# Dual Optimization for Kolmogorov Model Learning Using Enhanced Gradient Descent

Qiyou Duan, Hadi Ghauch, *Member, IEEE,*, and Taejoon Kim, *Senior Member, IEEE*

***Abstract*—Data representation techniques have made a substantial contribution to advancing data processing and machine learning (ML). Improving predictive power was the focus of previous representation techniques, which unfortunately perform rather poorly on the interpretability in terms of extracting underlying insights of the data. Recently, the Kolmogorov model (KM) was studied, which is an interpretable and predictable representation approach to learning the underlying probabilistic structure of a set of random variables. The existing KM learning algorithms using semi-definite relaxation with randomization (SDRwR) or discrete monotonic optimization (DMO) have, however, limited utility to big data applications because they do not scale well computationally. In this paper, we propose a computationally scalable KM learning algorithm, based on the regularized dual optimization combined with enhanced gradient descent (GD) method. To make our method more scalable to large-dimensional problems, we propose two acceleration schemes, namely, the eigenvalue decomposition (EVD) elimination strategy and an approximate EVD algorithm. Furthermore, a thresholding technique by exploiting the error bound analysis and leveraging the normalized Minkowski $\ell_1$-norm, is provided for the selection of the number of iterations of the approximate EVD algorithm. When applied to big data applications, it is demonstrated that the proposed method can achieve compatible training/prediction performance with significantly reduced computational complexity; roughly two orders of magnitude improvement in terms of the time overhead, compared to the existing KM learning algorithms. Furthermore, it is shown that the accuracy of logical relation mining for interpretability by using the proposed KM learning algorithm exceeds $80\%$.**

***Index Terms*—Kolmogorov model (KM), dual optimization, gradient descent (GD), scalability, large-dimensional dataset, big data, low latency, approximate eigenvalue decomposition (EVD).**

## I. Introduction

The digital era, influencing and reshaping the behaviors, performances, and standards, etc., of societies, communities, and individuals, has presented a big challenge for the conventional mode of data processing. Data consisting of numbers, words, and measurements becomes available in such huge volume, high velocity, and wide variety that it ends up outpacing human-oriented computing. It is urgent to explore the intelligent tools necessary for processing the staggering amount of data. Machine learning (ML), dedicated to providing insights into patterns in big data and extracting pieces of information hidden inside, arises and has been used in a wide variety of applications, such as computer vision [1], telecommunication [2], and recommendation systems [3]–[6]. Nevertheless, traditional ML algorithms become computationally inefficient and fail to scale up well as the dimension of data grows. A major issue that remains to be addressed is to find effective ML algorithms that perform well on both predictability and interpretability as well as are capable of tackling large-dimensional data with low complexity.

### A. Related Work

Data representation, providing driving forces to the advancing ML-based techniques, has lately attracted a great deal of interest because it transforms large-dimensional data into low-dimensional alternatives by capturing their key features and make them amenable for processing, prediction, and analysis. The gamut of data representation techniques including matrix factorization (MF) [7], [8], singular value decomposition (SVD)-based models [9], [10], nonnegative models (NNM) [11], and deep neural networks [12] have been shown to perform well in terms of predictive power (the capability of predicting the outcome of random variables that are outside the training set). Unfortunately, these techniques perform rather poorly on the interpretability (the capability of extracting additional information or insights that are hidden inside the data) because on the one hand, they are not developed to directly model the outcome of random variables; on the other hand, they fall under the black-box category which lacks transparency and accountability of predictive models [13]. Recently, a Kolmogorov model (KM) that directly represents a binary random variable as a superposition of elementary events in probability space was proposed [14]; KM models the outcome of a binary random variable as an inner product between two structured vectors, one probability mass function vector and one binary indicator vector. This inner product structure exactly represents an actual probability. Carefully examining association rules between two binary indicator vectors grants the interpretability of KM that establishes mathematically logical/causal relations between different random variables.

Previously, the KM learning was formulated as a coupled combinatorial optimization problem [14] by decomposing it into two subproblems: i) linearly-constrained quadratic program (LCQP) and ii) binary quadratic program (BQP), which can be alternatively solved by utilizing block coordinate descent (BCD). An elegant, low-complexity Frank-Wolfe (FW)

Q. Duan is with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong (e-mail: qyduan.ee@my.cityu.edu.hk).

H. Ghauch is with the Department of COMELEC, Telecom-ParisTech, Paris, France (e-mail: hadi.ghauch@telecom-paristech.fr).

T. Kim is with the Department of Electrical Engineering and Computer Science, The University of Kansas, Lawrence, KS 66045 USA (e-mail: taejoonkim@ku.edu).

The work of Taejoon Kim was supported in part by the National Science Foundation (NSF) under Grant CNS1955561, and in part by the Office of Naval Research (ONR) under Grant N00014-21-1-2472.

algorithm [15] was used to optimally solve the LCQP by exploiting the unit probability simplex structure. Whereas, it is known to be unpromising to find algorithms to exactly solve the BQP problems in polynomial time. To get around this challenge, relaxation methods for linear [16], [17], quadratic [18], second-order cone [19], [20], and semi-definite programming (SDP) [21], [22], were proffered to produce a feasible solution close to the optimal solution of the original problem. Among these relaxation methods, the semi-definite relaxation (SDR) has been shown to have a tighter approximation bound than that of others [23], [24]. Thus, an SDR with randomization (SDRwR) method [25] was employed to optimally solve the BQP of the KM learning in an asymptotic sense [14]. To address the high-complexity issue due to the reliance on the interior point methods, a branch-reduce-and-bound (BRB) algorithm based on discrete monotonic optimization (DMO) [26], [27] was proposed. However, the DMO approach only shows its efficacy in a low-dimensional setting and starts to collapse as the dimension increases. In short, the existing KM methods [14], [27] suffer from a similar drawback, namely, being unscalable. Unfortunately, the latter limitation hampers the application of them to large-scale datasets, for instance, the MovieLens 1 million (ML1M) dataset[1]. It is thus crucial to explore low-complexity and scalable methods for KM learning.

Duality often arises in linear/nonlinear optimization models in a wide variety of applications such as communication networks [28], economic markets [29], and structural design [30]. Simultaneously, the dual problem possesses some good mathematical, geometric, or computational structures that can be exploited to provide an alternative way of handling the intricate primal problems by using iterative methods, such as the first-order gradient descent (GD) [31], [32] and quasi-Newton method [33], [34]. It is for this reason that the first-order iterative methods are widely used when optimizing/training large-scale data representations (e.g., deep neural networks) and machine learning algorithms. We are motivated by these iterative first-order methods to effectively resolve the combinatorial challenge of KM learning.

### B. Overview of Methodologies and Contributions

We present a computationally scalable approach to the KM learning problem by proposing an enhanced GD algorithm and an approximate eigenvalue decomposition (EVD) with thresholding scheme based on dual optimization. Our main contributions are listed below.

- We provide a reformulation of the BQP subproblem of KM learning to a regularized dual optimization problem that ensures strong duality and is amenable to be solved by simple GD. Compared to the existing SDRwR [14] and DMO [27], the proposed dual optimization method proffers a more efficient and scalable solution to KM learning. This algorithmic approach is ideally suited to the KM learning, but is not limited thereto, and can be applied to any realistic problem involving BQP.
- Motivated by the fact that EVD is required at each iteration of GD, which introduces a computational bottleneck when applied to big data, an enhanced GD that eliminates the EVD computation when it is feasible is proposed to accelerate the computational speed. When the elimination is infeasible and EVD must be computed, we explore an approximate EVD based on the Lanczos method [35] by taking account of the fact that computing exact, entire EVD is usually unnecessary. We focus on analyzing the approximation error of the approximate EVD. A tractable thresholding scheme is then proposed to determine the number of iterations of the approximate EVD by exploiting the structure of the upper bound on the approximation error and utilizing the normalized Minkowski $\ell_1$-norm.
- Extensive numerical simulation results are presented to demonstrate the efficacy of the proposed KM learning algorithm. When applied to large-scale datasets (e.g., ML1M dataset), it is shown that the proposed method can achieve comparable training and prediction performance with significantly reduced computational cost of more than two orders of magnitude, compared to the existing KM learning algorithms. Finally, the interpretability of the proposed method is validated by exploiting the mathematically logical relations. We show that the accuracy of logical relation mining by using the proposed method exceeds $80\%$.

*Notation:* A bold lowercase letter $\mathbf{a}$ is a vector and a bold capital letter $\mathbf{A}$ is a matrix. $A(i,j)$, $\mathbf{A}(:,j)$, $\mathrm{trace}(\mathbf{A})$, $\mathrm{diag}(\mathbf{A})$, $\mathrm{rank}(\mathbf{A})$, $\lambda_{\max}(\mathbf{A})$, and $\sigma_{\max}(\mathbf{A})$ denote the $(i,j)$th entry, $j$th column, trace, main diagonal elements, rank, largest eigenvalue, and largest singular value of $\mathbf{A}$, respectively. $a(i)$ is the $i$th entry of $\mathbf{a}$, $\mathbf{a}(m:n) \triangleq [a(m),\cdots,a(n)]^T$, and $\mathrm{diag}(\mathbf{a})$ is a diagonal matrix with $\mathbf{a}$ on its main diagonal. $\langle \mathbf{X},\mathbf{Y}\rangle$ is the Frobenius inner product of two matrices $\mathbf{X}$ and $\mathbf{Y}$, i.e., $\langle \mathbf{X},\mathbf{Y}\rangle = \mathrm{trace}(\mathbf{X}^T\mathbf{Y})$. $\mathbf{X} \succeq \mathbf{0}$ indicates that the matrix $\mathbf{X}$ is positive semi-definite (PSD). $\mathbf{e}_i$ is the $i$th column of the identity matrix of appropriate size. $\mathbf{1}$ and $\mathbf{0}$ denote the all-one and all-zero vectors, respectively. $\mathbb{S}^{N\times N}$, $\mathbb{R}^N_+$, and $\mathbb{B}^N$ denote the $N\times N$ symmetric matrix space, nonnegative real-valued $N\times 1$ vector space, and $N\times 1$ binary vector space with each entry chosen from $\{0,1\}$, respectively. For $\mathbf{S}\in\mathbb{S}^{N\times N}$, $\boldsymbol{\lambda}(\mathbf{S}) \triangleq [\lambda_1(\mathbf{S}),\lambda_2(\mathbf{S}),\cdots,\lambda_N(\mathbf{S})]^T \in \mathbb{R}^{N\times 1}$ where $\lambda_n(\mathbf{S})$ is the $n$th eigenvalue of $\mathbf{S}$, $n=1,\ldots,N$. $\mathrm{supp}(\mathbf{a}) \triangleq \{i|a_i\neq 0, i\in\{1,\ldots,N\}\}$ is the support set of $\mathbf{a}\in\mathbb{R}^N$ and $|\mathcal{A}|$ denotes the cardinality of a set $\mathcal{A}$. Finally, $\mathcal{E}_1 \Rightarrow \mathcal{E}_2$ indicates that one outcome $\mathcal{E}_1$ completely implies another one $\mathcal{E}_2$.

