Matrix completability analysis via graph k-connectivity

Dehua Cheng[†]
University of Southern California

Natali Ruchansky[†]
University of Southern California

Yan Liu
University of Southern California

Abstract

The problem of low-rank matrix completion is continually attracting attention for its applicability to many real-world problems. Still, the large size, extreme sparsity, and non-uniformity of these matrices pose a challenge. In this paper, we make the observation that even when the observed matrix is not suitable for accurate completion there may be portions of the data where completion is still possible. We propose the `CompleteID` algorithm, which exploits the non-uniformity of the observation, to analyze the completability of the input instead of blindly applying completion. Balancing statistical accuracy with computational efficiency, we relate completability to *edge-connectivity* of the graph associated with the input partially-observed matrix. We develop the `MaxKCD` algorithm for finding *maximally k-edge-connected components* efficiently. Experiments across datasets from a variety of applications demonstrate not only the success of `CompleteID` but also the importance of completability analysis.

1 Introduction

Low-rank matrix completion is a tool that has been widely adopted across a variety of applications. The problem takes as input a partially-observed matrix and asks for a completion of the missing values such that the estimate is low rank. The large adoption of matrix completion in practice is supported by a body of elegant theoretical and practical results surrounding low-rank matrix completion [3, 4, 9, 10, 12, 13, 14, 31]. To provide guarantees on the quality of completion, the observations are typically assumed to sufficiently cover the matrix, both in quantity and uniformity. For example, Candès and Tao [5] showed that when $\Omega(nr\text{polylog}(n))$ entries are observed uniformly at random, an $n \times n$ incoherent matrix of rank r can be accurately completed with high probability. Unfortunately, it is often the case that real-world data does not satisfy the assumptions required for accurate completion; not only is the observed data highly sparse, but the entries are rarely observed uniformly at random [19]. In this case, blindly applying a completion algorithm does not guarantee accurate completion, and runs the danger of poor quality estimates.

However, even when the matrix as a whole is not well suited for existing completion algorithms, there may very well be submatrices that can still be completed reliably. By examining the sparsity and non-uniformity of the observed data, valuable information on the location of the *completable* submatrices van be extracted. Identifying such submatrices isolates parts of the data that can be completed accurately and more efficiently due to the reduced problem size. When the matrix estimate is used to guide an action, such recommendation, traffic routing, or experimentation, it is undoubtedly important to know which parts of the estimate can be relied upon. Moreover, knowing which parts of the data are completable also provides feedback on how information is distributed throughout the matrix with respect to the completion task. Such information can then be used to study the behavior of consumers, to guide additional data acquisition, and more.

In this work, we propose to incorporate an analysis of the completability of the input which runs in parallel to the matrix completion. The proposed framework, which we call `CompleteID` and depict in Figure 1, transforms the question of completability, i.e *does there exist a unique rank-k matrix that matches the observations?*, into a graph mining task on a bipartite graph built from the observation pattern which admits an efficient solution. Building upon previous results on matrix completability [15, 16, 24], we relate completability to *edge-connectivity* of the associated graph: A submatrix is considered as completable when the corresponding

Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84. Copyright 2018 by the author(s).

[†] These authors contributed equally to this work.

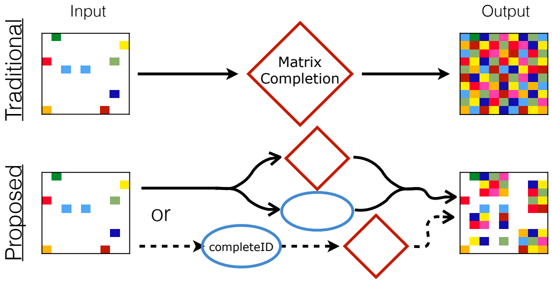


Figure 1: Traditional workflow (top) and two possible proposed workflows where completability is applied (1) as a post-facto analysis of which estimates are reliable; (2) a priori to guide completion.

subgraph is k-*edge-connected* for target rank k. This correspondence captures a slightly relaxed theoretical guarantee in favor of an efficient algorithm that makes application on web-scale data analysis possible. As a key module in CompleteID, we propose an efficient algorithm, called MaxKCD, to enumerate *maximal* k-*edge-connected components*; the corresponding completable submatrices that can either be completed later by off-the-shelf algorithms or serve as an indication of confidence on an existing prediction. Further, the algorithm we propose does not require an additional tuning parameter and is largely agnostic to completion algorithm used.

To the best of our knowledge, the CompleteID algorithm is the first scalable algorithm that discovers completable submatrices for real-world applications. We advocate for a new workflow that incorporates completability analysis alongside matrix prediction to any real-world application that is based on low-rank matrix completion, such as recommender systems. It has an array of benefits. First, the cold-start problem no longer requires manual handling, as the newly arrived user with insufficient data is identified as non-completable. Second, in the case that the overall completion accuracy is low, the work proposed here is able to identify entries that were overshadowed by the large error and can be accurately predicted nonetheless. Finally, even if the error is reasonable, there may be a disparity in the accuracy of the entries, giving a false sense that accurate recommendations are being made. This situation is dangerous as it has the propensity to lead to inaccurate recommendations that deter customers.

