Graph Sampling for Matrix Completion Using Recurrent First Eigenvector Computation

Fen Wang  
School of Telecommunication Engineering  
Xidian University, Xi’an, China  
fenwang@stu.xidian.edu.cn

Cheng Yang  
Department of EECS  
York University, Toronto, Canada  
 cyang@eecs.yorku.ca

Gene Cheung  
Department of EECS  
York University, Toronto, Canada  
 genec@yorku.ca

Yongchao Wang  
School of Telecommunication Engineering  
Xidian University, Xi’an, China  
ychwang@mail.xidian.edu.cn

Abstract

While matrix completion algorithms fill missing entries in a matrix given a subset of samples, how to best pre-select matrix entries to collect informative observations given a sampling budget is largely unaddressed. In this paper, we propose a fast graph sampling algorithm to select entries for matrix completion. Specifically, we first regularize the matrix reconstruction objective using a dual graph signal smoothness prior, resulting in a system of linear equations for solution. We then seek to maximize the smallest eigenvalue \( \lambda_{\text{min}} \) of the coefficient matrix by choosing appropriate samples, thus maximizing the stability of the linear system. To efficiently optimize this combinatorial problem, we derive a greedy sampling strategy, leveraging on Gershgorin circle theorem, that iteratively selects one sample at a time corresponding to the largest magnitude entry in the first eigenvector of a shifted graph Laplacian matrix. Our algorithm benefits computationally from warm start as the first eigenvectors of progressively shifted Laplacian matrices are computed recurrently for more samples. Towards computation-scalable sampling on large matrices, we further rewrite the coefficient matrix as a sum of two separate components, each of which exhibits attractive block-diagonal structure that we exploit for alternating block-wise sampling. Extensive experiments on both synthetic and real-world datasets show that our graph sampling algorithms substantially outperform existing sampling schemes for matrix completion, when combined with a range of modern matrix completion algorithms.

1 Introduction

Big data means not only that the volume of acquired data is large, but the dimensionality of the dataset can also be considerable. Matrix completion [20] studies this "curse of dimensionality" problem for the setting where two large dimensional item sets (e.g., viewers and movies in the famed Netflix challenge) are correlated within and across sets. Specifically, given a small subset of pairwise observations (e.g., limited viewers’ ratings on movies), a matrix completion algorithm reconstructs missing entries in the target matrix signal. Many matrix completion algorithms have been devised using different priors to regularize the under-determined inverse problem, such as low rank of the target matrix [10] and graph signal smoothness priors [17]. See [9] for an introductory exposition.

Preprint. Under review.
While matrix completion has been investigated intensively, how to pre-select matrix entries to collect which was further investigated in [14]. Adaptive sensing was proposed in [19] to select a subset of graphs. Specifically, columns of graphs [16, 17]. Under this assumption, [26, 27] identified a sampling set for matrix completion by selecting nodes from two factor graphs independently. However, the imposed structure severely limits the possible sampling patterns and is too restrictive to achieve a general sampling budget. More general sampling methods for single graphs are not applicable for matrix completion due to their high complexity [6, 35].

In contrast, in this paper we propose a fast unstructured graph sampling method for matrix completion. We first regularize the sampling objective with a dual graph smoothness prior, which was shown effective in completing missing matrix entries previously [17, 22]. This formulation leads to a system of linear equations for solution, which can be computed efficiently using conjugate gradient (CG) [21]. To maximize the stability of the linear system, we select samples to maximize the smallest eigenvalue $\lambda_{min}$ of the coefficient matrix, which we show to also mean minimizing the upper bound of the reconstructed matrix signal’s squared error.

Computing $\lambda_{min}$ for each candidate sample set and choosing the largest one would be expensive. Instead, leveraging on an insightful corollary of the Gershgorin circle theorem [23], we greedily select one sample at a time, where each selected sample corresponds to the largest magnitude entry in the first eigenvector of a shifted coefficient matrix, which also minimizes an augmented objective. Our algorithm benefits from warm start as the first eigenvectors of progressively shifted Laplacian matrices are computed recurrently during sampling using the locally optimal block preconditioned conjugate gradient (LOBPCG) method [18]. Towards scalable computation for large real-world datasets, we further divide the coefficient matrix into two matrices, each exhibiting attractive block-diagonal structure after permutation. We then propose an iterative sampling strategy that efficiently collects a variable number of samples block-wise on the two smaller factor graphs alternately. Extensive experiments on synthetic and real-world datasets show that our proposed graph sampling methods achieve smaller RMSE than competing sampling schemes for matrix completion [26, 30, 33], when combined with different state-of-the-art matrix completion methods [7, 24, 31].

## 2 Problem Formulation

In this section, we derive an objective function for matrix sampling using a dual graph signal smoothness prior [17, 22]. We first define the graph-based matrix completion problem in Section 2.1 and then formulate the graph spectral matrix sampling problem in Section 2.2.

### 2.1 Dual Graph Smoothness based Matrix Completion

Denote the original matrix signal and additive noise by $X$ and $N$ respectively, where $X, N \in \mathbb{R}^{m \times n}$. Denote also a sampling operator by $A_{\Omega} \in \{0, 1\}^{m \times n}$, so that given a sampling set $\Omega = \{(i, j) \mid i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}\}$, $A_{\Omega}$ can be defined as

$$A_{\Omega}(i, j) = \begin{cases} 1, & \text{if } (i, j) \in \Omega \\ 0, & \text{otherwise} \end{cases}$$

Given the above notations, the sampled noise-corrupted observation is $Y = A_{\Omega} \circ (X + N) \in \mathbb{R}^{m \times n}$, where $\circ$ denotes the element-wise matrix multiplication operator. We assume that elements in noise $N$ are zero-mean, independent and identically distributed (i.i.d.) noise with the same variance.

In Kalofolias et al. [17], it is assumed that the matrix signal $X$ is smooth with respect to two graphs. Specifically, columns of $X$ are smooth with respect to an undirected weighted row graph $G_r = (V_r, E_r, W_r)$ with vertices $V_r = \{1, \ldots, m\}$ and edges $E_r \subseteq V_r \times V_r$. Weight matrix $W_r$ specifies pairwise similarities among vertices in $G_r$. The combinatorial graph Laplacian matrix of row graph $G_r$ is $L_r = D_r - W_r$, where the degree matrix $D_r$ is a diagonal matrix with entries

$$D_r(i) = \sum_{j=1}^{m} W_r(i, j)$$

Denote the original matrix signal and additive noise by $X$ and $N$ respectively, where $X, N \in \mathbb{R}^{m \times n}$. Denote also a sampling operator by $A_{\Omega} \in \{0, 1\}^{m \times n}$, so that given a sampling set $\Omega = \{(i, j) \mid i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}\}$, $A_{\Omega}$ can be defined as

$$A_{\Omega}(i, j) = \begin{cases} 1, & \text{if } (i, j) \in \Omega \\ 0, & \text{otherwise} \end{cases}$$

Given the above notations, the sampled noise-corrupted observation is $Y = A_{\Omega} \circ (X + N) \in \mathbb{R}^{m \times n}$, where $\circ$ denotes the element-wise matrix multiplication operator. We assume that elements in noise $N$ are zero-mean, independent and identically distributed (i.i.d.) noise with the same variance.