## II. System Model and Preliminaries

In this section, we briefly discuss the concept of KM and its learning framework.

### A. Preliminaries

We consider a double-index set of binary random variables $X_{u,i}\in\{0,1\}$, $\forall (u,i)\in\mathcal{S}$, where $\mathcal{S}\triangleq\{(u,i)|(u,i)\in\mathcal{U}\times\mathcal{I}\}$ ($\mathcal{U}=\{1,\ldots,U\}$ and $\mathcal{I}=\{1,\ldots,I\}$ are the index sets of $u$

[1]https://grouplens.org/datasets/movielens/1m/

and $i$, respectively) denotes the set of all index pairs. Thus, $X_{u,i}$ can represent any two-dimensional learning applications (involving matrices) such as movie recommendation systems [11], DNA methylation for cancer detection [36], and beam alignment in multiple-antenna systems [37], [38]. We let $\Pr(X_{u,i} = 1) \in [0, 1]$ be the probability that the event $X_{u,i} = 1$ occurs. Since the random variable considered here is binary, the following holds $\Pr(X_{u,i} = 1) + \Pr(X_{u,i} = 0) = 1$. Without loss of generality, we can focus on one outcome, for instance, $X_{u,i} = 1$. Then, the $D$-dimensional KM of the random variable $X_{u,i}$ is given by

$$\Pr(X_{u,i} = 1) = \boldsymbol{\theta}_u^T \boldsymbol{\psi}_i, \ \forall (u, i) \in \mathcal{S}, \tag{1}$$

where $\boldsymbol{\theta}_u \in \mathbb{R}_+^D$ is the *probability mass function vector* and $\boldsymbol{\psi}_i \in \mathbb{B}^D$ is the *binary indicator vector*. Specifically, $\boldsymbol{\theta}_u$ is in the unit probability simplex $\mathcal{P} \triangleq \{\mathbf{p} \in \mathbb{R}_+^D | \mathbf{1}^T \mathbf{p} = 1\}$, i.e., $\boldsymbol{\theta}_u \in \mathcal{P}$, and $\boldsymbol{\psi}_i$ denotes the support set of $X_{u,i}$ (associated with the case when $X_{u,i} = 1$). The KM in (1) is built under a measurable probability space defined on $(\Omega, \mathcal{E})$ ($\Omega$ denotes the sample space and $\mathcal{E}$ is the event space consisting of subsets of $\Omega$) and satisfies the following conditions: i) $\Pr(E) \geq 0$, $\forall E \in \mathcal{E}$ (nonnegativity), ii) $\Pr(\Omega) = 1$ (normalization), and iii) $\Pr(\cup_{i=1}^{\infty} E_i) = \sum_{i=1}^{\infty} \Pr(E_i)$ for the disjoint events $E_i \in \mathcal{E}$, $\forall i$ (countable additivity) [39]. By (1), $X_{u,i}$ is modeled as stochastic mixtures of $D$ Kolmogorov elementary events. In addition, note that $\Pr(X_{u,i} = 0) = \boldsymbol{\theta}_u^T (\mathbf{1} - \boldsymbol{\psi}_i)$.

### B. KM Learning

Assume that the empirical probability of $X_{u,i} = 1$, denoted by $p_{u,i}$, is available from the training set $\mathcal{K} \triangleq \{(u, i) | u \in \mathcal{U}_{\mathcal{K}} \subseteq \mathcal{U}, i \in \mathcal{I}_{\mathcal{K}} \subseteq \mathcal{I}\} \subseteq \mathcal{S}$. Obtaining the empirical probabilities $\{p_{u,i}\}$ for the training set depends on the application and context in practical systems; we will illustrate an example for recommendation systems at the end of this section. The KM learning involves training, prediction, and interpretation as described below.

*1) Training:* The KM training proceeds to optimize $\{\boldsymbol{\theta}_u\}$ and $\{\boldsymbol{\psi}_i\}$ by solving the $\ell_2$-norm minimization problem:

$$\begin{aligned} \{\boldsymbol{\theta}_u^\star\}, \{\boldsymbol{\psi}_i^\star\} = & \underset{\{\boldsymbol{\theta}_u\}, \{\boldsymbol{\psi}_i\}}{\operatorname{argmin}} \sum_{(u,i) \in \mathcal{K}} (\boldsymbol{\theta}_u^T \boldsymbol{\psi}_i - p_{u,i})^2 \\ & \text{s.t. } \boldsymbol{\theta}_u \in \mathcal{P}, \ \boldsymbol{\psi}_i \in \mathbb{B}^D, \forall (u, i) \in \mathcal{K} \end{aligned}. \tag{2}$$

To deal with the coupled combinatorial nature of (2), a BCD method [14], [40] was proposed by dividing the problem in (2) into two subproblems: i) LCQP:

$$\boldsymbol{\theta}_u^{(\tau+1)} = \underset{\boldsymbol{\theta}_u \in \mathcal{P}}{\operatorname{argmin}} \, \boldsymbol{\theta}_u^T \mathbf{Q}_u^{(\tau)} \boldsymbol{\theta}_u - 2 \boldsymbol{\theta}_u^T \mathbf{w}_u^{(\tau)} + \varrho_u, \ \forall u \in \mathcal{U}_{\mathcal{K}}, \tag{3}$$

where $\mathbf{Q}_u^{(\tau)} \triangleq \sum_{i \in \mathcal{I}_u} \boldsymbol{\psi}_i^{(\tau)} \boldsymbol{\psi}_i^{(\tau)^T}$, $\mathbf{w}_u^{(\tau)} \triangleq \sum_{i \in \mathcal{I}_u} \boldsymbol{\psi}_i^{(\tau)} p_{u,i}$, $\varrho_u \triangleq \sum_{i \in \mathcal{I}_u} p_{u,i}^2$, $\mathcal{I}_u \triangleq \{i | (u, i) \in \mathcal{K}\}$, and $\tau$ is the index of BCD iterations, and ii) BQP:

$$\boldsymbol{\psi}_i^{(\tau+1)} = \underset{\boldsymbol{\psi}_i \in \mathbb{B}^D}{\operatorname{argmin}} \, \boldsymbol{\psi}_i^T \mathbf{S}_i^{(\tau+1)} \boldsymbol{\psi}_i - 2 \boldsymbol{\psi}_i^T \mathbf{v}_i^{(\tau+1)} + \rho_i, \forall i \in \mathcal{I}_{\mathcal{K}}, \tag{4}$$

where $\mathbf{S}_i^{(\tau+1)} \triangleq \sum_{u \in \mathcal{U}_i} \boldsymbol{\theta}_u^{(\tau+1)} \boldsymbol{\theta}_u^{(\tau+1)^T}$, $\mathbf{v}_i^{(\tau+1)} \triangleq \sum_{u \in \mathcal{U}_i} \boldsymbol{\theta}_u^{(\tau+1)} p_{u,i}$, $\rho_i \triangleq \sum_{u \in \mathcal{U}_i} p_{u,i}^2$, and $\mathcal{U}_i \triangleq \{u | (u, i) \in \mathcal{K}\}$. BCD has been successful in tackling coupled optimization problems in applications such as the transceiver design in wireless communications [28], [41]–[44]. The coupling among $\{\boldsymbol{\theta}_u\}$ and $\{\boldsymbol{\psi}_i\}$ in (2) makes BCD an ideal method to alternatively handle the coupled optimization problem. It has been studied that the BCD method converges to a local minimum of the original problem in (2) if a unique minimizer is found for both blocks, $\{\boldsymbol{\theta}_u\}$ and $\{\boldsymbol{\psi}_i\}$ [14], [45].

By exploiting the fact that the optimization in (3) was carried out over the unit probability simplex $\mathcal{P}$, a simple iterative FW algorithm [15] was employed to optimally solve (3), while the SDRwR was employed to asymptotically solve the BQP in (4) [25]. It is also possible to solve (4) directly without a relaxation and/or randomization, based on the DMO approach [27]. However, the DMO in [27] was shown only to be efficient when the dimension $D$ is small (e.g., $D \leq 8$); its computational cost blows up as $D$ increases (e.g., $D > 20$).

*2) Prediction:* Similar to other supervised learning methods, the trained KM parameters $\{\boldsymbol{\theta}_u^\star\}, \{\boldsymbol{\psi}_i^\star\}$ are used to predict probabilities over a test set $\mathcal{T}$ as

$$\hat{p}_{u,i} \triangleq \boldsymbol{\theta}_u^{\star T} \boldsymbol{\psi}_i^\star, \ \forall (u, i) \in \mathcal{T}, \tag{5}$$

where $\mathcal{T} \cap \mathcal{K} = \phi$ and $\mathcal{T} \cup \mathcal{K} = \mathcal{S}$.

*3) Interpretation:* KM offers a distinct advantage, namely, the interpretability by drawing on fundamental insights into the mathematically logical relations among the data. For two random variables $X_{u,i}$ and $X_{u,j}$ taken from the training set $\mathcal{K}$, i.e., $(u, i) \in \mathcal{K}$ and $(u, j) \in \mathcal{K}$, if the support sets of $\boldsymbol{\psi}_i^\star$ and $\boldsymbol{\psi}_j^\star$ satisfy $\text{supp}(\boldsymbol{\psi}_j^\star) \subseteq \text{supp}(\boldsymbol{\psi}_i^\star)$, then two logical relations between the outcomes of $X_{u,i}$ and $X_{u,j}$ can be inferred: the first outcome of $X_{u,i}$ implies the same one for $X_{u,j}$ while the second outcome of $X_{u,j}$ implies the second one for $X_{u,i}$, i.e., $X_{u,i} = 1 \Rightarrow X_{u,j} = 1$ and $X_{u,j} = 0 \Rightarrow X_{u,i} = 0$ [14, Proposition 1]. It is important to note that logical relations emerged from KM are based on the formalism of implications. Thus, they hold from a strictly mathematical perspective, and are general.

An implication of the introduced KM learning is illustrated by taking an example of movie recommendation systems as follows.