The contributions of the paper include:

- We study the novel problem of *completable submatrix discovery*: identify submatrices that are accurately completable. To solve the problem in a practical and theoretically supported manner, we identify a criterion for matrix completability by using *edge-connectivity* on the associated graph.
- We develop the first scalable solution, CompleteID, for the *completable submatrix discovery*. The CompleteID algorithm can be incorporated into real-world applications, without noticeable overhead, and empirical evaluation on both synthetic and real-world datasets demonstrate its success.

The paper is organized in the following manner: we briefly review the related works and notations in Section 2. In Section 3, we theoretically and empirically motivate edge-connectivity as the completability criterion. Section 4 gives the details of MaxKCD that efficiently finds k-connected components. Finally, we demonstrate the practical utility of CompleteID in Section 5. Due to the page limit, we defer the proofs and some additional experiments to the appendix.

2 Preliminaries and related work

Notation: Throughout the paper we will use $\mathbf{M}$ to refer to an $n \times m$ fully-known matrix of rank k. When the matrix is only partially known, we use $\mathbf{M}_\Omega$ to refer to the matrix that is equal to $\mathbf{M}$ on the entries specified by $\Omega \subseteq \{(i,j)|1 \leq i \leq n, 1 \leq j \leq m\}$ and zero elsewhere. When a completion algorithm is applied to a partially observed matrix $\mathbf{M}_\Omega$ with *target rank* k, the output estimate is denoted as $\hat{\mathbf{M}}$.

For a partially observed matrix $\mathbf{M}_\Omega$, the corresponding bipartite graph $G_\Omega = (V_1, V_2, E)$ is constructed as follows: create a vertex $i \in V_1$ for every row of $\mathbf{M}_\Omega$ and a vertex $j \in V_2$ for every column. Add an edge (i,j) to G_Ω if and only if $(i,j) \in \Omega$ (i.e., the entry is observed).

Related work: There are a large body of elegant theoretical and practical results for low-rank matrix completion [3, 4, 9, 10, 12, 13, 14, 31]. In order to provide guarantees on the quality of the completion, two types of assumption on the input have been established. The first set states that each low-rank component must be spread evenly across the matrix, for example, the notion of incoherence [4] is commonly adopted. The second set of assumptions concerns the locations of observed entries, requiring the pattern to sufficiently cover the matrix both in quantity and uniformity.

Most of the existing work on low-rank matrix completion focuses on 1) reducing the required number of observations under particular assumptions, or 2) improving the computational efficiency of the completion algorithm. There has been little work devoted to the case when the input has an insufficient number of observations, in which the low-rank solution might not be completable even with a known rank k. Existing work has incorporated side information [20, 22, 32], or taken an active approach by adding new entries [6, 23]. Unfortunately, active approaches are not feasible in many applications that are restricted in monetary and physical capabilities to acquire new information, nor in those that rely on human good will to provide it. Further, while the algorithm in [23] can identify completable entries, the results are dependent on the initialization and active capability.

A few works study the matrix completion problem in the similar setting of the pattern of observation as a deterministic input. In [15], Király et al. produce accuracy estimates for entries when the matrix has rank-1. The approach samples submatrices that contain a particular missing entry and uses the distribution of completions to produce an estimate of the confidence. Király et al. [16] also provided a necessary and sufficient condition for a partially-observed matrix to have a finite number of completions at rank k. This condition was used as a basis to an algorithm for identifying the missing entries in the partially observed matrix which can take only a finite number of possible values (called *finitely completable*). In another line of work work, Bhojanapalli and Jain [2] proved that the spectral gap on the associated bipartite graph can be used as a sufficient condition for the matrix completability with nuclear norm surrogate. Unfortunately, all works are not suitable for the purpose of identifying completable submatrices, as they are computationally demanding for large real-world matrices, let alone for our submatrix-search version. Further, many of these works are too coarse, providing only yes/no feedback on the matrix as a whole.

Several works that either explicitly or implicitly estimate the per-entry error margin on estimates produced by a particular matrix completion algorithm [6, 11, 27]. While differing in the precise details, each method constructs a particular model for the completion process. The variance of the estimate of each entry in this model is used to represent the confidence, and entries with very high confidence are likely to be completed accurately. In contrast to these works, the framework we propose incorporates completability in a manner that is independent of the completion algorithm used. In that sense, the algorithm we propose is more powerful in that it provides insights on the matrix completion task as whole, rather than the particular algorithm being applied.

3 Identifying completability

In this section, we present the proposed algorithm for identifying completable submatrices, which we call `CompleteID`. The focus of this section is on finding an appropriate criterion for matrix completability. For an algorithm to have both theoretical and practical importance, there are two factors to consider: (i) the criterion for matrix completability must be closely connected to the theoretical feasibility of recovery (identifiability), and (ii) the criterion must admit an efficient algorithm for finding completable submatrices. Existing results typically focus on developing a criterion that is strong for the first factor, while greatly sacrificing the second.