In Kalofolias et al. [17], it is assumed that the matrix signal $X$ is smooth with respect to two graphs. Specifically, columns of $X$ are smooth with respect to an undirected weighted row graph $G_r = (V_r, E_r, W_r)$ with vertices $V_r = \{1, \ldots, m\}$ and edges $E_r \subseteq V_r \times V_r$. Weight matrix $W_r$ specifies pairwise similarities among vertices in $G_r$. The combinatorial graph Laplacian matrix of row graph $G_r$ is $L_r = D_r - W_r$, where the degree matrix $D_r$ is a diagonal matrix with entries

$$D_r(i) = \sum_{j=1}^{m} W_r(i, j)$$
To simplify notations, we first denote $L_\alpha$ where $l$.

To solve the unconstrained QP problem (2), we take the derivative of $\tilde{\phi}$ where $\phi$.

The stability of the linear solution in (3) is determined by the condition number of matrix $L_c$ to promote matrix low rankness, it has been shown through extensive experiments that the dual graph signal smoothness prior does enable good matrix completion performance [17].

In movie recommendation systems, the row graph is a social relationship graph among viewers. We can formulate now the matrix completion problem with dual graph Laplacian regularization (GLR) as follows:

$$\min_X f(X) = \frac{1}{2}\|A_\Omega \odot (X - Y)\|_F^2 + \frac{\alpha}{2} \text{Tr} (X^T L_r X) + \frac{\beta}{2} \text{Tr} (XL_c X^T)$$

where $\alpha$ and $\beta$ are parameters trading off the first fidelity term with the two signal smoothness priors. Though $\phi$ does not explicitly include a nuclear norm term ($l_1$-norm of the singular values of the target matrix) to promote matrix low rankness, it has been shown through extensive experiments that the dual graph signal smoothness prior does enable good matrix completion performance [17].

To solve the unconstrained QP problem (2), we take the derivative of $f(X)$ with respect to $X$, set it to 0 and solve for $X$, resulting in a system of linear equations for unknown vec($X^*$):

$$(\tilde{A}_\Omega + \alpha I_n \otimes L_r + \beta L_c \otimes I_m) \text{vec}(X^*) = \text{vec}(Y)$$

where $\tilde{A}_\Omega = \text{diag}(\text{vec}(A_\Omega))$, and vec($\cdot$) means a vector form of an matrix by stacking its columns. See Appendix A for a detailed derivation.

Since the coefficient matrix $Q = \tilde{A}_\Omega + \alpha I_n \otimes L_r + \beta L_c \otimes I_m$ is symmetric, sparse and positive definite (PD), (4) can be solved efficiently using a plethora of mature numerical linear algebra methods such as conjugate gradient (CG) [34]. This is one notable appeal of formulating the matrix completion problem using the dual graph signal smoothness prior in (2), where computing its solution requires only solving a system of linear equations. See Appendix B for a detailed description of $Q$.

### 2.2 Sampling for Matrix Completion

Previous works in matrix completion [7, 17, 22] assume that limited samples are provided as input and focus on the reconstruction of missing matrix entries. However, sampling is often expensive and/or time-consuming [32], and pre-selection of the most beneficial samples given a sampling budget is an important issue. We next derive a sampling objective based on mean squared error (MSE).

The stability of the linear solution in (3) is determined by the condition number of matrix $Q$, which is the ratio of the largest eigenvalue $\lambda_{\max}$ of $Q$ to its smallest eigenvalue $\lambda_{\min}$. Given that $\lambda_{\max}(Q)$ is upper-bounded for a degree-constrained graph (see Appendix C for a proof), to maximize stability, we seek to maximize $\lambda_{\min}(Q)$ through sampling and propose the following sampling objective

$$\max_{\Omega} g(\Omega) = \lambda_{\min} (\tilde{A}_\Omega + \alpha I_n \otimes L_r + \beta L_c \otimes I_m)$$

It turns out that maximizing this objective (4) also means minimizing the MSE upper bound, as stated formally in the lemma below. See Appendix D for a proof.

**Lemma 1.** Given dual graph Laplacians $L_r$ and $L_c$, assuming ground truth signal $X$ is corrupted by independent additive noise $N$, MSE of the reconstructed signal $X^*$ with respect to the original signal $X$ is upper-bounded by

$$\|\text{vec}(X^*) - \text{vec}(X)\|_2 \leq \frac{\rho}{\lambda_{\min}(Q)} + \text{vec}(N)$$

where $\rho = \|\text{vec}(\alpha I_n \otimes L_r + \beta L_c \otimes I_m)\|_2$ vec$(X + N)$

### 3 Fast Sampling on Product Graph via Gershgorin Circle Theorem

To simplify notations, we first denote $L = \alpha I_n \otimes L_r + \beta L_c \otimes I_m$ and interpret it as the Laplacian of a scaled Cartesian product graph. Thus optimization (4) becomes the maximization of $\lambda_{\min}$ for

1 Cartesian product between two matrices $L_r$ and $L_c$ is defined by: $L_r \odot L_c = I_n \otimes L_r + L_c \otimes I_m$
We first review GCT and its corollary [23], which will lead to a lightweight sampling method. We justify our strategy from another perspective using the following lemma, which shows that the where

\[ K = |S|, \ k_t = S(t) \] and \( e_t \) is an indicator vector with \( e_t(t) = 1 \) and \( e_t(k) = 0, k \neq t \).

Finding an optimal \( \Omega \) to maximize \( \lambda_{min}(Q) \) is combinatorial in nature. Towards a low-complexity sampling strategy, we take a greedy approach, where we iteratively add a locally optimal sample to a selected sample set until the sample budget is exhausted. Hence, assuming we have collected \( t - 1 \) samples in \( S_{t-1} \), at the \( t \)-th iteration, we solve the following local optimization problem:

\[
k_t^* = \arg\max_{k_t \in S_{t-1}} \lambda_{min}(L_{t-1} + e_{k_t}e_{k_t}^\top)
\]

where \( k \in \{1, \ldots, K\}, S_t = S_{t-1} \cup k_t^* \) with \( S_0 = \emptyset \), and \( L_t = L_{t-1} + e_{k_t}e_{k_t}^\top \) with \( L_0 = L \).

To find an optimal solution \( k_t^* \) in (7) for each new sample, one can compute \( \lambda_{min} \) of the shifted Laplacian \( L_{t-1} + e_{k_t}e_{k_t}^\top \) corresponding to all candidate nodes \( k_t \in S_{t-1} \) and identify the largest one, which is computation-intensive. Instead, we circumvent multiple computation of eigenvalues for candidates using a strategy based on Gershgorin circle theorem (GCT).

### 3.1 Gershgorin Circle Theorem based Sampling

We first review GCT and its corollary [23], which will lead to a lightweight sampling method.

**Theorem 1.** Given an \( n \times n \) matrix \( A \) be with entries \( a_{ij} \), define the \( i \)-th Gershgorin disc \( D(a_{ii}, R_i) \), corresponding to the \( i \)-th row of \( A \), with center \( a_{ii} \) and radius \( R_i = \sum_{j \neq i} |a_{ij}| \). Each eigenvalue \( \lambda \) of \( A \) lies within at least one Gershgorin disc, i.e.,

\[
\exists i \mid a_{ii} - R_i \leq \lambda \leq a_{ii} + R_i
\]

**Corollary 1.** If the largest magnitude component of an eigenvector \( \phi \) is at index \( i \), then its corresponding eigenvalue \( \lambda \) must be within the \( i \)-th Gershgorin disc \( D(a_{ii}, R_i) \).