**Illustrative Example:** Suppose there are two users ($U = 2$) who have rated two movie items ($I = 2$). In this example, $X_{u,i} = 1$ denotes the event that user $u$ likes the movie item $i$, $\forall u \in \{1, 2\}, \forall i \in \{1, 2\}$. Then, $\Pr(X_{u,i} = 1)$ denotes the probability that user $u$ likes item $i$ (conversely, $\Pr(X_{u,i} = 0)$ denotes the probability that user $u$ dislikes item $i$). Suppose $D = 4$ in (1). Then, the four elementary events can represent four different movie genres including i) Comedy, ii) Thriller, iii) Action, and iv) Drama. The empirical probability corresponding to $X_{u,i} = 1$ can be obtained by

$$p_{u,i} \triangleq \frac{r_{u,i}}{r_{\max}}, \tag{6}$$

where $r_{u,i}$ denotes the rating score that user $u$ has provided for item $i$ and $r_{\max}$ is the maximum rating score. In a 5-star rating system ($r_{\max} = 5$), we consider the following matrix as

an example:

$$\begin{bmatrix} p_{1,1} & p_{1,2} \\ p_{2,1} & p_{2,2} \end{bmatrix} = \begin{bmatrix} 0.8 & 0.4 \\ * & 0.6 \end{bmatrix}, \tag{7}$$

where $p_{2,1}$ is unknown (as in the '*' entry) and $\{p_{1,1}, p_{1,2}, p_{2,2}\}$ constitutes the training set of empirical probability where $\mathcal{K} = \{(1,1),(1,2),(2,2)\}$. By solving the KM learning problem in (2) for the empirical probabilities provided in (7), one can find the optimal model parameters, $\{\boldsymbol{\theta}_u^\star\}$ and $\{\boldsymbol{\psi}_i^\star\}$ (an optimal solution to (2)), which is given by

$$\boldsymbol{\theta}_1^\star = [0.4\ 0.2\ 0.1\ 0.3]^T,\ \boldsymbol{\theta}_2^\star = [0.1\ 0.3\ 0.1\ 0.5]^T;$$
$$\boldsymbol{\psi}_1^\star = [1\ 0\ 1\ 1]^T,\ \boldsymbol{\psi}_2^\star = [0\ 0\ 1\ 1]^T.$$

Then, we can predict $p_{2,1}$ ($\mathcal{T} = \{(2,1)\}$) by using the learned KM parameters $\boldsymbol{\theta}_2^\star$ and $\boldsymbol{\psi}_1^\star$ as $\hat{p}_{2,1} = \boldsymbol{\theta}_2^{\star T}\boldsymbol{\psi}_1^\star = 0.7$. In this example, the following inclusion holds $\text{supp}(\boldsymbol{\psi}_2^\star) \subset \text{supp}(\boldsymbol{\psi}_1^\star)$. Thus, if a certain user (user 1 or 2) likes movie item 1, this logically implies that the user also likes movie item 2.

*Remark 1:* In contrast to the KM in (1), the state-of-the-art method, MF [7], [8], considers an inner product of two arbitrary vectors without having implicit or desired structures in place. While NNM [11] has a similar structure as (1), the distinction is that NNM relaxes the binary constraints on $\boldsymbol{\psi}_i$ to a nonnegative box, i.e., $\boldsymbol{\psi}_i \in [\mathbf{0}, \mathbf{1}]$, and thus sacrifices the highly interpretable nature of KM. Unlike the existing data representation techniques, the KM can exactly represent the outcome of random variables in a Kolmogorov sense. As illustrated in Section V, this in turn improves the prediction performance of the KM compared to other existing data representation techniques. Despite its predictability benefit, the existing KM learning methods [14], [27], however, suffer from high computational complexity and a lack of scalability. In particular, the LCQP subproblem, which can be efficiently solved by the FW algorithm, has been well-investigated, while resolving the BQP introduces a major computational bottleneck. It is thus of great importance to study more efficient and fast KM learning algorithms that are readily applicable to large-scale problems.

## III. Proposed Method

To scale KM learning, we propose an efficient, first-order method to the BQP subproblem in (4).

### A. Dual Problem Formulation

We transform the BQP subproblem in (4) to a dual problem. To this end, we formulate an equivalent form to the BQP in (4) as

$$\min_{\mathbf{x}\in\{+1,-1\}^D} \mathbf{x}^T\mathbf{A}_0\mathbf{x} + \mathbf{a}^T\mathbf{x}, \tag{8}$$

where $\rho_i$ in (4) is ignored in (8), $\mathbf{x} = 2\boldsymbol{\psi}_i - \mathbf{1} \in \{+1,-1\}^D$, $\mathbf{A}_0 = \frac{1}{4}\mathbf{S}_i$, and $\mathbf{a} = \frac{1}{2}\mathbf{S}_i^T\mathbf{1} - \mathbf{v}_i$. For simplicity, the iteration index $\tau$ is omitted hereinafter. By introducing $\mathbf{X}_0 = \mathbf{x}\mathbf{x}^T$ and $\mathbf{X} = \begin{bmatrix} 1 & \mathbf{x}^T \\ \mathbf{x} & \mathbf{X}_0 \end{bmatrix} \in \mathbb{S}^{(D+1)\times(D+1)}$, the problem in (8) can be rewritten as

$$\min_{\mathbf{x},\mathbf{X}_0} \quad \langle \mathbf{X}_0, \mathbf{A}_0\rangle + \mathbf{a}^T\mathbf{x}, \tag{9a}$$
$$\text{s.t.} \quad \text{diag}(\mathbf{X}_0) = \mathbf{1}, \tag{9b}$$
$$\mathbf{X} \succeq \mathbf{0}, \tag{9c}$$
$$\text{rank}(\mathbf{X}) = 1. \tag{9d}$$

Solving (9) directly is NP-hard due to the rank constraint in (9d), thus we turn to convex relaxation methods. The SDR to (9) can be expressed in a homogenized form with respect to $\mathbf{X}$ as

$$\min_{\mathbf{X}} \quad f(\mathbf{X}) \triangleq \langle \mathbf{X}, \mathbf{A}\rangle, \tag{10a}$$
$$\text{s.t.} \quad \langle \mathbf{B}_i, \mathbf{X}\rangle = 1,\ i = 1,\ldots,D+1, \tag{10b}$$
$$\mathbf{X} \succeq \mathbf{0}, \tag{10c}$$

where $\mathbf{A} = \begin{bmatrix} 0 & (1/2)\mathbf{a}^T \\ (1/2)\mathbf{a} & \mathbf{A}_0 \end{bmatrix} \in \mathbb{S}^{(D+1)\times(D+1)}$ and $\mathbf{B}_i = [\mathbf{0}_1\ \cdots\ \mathbf{0}_{i-1}\ \mathbf{e}_i\ \mathbf{0}_{i+1}\ \cdots\ \mathbf{0}_{D+1}] \in \mathbb{R}^{(D+1)\times(D+1)}$. Note that the diagonal constraint in (9b) has been equivalently transformed to $D+1$ equality constraints in (10b). While the problem in (9) is combinatorial due to the rank constraint, the relaxed problem in (10) is a convex SDP. Moreover, the relaxation is done by dropping the rank constraint.

We further formulate a regularized SDP formulation of (10) as

$$\begin{aligned} \min_{\mathbf{X}} \quad & f_\gamma(\mathbf{X}) \triangleq \langle \mathbf{X}, \mathbf{A}\rangle + \frac{1}{2\gamma}\|\mathbf{X}\|_F^2, \\ \text{s.t.} \quad & \langle \mathbf{B}_i, \mathbf{X}\rangle = 1,\ i = 1,\ldots,D+1, \\ & \mathbf{X} \succeq \mathbf{0}, \end{aligned} \tag{11}$$

where $\gamma > 0$ is a regularization parameter. With a Frobenius-norm term regularized, the strict convexity of (11) is ensured, which in turn makes strong duality hold for the feasible dual problem of (11). In this work, we leverage this fact that the duality gap is zero for (11) (a consequence of strong duality) to solve the dual problem. Using a larger regularization parameter $\gamma$ yields better quality of the solution to (10), but at the cost of slower convergence. In addition, the two problems in (10) and (11) are equivalent as $\gamma \to \infty$. The choice of $\gamma$ will be further discussed in Section V-A.

Given the regularized SDP formulation in (11), its dual problem and the gradient of the objective function are of interest.

*Lemma 1:* Suppose the problem in (11) is feasible. Then, the dual problem of (11) is given by

$$\max_{\mathbf{u}\in\mathbb{R}^{D+1}} \quad d_\gamma(\mathbf{u}) \triangleq -\mathbf{u}^T\mathbf{1} - \frac{\gamma}{2}\|\Pi_+(\mathbf{C}(\mathbf{u}))\|_F^2, \tag{12}$$

where $\mathbf{u} \in \mathbb{R}^{D+1}$ is the vector of Lagrange multipliers associated with each of the $D+1$ equality constraints of (11), $\mathbf{C}(\mathbf{u}) \triangleq -\mathbf{A} - \sum_{i=1}^{D+1} u_i\mathbf{B}_i$, and $\Pi_+(\mathbf{C}(\mathbf{u})) \triangleq \sum_{i=1}^{D+1}\max(0, \lambda_i(\mathbf{C}(\mathbf{u})))\mathbf{p}_i\mathbf{p}_i^T$, in which $\lambda_i(\mathbf{C}(\mathbf{u}))$ and $\mathbf{p}_i$, $i = 1,\ldots,D+1$, respectively, are the eigenvalues and corresponding eigenvectors of $\mathbf{C}(\mathbf{u})$. The gradient of $d_\gamma(\mathbf{u})$

**Algorithm 1** GD for Solving the Dual Problem in (14)

**Input:** $\mathbf{A}$, $\{\mathbf{B}_i\}_{i=1}^{D+1}$, $D$, $\mathbf{u}_0$, $\gamma$, $\epsilon$ (*tolerance threshold value*), and $I_{\max}$ (*maximum number of iterations*).
**Output:** $\mathbf{u}^\star$.
1: **for** $i = 0, 1, 2, \ldots, I_{\max}$ **do**
2: Calculate the gradient: $\nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i)$.
3: Compute the descent direction: $\Delta\mathbf{u}_i = -\nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i)$.
4: Find a step size $t_i$ (via *backtracking line search*), and $\mathbf{u}_{i+1} = \mathbf{u}_i + t_i\Delta\mathbf{u}_i$.
5: **if** $\|t_i\Delta\mathbf{u}_i\|_2 \leq \epsilon$ **then** terminate and **return** $\mathbf{u}^\star = \mathbf{u}_{i+1}$.
6: **end if**
7: **end for**

with respect to $\mathbf{u}$ is

$$\nabla_{\mathbf{u}} d_\gamma(\mathbf{u}) = -\mathbf{1} + \gamma\Phi[\Pi_+(\mathbf{C}(\mathbf{u}))], \tag{13}$$

where $\Phi[\Pi_+(\mathbf{C}(\mathbf{u}))] \triangleq [\langle\mathbf{B}_1, \Pi_+(\mathbf{C}(\mathbf{u}))\rangle, \cdots, \langle\mathbf{B}_{D+1}, \Pi_+(\mathbf{C}(\mathbf{u}))\rangle]^T \in \mathbb{R}^{D+1}$.

*Proof:* See Appendix A.

It is well known that $d_\gamma(\mathbf{u})$ in (12) is a strongly concave (piecewise linear) function, thereby making the Lagrange dual problem (12) a strongly convex problem having a unique global optimal solution [31]. Furthermore, the special structure of $\mathbf{C}(\mathbf{u})$ of Lemma 1, i.e., being symmetric, allows us to propose computationally efficient and scalable KM learning algorithms which can be applied to handle large-scale datasets with low latency.

