we discuss data generation and mask setting. For ReLU-S and MCT-S patterns, we generate a rank r matrix $\mathbf{X}^*$ by sampling two factors $\mathbf{U} \in \mathbb{R}^{m \times r}$ and $\mathbf{V} \in \mathbb{R}^{n \times r}$ with i.i.d. Gaussian entries and set $\mathbf{X}^* = \mathbf{U}\mathbf{V}^\top$. In the GS-S case, seeking to mimic the structure in RS, we sample $\mathbf{U} \in \mathbb{R}^{m \times r}$ and $\mathbf{V} \in \mathbb{R}^{n \times r}$ with i.i.d. random entries uniformly distributed in the interval $[0, 1]$ and then modify their multiplications such that each x_{ij} lie between 1 and 5.

To ensure fair comparison, we set the parameters such that all methods observe about 50 % of the entries. As discussed in Section III , the matrices for ReLU-S and MCT-S were generated using Gaussian entries while GS-S used randomly sampled uniform entries. The sampling schemes were explained in Section III.

We follow the MATLAB implementations provided by the authors and observe that the default parameters provide near-optimal performance; see Table I.

FPCA	μ = 10^{-8}, xtol = 10^{-6}, $\mathrm{iter_{max}}$= 500, τ= 1
NNLS	μ = 10^{-8}, xtol= 10^{-4}, $\mathrm{iter_{max}}$= 100, τ= 1
R2RILS	$\mathrm{t_{out}}$= 40, $\mathrm{t_{in}}$= 150, rtol $= 10^{-11}$, $\mathrm{t_{max}}$= 40
GNMR	$\mathrm{t_{out}}$= 100, $\mathrm{t_{in}}$= 2000, $\mathrm{rel_{res}} = 10^{-11}$

TABLE I: Parameter values for MC algorithms.

Let $\mathbf{X}^*$ denote true unknown matrix and $\hat{\mathbf{X}}$ be the output of the above algorithms. The prediction accuracy of $\hat{\mathbf{X}}$ is defined by the normalized root mean square error:

$$\text{NRMSE} := \frac{\|\hat{\mathbf{X}} - \mathbf{X}^*\|_{\mathrm{F}}}{\|\mathbf{X}^*\|_{\mathrm{F}}} \text{ and}$$
$$\text{Log-NRMSE} := \log_{10} \text{NRMSE}.$$

For each problem with $m \times n$ matrix $\mathbf{X}^*$ with rank r, we solve 50 independently created MC problems for ReLU-S and MCT-S and 20 independent problems for GS-S experiments. Table I contains additional stopping criterion on the tolerance and the number of iterations used for convergence of the algorithms. We will consider an NRMSE of the order of 10^{-4} or higher

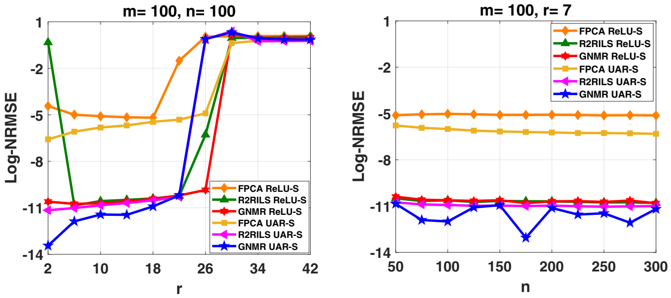

Fig. 2: Performance of MC algorithms under ReLU-S and UAR-S schemes.

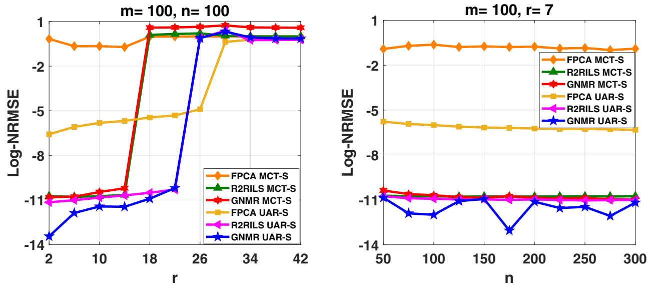

Fig. 3: Performance of MC algorithms under MCT-S and UAR-S schemes.

as a failure case, successful otherwise. We note that in these artificial settings without added noise, recovery at machine precision is theoretically possible (NRMSE 10^{-11} or smaller), however we haven't used this as our success threshold, since it would not be appropriate in more realistic scenarios.