To achieve a good trade-off between theoretical and practical factors, we propose using the concept of *edge-connectivity* on the bipartite graph G_Ω associated with the input $\mathbf{M}_\Omega$, as the criterion for matrix completability. Hence, we can transform the problem of identifying completable submatrices in $\mathbf{M}_\Omega$ into the problem of *identifying k-connected components* in G_Ω.

The proposed approach, called `CompleteID`, is composed of three steps: (1) transform the input matrix $\mathbf{M}_\Omega$ into a graph G_Ω, (2) decompose G_Ω into maximal k-connected components using the proposed `MaxKCD` algorithm, (3) map the vertices in each component to rows and columns indexing completable submatrices. In the remainder of this section, we motivate the k-connectivity criterion both theoretically and empirically. Note that we assume the underlying low-rank matrix is drawn from some continuous distribution, which avoids the extreme coherent matrices such as the permutation matrices. Similar assumption is also adopted by existing works [15, 16].

3.1 Connectivity for completability

High edge-connectivity in a graph ensures that there are many independent paths between the vertices. For example, if a graph is 3-edge connected, it means that there exist three vertex-independent paths between every pair of vertices. A graph with high edge connectivity can be considered robust in the sense that edges can be removed without disconnecting the graph. In the context of matrix completion, a set of independent paths in the bipartite graph G_Ω can be thought of as a set of constraints on the relationship between the rows or columns of $\mathbf{M}_\Omega$. Intuitively, more independent paths impose more constraints and render it easier to recover the missing entries.

For example, consider the case when the target rank is $k = 1$, i.e., the matrix $\mathbf{M}$ is assumed to be rank-1. In this case, k-edge-connectivity reduces simply to graph connectivity; if G_Ω is connected then each entry of $\mathbf{M}_\Omega$ will be in the same row or column as at least one other entry. This overlap can be viewed as fixing the scaling parameter in the rank-1 factors, making it possible to recover the missing values in $\mathbf{M}_\Omega$. The intuition is in line with theoretical results by Király et al. [16] showing that if G_Ω is connected, then there is a single

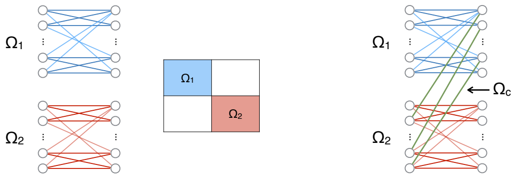
	`CompleteID`		`Density`		`Quasi`		`Triangle`		`AC`	
	Pr	Rec	Pr	Rec	Pr	Rec	Pr	Rec	Pr	Rec
CAG	0.93	1.00	0.27	1.00	0.00	1.00	1.00	0.42	0.74	0.52
Celegans	0.91	1.00	0.32	1.00	0.00	1.00	0.99	0.53	1.00	0.47
CAG364	0.98	1.00	0.34	1.00	0.00	1.00	0.99	0.66	1.00	0.73

Table 1: Evaluation of k-connectivity criterion with respect to `Exact` and baselines. *Pr* as Precision, *Rec* as Recall, and 0.00 as $< 1e-3$.

rank-1 matrix $\hat{\mathbf{M}}$ that is consistent with $\mathbf{M}_\Omega$ on Ω. For general rank $k > 1$, the following proposition holds:

Proposition 1. *If the associated bipartite graph of a partially-observed matrix is not k-edge-connected, there are an infinite number of rank-k matrices that consistent with the observation.*

Proof. The key idea is to consider the number of entries that need to be observed in order to fix ambiguity in the recover. For example, consider an $n \times m$ matrix $\mathbf{M}$ of rank k which can be written as a product of two factors: $\mathbf{A}$ of size $n \times k$ and $\mathbf{B}$ of size $m \times k$ such that $\mathbf{M} = \mathbf{A}\mathbf{B}^\top$. Now consider a partition of $\mathbf{M}_\Omega$ into $\mathbf{M}_{\Omega_1}$ of size $n_1 \times m_1$ and $\mathbf{M}_{\Omega_2}$ of size $n_2 \times m_2$ where $n_1 + n_2 = n$, $m_1 + m_2 = m$, and $\Omega = \{\Omega_1, \Omega_2\}$. Figure 2a gives a visual of $\mathbf{M}_\Omega$ and G_Ω. Observe that G_Ω is composed of two components that are disconnected, since there is no overlap between Ω_1 and Ω_2.