This corollary implies that \( \lambda_{min} \) of matrix \( L_{t-1} \) must reside in the \( i^* \)-th Gershgorin disc, where \( i^* = \arg\max_{i} |\phi(i)| \) and \( \phi \) is the first eigenvector of \( L_{t-1} \) corresponding to \( \lambda_{min} \). By (7), \( e_{k_t}e_{k_t}^\top \) shifts the center of the \( k_t \)-th Gershgorin disc of \( L_{t-1} \) to the right by 1. Our strategy is then to right-shift the Gershgorin disc corresponding to the largest magnitude entry \( k_t^* \in S_{t-1} \) in \( \phi \) which contains \( \lambda_{min} \), thus promoting a larger \( \lambda_{min} \) in \( L_t \); i.e., select sample \( k_t^* \) where

\[
k_t^* = \arg\max_{k_t \in S_{t-1}} |\phi(k_t)|, \quad \text{where } L_{t-1}\phi = \lambda_{min}(L_{t-1}) \phi
\]

We justify our strategy from another perspective using the following lemma, which shows that the sample corresponding to the largest magnitude component in \( \phi \) induces the largest increase in \( \lambda_{min} \) in the limit.

**Lemma 2.** An optimal solution for the problem

\[
\arg\max_{k_t \in S_{t-1}} \lim_{\delta \to 0} \lambda_{min}(L_{t-1} + \delta e_{k_t}e_{k_t}^\top)
\]

is \( k_t^* = \arg\max_{k_t \in S_{t-1}} |\phi(k_t)| \), where \( \phi \) is the first eigenvector of \( L_{t-1} \) corresponding to \( \lambda_{min}(L_{t-1}) \).

See Appendix 2 for a detailed proof. Thus by computing \( k_t^* \) using (10), we are solving optimization (10), which is a proxy approximating original (7). Computing \( k_t^* \) in (10) is computationally light, since the first eigenvector \( \phi \) is computed only once to identify a new sample among candidates in \( S_{t-1} \).

\footnote{By restricting \( k_t^* \in S_{t-1} \), one cannot guarantee that \( |\phi(k_t^*)| \geq |\phi(i)| \), \( \forall i \), but in practice, an already chosen sample \( j \in S_{t-1} \) with self-loop at node \( j \) in \( L_{t-1} \) is very unlikely to have the largest magnitude \( |\phi(j)| \).}
3.2 Fast Repeated Eigenvector Computation with Warm Start

In this paper, we adopt the LOBPCG method to compute first eigenvector $\phi$ of matrix $L_{t-1}$ in (9), which is very efficient for large sparse matrix [6]. With an initial input $x_0$, in each iteration, LOBPCG works as follows: (1) Multiply $L$ with $x_i \in \mathbb{R}^{mn}$ (guess of the first eigenvector in $i$-th iteration) with complexity is $O(n|E_r| + m|E_r|)$; (2) Perform a Rayleigh-Ritz step [18] to solve an eigenvalue problem with complexity $O(r^3)$, where $r$ is the number of computed eigenvectors. In our problem, $r = 1$. Hence, for identifying a new sample, the complexity of LPBPCG in our problem is $O((n|E_r| + m|E_r|)R)$, where $R$ is the iteration number. Our proposed algorithm benefits computationally from warm start when deploying LOBPCG: we use the estimated first eigenvector $\tilde{Q}$ for warm start. The small change between $L_t$ and $L_{t-1}$ (the Frobenius norm difference is only 1) ensures a good initial guess, reducing the number of iterations until convergence.

We write the pseudo code of our sampling strategy in Algorithm 1 called Gershgorin circle shift (GCS)-based sampling.

Given a sampling budget $K$, the complexity of our proposed GCS method is $O(K(n|E_r| + m|E_r|)R + Kmn)$. If the row and column graphs are both sparse such that $|E_r| = O(m)$ and $|E_t| = O(n)$, then the complexity is $O(KRmn)$. Though the spectral proxy based sampling method in [6] also computes the first eigenvector of a submatrix of $L^p$ via LOBPCG, it does not benefit from warm start, and the complexity of computing $L^p x$ will be higher than $L x$ by at least by a factor $p$.

Algorithm 1 Proposed GCS sampling

*pseudo code*

Input: Sample budget $K$; $L = \alpha I_n \otimes L_r + \beta L_c \otimes I_m$; random vector $v$

Initialization: $S = \emptyset$

1: While $|S| < K$
2: compute the first eigenvector $\phi$ of $L$ with initial guess $v$
3: $i^* \leftarrow \max_{i \in S} |\phi(i)|$
4: $S \leftarrow S \cup \{i^*\}$
5: Update $L = L + e_{i^*}e_{i^*}^\top$ and $v = \phi$
6: end While
7: return $S$

4 Iterative Spectral Graph Sampling for Matrix Completion

Though we identify samples using LOBPCG to compute first eigenvectors repeatedly with warm start, sampling on a graph with $mn$ nodes is still expensive for large real-world matrix completion datasets. We thus propose an efficient block-wise sampling method for matrix completion problem. Towards a simpler presentation, we omit $\Omega$ in $A_\Omega$ in the sequel. First, we divide $Q$ into two terms:

$$Q = \left( q\tilde{A} + \alpha I_n \otimes L_r \right) + \left( (1 - q)\tilde{A} + \beta L_c \otimes I_m \right) \equiv Q_1 + Q_2$$

where $0 < q < 1$ is a split parameter.

$Q_1$ and $Q_2$ have block-diagonal structures (with appropriate permutations), and each selected sample affects respective $\lambda_{\text{min}}$'s of $Q_1$ and $Q_2$. From Gershgorin circle perspective, each selected sample shifts one disc in $\alpha I_n \otimes L_r$ and $\beta L_c \otimes I_m$ by $q$ and $1 - q$, respectively. To see the connection clearly, we first write $Q_1 \in \mathbb{R}^{mn \times mn}$ as:

$$Q_1 = q\tilde{A} + \alpha I_n \otimes L_r = q \begin{bmatrix} \tilde{A}_1 & 0 & \cdots & 0 \\ 0 & \tilde{A}_2 & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & \tilde{A}_n \end{bmatrix} + \alpha \begin{bmatrix} L_r & 0 & \cdots & 0 \\ 0 & L_r & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & L_r \end{bmatrix}$$

where $\tilde{A}_j \in \mathbb{R}^{m \times m}$ is the $j$-th diagonal block of $\tilde{A}$. When entry $(i, j)$ of the target matrix is sampled, $\tilde{A}_j(i, i) = 1$. Similarly, we can write $Q_2 \in \mathbb{R}^{mn \times mn}$ as:

$$Q_2 = (1 - q)\tilde{A} + \beta L_c \otimes I_m = (1 - q)\tilde{A} + \beta \begin{bmatrix} L_c(1, 1)I_m & L_c(1, 2)I_m & \cdots & L_c(1, n)I_m \\ L_c(2, 1)I_m & L_c(2, 2)I_m & \cdots & L_c(2, n)I_m \\ \vdots & \vdots & \ddots & \vdots \\ L_c(n, 1)I_m & L_c(n, 2)I_m & \cdots & L_c(n, n)I_m \end{bmatrix}$$