In Figure 1, we compare the performance of convex optimizers NNLS and CVX for MC under the ReLU-S and UAR-S settings. Interestingly, despite being a powerful convex solver, CVX recovers matrices only for very low-rank structures (above 2) under the ReLU-S pattern. NNLS completely fails to recover the matrices under this pattern. Our initial experiments indicated that these convex optimizers gave poor results under MCT-S and GS-S as well. Hence, we leave as future work a more extensive investigation into why these convex optimization approaches fail to consistently recover matrices when the sampling is MNAR and dependent on $\mathbf{X}^*$.

In Figure 2, we find that algorithms FPCA, R2RILS and GNMR recover low-rank matrices sampled under the ReLU-S pattern successfully when r and n are varied separately. Furthermore, the non-convex approaches R2RILS and GNMR recover the matrices under this pattern till ranks similar to that in the case of UAR-S. On the other hand, FPCA, which solves the convex relaxation, begins to fail at smaller matrix rank with ReLU-S than with UAR-S.

Under the MCT-S pattern in Figure 3, the non-convex approaches R2RILS and GNMR with pre-specified rank r recover low rank matrices successfully. FPCA, on the other hand, fails to recover matrices sampled under this pattern for even very low rank structures. The non-convex approaches although successful, have the drawback that matrices are recovered for structures of significantly lesser ranks than in the case of under UAR-S.

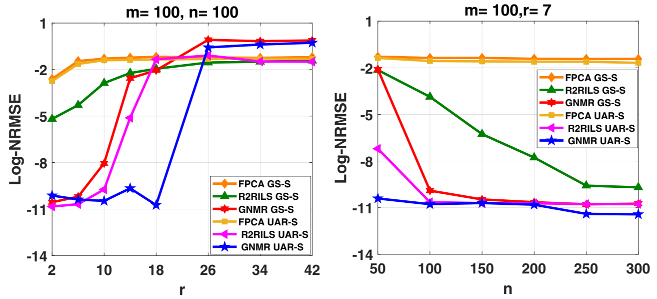

Fig. 4: Performance of MC algorithms under GS-S and UAR-S schemes.

Figure 4 shows the performance of FPCA, R2RILS and GNMR under GS-S pattern. One can notice that FPCA completely fails to recover the underlying matrix for both GS-S and UAR-S patterns. We believe that the failure is due to the large set of group structures in $\mathbf{X}$ in both cases while being independent of other entries in UAR-S. This is consistent with the group-specific RS in the literature. For example, [7] showed that convex MC methods can lead to a larger value of the prediction error or even failure if the data have some non-uniform group structures. We want to highlight here that the performance on UAR-S data in Figure 4 is different from that in earlier figures because of different data generation procedures. While comparing the performance of ReLU-S and MCT-S against UAR-S, we sample uniformly over the data matrix which is generated using low rank factorization where the matrices $\mathbf{U}$ and $\mathbf{V}$ have i.i.d. Gaussian entries. When comparing with GS-S, we use another UAR-S setting where the sampling is uniformly done over the data matrix generated using random i.i.d. entries and rescaled to values between 1 and 5. The entries are not normalized, unlike the UAR-S setting for the ReLU-S and MCT-S. In GS-S, we observe that FPCA fails even under the UAR-S setting, and the R2RILS performance is diminished. Similar to ReLU-S and MCT-S experiments, GNMR outperforms the rest of the algorithms under GS-S pattern. GNMR recovers the matrix even when the mask Ω is both data-dependent and MNAR for r and n varied separately.

We observe that across all three sampling schemes described in the paper, the non-convex approaches R2RILS and GNMR perform well across the settings, for r and n varied individually. GNMR performs better than the other algorithms in the GS-S setting where the structure closely resembles the nature of data in recommender systems.

V. Conclusion

In this paper, we studied MC problem using three realistic sampling patterns which are data-dependent, i.e., where Ω depends on $\mathbf{X}^*$. Our extensive numerical evaluations showed that MC is successful for different truncated sampling patterns, despite the standard UAR-S assumption being violated. In particular, our results indicate that non-convex SoTA algorithms

R2RILS and GNMR consistently recover matrices under such realistic sampling patterns. Across all experimental settings, we find that GNMR consistently outperforms other algorithms described in this paper. Although initially designed for UAR-S, it performs well for the MC with data-dependent observations.

VI. Acknowledgements

The authors thank Avi Tachna-Fram and Anya Martin for their early contributions. This work was supported by ARO YIP award W911NF1910027 and NSF BIGDATA award IIS-1838179.

References

[1] A. Agarwal, M. Dahleh, D. Shah, and D. Shen. Causal matrix completion. *arXiv preprint arXiv:2109.15154*, 2021.

[2] S. Athey, M. Bayati, N. Doudchenko, G. Imbens, and K. Khosravi. Matrix completion methods for causal panel data models. *J. Am. Stat. Assoc.*, 2021.

[3] J. Bai and S. Ng. Matrix completion, counterfactuals, and factor analysis of missing data. *J. Am. Stat. Assoc.*, 2021.