(a) Ω_1 , Ω_2 in $\mathbf{M}_\Omega$ and G_Ω (b) with k edges as Ω_C

With this Ω, the factor $\mathbf{A}$ can be divided into two parts: $\mathbf{A}_1 \in \mathbb{R}^{n_1 \times k}$ and $\mathbf{A}_2 \in \mathbb{R}^{n_2 \times k}$ with $n = n_1 + n_2$. Similarly for $\mathbf{B}$ with $m = m_1 + m_2$. Denote by $\mathbf{M}(\mathbf{W})$ the family of matrices parameterized by a $k \times k$ matrix $\mathbf{W}$, laying in the span of $\mathbf{A}_2$ and $\mathbf{B}_2$, that is applied to $\mathbf{A}_2$ and $\mathbf{B}_2$ as shown below:

$$\mathbf{M}(\mathbf{W}) = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2\mathbf{W} \end{bmatrix} \begin{bmatrix} \mathbf{B}_1^\top & \mathbf{W}^\dagger\mathbf{B}_2^\top \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1\mathbf{B}_1^\top & \mathbf{A}_1\mathbf{W}^\dagger\mathbf{B}_2^\top \\ \mathbf{A}_2\mathbf{W}\mathbf{B}_1^\top & \mathbf{A}_2\mathbf{B}_2^\top \end{bmatrix}$$

Observe that $\mathbf{M}_\Omega$ is identical for every matrix in the family, regardless of $\mathbf{W}$. In other words, given $\mathbf{M}_\Omega$ there are infinitely many completions $\hat{\mathbf{M}}$ that match on Ω. To have any hope of recovering the true matrix $\mathbf{M}$, there must be some additional known entries outside of Ω_1 and Ω_2 to fix the degrees of freedom in $\mathbf{W}$, call these entries Ω_c. It can be shown that as long as $|\Omega_c| < k$, which is the minimum degrees of freedom for $\mathbf{W}$, then $\mathbf{M}_\Omega$ has infinitely many completions. Entries in Ω_c corresponds to edges in the cut on G_Ω as illustrated in Figure 2b; hence, Ω_c induces k-connectivity on G_Ω. □

The intuition is to consider a partition of the graph into two components corresponding to submatrices all of whose degrees of freedom are fixed. It can be shown, that in order to fix all degrees of freedom in the matrix at least k additional entries (edges) are needed, translating to k-edge-connectivity.

When entries are observed uniformly at random, the edge-connectivity requirement also nicely aligns with the known results [5], which can be proved by applying Chernoff Bounds on the graph cut over all of the possible graph partitions.

Proposition 2. *For a partially observed $n \times n$ matrix, if the entries Ω is observed uniformly at random, then there exists a constant C, such that when $|\Omega| \geq nk + C(nk)^{0.5}\log(n/\delta)$, the corresponding bipartite graph is k-edge-connected with probability at least $1 - \delta$.*

Aside from k-connectivity, there are other graph-theoretic concepts that may be considered as a criterion for matrix completability, such as density. Depending on the particular end-goal, various density measures and computational models have been defined, with the state of the art including degree-density [8], edge-surplus [30], k-clique [29], and various notions related to k-clique that emphasize cohesiveness, such as k-plex. Not only do these definitions have no known relation to completability, but most of them are also NP-hard to enumerate [1]. In contrast, the k-connected components can be mined in polynomial time even in the worst case [7, 26].

3.2 Empirical evidence for connectivity as a completability criterion

As discussed in Section 2, the algorithm proposed in [16] which we call `Exact`, is computationally expensive; however, its theoretical guarantee provides an opportunity to validate our choice of k-connectivity as a criterion for completability. To do this, we compare the entries deemed completable by `Exact` to those that participate in k-connected components. Limited by the poor scalability of `Exact`, we select several small partially-observed matrices from the UF sparse matrix collection, and run `Exact` and `CompleteID` on the Ω corresponding to each dataset with a range of values for k.[1] Table 1 shows high precision and recall of `CompleteID` with respect to `Exact`, validating that k-connectivity is a good and efficient proxy for `Exact`.

We also compare with several baselines, including the `ActiveCompletion` algorithm (`AC`) proposed in [23] with the query budget set to zero, and four state-of-the-art algorithms for dense subgraph discovery: `Density` [8], `Quasi` [30], and `Triangle` [29].[2] We note that while the density-based algorithms have no theoretical ties to completability, in principle, higher density increases the likelihood of completability.

We find that `ActiveCompletion` achieves high precision but low recall; in fact, as k grows, the algorithm does not find any completable entries at all. These results held even when we allowed a small query budget, and are a result of the large reliance on the starting point and an active approach. Similarly, the `Triangle` algorithm failed to identify a vast majority of the completable entries; this behavior is exacerbated as the size and rank of the matrix grow. In contrast both `Density` and `Quasi` achieve a high recall but a low precision.In other words, these algorithms falsely label many entries as completable. The `Graph-Triangle` algorithm is not shown since it failed to identify any components.

These results demonstrate that identifying completable submatrices is a non-trivial task, and that k-CC decomposition offer a promising approach for this problem by achieving both high precision and recall. Not only does the k-CCs capture the majority of completable entries, but it also does not give false confidence with false positives.