5
It is known that there exists a permutation matrix $P$ such that $P(\mathbf{L}_m \otimes \mathbf{I}_m)P^T = \mathbf{I}_m \otimes \mathbf{L}_c$ [12]. The corresponding sampling matrix $\hat{A} = P\hat{\mathbf{A}}P^T$ for $\mathbf{I}_m \otimes \mathbf{L}_c$ is also a diagonal matrix. Combined with the block-diagonal property of $\mathbf{I}_m \otimes \mathbf{L}_c$, we write the permuted version of $Q_2$ as follows:

$$
\hat{Q}_2 = PQ_2P^T = (1-q)\hat{A} + \mathbf{I}_m \otimes \mathbf{L}_c = \begin{bmatrix}
(1-q)\hat{A}_1 + \beta\mathbf{L}_c \\
(1-q)\hat{A}_2 + \beta\mathbf{L}_c \\
\vdots \\
(1-q)\hat{A}_m + \beta\mathbf{L}_c
\end{bmatrix}
$$

where $\hat{A}_i \in \mathbb{R}^{n \times n}$ is the $i$-th diagonal block of $\hat{A}$.

We see that if entry $(i, j)$ of the target matrix is sampled, the $i$-th row of the $j$-th unitary rectangular row in matrix $Q_2$ will be shifted by $1-q$, which corresponds to shifting the $j$-th disc in the $i$-th block in matrix $Q_2$. Therefore, sampling entry $(i, j)$ implies $\hat{A}_i(j, j) = 1$. Combined with the property of $Q_1$, we know that sampling data at $(i, j)$ will promote the smallest eigenvalue of $j$-th (i-th) diagonal block of $Q_1$ ($Q_2$) by letting $\hat{A}_i(i, i) = 1$ ($\hat{A}_i(j, j) = 1$). Based on this connection, we propose an iterative sampling strategy to select entries based on the shifted Laplacian matrix of each group or cluster, which is the diagonal block matrix in $Q_1$ or $Q_2$, respectively. Specifically, we start sampling the matrix signal from the first column ($j = 1$) based on the Laplacian matrix of the first cluster in $Q_1$. If its first eigenvector has the largest energy at the $i$-th index, we will sample data at $(i, 1)$ and then proceed sampling based on the shifted Laplacian matrix of the $i$-th group in $Q_2$.

We continue to choose samples alternating between clusters and groups until the sampling budget is exhausted.

Since our method using LOBPCG benefits from warm start, the computation complexity can be further reduced if we choose more than one sample from the same cluster (or group). We thus introduce a warm start parameter $\zeta$ to trade off sampling performance and computation complexity; its sensitivity will be studied in the results section. Detailed iterative sampling pseudo-code is shown in Algorithm 2 called iterative Gershgorin circle shift (IGCS)-based sampling. As we analyzed in Section 5, our proposed GCS method has complexity $O(KRnm)$, while the complexity of IGCS is just $O(KR_0c)$, where $R_0$ is the convergence iteration number of LOBPCG in IGCS and $c = \max\{m, n\}$. Therefore, with iterative sampling, the complexity is reduced by at least a factor $\min\{m, n\}$ using the IGCS.

### 5 Graph Construction

There exist many methods to construct finite graphs from data, so that the observed signal(s) are smooth (low-pass) with respect to the constructed graphs [16, 17]. For completeness, we overview methods we chose to construct row and column graphs using which we select samples. We stress that our work focuses on sampling; the discussion here merely demonstrates that our graph sampling schemes can be practically realized in combination with existing graph learning methods.

- **G1: Feature-based graph.** As done in graph-based matrix completion methods [7, 22], when the user/l item profiles (e.g., age, gender and occupation of users and genre of the items) are available,
we first deploy our proposed IGCS sampling method on a large real-world dataset.

We list partial profiles of some simulated datasets.

we construct a weighted nearest neighbor graph using GSPBox [29].

- **G2: Content-based graph from observed information.** When features of data points are not available, we construct row and column graphs only from partial matrix entries, extending method used in [17]. Specifically, assuming first that the partially observed matrix is $\mathbf{Z} = \mathbf{A}_p \circ \mathbf{X}$ for a given sampling set $\Gamma$, for each pair of users $\{i, j\}$ their partial ratings are in the $i$-th and $j$-th rows of matrix $\mathbf{Z}$, denoted by $z_i$ and $z_j$. We then compute the inter-node distance as $d_{ij} = ||z_i(R_{ij}) - z_j(R_{ij})||_2/\sqrt{||R_{ij}||}$, where $R_{ij} = R_i \cap R_j$, and $R_i$ is the set of items rated by user $i$. If $|R_{ij}| = 0$, we set $d_{ij} = \infty$. We then compute the edge weight between users $i$ and $j$ as $w_{ij} = \exp\{-\frac{(d_{ij} - d_{\min})^2}{\gamma}\}$ if $d_{ij} \leq d_i^*$, where $d_{\min} = \min_{\{i,j\}} d_{ij}$ and $d_i^*$ is the threshold of user $i$ for sparsifying $\mathbf{W}_i$; otherwise, we set $w_{ij} = 0$. Likewise, the item graph is constructed similarly using column ratings in observed matrix $\mathbf{Z}$.

6 Experiments

We present experimental results of our proposed sampling methods and other competing schemes, combined with several state-of-the-art matrix completion methods. In all experiments, we set $\alpha = 0.5$ and $\alpha = \beta = 0.1$ for our method.

We list partial profiles of some simulated datasets in Table 1. See Appendix [1] for the detailed version.

- **Intelligent sampling improves matrix completion performance**

We first deploy our proposed IGCS sampling method on a large real-world dataset ML100K [5] to collect samples. Since the typically used ratio between training and testing in ML100K is 80% to 20%, in our experiments, we first randomly generate 60K samples as the initial known data, and then proceed to sample the following 20K data based on our proposed IGCS method or random sampling on different graphs (G2) is constructed from those 60K given samples based on the second method in Section 5. Average RMSE combined with different matrix completion (MC) methods and graphs are listed in Table 3, where the best performance number for each MC method is marked in boldface. In the widely used feature-based graph (G1), our proposed IGCS sampling (right side) achieves better performance than random sampling (left side) using almost all MC methods. Further, when we select entries on the content-based graph (G2), our IGCS method substantially outperforms random sampling. Note that when G2 is used instead of G1, RMSE for random sampling is almost the same for every graph-based MC method. Hence, we can conclude that the performance improvement using G2 is due to the more informative samples intelligently collected using our proposed IGCS method.

We next test the IGCS method on various well-known real-world datasets, combined with GRALS MC method. Note that features needed to construct G1 are not available for most datasets, and so we use G2 as the underlying graph for sampling. For datasets Flixter, YahooMusic, Douban, typically random 90/10 training/test splits are used, and thus we choose 80% entries in the given training set as the initial samples to construct G2 and then use our IGCS method to sample the next 20% data. For other datasets, we first randomly generate 90/10 training/test split and then use the same method to collect samples and compute RMSE. The percentage of initial samples in datasets ML1M and ML10M (submatrix) is set to be 90%, since the training set is too large. Resulting RMSEs of the proposed IGCS sampling with different $\zeta$ are shown in Table 4, where the best RMSE value of each dataset is marked in boldface. Table 4 confirms that IGCS outperforms random sampling in every dataset with various data size, density and rating level. Moreover, when the warm start parameter $\zeta$ becomes larger, the RMSE value of our method deteriorates only slightly, but still significantly outperforms random sampling in most datasets.