### *B. Fast GD Methods For The Dual Problem*

*1) GD:* The dual problem in (12), having a strongly concave function $d_\gamma(\mathbf{u})$, is equivalent to the following unconstrained convex minimization problem

$$\min_{\mathbf{u}\in\mathbb{R}^{D+1}} \quad h_\gamma(\mathbf{u}) \triangleq \mathbf{u}^T\mathbf{1} + \frac{\gamma}{2}\|\Pi_+(\mathbf{C}(\mathbf{u}))\|_F^2, \tag{14}$$

with the gradient being $\nabla_{\mathbf{u}} h_\gamma(\mathbf{u}) = \mathbf{1} - \gamma\Phi[\Pi_+(\mathbf{C}(\mathbf{u}))]$. We first introduce a GD, which is detailed in Algorithm 1, to solve (14). Note that, due to the fact that the dual problem in (14) is unconstrained, a simple GD method is proposed here: indeed, we would need a projected GD method if there is constraint included, for which the computational complexity would be much larger because of the projection at each iteration.

In Algorithm 1, only the gradient of $h_\gamma(\mathbf{u}_i)$, i.e., $\nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i)$, is required to determine the descent direction. It is therefore a more practical and cost-saving method compared to standard Newton methods which demand the calculation of second-order derivatives and the inverse of the Hessian matrix. Moreover, Algorithm 1 does not rely on any approximation of the inverse of the Hessian matrix such as the quasi-Newton methods [46]. To find a step size in Step 4, we apply the backtracking line search method [47], which is based on the Armijo-Goldstein condition [48]. The algorithm is terminated when the pre-designed stopping criterion (for instance, $\|t_i\Delta\mathbf{u}_i\|_2 \leq \epsilon$ in Step 5, where $\epsilon > 0$ is a predefined tolerance) is satisfied. Finally, the computational complexity of Algorithm 1 is dominated by the EVD of a $(D+1)\times(D+1)$ matrix, needed to compute $\nabla_{\mathbf{u}} h_\gamma(\mathbf{u})$ in Step 2, which is given as $\mathcal{O}((D+1)^3)$.

**Algorithm 2** Enhanced GD with EVD Elimination

**Input:** $-\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$, $\{\mathbf{B}_i\}_{i=1}^{D+1}$, $D$, $\mathbf{u}_0$ (*with equal entries*), $\gamma$, $\epsilon$, and $I_{\max}$.
**Output:** $\mathbf{u}^\star$.
1: **for** $i = 0, 1, 2, \ldots, I_{\max}$ **do**
2: Calculate the gradient with EVD elimination:
3: **if** All $D+1$ elements of $\mathbf{u}_i$ are the same **then** ***Phase I:***
4: $\boldsymbol{\lambda}(\mathbf{C}(\mathbf{u}_i)) = \boldsymbol{\lambda}(-\mathbf{A}) - \mathbf{u}_i$ where $\boldsymbol{\lambda}(-\mathbf{A}) = \text{diag}(\mathbf{\Lambda})$.
5: Find the index set $\mathcal{I}_\lambda \triangleq \{j|\lambda_j(\mathbf{C}(\mathbf{u}_i)) > 0, j = 1, \ldots, D+1\}$.
6: **if** $\mathcal{I}_\lambda = \emptyset$ **then** *Phase I-A:* $\nabla h_\gamma(\mathbf{u}_i) = \mathbf{1}$.
7: **else** *Phase I-B:*
8: $\nabla h_\gamma(\mathbf{u}_i) = \mathbf{1} - \gamma\Phi[\Pi_+(\mathbf{C}(\mathbf{u}_i))]$, $\Pi_+(\mathbf{C}(\mathbf{u}_i)) = \sum_{j\in\mathcal{I}_\lambda}\lambda_j(\mathbf{C}(\mathbf{u}_i))\mathbf{V}(:,j)\mathbf{V}(:,j)^T$.
9: **end if**
10: **else** ***Phase II:***
11: **if** $\lambda_{\max}(-\mathbf{A}) + \lambda_{\max}(-\text{diag}(\mathbf{u}_i)) \leq 0$ **then** *Phase II-A:* $\nabla h_\gamma(\mathbf{u}_i) = \mathbf{1}$.
12: **else** *Phase II-B:*
13: $\mathbf{C}(\mathbf{u}_i) = \mathbf{V}_\mathbf{C}\mathbf{\Lambda}_\mathbf{C}\mathbf{V}_\mathbf{C}^T$, $\boldsymbol{\lambda}(\mathbf{C}(\mathbf{u}_i) = \text{diag}(\mathbf{\Lambda}_\mathbf{C})$.
14: $\nabla h_\gamma(\mathbf{u}_i) = \mathbf{1} - \gamma\Phi[\Pi_+(\mathbf{C}(\mathbf{u}_i))]$, $\Pi_+(\mathbf{C}(\mathbf{u}_i)) = \sum_{j\in\mathcal{I}_\lambda}\lambda_j(\mathbf{C}(\mathbf{u}_i))\mathbf{V}_\mathbf{C}(:,j)\mathbf{V}_\mathbf{C}(:,j)^T$.
15: **end if**
16: **end if**
17: Compute the descent direction: $\Delta\mathbf{u}_i = -\nabla h_\gamma(\mathbf{u}_i)$.
18: Find a step size $t_i$ (via *backtracking line search*), and $\mathbf{u}_{i+1} = \mathbf{u}_i + t_i\Delta\mathbf{u}_i$.
19: **if** $\|t_i\Delta\mathbf{u}_i\|_2 \leq \epsilon$ **then** terminate and **return** $\mathbf{u}^\star = \mathbf{u}_{i+1}$.
20: **end if**
21: **end for**

*2) Enhanced GD:* In Algorithm 1, an EVD of $\mathbf{C}(\mathbf{u}_i)$ is required at each iteration to determine $\Pi_+(\mathbf{C}(\mathbf{u}_i))$ and $\nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i)$. However, it is difficult to employ EVD per iteration as they require high computational cost ($\mathcal{O}(UI(D+1)^3)$) when large-scale datasets are involved (with very large $U$, $I$, and $D$). It is critical to reduce the computational cost of Algorithm 1 by avoiding the full computation of EVD or even discarding them.

In relation to the original SDP problem in (10), we can understand the PSD constraint in (10c) is now penalized as the penalty term in $h_\gamma(\mathbf{u})$, i.e., $\frac{\gamma}{2}\|\Pi_+(\mathbf{C}(\mathbf{u}))\|_F^2$. Thus, one of the key insights we will use is that: i) if the PSD constraint is not satisfied, the penalty term equals to zero, simplifying the objective function as $h_\gamma(\mathbf{u}) = \mathbf{u}^T\mathbf{1}$; in this case, the gradient is simply $\nabla_{\mathbf{u}} h_\gamma(\mathbf{u}) = \mathbf{1}$, eliminating the computation of EVD, and ii) if the PSD constraint is satisfied, the penalty term becomes nonzero and it requires the computation of EVD to find out $\nabla_{\mathbf{u}} h_\gamma(\mathbf{u})$. This fact leads to the following proposition showcasing the rule of updating $\mathbf{u}_{i+1}$ for the enhanced GD.

*Proposition 1:* The enhanced GD includes two cases de-

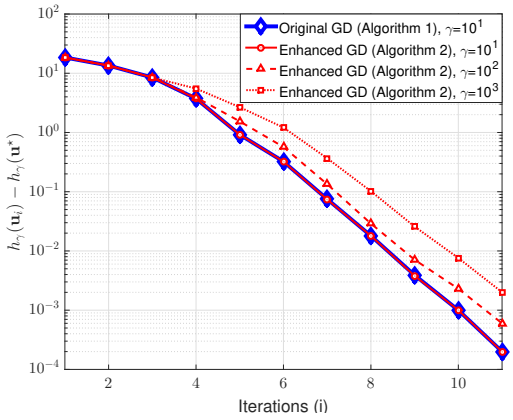



Fig. 1. Convergence rate comparison of GD in Algorithm 1 and the enhanced GD with EVD elimination in Algorithm 2 when $D = 4$.

pending on the condition of the PSD constraint as

$$\begin{cases} \text{Case A: if the PSD constraint does not meet} \\ \qquad\qquad \Rightarrow \mathbf{u}_{i+1} = \mathbf{u}_i - t_i \mathbf{1} \\ \text{Case B: if the PSD constraint meets} \\ \qquad\qquad \Rightarrow \mathbf{u}_{i+1} = \mathbf{u}_i - t_i \nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i) \end{cases}.$$

The key is to check if the PSD constraint in Proposition 1 is satisfied or not without the need of computing EVD. We propose a simple sufficient condition, based on the Weyl's inequality [49], as demonstrated in the proposed Algorithm 2.

In Algorithm 2, we focus on modifying Step 2 in Algorithm 1 by using an initial $\mathbf{u}_0$ with equal entries (for instance, $\mathbf{u}_0 = \mathbf{1}$) and exploiting the fact that $\nabla h_\gamma(\mathbf{u}_i) = \mathbf{1}$ if $\mathbf{C}(\mathbf{u}_i)$ is not PSD (Case A in Proposition 1) to reduce the computational cost of EVD. Step 4 in Algorithm 2 is due to the fact that the $k$th eigenvalue of $\mathbf{A} + \alpha\mathbf{I}$ ($\alpha \in \mathbb{R}$) is $\lambda_k(\mathbf{A}) + \alpha$. One of the key insights we leverage is that the choice of the sequence of gradient directions, i.e., $\nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i)$, $i = 0, 1, \ldots$, does not alter the optimality of the dual problem in (14). We approach the design of $\mathbf{u}_0$ with the goal of eliminating the computation of EVD to the most extent. Moreover, in Step 11 of Algorithm 2, the condition $\lambda_{\max}(-\mathbf{A}) + \lambda_{\max}(-\text{diag}(\mathbf{u}_i)) \leq 0 \Rightarrow \lambda_{\max}(\mathbf{C}(\mathbf{u}_i)) \leq 0$ (Case A in Proposition 1), holds because of the Weyl's inequality [49]. Note that we accelerate the original GD by reducing the computation of EVD from two different perspectives: one is from a better designed initial point $\mathbf{u}_0$ and another one is taking into acount the charateristics of $\mathbf{C}(\mathbf{u}_i)$, i.e., $\mathbf{C}(\mathbf{u}_i)$ is PSD or not. The EVD of $\mathbf{C}(\mathbf{u}_i)$ is required only when both the conditions "all the elements of $\mathbf{u}_i$ are the same" and "$\lambda_{\max}(-\mathbf{A}) + \lambda_{\max}(-\text{diag}(\mathbf{u}_i)) \leq 0$" are violated, as in *Phase II-B*. The effectiveness of the proposed enhanced GD will be validated by using numerical results in Section V.