4 Finding k-connected components

Having established that edge connectivity is a good criterion for completability, we now present an algorithm for finding k-connected components (k-CCs) in a graph. The problem can be stated as:

Problem 3. *Given a graph $G = (V, E)$ and a positive integer k, find a partition of the vertices $V = \bigcup_{i=1}^{m} V_i$ with minimum m such that if $|V_i| > 1$ then the subgraph induced by V_i is a maximal k-connected subgraph.*

The algorithm we propose, called `MaxKCD`, is an improvement upon recent literature [7, 26] which modifies the Stoer-Wagner algorithm [25] for finding a global min cut. `MaxKCD` works by iteratively finding a cut of size less than k in G until no such a cut exists and only k-CCs or singleton vertices remain; the pseudocode is shown in Algorithm 1.

Algorithm 1: Maximal k-CC Decomposition (`MaxKCD`)

```
Input  : Graph $G = (V, E)$ and the target cut size $k$
Output : The vertex partition $\Phi$
Initialize $\Phi = \emptyset$ and $\Gamma_0$ to be the whole graph $G$
while $\Gamma_i \neq \emptyset$ do
    Initialize $\Gamma_{i+1} = \emptyset$
    forall $G' = (V', E') \in \Gamma_i$ do
        Find a cut $C = (V_1', V_2') = \texttt{kCut}(G', k)$
        if $C = \emptyset$ or $|V'| = 1$ then
            Add $V'$ to $\Phi$
        else
            Add $G'_{V_1}$ and $G'_{V_2}$ to $\Gamma_{i+1}$
        end
    end
end
return $\Phi$
```

Our main contribution in `MaxKCD` is the `kCut` subroutine[3] which utilizes three core operations: `EarlyStop`, `ForceContraction`, and `Batch-EarlyMerge`. The `Batch-EarlyMerge` operation is inspired by the observation that the *maximal adjacency search* (`MAS`) subroutine in the original Stoer-Wagner algorithm can be accelerated by a batch-like approach; the acceleration allows `kCut` to be more efficient while still retaining accuracy for Problem 3. We also incorporate a vertex-merging technique, `Force-Contraction`, which is a more aggressive version than those used in previous work. Finally, we use the `EarlyStop` routine proposed in [26] to limit the number of recursive iterations. As an interesting side benefit, the proposed `kCut` algorithm can be used to speed up the original Stoer-Wagner algorithm by actively updating k to be the current smallest cut, starting from infinity. Due to space constraints, we include the implementation details in Appendix A, proofs of correctness in Appendix B. Note that some of the techniques we mentioned have been discovered in the graph mining literature[26, 7, 1], where we incorporate them together to achieve best performance.

Computational Complexity Since the `kCut` algorithm can be used with binary search to solve the global min-cut problem, the worst-case computational complexity of `kCut` is the same as the Stoer-Wagner algorithm: $O\left(|V||E| + |V|^2 \log |V|\right)$. However, unlike

[1] Dataset available at http://www.cise.ufl.edu/research/sparse/matrices/

[2] Code available at https://github.com/giannisnik/k-clique-graphs-dense-subgraphs

[3] The source code of our C++ implementation can be found at `https://github.com/USC-Melady/Graph-Cut`.

	n	m	$\|\Omega\|$
Netflix	480K	17K	100M
Amazon	6.6M	2.4M	29M
Foursquare	45K	17K	1.2M
DBLP	122K	36K	1.8M
Music	5K	210K	16M
LibimSeTi	135K	168K	17M
Gisette	6K	5K	3.9M

Table 2: Dataset details.

the Stoer-Wagner algorithm, the empirical runtime of the kCut algorithm is much less than its worst-case. If the input graph is sparse and loosely connected, like those studied in real-world applications of matrix completion, EarlyStop is likely to terminate the algorithm in the first few calls. On the other hand, if the input graph is tightly connected, Batch-EarlyMerge ensures that each recursive iteration will be conducted efficiently, and together with ForceContraction, the graph size shrinks quickly. In our experiments, less than 10 recursive calls are observed even in graphs with million of vertices and billions of edges, with the empirical runtime scaling with $|E|\log|V|$.Note that the overall running time of the MaxKCD algorithm is a small fraction of the running time of the common matrix completion algorithms.

5 Experiments

In this section, we evaluate the performance of CompleteID on both real and synthetic data, study the interplay of connectivity and completability, and demonstrate the variety of insights it offers.

Setup: In all our experiments, we use a pattern of observed entries Ω from sparse real-world data (shown in Table 2) to capture sampling patterns that occur in practice as opposed to artificial settings. We experiment with the real matrix values and, for a more controlled setting, with synthetic ones of fixed rank. Synthetic matrices are generated following a standard Gaussian, with additive Gaussian noise having standard deviation 0.1.

Given a partially observed matrix $\mathbf{M}_\Omega$, we estimate the missing entries by fitting a rank k_{MC} matrix to the observed entries with alternating minimization (chosen for it computational efficiency) [17]. For an output estimate $\hat{\mathbf{M}}$ we compare the RMSE on the held-out test set, separately over the completable and non-completable entries as selected by CompleteID.