- **Performance superiority to other sampling strategies on small datasets**

Classical graph sampling methods like [6] are not applicable for the aforementioned large real-world datasets because of their high complexity. To compare our method with classical graph sampling strategies, we conduct experiments on small-size datasets, where the matrix is completed by dual smoothness based method [2]. Two relatively low-complexity classical methods called graph weight
coherence (GWC) [30] and localized operator coverage (LOC) [33] are implemented, which select samples on matrix \( L_{tr} = L_r \otimes L_c \otimes L_m \) directly. Authors in [26] proposed a structured sampling method to collect data assuming the matrix signal is \( (\eta_1, \eta_2) \)-bandlimited on the product graph.

In experiments, we set two combinations (11, 9) and (9, 11) as bandwidth prior, and then artificially increased the parameter \( L = |L_1| + |L_2| \) to get rectangular output. where \( L_1 \) and \( L_2 \) are its selected row and column indices, respectively. After sampling, we recorded its sample size \( |L_1| \times |L_2| \) and corresponding reconstruction error. The RMSE value on noiseless and noisy synthetic Netflix dataset are shown in Fig. 1 (a) (b). We observe that our proposed GCS outperforms all competitors. Though with parameter (11, 9), the PG sampling has comparable RMSE value, its performance deteriorates drastically by just swapping the parameters to (9, 11). This implies that the method is very sensitive to bandwidth estimates. Further, our proposed iterative sampling method IGCS achieves good performance compared with GCS and LOC with much lower complexity \( O(KR_0 \max\{m, n\}) \). Recall that the complexity of GCS is \( O(KR_{mn}) \) and LOC is \( O(KJ) \) with \( J \) as the number of non-zero entries in \( L^4 \) which is at least \( O(dmn) \).

We also simulate sampling methods on the densest submatrix \((100 \times 200)\) from two real-world datasets ML100K and ML10M. By assuming that the information outside this submatrix is given as prior, we construct graph \( G_2 \) via strategy in Section 5 also used in [17]. Since the ground truth submatrix is sparse, the sampling method must be constrained to sample on the sparse entries, for a fair RMSE comparison. The PG method proposed in [26] is structured and thus cannot satisfy this requirement. Further, LOC requires computation of a Chebyshev polynomial graph filter before sampling, which always results in out-of-memory error using our constructed graph. Thus, we show only executable sampling schemes for comparison. The resulting RMSE in Fig. 1(c) (d) show that our proposed IGCS has the lowest RMSE compared to GWC, GCS and random sampling.

| Table 3: RMSE for ML100K using random / IGCS sampling combined with different MC methods. Graph-based strategies are marked with √. |
|---------------------------------|---|---|---|
| MC methods | \( G^T \) | G1 | G2 |
| IMC [36] | 1.590 | 1.590 | 1.590 |
| SVT [5] | 1.021 | 1.021 | 1.021 |
| GRALS [31] | 0.947 | 0.945 | 0.893 |
| GMC [17] | 1.036 | 1.118 | 1.054 |
| GC-MC [7] | 0.899 | 0.899 | 0.858 |
| NMC [2] | - | 0.892 | 0.861 |

| Table 4: GRALS RMSE for different datasets using IGCS sampling with different \( \zeta \)'s. |
|---------------------------------|---|---|---|---|---|---|
| dataset | random | \( \zeta = 1 \) | \( \zeta = 3 \) | \( \zeta = 5 \) | \( \zeta = 7 \) |
| Flixster | 1.029 | 0.932 | 1.057 | 1.046 | 1.045 |
| Douban | 0.744 | 0.715 | 0.720 | 0.736 | 0.730 |
| YahooMusic | 96.987 | 59.172 | 44.546 | 52.391 | 47.082 |
| ML1H | 0.905 | 0.829 | 0.833 | 0.835 | 0.838 |
| Book-Crossing | 3.987 | 3.578 | 3.704 | 3.804 | 4.185 |
| ML10M | 0.706 | 0.655 | 0.665 | 0.665 | 0.656 |
| Jester | 0.214 | 0.160 | 0.162 | 0.162 | 0.165 |
| FilmTrust | 0.820 | 0.668 | 0.735 | 0.711 | 0.742 |

Figure 1: RMSE of different sampling methods for matrix completion on Synthetic Netflix (SN) [17], ML100K (100 \times 200) and ML10M (100 \times 200) datasets. The matrix is completed by dual smoothness based method [2].

Table 2: RMSE and IGCS sampling time on ML100K.

| graph | MC methods | random | \( \zeta = 1 \) | \( \zeta = 3 \) | \( \zeta = 5 \) | \( \zeta = 7 \) |
|-------|------------|---|---|---|---|---|
| G1    | GRALS      | 0.947 | 0.927 | 0.935 | 0.934 | 0.931 |
|       | GC-MC      | 0.898 | 0.889 | 0.895 | 0.897 | 0.891 |
|       | NMC        | 0.892 | 0.880 | 0.888 | 0.889 | 0.886 |
|       | Time (10^5s) | - | 1.104 | 0.503 | 0.375 | 0.329 |
| G2    | GRALS      | 0.945 | 0.871 | 0.870 | 0.882 | 0.882 |
|       | GC-MC      | 0.899 | 0.839 | 0.840 | 0.847 | 0.851 |
|       | NMC        | 0.892 | 0.840 | 0.845 | 0.843 | 0.852 |
|       | Time (10^5s) | - | 1.216 | 0.573 | 0.441 | 0.388 |
Faster sampling with different warm start

In this experiment, we implement our proposed IGCS using different warm start parameter $\zeta$ on different graphs and with different MC methods. The resulting RMSE values and sampling times are shown in Table 2 along with random sampling for comparison. The simulated dataset is ML100K, and the experiments are performed in a laptop with Intel Core i7-8750H and 16GB of RAM on Windows 10 to record execution time. In Table 2, the best performance numbers are marked in boldface. Table 2 shows that with increased $\zeta$, execution time of our method decreases rapidly, while the performance becomes slightly worse for different graphs and MC methods and always outperforms random sampling.