Notice that Step 2 in computing $\nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i)$ of Algorithm 1 has been transformed into two different phases (each phase includes two sub-phases) in Algorithm 2. Algorithm 2 executes the four sub-phases in order and irreversibly. To be specific, the algorithm first enters *Phase I* at the initial iteration and ends up with *Phase II*. Once the algorithm enters *Phase II*, there is no way to return back to *Phase I*. The duration of four sub-phases varies with the characteristics of $\mathbf{C}(\mathbf{u}_i)$, which depends on $D$ and the dataset. An example will be taken to illustrate the duration of phases in Algorithm 2 in Section V-A. Algorithms 1 and 2 are based on GD, and thus the enhanced GD does not alter the convergency of Algorithm 1 [31].

**Algorithm 3** Randomization

**Input:** $\mathbf{A}$, $\Pi_+(\mathbf{C}(\mathbf{u}^\star)) = \mathbf{V}_+\mathbf{\Lambda}_+\mathbf{V}_+^T$, $D$, $\gamma$, and $I_{\text{rand}}$ (*the number of randomizations*).
**Output:** $\hat{\psi}$ (an approximate solution to the BQP in (4)).
1: Obtain $\mathbf{L} = \mathbf{V}_+\sqrt{\gamma\mathbf{\Lambda}_+}$ and $\mathbf{L}\mathbf{L}^T = \mathbf{X}^\star$.
2: **for** $\ell = 1, 2, \ldots, I_{\text{rand}}$ **do**
3: Generate an independent and identically distributed (i.i.d.) Gaussian random vector: $\boldsymbol{\xi}_\ell \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D+1})$.
4: Random sampling: $\tilde{\boldsymbol{\xi}}_\ell = \mathbf{L}\boldsymbol{\xi}_\ell$.
5: Discretization: $\tilde{\mathbf{x}}_\ell = \text{sign}(\tilde{\boldsymbol{\xi}}_\ell)$.
6: **end for**
7: Determine $\ell^\star = \text{argmin}_{\ell=1,\ldots,I_{\text{rand}}} \tilde{\mathbf{x}}_\ell^T \mathbf{A} \tilde{\mathbf{x}}_\ell$.
8: Approximation: $\hat{\mathbf{x}} = \tilde{\mathbf{x}}_{\ell^\star}(1) \cdot \tilde{\mathbf{x}}_{\ell^\star}(2 : D+1)$ and $\hat{\boldsymbol{\psi}} = (\hat{\mathbf{x}} + \mathbf{1})/2$.

*Proposition 2 (Convergence Rate of the Enhanced GD):* Let $\mathbf{u}^\star$ be the optimal solution to the strongly convex problem in (14). Then if we run Algorithm 2 for $i$ iterations, it will yield a solution $h_\gamma(\mathbf{u}_i)$ which satisfies

$$h_\gamma(\mathbf{u}_i) - h_\gamma(\mathbf{u}^\star) \leq \mathcal{O}(c^i), \ 0 < c < 1, \ i = 1, 2, \ldots$$

Intuitively, this means that the enhanced GD is guaranteed to converge with the convergence rate $\mathcal{O}(c^i)$.

*Remark 2:* Both Algorithms 1 and 2, which are based on the original GD [31], [32], result in the same update sequences $\{\mathbf{u}_i\}$. This phenomenon is captured in Fig. 1, in which the optimality gap (i.e., $h_\gamma(\mathbf{u}_i) - h_\gamma(\mathbf{u}^\star)$) as a function of the iteration number $i$ is depicted for Algorithms 1 and 2. In terms of flops, however, Algorithm 2 is more efficient than Algorithm 1. This leads to a dramatic reduction in the running time of Algorithm 2 since we mainly move on the direction obtained without the computation of EVD. Furthermore, the asymptotic error bound in Proposition 1 is unassociated with $\gamma$, in which the bound converges to zero as $i$ tends to infinity. This asymptote is captured by the slop of the error decrease as $\log(h_\gamma(\mathbf{u}_i) - h_\gamma(\mathbf{u}^\star)) \leq \mathcal{O}(\log(c) \cdot i)$, where $\log(c) < 0$ defines the asymptotic slop and is independent of $\gamma$. We utilize simulation curves to show the effect of $\gamma$ on the convergence rate of the enhanced GD in Fig. 1. It can be observed that a larger $\gamma$ leads to a slower convergence (i.e., a larger shift of the red curves to the right). Nevertheless, $\gamma$ needs to be chosen by considering the tradeoff between the training performance of KM and the computational cost as illustrated in Section V-A.

### C. Randomization

The solution to the dual problem in (14) (or equivalently (12)) produced by Algorithm 2, is not yet a feasible solution to the BQP in (4). A randomization procedure [50] can be employed to extract a feasible binary solution to (4) from the

**Algorithm 4** Dual Optimization for KM learning with Enhanced GD

**Input:** $\mathcal{U}_\mathcal{K}$, $\mathcal{I}_\mathcal{K}$, $\mathcal{K}$, $\{p_{u,i}\}_{(u,i)\in\mathcal{K}}$, and $I_{\text{BCD}}$. Initialize $\{\boldsymbol{\theta}_u^{(1)} \in \mathcal{P}\}_{u\in\mathcal{U}_\mathcal{K}}$.

**Output:** $\{\boldsymbol{\theta}_u^\star\}_{u\in\mathcal{U}_\mathcal{K}}$, $\{\boldsymbol{\psi}_i^\star\}_{i\in\mathcal{I}_\mathcal{K}}$.

1: **for** $\tau = 1, 2, \ldots, I_{\text{BCD}}$ **do**
2: Update $\{\boldsymbol{\psi}_i^{(\tau)}\}_{i\in\mathcal{I}_\mathcal{K}}$:
3: **for** $i \in \mathcal{I}_\mathcal{K}$ **do**
4: Obtain $\mathbf{u}_i^\star$ from Algorithm 2.
5: Recover $\boldsymbol{\psi}_i^{(\tau)}$ from Algorithm 3.
6: **end for**
7: Update $\{\boldsymbol{\theta}_u^{(\tau)}\}_{u\in\mathcal{U}_\mathcal{K}}$:
8: **for** $u \in \mathcal{U}_\mathcal{K}$ **do**
9: Obtain $\boldsymbol{\theta}_u^{(\tau)}$ from the FW algorithm [15].
10: **end for**
11: **end for**
12: **return** $\{\boldsymbol{\theta}_u^\star = \boldsymbol{\theta}_u^{(I_{\text{BCD}})}\}_{u\in\mathcal{U}_\mathcal{K}}$ and $\{\boldsymbol{\psi}_i^\star = \boldsymbol{\psi}_i^{(I_{\text{BCD}})}\}_{i\in\mathcal{I}_\mathcal{K}}$.

SDP solution $\mathbf{X}^\star$ of (11). One typical design of the randomization procedure for BQP is to generate feasible points from the Gaussian random samples via rounding [51]. The Gaussian randomization procedure provides a tight approximation with probability $1 - \exp(-\mathcal{O}(D))$, asymptotically in $D$ [52]. By leveraging the fact that the eigenvalues and corresponding eigenvectors of $\Pi_+(\mathbf{C}(\mathbf{u}))$ can be found by Steps 13 and 14 of Algorithm 2, we have

$$\mathbf{X}^\star = \gamma\Pi_+(\mathbf{C}(\mathbf{u}^\star)) = \gamma\mathbf{V}_+\boldsymbol{\Lambda}_+\mathbf{V}_+^T = \mathbf{L}\mathbf{L}^T,$$

where the first equality follows from (23) and (26), $\Pi_+(\mathbf{C}(\mathbf{u})) \triangleq \mathbf{V}_+\boldsymbol{\Lambda}_+\mathbf{V}_+^T$, and $\mathbf{L} = \mathbf{V}_+\sqrt{\gamma\boldsymbol{\Lambda}_+}$. A detailed randomization procedure is provided in Algorithm 3.

In Step 8 of Algorithm 3, the $D$-dimensional vector $\hat{\mathbf{x}}$ is first recovered from a $(D+1)$-dimensional vector $\tilde{\mathbf{x}}_{\ell^\star}$ by considering the structure of $\mathbf{X}^\star$ in (9), and then used to approximate the BQP solution based on (8). Also note that the randomization performance improves with $I_{\text{rand}}$. In practice, we only need to choose a sufficient but not excessive $I_{\text{rand}}$ (for instance, $50 \le I_{\text{rand}} \le 100$) achieving a good approximation for the BQP solution. Moreover, its overall computational complexity is much smaller than the conventional randomization algorithms [14], [50], [51] because our proposed Algorithm 3 does not require the computation of the Cholesky factorization.

### D. Overall KM Learning Algorithm

Incorporating Algorithm 2 and Algorithm 3, the overall KM learning framework is described in Algorithm 4.

Note that the index of BCD iterations $\tau$ that has been omitted is recovered here and $I_{\text{BCD}}$ denotes the total number of BCD iterations for KM learning. In Algorithm 4, the BCD method is adopted to refine $\{\boldsymbol{\psi}_i^{(\tau)}\}_{i\in\mathcal{I}_\mathcal{K}}$ and $\{\boldsymbol{\theta}_u^{(\tau)}\}_{u\in\mathcal{U}_\mathcal{K}}$ until it converges to a stationary point of (2). In fact, the proof of convergence (to stationary solution) for Algorithm 4 is exactly the same as that of Algorithm 1 in [14]. In practice, we can use $I_{\text{BCD}}$ to control the termination of Algorithm 4.

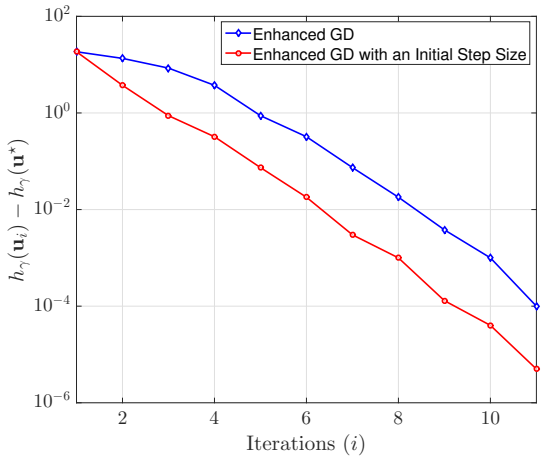



Fig. 2. Convergence rate comparison of the enhanced GD with EVD elimination in Algorithm 2 and that with an initial step size when $D = 4$.

## IV. Approximate EVD and Error Analysis

In this section, several techniques are discussed to further accelerate Algorithm 2.