[4] L. Balzano and R. Nowak. Blind calibration of sensor networks. In *the 6th international conference on Information Processing in Sensor Networks*, 2007.

[5] J. Bauch, B. Nadler, and P. Zilber. Rank 2r iterative least squares: efficient recovery of ill-conditioned low rank matrices from few entries. *SIAM Journal on Mathematics of Data Science*, 3(1):439–465, 2021.

[6] S. Bhattacharya and S. Chatterjee. Matrix completion with data-dependent missingness probabilities. *arXiv preprint arXiv:2106.02290*, 2021.

[7] X. Bi, A. Qu, J. Wang, and X. Shen. A group-specific recommender system. *J. Am. Stat. Assoc.*, 2017.

[8] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Found. Comput. Math*, 2009.

[9] Y. Chen, S. Bhojanapalli, S. Sanghavi, and R. Ward. Completing any low-rank matrix, provably. *The Journal of Machine Learning Research*, 16(1):2999–3034, 2015.

[10] J. Grand-Clement. *Robust and Interpretable Sequential Decision-Making for Healthcare*. Columbia University, 2021.

[11] M. Grant and S. Boyd. Cvx: Matlab software for disciplined convex programming, version 2.1, 2014.

[12] P. Jain, P. Netrapalli, and S. Sanghavi. Low-rank matrix completion using alternating minimization. In *the 45th annual ACM symposium on Theory of computing*, 2013.

[13] H. Ji, C. Liu, Z. Shen, and Y. Xu. Robust video denoising using low rank matrix completion. In *IEEE Computer Society Conference on CVPR*, 2010.

[14] R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. *IEEE transactions on information theory*, 56(6):2980–2998, 2010.

[15] L. Kong, M. Xia, X.-Y. Liu, M.-Y. Wu, and X. Liu. Data loss and reconstruction in sensor networks. In *IEEE INFOCOM*, 2013.

[16] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[17] C. Kümmerle and C. M. Verdun. Escaping saddle points in ill-conditioned matrix completion with a scalable second order method. *arXiv preprint arXiv:2009.02905*, 2020.

[18] R. J. Little and D. B. Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

[19] S. Ma, D. Goldfarb, and L. Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 2011.

[20] W. Ma and G. H. Chen. Mnar in matrix completion: The effectiveness of estimating missingness probabilities under a low nuclear norm assumption. *NeurIPS*, 32, 2019.

[21] K. Ni, N. Ramanathan, M. N. H. Chehade, L. Balzano, S. Nair, S. Zahedi, E. Kohler, G. Pottie, M. Hansen, and M. Srivastava. Sensor network data fault types. *ACM Trans. Sens. Netws. (TOSN)*, 2009.

[22] D. L. Pimentel-Alarcón, N. Boston, and R. D. Nowak. A characterization of deterministic sampling patterns for low-rank matrix completion. *IEEE Journal of Selected Topics in Signal Processing*, 2016.

[23] T. Schnabel, A. Swaminathan, A. Singh, N. Chandak, and T. Joachims. Recommendations as treatments: Debiasing learning and evaluation. In *ICML (PMLR)*, 2016.

[24] A. Shapiro, Y. Xie, and R. Zhang. Matrix completion with deterministic pattern: A geometric perspective. *IEEE Transactions on Signal Processing*, 2018.

[25] A. Sportisse, C. Boyer, and J. Josse. Imputation and low-rank estimation with mnar data. *Stat. Comput.*, 2020.

[26] G. D. Thurston and J. D. Spengler. A quantitative assessment of source contributions to inhalable particulate matter pollution in metropolitan boston. *Atmospheric Environment (1967)*, 1985.

[27] N. Timor, G. Vardi, and O. Shamir. Implicit regularization towards rank minimization in relu networks. *arXiv preprint arXiv:2201.12760*, 2022.

[28] K.-C. Toh, M. J. Todd, and R. H. Tütüncü. Sdpt3—a matlab software package for semidefinite programming. *Optim. Meth. Software*, 1999.

[29] K.-C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of optimization*, 2010.

[30] C. Yang, L. Ding, Z. Wu, and M. Udell. Tenips: Inverse propensity sampling for tensor completion. In *International Conference on AISTATS*. PMLR, 2021.

[31] X. Zheng, H. Ding, H. Mamitsuka, and S. Zhu. Collaborative matrix factorization with multiple similarities for predicting drug-target interactions. In *the 19th ACM SIGKDD intern. conf. Knowl. Discov. Data Min.*, 2013.

[32] P. Zilber and B. Nadler. Gnmr: A provable one-line algorithm for low rank matrix recovery. *ArXiv*, abs/2106.12933, 2021.