References

[1] Filmtrust dataset. https://www.librec.net/datasets.html Accessed: 2019-5-23.
[2] IMC code. http://lamda.nju.edu.cn/xum/code/Maxide.zip Accessed: 2019-5-23.
[3] Jester dataset. https://grouplens.org/datasets/jester/ Accessed: 2019-5-23.
[4] Book-crossing dataset. http://www2.informatik.uni-freiburg.de/~cziegler/BX/ Accessed: 2019-5-23.
[5] Movielens. https://grouplens.org/datasets/movielens/ Accessed: 2019-5-23.
[6] A. Anis, A. Gadde, and A. Ortega. Efficient sampling set selection for bandlimited graph signals using graph spectral proxies. In IEEE Transactions on Signal Processing, volume 64, no.14, pages 3775–3789, July 2016.
[7] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. arXiv preprint arXiv:1706.02263, 2017.
[8] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. SIAM Journal on Optimization, 20(4):1956–1982, 2010.
[9] E. Candes and Y. Plan. Matrix completion with noise. In Proceedings of the IEEE, volume 98, no.6, pages 925–936, April 2010.
[10] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. Foundations of Computational mathematics, 9(6):717, 2009.
[11] Shayok Chakraborty, Jiayu Zhou, Vineeth Balasubramanian, Sethuraman Panjtanathan, Ian Davidson, and Jieping Ye. Active matrix completion. In 2013 IEEE 13th International Conference on Data Mining, pages 81–90. IEEE, 2013.
[12] Harold V Henderson and Shayle R Searle. The vec-permutation matrix, the vec operator and kronecker products: A review. Linear and multilinear algebra, 9(4):271–288, 1981.
[13] Roger A Horn, Roger A Horn, and Charles R Johnson. Matrix analysis. Cambridge university press, 1990.
[14] Sheng-Jun Huang, Miao Xu, Ming-Kun Xie, Masashi Sugiyama, Gang Niu, and Songcan Chen. Active feature acquisition with supervised matrix completion. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 1571–1579. ACM, 2018.
[15] Weiyu Huang, Antonio G Marques, and Alejandro Ribeiro. Matrix completion via graph signal processing. In 43rd IEEE Int. Conf. Acoust., Speech and Signal Process., Calgary, AB, pages 15–20, 2018.
[16] Weiyu Huang, Antonio G Marques, and Alejandro R Ribeiro. Rating prediction via graph signal processing. IEEE Transactions on Signal Processing, 66(19):5066–5081, 2018.
[17] Vassilis Kalofolias, Xavier Bresson, Michael Bronstein, and Pierre Vandergheynst. Matrix completion on graphs. arXiv preprint arXiv:1408.1717, 2014.
[18] Andrew V Knyazev. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM journal on scientific computing*, 23(2):517–541, 2001.

[19] Akshay Krishnamurthy and Aarti Singh. Low-rank matrix and tensor completion via adaptive sampling. In *Advances in Neural Information Processing Systems*, pages 836–844, 2013.

[20] Guangcan Liu, Qingshan Liu, and Xiaotong Yuan. A new theory for matrix completion. In *Advances in Neural Information Processing Systems*, pages 785–794, 2017.

[21] Martin Fodslette Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4):525–533, 1993.

[22] Federico Monti, Michael Bronstein, and Xavier Bresson. Geometric matrix completion with recurrent multi-graph neural networks. In *Advances in Neural Information Processing Systems*, pages 3697–3707, 2017.

[23] Yuji Nakatsukasa, Naoki Saito, and Ernest Woei. Mysteries around the graph laplacian eigenvalue 4. *Linear Algebra and its Applications*, 438(8):3231–3246, 2013.

[24] D. M. Nguyen, E. Tsigilianni, and N. Deligiannis. Extendable neural matrix completion. In *IEEE ICASSP*, pages 6328–6332, April 2018.

[25] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst. Graph signal processing: Overview, challenges, and applications. In *Proceedings of the IEEE*, volume 106, no.5, pages 808–828, May 2018.

[26] Guillermo Ortiz-Jiménez, Mario Coutino, Sundeep Prabhakar Chepuri, and Geert Leus. Sampling and reconstruction of signals on product graphs. *arXiv preprint arXiv:1807.00145*, 2018.

[27] Guillermo Ortiz-Jiménez, Mario Coutino, Sundeep Prabhakar Chepuri, and Geert Leus. Sparse sampling for inverse problems with tensors. *arXiv preprint arXiv:1806.10976*, 2018.

[28] Jiahao Pang and Gene Cheung. Graph laplacian regularization for image denoising: Analysis in the continuous domain. *IEEE Transactions on Image Processing*, 26(4):1770–1785, 2017.

[29] Nathanael Perraudin, Johan Paratte, David I. Shuman, Vassilis Kalofolias, Pierre Vandergheynst, and David K. Hammond. GSPBOX: A toolbox for signal processing on graphs. *arXiv preprint arXiv:1408.5781*, 2014.

[30] G. Puy, N. Tremblay, R. Gribonval, and P. Vandergheynst. Random sampling of bandlimited signals on graphs. In *Applied and Computational Harmonic Analysis*, volume 44, no.2, pages 446–473, March 2018.

[31] Nikhil Rao, Hsiang-Fu Yu, Pradeep K Ravikumar, and Inderjit S Dhillon. Collaborative filtering with graph information: Consistency and scalable methods. In *Advances in Neural Information Processing Systems*, pages 2107–2115, 2015.

[32] Neil Rubens, Mehdi Elahi, Masashi Sugiyama, and Dain Kaplan. Active learning in recommender systems. In *Recommender systems handbook*, pages 809–846. Springer, 2015.

[33] Akie Sakiyama, Yuichi Tanaka, Toshihisa Tanaka, and Antonio Ortega. Eigendecomposition-free sampling set selection for graph signals. *arXiv preprint arXiv:1809.01827*, 2018.

[34] Jonathan Richard Shewchuk et al. An introduction to the conjugate gradient method without the agonizing pain, 1994.

[35] Fen Wang, Gene Cheung, and Yongchao Wang. Fast sampling of graph signals with noise via neumann series conversion. In *IEEE ICASSP*, pages 5456–5460, May 2019.

[36] Miao Xu, Rong Jin, and Zhi-Hua Zhou. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in Neural Information Processing Systems*, pages 2301–2309, 2013.
Graph Sampling for Matrix Completion Using Recurrent First Eigenvector Computation — Supplementary Materials

A Derivations of linear equation

We first compute the derivative of $f(X)$ with respect to the optimization variable $X$:

$$
\frac{\partial f(X)}{\partial X} = A \odot (X - Y) + \alpha L_r X + \beta XL_c
$$

(14)

whose vector form by using the property $\text{vec}(AB) = (I_n \otimes A)\text{vec}(B) = (B^T \otimes I_m)\text{vec}(A)$ is:

$$
\frac{\partial f[\text{vec}(X)]}{\partial [\text{vec}(X)]} = (\tilde{A} \odot \alpha I_n \otimes L_r + \beta L_c \otimes I_m)\text{vec}(X) - \text{vec}(Y)
$$

(15)

Note that $A \odot A = A$ and $Y = A \odot (X + N)$, so $\text{vec}(A \odot Y) = \text{vec}(Y)$. To obtain an optimal solution to the original problem, we set $\partial f(X)/\partial X = 0$, which in vector form leads to

$$
(\tilde{A} \otimes 2)\text{vec}(X^*) = \text{vec}(Y)
$$

(16)