### A. Initial Step Size

A good initial step size $t_0$ is crucial for the convergence speed of the enhanced GD. In Phase I-A of Algorithm 2, we have

$$\boldsymbol{\lambda}(\mathbf{C}(\mathbf{u}_{i+1})) = \boldsymbol{\lambda}(-\mathbf{A}) - \mathbf{u}_{i+1} = \boldsymbol{\lambda}(-\mathbf{A}) - \mathbf{u}_i + t_i\mathbf{1}.$$

If $\lambda_{\max}(\mathbf{C}(\mathbf{u}_{i+1})) > 0$, the following holds $t_i > u_i - \lambda_{\max}(-\mathbf{A})$ where $u_i \triangleq u_i(1) = \cdots = u_i(D+1)$. Therefore, in the first iteration of Phase I-A, we can set an appropriate step size $t_0 > u_0 - \lambda_{\max}(-\mathbf{A})$ so that $\mathbf{C}(\mathbf{u}_1) = -\mathbf{A} - \text{diag}(\mathbf{u}_1)$ has at least one positive eigenvalue, where $\mathbf{u}_1 = \mathbf{u}_0 - t_0\mathbf{1}$. With the above modification of Algorithm 2, we can reduce the execution time spent in Phase I-A, and thus, the total number of iterations required by the enhanced GD can be reduced as shown in Fig. 2. Moreover, the choice of $\mathbf{u}_0$ does not affect the overall performance in terms of the computational cost.

### B. Approximate EVD

Compared to the original GD in Algorithm 1, the enhanced GD in Algorithm 2 has reduced the costly EVD substantially. Nevertheless, the EVD is still necessary in Algorithm 2 when the algorithm enters into *Phase II-B*. In order to further accelerate the algorithm, we employ and modify the Lanczos method to numerically compute the approximate EVD of $\mathbf{C}(\mathbf{u}_i)$ in Algorithm 2.

The Lanczos algorithm [53] is a special case of the Arnoldi method [54] when the matrix is symmetric. In principle, it is based on an orthogonal projection of $\mathbf{C}(\mathbf{u}_i)$ onto the Krylov subspace $\mathcal{K}_m \triangleq \text{span}\{\mathbf{p}, \mathbf{C}(\mathbf{u}_i)\mathbf{p}, \cdots, \mathbf{C}(\mathbf{u}_i)^{m-1}\mathbf{p}\}$ where $m$ denotes the dimension of Krylov subspace. An algorithmic description of a modified Lanczos method is presented in Algorithm 5.

**Algorithm 5** Modified Lanczos Algorithm

**Input:** $\mathbf{C}(\mathbf{u}_i)$, $D$, and $\delta$ (*threshold value*). Choose an initial unit-norm vector $\mathbf{p}_1 \in \mathbb{R}^{D+1}$. Set $\beta_1 = 0$, $\mathbf{p}_0 = \mathbf{0}$, and $\mathbf{H}_m = \mathbf{0}_{(D+1)\times(D+1)}$ ($\mathbf{0}_{(D+1)\times(D+1)}$ denotes the all-zero matrix of dimension $(D+1)\times(D+1)$).

**Output:** $\mathbf{P}_m = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_m]$ and $\mathbf{H}_m$.

1: **for** $j = 1, 2, \ldots, D+1$ **do**
2: $\quad \mathbf{w}_j = \mathbf{C}(\mathbf{u}_i)\mathbf{p}_j - \beta_j \mathbf{p}_{j-1}$.
3: $\quad \alpha_j = \langle \mathbf{w}_j, \mathbf{p}_j \rangle$ and $H_m(j,j) = \alpha_j$ ($\alpha_j$ forms the main diagonal of $\mathbf{H}_m$).
4: $\quad \mathbf{w}_j = \mathbf{w}_j - \alpha_j \mathbf{p}_j$ and $\beta_{j+1} = \|\mathbf{w}_j\|_2$.
5: $\quad$ **if** $\beta_{j+1} \leq \delta$ **then** terminate and **return**
6: $\quad\quad m = j$ and $\mathbf{H}_m = \mathbf{H}_m(1:m, 1:m)$.
7: $\quad$ **else**
8: $\quad\quad H_m(j, j+1) = H_m(j+1, j) = \beta_{j+1}$ ($\beta_{j+1}$ forms the super- and sub-diagonal of $\mathbf{H}_m$).
9: $\quad\quad \mathbf{p}_{j+1} = \mathbf{w}_j / \beta_{j+1}$.
10: $\quad$ **end if**
11: **end for**

Different from the Arnoldi method, the matrix $\mathbf{H}_m \in \mathbb{R}^{m\times m}$ constructed by Algorithm 5 is tridiagonal and symmetric, i.e., the entries of $\mathbf{H}_m$ in Algorithm 5 satisfy that $H_m(i,j) = 0$, $1 \leq i < j-1$, and $H_m(j, j+1) = H_m(j+1, j)$, $j = 1, 2, \ldots, m$. Also, Algorithm 5 iteratively builds an orthonormal basis, i.e., $\mathbf{P}_m \in \mathbb{R}^{(D+1)\times m}$, for $\mathcal{K}_m$ such that $\mathbf{P}_m^T \mathbf{C}(\mathbf{u}_i)\mathbf{P}_m = \mathbf{H}_m$ and $\mathbf{P}_m^T \mathbf{P}_m = \mathbf{I}_m$, where $m \leq D+1$. Let $(\vartheta_i, \mathbf{q}_i)$, $i = 1, \ldots, m$, be the eigenpairs of $\mathbf{H}_m$. Then, the eigenvalues/eigenvectors of $\mathbf{C}(\mathbf{u}_i)$ can be approximated by the Ritz pairs $(\vartheta_i, \mathbf{P}_m \mathbf{q}_i)$, i.e.,

$$\hat{\lambda}_i = \vartheta_i, \ \hat{\mathbf{v}}_i = \mathbf{P}_m \mathbf{q}_i, \ i = 1, \ldots, m. \tag{15}$$

With the increase of the dimension of Krylov subspace $m$, the approximation performance improves at the price of additional computations. Thus, in practice, we adopt the value of $m$ balancing the tradeoff between the accuracy of approximation and the computational complexity.

### C. Analysis of Approximation Error and Thresholding Scheme

In this subsection, we analyze the approximation error of the approximate EVD and propose a thresholding scheme for selecting an appropriate $m$ in Algorithm 5. The main results are provided in the following lemmas.

*Lemma 2:* Let $(\vartheta_i, \mathbf{q}_i)$ be any eigenpair of $\mathbf{H}_m$ and $(\hat{\lambda}_i = \vartheta_i, \hat{\mathbf{v}}_i = \mathbf{P}_m \mathbf{q}_i)$ in (15) is an approximated eigenpair (Ritz pair) of $\mathbf{C}(\mathbf{u}_i)$ in Algorithm 5. Then the following holds:

i) The residual error $r_e(\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i, \hat{\lambda}_i\hat{\mathbf{v}}_i) \triangleq \|\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i - \hat{\lambda}_i\hat{\mathbf{v}}_i\|_2$ is upper bounded by

$$r_e(\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i, \hat{\lambda}_i\hat{\mathbf{v}}_i) \leq \beta_{m+1}. \tag{16}$$

ii) The maximum approximation error of eigenvalues of $\mathbf{C}(\mathbf{u}_i)$ is bounded by

$$\max_i |\lambda_i - \hat{\lambda}_i| \leq \beta_{m+1}, \ i \in \{1, \ldots, m\}, \tag{17}$$

where $\lambda_i$ is the associated true eigenvalue of $\mathbf{C}(\mathbf{u}_i)$.

TABLE I
TIME CONSUMPTION (IN SECONDS) COMPARISON OF SOLVING THE BQP UNDER DIFFERENT $\gamma$ FOR (D1) WHEN $D = 16$

| Algorithm \ $\gamma$ | $10^1$ | $10^2$ | $10^3$ |
|---|---|---|---|
| Original GD | $6.51 \times 10^{-1}$ | 1.24 | 2.79 |
| Enhanced GD | $6.79 \times 10^{-2}$ | $1.53 \times 10^{-1}$ | $2.62 \times 10^{-1}$ |

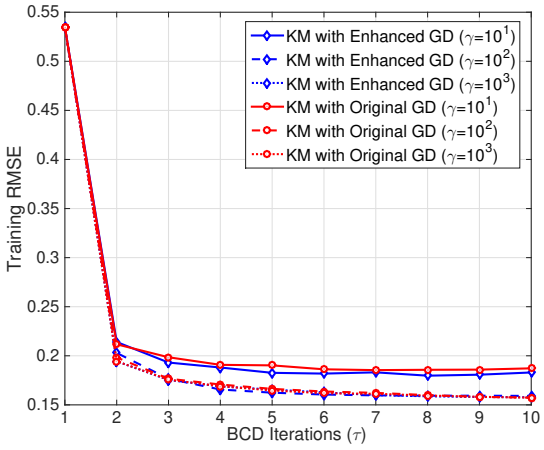



Fig. 3. The effect of $\gamma$ in (11) on the KM training performance when $D = 16$.

iii) The minimum approximation error of eigenvalues of $\mathbf{C}(\mathbf{u}_i)$ is bounded by

$$\min_i |\lambda_i - \hat{\lambda}_i| \leq \beta_{m+1}|q_i(m)|, \ i \in \{1, \ldots, m\}. \tag{18}$$

*Proof:* See Appendix B.

Lemma 2 indicates that the error bounds of the approximate eigenvalues of $\mathbf{C}(\mathbf{u}_i)$ by using the approximate EVD in Algorithm 5 largely depends on $\beta_{m+1}$. Indeed, the upper bounds in (17) and (18) are quite tight as will be seen in Section V. Inspired by Lemma 2, finding an upper bound of $\beta_{m+1}$ that only depends on the trace of $\mathbf{C}(\mathbf{u}_i)$ is of interest.

*Lemma 3:* $\beta_{m+1}$ in Algorithm 5 is upper bounded by

$$\beta_{m+1} \leq 2m\big((\sigma_{\text{max,UB}} - \sigma_{\text{max,LB}}) + \hat{\sigma}_{\text{max,Minkowski}}\big), \tag{19}$$

where $\sigma_{\text{max,UB}} = \frac{\text{trace}(\mathbf{C}(\mathbf{u}_i))}{D+1} + \big(\big(\frac{\text{trace}(\mathbf{C}(\mathbf{u}_i)^2)}{D+1} - \frac{\text{trace}^2(\mathbf{C}(\mathbf{u}_i))}{(D+1)^2}\big)\cdot D\big)^{\frac{1}{2}}$ and $\sigma_{\text{max,LB}} = \frac{\text{trace}(\mathbf{C}(\mathbf{u}_i))}{D+1} + \big(\big(\frac{\text{trace}(\mathbf{C}(\mathbf{u}_i)^2)}{D+1} - \frac{\text{trace}^2(\mathbf{C}(\mathbf{u}_i))}{(D+1)^2}\big)/D\big)^{\frac{1}{2}}$ are the upper and lower bounds on the largest singular value of $\mathbf{C}(\mathbf{u}_i)$, respectively[2], and $\hat{\sigma}_{\text{max,Minkowski}} \triangleq \frac{1}{D+1} \sum_{\ell=1}^{D+1}\sum_{j=1}^{D+1} |C(\ell, j)|$ is a normalized Minkowski $\ell_1$-norm of $\mathbf{C}(\mathbf{u}_i)$ ($C(\ell, j)$ denotes the $(\ell, j)$th entry of $\mathbf{C}(\mathbf{u}_i)$ for simplicity).