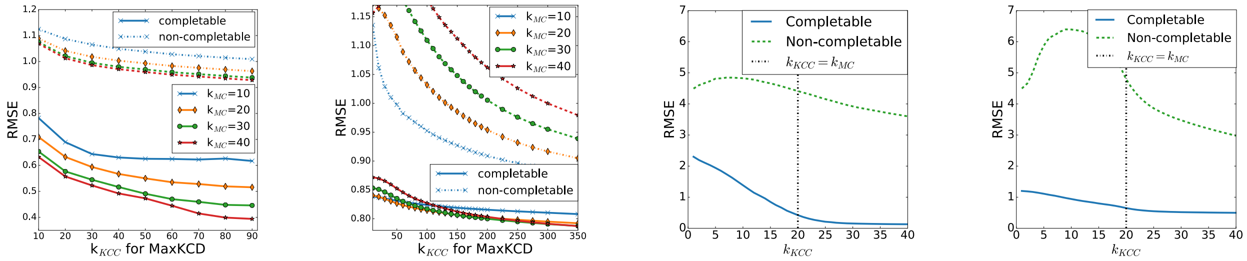
5.1 Traditional completion task

For two large-scale collaborative filtering datasets, Netflix and Amazon [18] , we select 99% of the observed entries of Amazon and Netflix as the training set and evaluate the error over the remaining held-out test set. We explore the behavior of CompleteID under a variety of model settings parameterized by k_{KCC}, the edge-connectivity for MaxKCD ,and k_{MC} the rank for the completion algorithm. The target rank k_{MC} ranges in $\{10, 20, 30, 40\}$, and k_{KCC} in $[10, 90]$ for Amazon and $[10, 350]$ for Netflix.

Figures 3a and 3b show the RMSE on the completable and non-completable entries. For any given k_{KCC} and k_{MC}, the RMSE on the completable entries is significantly lower than on non-completable ones. Moreover, for a given k_{MC}, as k_{KCC} increases, the RMSE on the completable entries continues to drop. This decay indicates that the CompleteID algorithm serves as a smooth measure for completability, meaning that we need not be bounded by $k_{MC} = k_{KCC}$ in real world applications. With $k_{KCC} = 350$, there are still over 55% of test entries that are completable on Netflix, while Amazon is much sparser with around 3% entries that are completable when $k_{KCC} = 90$. In Figure 3b) we see an interesting flipping behavior on the completable entries with different k_{MC}. The fact that smaller k_{MC} give better accuracy when k_{KCC} is small can be explained by overfitting; however, as k_{KCC} increases, we see a faster decay in RMSE for larger k_{MC}, resulting in a better performance. This behavior demonstrates that the CompleteID algorithm can be used to deploy more complex models to improve local performance without the consequence of overfitting.

5.2 Controlled matrix completion

In this section, we mimic the collaborative filtering task on DBLP, Music, LibimSeTi, and Gisette, in a controlled setting where we generate a synthetic matrix of rank k as described in the beginning of this section. The synthetic matrix allows us to use the full set of observed entries for training, and the rest for testing. Table 3 shows the RMSE for a synthetic matrix of rank $k = 20$ and $k_{KCC} = k_{MC} = 20$. In addition, Figure 3c and Figure 3d show the RMSE over $k_{KCC} \in [1, 40]$ on the DBLP and LibimSeTi dataset; the coordinate at $x = k_{KCC} = 20$ is the same value that is in Table 3. The completability ranges from 96% to 55% for DBLP and 99% to 87% for LibimSeTi. Consistent with the previous experiment, the error over completable entries is lower than over non-completable ones across all datasets. Further, we see that the error decreases as k_{KCC} grows, but hits a meaningful point around $k_{KCC} = 20$ which is the matrix rank.



(a) Amazon

(b) Netflix

(c) DBLP $k_{MC} = 20$

(d) LibimSeTi $k_{MC} = 20$

Figure 3: The RMSE of completable and non-completable entries on the test set as a function of k_{KCC}, with varying k_{MC} for Amazon and Netflix, and $k_{MC} = 20$ for DBLP and LibimSeTi.

	DBLP	Music	LibimSeTi	Gisette
Compl.	0.42	0.29	0.38	0.11
Non-Compl.	4.41	2.96	5.27	5.81

Table 3: Completability over small datasets with rank-20 synthetic matrix and $k_{KCC} = k_{MC} = 20$.

5.3 Comparison with baselines

As discussed in Section 3, most related completability algorithms are limited by their computational inefficiency, hence we include a comparison with the best performing density-based baseline, `SmartDensity`. For each unknown entry (i, j) in the test set, we calculate the number of known entries in the i-th row and j-th column as a measure of completability; the higher the better. Then, we mark the top N entries as completable, where N is the number of completable entries discovered by `CompleteID` (the knowledge of N merits the name *Smart*).