B The property of matrix $Q$

Note that $\lambda_{\min}(I_n \otimes L_r) = 0$ since $\lambda_{\min}(L_r) = 0$ and $\lambda_{\min}(L_c \otimes I_m) = \lambda_{\min}(I_m \otimes L_c) = 0$, so $\lambda_{\min}$ of $\alpha I_n L_r + \beta L_c \otimes I_m$ is at least 0 based on Weyl’s inequality on eigenvalues that $\lambda_{\min}(A + B) \geq \lambda_{\min}(A) + \lambda_{\min}(B)$. Moreover, the vectorized sampling operator $\tilde{A} \otimes 2$ is positive semi-definite (PSD). Combined with the invertibility of $Q$, we know that $Q = \tilde{A} \otimes 2 + \alpha I_n \otimes L_r + \beta L_c \otimes I_m$ is symmetric and positive definite (PD), and the optimal solution to problem (16) in closed form is:

$$
\text{vec}(X^*) = (\tilde{A} \otimes 2 + \alpha I_n \otimes L_r + \beta L_c \otimes I_m)^{-1} \text{vec}(Y) = Q^{-1}\text{vec}(Y)
$$

(17)

C The upper-bound of $\lambda_{\max}(Q)$

Reusing the notations in Section 3, let $L = \alpha I_n \otimes L_r + \beta L_c \otimes I_m$, and $Q = \tilde{A} \otimes 2 + L$. Specifically,

$$
L = \alpha \begin{bmatrix}
L_r & 0 & \cdots & 0 \\
0 & L_r & \vdots & \\
\vdots & \vdots & \ddots & 0 \\
0 & \cdots & \cdots & L_r
\end{bmatrix} + \beta \begin{bmatrix}
L_c(1, 1)I_m & L_c(1, 2)I_m & \cdots & L_c(1, n)I_m \\
L_c(2, 1)I_m & L_c(2, 2)I_m & \cdots & L_c(2, n)I_m \\
\vdots & \vdots & \ddots & \vdots \\
L_c(n, 1)I_m & L_c(n, 2)I_m & \cdots & L_c(n, n)I_m
\end{bmatrix}
$$

(18)

For $k = i + m \times (j - 1)$ with $i \in \mathcal{V}_r$ and $j \in \mathcal{V}_c$, the $k$-th row of matrix $L$ (denoted by $s_k$) is a combination of the $i$-th row of $L_r$ (denoted by $v_i$) and the $j$-th row of $L_c$ (denoted by $t_j$). It is easy to see that $s_k(k) = \alpha v_i(i) + \beta t_j(j)$ and $\sum_{l=1;l\neq k}^m |s_k(l)| = \alpha \sum_{l=1;l\neq i}^m |v_i(l)| + \beta \sum_{l=1;l\neq j}^m |t_j(l)|$. Since $v_i(i) = \sum_{l=1;l\neq i}^m |v_i(l)|$ and $t_j(j) = \sum_{l=1;l\neq j}^m |t_j(l)|$ by the definitions of combinatorial Laplacian matrix $L_r$ and $L_c$, we know that $s_k(k) = \sum_{l=1;l\neq k}^m |s_k(l)|, \forall k \in \{1, 2, \ldots, mn\}$. Based on the Gershgorin circle theorem presented in Section 3.1, we know that the eigenvalues of matrix $L$ are all bounded in $[0, 2\max_k\{s_k(k)\}]$, where 0 is the lower bound of all left ends of Gershgorin discs, and $2\max_k\{s_k(k)\}$ is the upper bound of all right ends of those discs. Note that $v_i(i) = D_r(i, i)$ and $t_j(j) = D_c(j, j)$. Assuming $\max_i\{D_r(i, i)\} \leq d_r$ and $\max_j\{D_c(j, j)\} \leq d_c$ (degree constrained graphs), we will have

$$
\max_k\{s_k(k)\} = \alpha \max_i\{v_i(i)\} + \beta \max_j\{t_j(j)\} = \alpha \max_i\{D_r(i, i)\} + \beta \max_j\{D_c(j, j)\} \leq \alpha d_r + \beta d_c
$$

(19)

Therefore, $\lambda_{\max}(L)$ will be upper-bounded by $2\max_k\{s_k(k)\}$, i.e., $2\alpha d_r + 2\beta d_c$. Because $\tilde{A} \otimes 2$ is a diagonal matrix with diagonal entries 0 or 1, $\lambda_{\max}(Q)$ will be upper-bounded by $2\alpha d_r + 2\beta d_c + 1$ if the two factor graphs are degree-bounded by $d_r$ and $d_c$, respectively.
D  Proof of Lemma 1

It is observed that in vector form, \( \text{vec}(Y) = \tilde{A}_\Omega [\text{vec}(X + N)] \). Thus, a solution to the system of linear equations (3) can be written as

\[
\text{vec}(X^*) = Q^{-1} \text{vec}(Y) = Q^{-1} \tilde{A}_\Omega [\text{vec}(X + N)]
\]

Thus the squared error of estimator \( \text{vec}(X^*) \) with respect to \( \text{vec}(X) \) is

\[
||\text{vec}(X^*) - \text{vec}(X)||_2
\]
\[
= ||\text{vec}(N) - Q^{-1} (\alpha I_n \otimes L_r + \beta L_c \otimes I_m) \text{vec}(X + N)||_2
\]
\[
\leq ||Q^{-1} (\alpha I_n \otimes L_r + \beta L_c \otimes I_m) \text{vec}(X + N)||_2 + ||\text{vec}(N)||_2
\]
\[
\leq ||Q^{-1}||_2 ||(\alpha I_n \otimes L_r + \beta L_c \otimes I_m) \text{vec}(X + N)||_2 + ||\text{vec}(N)||_2
\]
\[
= \rho ||Q^{-1}||_2 + ||\text{vec}(N)||_2
\]

From this inequality, we see that the sampling set \( \Omega \) only influences matrix \( Q \) in this upper bound. Moreover, since \( Q \) is symmetric and positive definite, we will have

\[
||Q^{-1}||_2 = \sqrt{\lambda_{\max}\{(Q^{-1})^T Q^{-1}\}} = \lambda_{\max}(Q^{-1}) = \frac{1}{\lambda_{\min}(Q)}
\]

Hence, we arrive at the following

\[
||\text{vec}(X^*) - \text{vec}(X)||_2 \leq \frac{\rho}{\lambda_{\min}(Q)} + ||\text{vec}(N)||_2
\]

E  Proof of Lemma 2

At step \( t - 1 \) of the greedy sampling method, from previous steps we have collected a sampling set \( S_{t-1} \) and constructed a corresponding shifted Laplacian matrix \( L_{t-1} \) in the following form:

\[
L_{t-1} = L + \sum_{j \in S_{t-1}} e_je_j^T
\]

Assuming we further add new sample with shift \( \delta \), we will get \( \tilde{L}_t = L_{t-1} + \delta e_k e_k^T \), where \( k \in S_{t-1} \). Using Rayleigh quotient theorem [13], we write \( \lambda_{\min} \) of matrix \( \tilde{L}_t \) as

\[
\lambda_{\min}(\tilde{L}_t) = \min_{x} \frac{x^T L_{t-1} x + \delta x^T e_k e_k^T x}{x^T x} = \min_{x} \frac{x^T L_{t-1} x + \delta x(k_t)^2}{x^T x}
\]

with minimizer to be the first eigenvector of \( L_{t-1} \) when \( \delta \to 0 \), i.e., \( \lim_{\delta \to 0} x^* = \phi \). Therefore,

\[
\lim_{\delta \to 0} \lambda_{\min}(\tilde{L}_t) = \lambda_{\min}(L_{t-1}) + \frac{\delta \phi(k_t)^2}{\phi} \phi
\]