*Proof:* See Appendix C.

Lemma 3 gives us an upper bound of $\beta_{m+1}$ that does not require a computation of EVD and can be readily employed as a stopping condition in Step 5 of Algorithm 5. In particular, it proposes to use the normalized Minkowski $\ell_1$-norm $\hat{\sigma}_{\text{max,Minkowski}}$. Notice that we introduced $\hat{\sigma}_{\text{max,Minkowski}}$ in Appendix C (Proof of Lemma 3) to further upper bound

[2]For the symmetric matrix $\mathbf{C}(\mathbf{u}_i) \in \mathbb{S}^{(D+1)\times(D+1)}$, its largest singular value $\sigma_{\max}(\mathbf{C}(\mathbf{u}_i))$ is the same as the absolute value of its eigenvalue with the largest modulus.

TABLE II
THE DURATION OF FOUR SUB-PHASES IN ALGORITHM 2 BASED ON (D1) WHEN $D = 8$

| Phase | I-A | I-B | II-A | II-B |
|---|---|---|---|---|
| Duration (%) | 37 | 9 | 8 | 46 |

$\sigma_{\max}(\mathbf{C}(\mathbf{u}_i))$ in (31), which gives a good approximation of $\sigma_{\max}(\mathbf{C}(\mathbf{u}_i))$. It is important to note that the upper bound in (19) depends only on the traces and the absolute value of entries of $\mathbf{C}(\mathbf{u}_i)$, whose computational cost is extremely low compared to that of EVD. Moreover, in Appendix C, a useful property of $\hat{\sigma}_{\text{max,Minkowski}}$ is leveraged, namely, $\sigma_{\text{max,LB}} \leq \hat{\sigma}_{\text{max,Minkowski}} \leq \sigma_{\text{max,UB}}$ [55].

Lemma 3 motivates us to adjust the number of iterations $m$ of Algorithm 5 by proposing a low-complexity yet reasonable threshold, which exploits the structure of the upper bound on $\beta_{m+1}$ in (19). Therefore, we propose to use a threshold value provided as

$$\delta = \frac{1}{aD \ln D}((\sigma_{\text{max,UB}} - \sigma_{\text{max,LB}}) + \hat{\sigma}_{\text{max,Minkowski}}), \qquad (20)$$

where $a > 0$ is a controlled parameter. Unlike the prior works which choose $m$ in a greedy manner, this thresholding scheme determines $m$ by controlling the approximation error below $\delta$, leading to a balance between the accuracy of approximation and the computational complexity. This will be further investigated in Section V.

## V. NUMERICAL RESULTS

We now present the simulation results demonstrating the superiority of the proposed methods compared with the conventional KM (e.g., the KM with SDRwR [14] and DMO [27]) and existing data representation techniques (e.g., NNM [11], MF [7], and SVD++ [9]) in terms of the computational cost, training and prediction performance, and interpretability. Three datasets for experiments are mainly considered, including (D1) an artificially generated toy dataset (for training only): $\mathcal{K} = \{(u,i)|u \in \{1,\ldots,20\}, i \in \{1,\ldots,40\}\}$ $(U = 20, I = 40)$ and $\{p_{u,i}\}_{(u,i)\in\mathcal{K}}$ are independent and uniformly distributed on the unit interval $[0,1]$, (D2) the MovieLens 100K dataset[3] (ML100K) with $U = 943$ users and $I = 1682$ movie items, and (D3) the MovieLens 1 million dataset (ML1M) with $U = 6040$ users and $I = 3900$ movie items. For latter two MovieLens datasets, we divide each one of both into 80% for training and the remaining 20% for testing. The empirical probabilities of the training set, i.e., $\{p_{u,i}\}$, are obtained by $p_{u,i} = r_{u,i}/r_{\max}$, $(u,i) \in \mathcal{K}$, as in (6).

### A. Computational Cost and Training Performance

We evaluate the computational cost and training performance of the proposed KM learning with the enhanced GD (i.e., Algorithm 4). Throughout the paper, the computational cost is calculated by averaging the total running time in seconds (measured by "cputime" in MATLAB running on a PC with an Intel Core i7-7700 3.6 GHz CPU and 16 GB RAM) over the number of BCD iterations. We adopt the training root-mean-square error (RMSE), which is defined as $E_{\text{train}} \triangleq \sqrt{\frac{1}{|\mathcal{K}|}\sum_{(u,i)\in\mathcal{K}} |p_{u,i} - \boldsymbol{\theta}_u^{\star T}\boldsymbol{\psi}_i^{\star}|^2}$, as a metric.

We first investigate the effect of the regularization parameter $\gamma$ in (11) on the KM learning performance. In Fig. 3, the training RMSE is evaluated under different parameter settings of $\gamma$ for the two proposed GD-based methods based on the artificial dataset (D1). It can be seen from Fig. 3 that the

[3]https://grouplens.org/datasets/movielens/100k/

TABLE III
TIME CONSUMPTION (IN SECONDS) COMPARISON OF THE KM LEARNING ($D = 8$)

| Dataset | Algorithm \ Subproblem | 1. LCQP | 2. BQP |
|---|---|---|---|
| (D1) | SDRwR | $1.51 \times 10^{-1}$ | 7.36 |
| | DMO | $1.43 \times 10^{-1}$ | 1.85 |
| | Original GD | $1.41 \times 10^{-1}$ | $1.12 \times 10^{-1}$ |
| | Enhanced GD | $1.37 \times 10^{-1}$ | $8.60 \times 10^{-2}$ |
| (D2) | SDRwR | 7.05 | $3.08 \times 10^{+2}$ |
| | DMO | 7.11 | $7.80 \times 10^{+1}$ |
| | Original GD | 7.11 | $2.24 \times 10^{+1}$ |
| | Enhanced GD | 7.10 | $2.19 \times 10^{+1}$ |

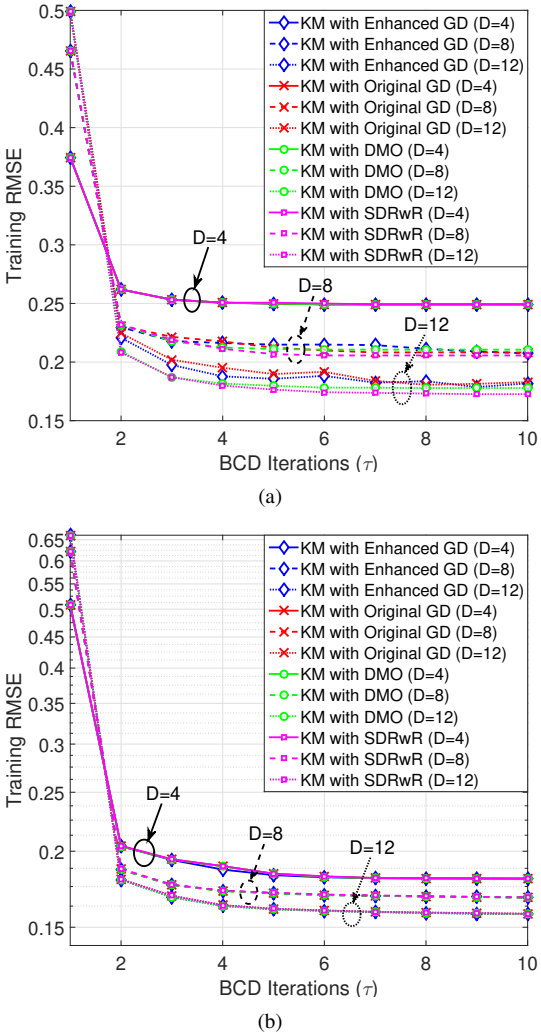



Fig. 4. Training RMSE vs. BCD iterations ($\tau$ in Algorithm 4): (a) the artificial dataset (D1), (b) ML100K (D2).

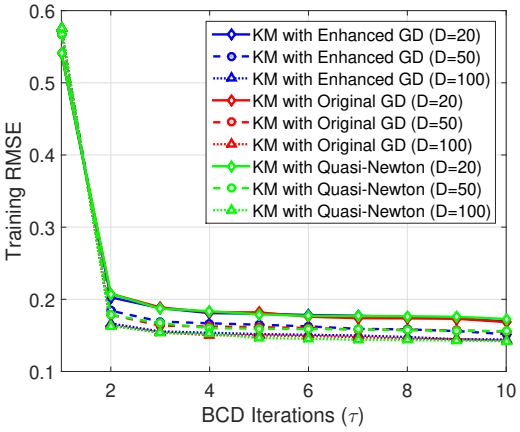



Fig. 5. KM training performance comparison between the quasi-Newton method and the proposed first-order methods based on (D1).

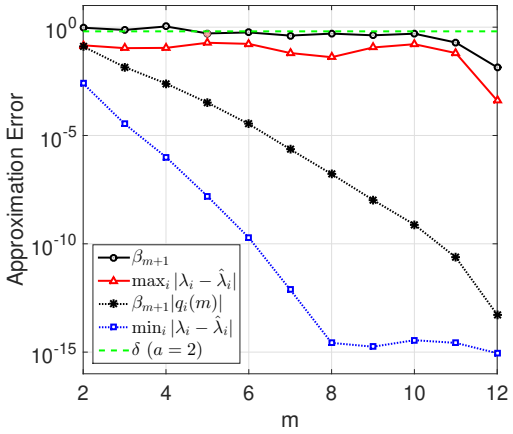



Fig. 6. Approximation error and upper bounds of the approximate EVD with thresholding when $D = 12$.

larger the $\gamma$ value, the better the training performance of KM is, but this is achieved with an increased computational cost as shown in Table I. Moreover, it is indistinguishable in terms of the training error when $\gamma$ is increased from $10^2$ to $10^3$. It is thus a tradeoff between accuracy and complexity in the choice of $\gamma$. We choose $\gamma = 100$ and fix it for subsequent numerical experiments.

In particular, we take the case when $D = 8$ and (D1) is considered as an example to show the duration of each phase in the enhanced GD (Algorithm 2). In Table II, the duration of each phase is measured by using the ratio between the number of iterations spent by the phase and the total number of iterations required by Algorithm 2 and averaged by taking $10^4$ realizations of (D1). Observed from Table II, $54\%$ of iterations (including Phase I-A, I-B, and II-A) of Algorithm 2 do not require the computation of EVD, which results in a significant reduction of the computational cost compared to the original GD as will be shown in Table IV (i.e., one or two orders of magnitude improvement in terms of the time overhead).