We tested `CompleteID` and `SmartDensity` on the Netflix data. Instead of the randomly splitting the entries into training and testing, we used the timestamps available on matrix entries and select the first 25% (or 50%) of the observed ratings as training, and the next 1% of the observed entries as testing. The `RMSE` on the completable entries for both `CompleteID` and `SmartDensity` is shown in Figures 4a and 4b. We see that the `RMSE` over the completable entries discovered by `CompleteID` is significantly lower than over those discovered by `SmartDensity` which is slightly better than the total `RMSE`. Moreover, the performance of `SmartDensity` degrades, implying that identifying completable submatrices is a non-trivial problem that cannot be addressed solely with a density-based approach. Finally, we also compared with `ActiveCompletion` [23] of Section 3. We found that `ActiveCompletion` identifies only a small subset of the completable entries identified by `CompleteID`. For example, on Netflix the algorithm did not find any completable regions for $k > 90$. This is due to its dependence on proper initialization and the ability to be active by adding entries.

5.4 Completability over time

Finally, we study the accuracy over the completable and non-completable entries with time for two datasets, Netflix and Foursquare. For Netflix we use the real data values, and for Foursquare we generate a synthetic rank-10 matrix. Both datasets contain timestamps which we use to divide the data into training and testing. At each timestamp t, we use the entries that appeared before t as training and the entries appears within $[t, t + \Delta t]$ as testing – mimicking the pattern of observed entries encountered in real-world applications. The time is measured by *days*, and the length of the test set Δt is 30 days for Netflix and 3 days for Foursquare. For both matrices we show results for fixed $k_{MC} = k_{KCC} = 10$.

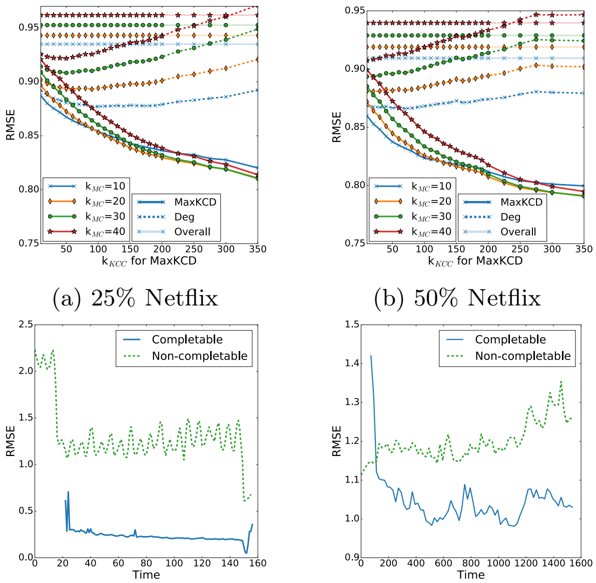
Figures 4d and 4c show the `RMSE` as a function of time with the percentage of completable entries varying from 0% to ≈ 80%. We observe that over time, the test entries that fall in submatrices deemed completable by `MaxKCD` have significantly less error than those outside the submatrices. These results demonstrate not only that `CompleteID` successfully identifies completable submatrices, but that the knowledge of these submatrices carries importance in practice.

5.5 Further experiments

In this section, we highlight several additional experiments and with more results featured in Appendix C.

Alternative completion algorithms: We selected five algorithms with the most competitive efficiency and accuracy: `LMaFit`, `OptSpace`, `LRGeom`, `Riemann`, and `VBMC`. [4] The interested reader can refer to [34]

[4] Source code was obtained from https://sites.google.com/site/lowrankmodeling/



(a) 25% Netflix

(b) 50% Netflix

(c) Foursquare

(d) Netflix

Figure 4: Figures (a) and (b) show a comparison between `CompleteID` and `SmartDensity` with 25% and 50% of Netflix. Figures (c) and (d) show the `RMSE` of completable and non-completable entries as a function of time for Netflix and Foursquare with $k = 10$.

for a comparison. We found that using all algorithms, the error over the entries selected by `CompleteID` as completable is significantly smaller than over the non-completable ones, which demonstrates the robustness of `CompleteID`.

Efficiency of `MaxKCD`: The overall running time of the `MaxKCD` algorithm is a small fraction of the running time of common matrix completion algorithms. Therefore, the utility of our framework is not limited by the computational efficiency of `MaxKCD`. However, the empirical runtime is of interest since `MaxKCD` can be used independently for graph maximal k-connected components decomposition and global minimum cut algorithms. With our aggressive graph contraction schemes, including both `Batch-EarlyMerge` and `ForceContraction`, the overall running time is all less than 6 minutes on both Amazon and Netflix datasets with $k \geq 10$.

6 Conclusion

In this work, we argue that an analysis of the *completability* of a partially-observed matrix should be carried out alongside the actual completion. While real-world matrices are not typically completable as a whole, we make the observation that there may still exist portions of the data that can still be completed accurately. Information of such completable regions enables a more principled completion process, providing feedback on the structure of the observed data, the accuracy of the estimate through the data, and whether issues such as overfitting and cold-start are of concerns. We propose the problem of identifying completable submatrices and the `CompleteID` framework, which features the first scalable algorithm for the problem. A key component of our work is the exposure of edge-connectivity as a practical and theoretically supported surrogate for completability.