With previous collected \( S_{t-1} \), \( \lambda_{\min}(L_{t-1}) \) is known as a priori, so an optimal sampling solution of maximizing \( \lambda_{\min}(\tilde{L}_t) \) when \( \delta \to 0 \) is

\[
k_t^* = \arg\max_{k \in S_{t-1}} \lambda_{\min}(\tilde{L}_t) = \arg\max_{k \in S_{t-1}} \frac{\delta \phi(k_t)^2}{\phi} = \arg\max_{k \in S_{t-1}} |\phi(k_t)|
\]

where the last equality holds since \( \phi^T \phi \) is the same for different sampling indices.
Table 5: Profiles of experimented datasets.

| dataset         | Exp. users | items | features | entries | density | entry levels |
|-----------------|------------|-------|----------|---------|---------|--------------|
| Synthetic Netflix [17] | 200        | 100   |          | 20,000  | 100%    | 1,2,...,5    |
| ML100K [5]      | 100        | 200   |          | 12566   | 62.83%  | 1,2,...,5    |
| ML10M [5]       | 1000       | 500   |          | 345904  | 69.18%  | 0.5,1,...,5  |
| Douban [7]      | 3000       | 3000  |          | 136,891 | 1.52%   | 1,2,...,5    |
| Flixter [7]     | 3000       | 3000  |          | 26,173  | 0.29%   | 0.5,1,...,5  |
| YahooMusic [7]  | 3000       | 3000  |          | 1,000,209 | 4.47%  | 1,2,...,5    |
| Book-Crossing   | 1000       | 1000  |          | 3166    | 0.32%   | (0,1)        |
| ML1M            | 6040       | 3706  | ✓        | 1,000,209 | 4.47%  | 1,2,...,5    |
| Jester          | 1000       | 100   |          | 31880   | 3.19%   | 0.5,1,...,4  |
| FilmTrust       | 1000       | 1000  | ✓        | 31880   | 3.19%   | 0.5,1,...,4  |

Table 6: RMSE and IGCS sampling with different $\zeta$'s on ML100K.

| graph | MC methods | $\zeta$ = 1 | $\zeta$ = 2 | $\zeta$ = 3 | $\zeta$ = 4 | $\zeta$ = 5 | $\zeta$ = 6 | $\zeta$ = 7 | $\zeta$ = 8 | $\zeta$ = 9 |
|-------|------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| G1    | GRALS      | 0.927        | 0.928        | 0.935        | 0.929        | 0.934        | 0.934        | 0.934        | 0.934        | 0.932        |
|       | GC-MC      | 0.889        | 0.891        | 0.895        | 0.893        | 0.897        | 0.891        | 0.898        | 0.898        | 0.892        |
|       | NMC        | 0.880        | 0.887        | 0.888        | 0.889        | 0.889        | 0.880        | 0.886        | 0.890        | 0.886        |
|       | Time (10^3 s) | 1.104        | 0.662        | 0.503        | 0.427        | 0.375        | 0.340        | 0.320        | 0.300        | 0.284        |
| G2    | GRALS      | 0.871        | 0.871        | 0.870        | 0.871        | 0.882        | 0.889        | 0.882        | 0.887        | 0.898        |
|       | GC-MC      | 0.839        | 0.840        | 0.840        | 0.846        | 0.847        | 0.839        | 0.851        | 0.860        | 0.860        |
|       | NMC        | 0.840        | 0.842        | 0.845        | 0.844        | 0.843        | 0.842        | 0.852        | 0.857        | 0.856        |
|       | Time (10^3 s) | 1.216        | 0.734        | 0.573        | 0.491        | 0.441        | 0.409        | 0.388        | 0.372        | 0.356        |

F Detailed profiles of all experimental datasets

Please refer to Table 5.

- For datasets Synthetic Netflix, Douban, Flixter, YahooMusic and ML1M, we use their widely used size for experiments.
- For ML100K, we use its typical size for experiments in Tab. 3/2 and use its submatrix with size $100 \times 200$ for experiments in Fig. 1.
- For ML10M, we use its densest submatrix with size $100 \times 200$ for experiments in Fig. 1 and its submatrix with size $1000 \times 500$ for experiments in Tab. 4. Note that the original matrix size for ML10M is $69878 \times 10677$.
- For datasets Book-Crossing, Jester and FilmTrust, we randomly generate a submatrix with reasonable size of the original online large rating matrix for experiments in Tab. 4.

G Experimental results with more parameter settings

Please refer to Table 6 and 7.

Table 7: RMSE of the proposed IGCS sampling on different datasets with different $\zeta$'s. The matrix completion method is GRALS method.

| dataset       | random | $\zeta$ = 1 | $\zeta$ = 3 | $\zeta$ = 5 | $\zeta$ = 7 | $\zeta$ = 9 |
|---------------|--------|-------------|-------------|-------------|-------------|-------------|
| Flixter       | 1.029  | 0.932       | 1.057       | 1.046       | 1.045       | 1.015       |
| Douban        | 0.744  | 0.715       | 0.720       | 0.736       | 0.730       | 0.736       |
| YahooMusic    | 96.987 | 59.172      | 52.391      | 47.082      | 48.258      |             |
| ML1M          | 0.905  | 0.829       | 0.833       | 0.835       | 0.838       | 0.843       |
| Book-Crossing | 3.987  | 3.578       | 3.704       | 3.804       | 4.185       | 4.597       |
| Jester        | 0.706  | 0.655       | 0.656       | 0.656       | 0.656       | 0.662       |
| FilmTrust     | 0.214  | 0.160       | 0.162       | 0.162       | 0.165       | 0.167       |
H Detailed parameter settings of competing sampling methods and matrix completion methods

1. PG sampling: we set the bandwidth prior to be \((\eta_1, \eta_2) = (11, 9)\) or \((\eta_1, \eta_2) = (11, 9)\), and then artificially increased the parameter \(L\), defined as row-column sum in paper [26]. After sampling, we recorded its output sample size and reconstruction error correspondingly.

2. LOC sampling: this eigen-decomposition-free sampling method required the bandwidth prior as input, which was set to be 1000 in our experiments. Other settings were the same as done in [33].

3. GWC sampling: we used the ‘estimated’ setting for computing the optimal random sampling probability. The bandwidth was also set to be 1000 for this method. The function ‘datasample’ was adopted for selecting samples based on the optimal probability without replacement [30].

Though we implemented several modern matrix completion methods, our main contribution was on the matrix sampling component, combined with which the matrix completion performance could be improved substantially.

- IMC completion: we use the public code related to [36] in [2] directly.
- SVT completion: we use the public code related to [8] directly.
- GMC completion: regularization parameters used in paper [17] were set to be \(\gamma_n = 3\), \(\gamma_r = \alpha = 0.1\) and \(\gamma_c = \beta = 0.1\).
- GRALS completion: in the simulations, we assume the rank to be 5 for GRALS, and its Laplacian matrix input were computed from \(L_{ih} = L_c + 0.1I_n\) and \(L_{iw} = L_r + 0.1I_m\), as used in [26].
- GC-MC completion: to achieve the similar performance demonstrated in paper [7], we also set the training epochs to be 1000. Other parameters were the same as used in [7].
- NMC completion: we adopt the same parameter settings as in [24] where the number of training epochs 10000.