While we have shown that incorporating completability analysis into the matrix completion workflow is an impactful and promising direction, there are various avenues that are open for investigation. One major component is the development of a deeper theoretical understanding of matrix completability. While several works have studied the problem [19, 23, 16, 24], many open questions remain.

Acknowledgments

This work is supported in part by the Adobe University Marketing Research Award, U. S. Army Research Office under grant Number W911NF-15-1-0491, NSF Research Grant IIS-1254206 and IIS-1619458. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agency, or the U.S. Government.

References

[1] T. Akiba, Y. Iwata, and Y. Yoshida. Linear-time enumeration of maximal k-edge-connected subgraphs in large networks by random contraction. In *CIKM*, 2013.

[2] S. Bhojanapalli and P. Jain. Universal matrix completion. In *ICML*, 2014.

[3] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J Optimiz*, 2010.

[4] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *FoCM*, 2009.

[5] E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[6] S. Chakraborty, J. Zhou, V. Balasubramanian, S. Panchanathan, I. Davidson, and J. Ye. Active matrix completion. In *ICDM*, 2013.

[7] L. Chang, J. X. Yu, L. Qin, X. Lin, C. Liu, and W. Liang. Efficiently computing k-edge connected

components via graph decomposition. In *SIGMOD*, 2013.

[8] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In *APPROX*, 2000.

[9] Y. Chen, S. Bhojanapalli, S. Sanghavi, and R. Ward. Coherent matrix completion. In *ICML*, 2014.

[10] M. A. Davenport and J. Romberg. An overview of low-rank matrix recovery from incomplete observations. *J-STSP*, 2016.

[11] F. Fazayeli, A. Banerjee, J. Kattge, F. Schrodt, and P. B. Reich. Uncertainty quantified matrix completion using bayesian hierarchical matrix factorization. In *ICMLA*, 2014.

[12] R. Ge, J. D. Lee, and T. Ma. Matrix completion has no spurious local minimum. In *NIPS*, 2016.

[13] T. Hastie, R. Mazumder, J. D. Lee, and R. Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *JMLR*, 2015.

[14] R. H. Keshavan, S. Oh, and A. Montanari. Matrix completion from a few entries. In *ISIT*, 2009.

[15] F. Kiraly and L. Theran. Error-minimizing estimates and universal entry-wise error bounds for low-rank matrix completion. In *NIPS*, 2013.

[16] F. J. Király, L. Theran, and R. Tomioka. The algebraic combinatorial approach for low-rank matrix completion. *Journal of Machine Learning Research*, 16:1391–1436, 2015.

[17] Y. Koren, R. Bell, C. Volinsky, et al. Matrix factorization techniques for recommender systems. *Computer*, 2009.

[18] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*, 2013.

[19] R. Meka, P. Jain, and I. S. Dhillon. Matrix completion from power-law distributed samples. In *NIPS*, 2009.

[20] A. K. Menon, K.-P. Chitrapura, S. Garg, D. Agarwal, and N. Kota. Response prediction using collaborative filtering with hierarchies and side-information. In *KDD*, 2011.

[21] H. Nagamochi and T. Watanabe. Computing k-edge-connected components of a multigraph. *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, 1993.

[22] I. Porteous, A. U. Asuncion, and M. Welling. Bayesian matrix factorization with side information and dirichlet process mixtures. In *AAAI*, 2010.

[23] N. Ruchansky, M. Crovella, and E. Terzi. Matrix completion with queries. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1025–1034. ACM, 2015.

[24] A. Singer and M. Cucuringu. Uniqueness of low-rank matrix completion by rigidity theory. *SIAM Journal on Matrix Analysis and Applications*, 31(4):1621–1641, 2010.

[25] M. Stoer and F. Wagner. A simple min-cut algorithm. *JACM*, 1997.

[26] H. Sun, J. Huang, Y. Bai, Z. Zhao, X. Jia, F. He, and Y. Li. Efficient k-edge connected component detection through an early merging and splitting strategy. *Knowl-Based Syst.*, 2016.

[27] D. J. Sutherland, B. Póczos, and J. Schneider. Active learning and search on low-rank matrices. In *KDD*, 2013.

[28] Y. H. Tsin. Yet another optimal algorithm for 3-edge-connectivity. *Journal of Discrete Algorithms*, 7(1):130–146, 2009.

[29] C. Tsourakakis. The k-clique densest subgraph problem. In *WWW*, 2015.

[30] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli. Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. In *KDD*, 2013.

[31] Z. Wen, W. Yin, and Y. Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 2012.

[32] M. Xu, R. Jin, and Z.-H. Zhou. Speedup matrix completion with side information: Application to multi-label learning. In *NIPS*, 2013.

[33] R. Zhou, C. Liu, J. X. Yu, W. Liang, B. Chen, and J. Li. Finding maximal k-edge-connected subgraphs from a large graph. In *EDBT*, 2012.

[34] X. Zhou, C. Yang, H. Zhao, and W. Yu. Low-rank modeling and its applications in image analysis. *CSUR*, 2014.