Table III demonstrates the computational complexity of the overall KM learning (LCQP + BQP) on the datasets (D1) and (D2), respectively, when $D = 8$. In particular, the FW algorithm is fixed for solving the LCQP while different algorithms are applied to solving the BQP. It reveals that the computational cost of solving the LCQP in (3) via the FW algorithm is negligible compared to that of the BQP in (4), especially, via DMO or SDRwR. It can be seen that enhanced GD results in improved time complexity while it is clear that SDRwR and DMO are not scalable even for (D2), ML100K. The time complexity for solving the BQP with the varying $D$ can be found in Table IV. Seen from Table IV, the DMO shows benefits when $D$ is small, but its computational cost blows up as $D$ increases since the DMO is based on the branch-and-bound, which is very close to the exhaustive search in the worst case. For our proposed methods, the improvement on the computational cost of the enhanced GD compared to the original GD is marginal when $D$ is small. However, as $D$ grows, the benefit of the enhanced GD becomes significant. Fig. 4 displays the training RMSE comparison, which demonstrates that the proposed enhanced GD achieves similar, good training performance to the other approaches while reducing the computational complexity by several orders of magnitude as shown in Table IV. Furthermore, we compare the performance between the quasi-Newton method [56] and our proposed first-order methods based on (D1). It can be seen from Fig. 5 that the KM with quasi-Newton method achieves similar training performance as the gradient-based methods, while it consumes more computation time even compared with the original GD as shown in Table IV. This is due to the fact that, in addition to calculating the gradient $\nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i)$ as in Algorithm 1, the quasi-Newton methods need to compute the approximated inverse of the Hessian matrix $\mathbf{H} \approx (\nabla^2_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i))^{-1}$ to obtain the descent direction $(\Delta \mathbf{u}_i = -\mathbf{H} \nabla_{\mathbf{u}_i} h_\gamma(\mathbf{u}_i))$.

Next, we evaluate the performance of the enhanced GD with approximate EVD and thresholding technique presented in Section IV-C and compare it with Algorithm 2 (i.e., the enhanced GD with exact EVD) in Table IV and Figs. 6–7. The computational cost of the enhanced GD with approximate EVD and thresholding can be reduced significantly compared to that of exact EVD, especially when $D$ is large. It is worth noting that the time overhead of the enhanced GD with approximate EVD and thresholding does not increases substantially compared to the original GD (even the enhanced GD with exact EVD) as D grows from $D = 20$ to $D = 100$ for (D2). The same trend is observed for (D3). In Fig. 6, we show the upper bounds of the approximation error of the approximate EVD with respect to $m$. It demonstrates that $\beta_{m+1}$ and $\beta_{m+1}|q_i(m)|$ in Lemma 2 provide tight upper bounds of $\max_i |\lambda_i - \hat{\lambda}_i|$ and $\min_i |\lambda_i - \hat{\lambda}_i|$, respectively. It also shows that the threshold value $\delta$ in (20), based on Lemma 3, guides a good choice of $m$. For instance of Fig. 6, the approximate EVD terminates when $m = 5$ (the point $\beta_{m+1} \leq \delta$) with the approximation error of the dominant eigenvalue far below $10^{-5}$. Moreover, as depicted in Fig. 7, the training performance of the approximate EVD with thresholding is very close to that of the exact EVD.

TABLE IV
TIME CONSUMPTION (IN SECONDS) COMPARISON OF SOLVING THE BQP

| Dataset | Algorithm \ $D$ | 4 | 8 | 12 | 20 | 50 | 100 |
|---|---|---|---|---|---|---|---|
| (D1) | SDRwR[a] | 7.11 | 7.36 | 7.62 | - | - | - |
| | DMO[a] | $1.21 \times 10^{-1}$ | 1.85 | $1.00 \times 10^{+3}$ | - | - | - |
| | Quasi-Newton[b] | - | - | - | 1.91 | $1.76 \times 10^{+1}$ | $8.04 \times 10^{+1}$ |
| | Original GD | $8.16 \times 10^{-2}$ | $1.12 \times 10^{-1}$ | $1.32 \times 10^{-1}$ | 1.47 | $1.48 \times 10^{+1}$ | $6.76 \times 10^{+1}$ |
| | Enhanced GD (with exact EVD) | $8.10 \times 10^{-2}$ | $8.60 \times 10^{-2}$ | $1.07 \times 10^{-1}$ | $1.63 \times 10^{-1}$ | $5.34 \times 10^{-1}$ | 2.56 |
| | Enhanced GD (with approximate EVD & thresholding)[c] | - | - | - | $1.55 \times 10^{-1}$ | $1.98 \times 10^{-1}$ | $2.87 \times 10^{-1}$ |
| (D2) | SDRwR[a] | $2.99 \times 10^{+2}$ | $3.08 \times 10^{+2}$ | $3.18 \times 10^{+2}$ | - | - | - |
| | DMO[a] | $2.02 \times 10^{+1}$ | $7.80 \times 10^{+1}$ | $7.57 \times 10^{+3}$ | - | - | - |
| | Original GD | $2.14 \times 10^{+1}$ | $2.24 \times 10^{+1}$ | $2.41 \times 10^{+1}$ | $1.02 \times 10^{+2}$ | $5.53 \times 10^{+2}$ | $5.10 \times 10^{+3}$ |
| | Enhanced GD (with exact EVD) | $2.10 \times 10^{+1}$ | $2.19 \times 10^{+1}$ | $2.30 \times 10^{+1}$ | $2.67 \times 10^{+1}$ | $5.49 \times 10^{+1}$ | $2.59 \times 10^{+2}$ |
| | Enhanced GD (with approximate EVD & thresholding)[c] | - | - | - | $2.58 \times 10^{+1}$ | $3.20 \times 10^{+1}$ | $3.87 \times 10^{+1}$ |
| (D3)[d] | Enhanced GD (with exact EVD) | - | $5.88 \times 10^{+2}$ | $6.01 \times 10^{+2}$ | $8.14 \times 10^{+2}$ | - | - |
| | Enhanced GD (with approximate EVD & thresholding) | - | $5.84 \times 10^{+2}$ | $5.92 \times 10^{+2}$ | $6.89 \times 10^{+2}$ | - | - |

[a] The missed entries ('-') are due to the extraordinary high computational cost of the SDRwR and DMO when $D$ is large.
[b] For the quasi-Newton method, we utilize (D1) and focus on the cases when $D$ is large.
[c] The missed entries exist because we focus on evaluating the performance of the enhanced GD with approximate EVD & thresholding when $D$ is large.
[d] For the large-scale (D3), our focus has been switched to the proposed scalable enhanced GD and the simulations are done only for $D = 8, 12, 20$.

TABLE V
PREDICTION PERFORMANCE EVALUATION (NRMSE COMPARISON) BASED ON (D2) AND (D3)

| Dataset | Algorithm \ $D$ | 4 | 8 | 16 | 24 |
|---|---|---|---|---|---|
| (D2) | KM-Enhanced GD (Algorithm 4) | 0.1978 ($\lambda_u = 10, \mu_i = 0$) | 0.1963 ($\lambda_u = 30, \mu_i = 0$) | 0.1946 ($\lambda_u = 40, \mu_i = 0$) | 0.1891 ($\lambda_u = 60, \mu_i = 0$) |
| | KM-Enhanced GD with approximate EVD & thresholding | 0.2045 ($\lambda_u = 10, \mu_i = 0$) | 0.1999 ($\lambda_u = 30, \mu_i = 0$) | 0.1961 ($\lambda_u = 60, \mu_i = 0$) | 0.1914 ($\lambda_u = 60, \mu_i = 0$) |
| | NNM | 0.1944 ($\lambda_u = 10, \mu_i = 0$) | 0.2255 ($\lambda_u = 20, \mu_i = 0$) | 0.2057 ($\lambda_u = 40, \mu_i = 0$) | 0.2118 ($\lambda_u = 50, \mu_i = 10$) |
| | MF[d,e] | 0.2292 | 0.2287 ($k = 10$) | - | 0.2269 ($k = 40$) |
| | SVD++[d] | 0.2284 | 0.2277 ($k = 10$) | 0.2270 ($k = 20$) | 0.2266 ($k = 50$) |
| (D3) | KM-Enhanced GD (Algorithm 4) | 0.1812 ($\lambda_u = 0, \mu_i = 2$) | 0.1768 ($\lambda_u = 10, \mu_i = 1.5$) | 0.1716 ($\lambda_u = 20, \mu_i = 1.5$) | 0.1629 ($\lambda_u = 30, \mu_i = 2$) |
| | KM-Enhanced GD with approximate EVD & thresholding | 0.2478 ($\lambda_u = 10, \mu_i = 2$) | 0.1812 ($\lambda_u = 10, \mu_i = 0.5$) | 0.1765 ($\lambda_u = 10, \mu_i = 4$) | 0.1684 ($\lambda_u = 10, \mu_i = 2.5$) |
| | NNM | 0.1798 ($\lambda_u = 0, \mu_i = 1.5$) | 0.1776 ($\lambda_u = 10, \mu_i = 2.5$) | 0.1765 ($\lambda_u = 10, \mu_i = 1.5$) | 0.1758 ($\lambda_u = 10, \mu_i = 3$) |
| | MF[d,e] | - | 0.2143 ($k = 10$) | - | - |
| | SVD++[d,e] | - | 0.2130 ($k = 10$) | 0.2128 ($k = 20$) | - |

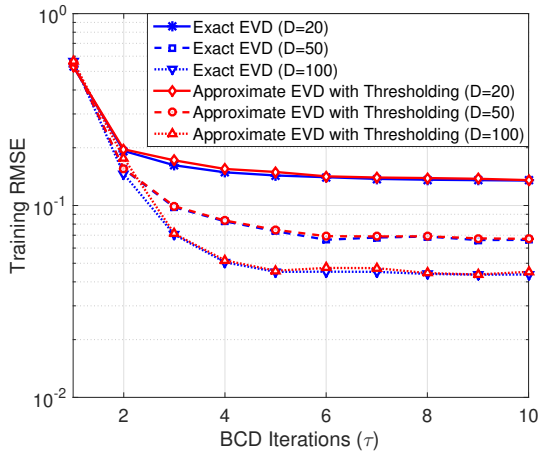
[d] The results of MF and SVD++ are taken from the following repository: *http://www.mymedialite.net/examples/datasets.html*.
[e] The missed entries ('-') are due to the unavailability of the corresponding RMSE result of the repository.

### *B. Prediction Performance*

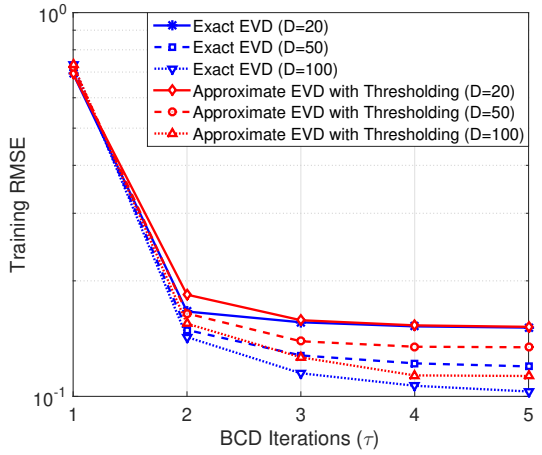
We assess the prediction performance of the proposed methods against the existing methods, including NNM [11], MF [7], and SVD++ [9], based on the ML100K (D2) and the ML1M (D3) datasets. We adopt the normalized RMSE (NRMSE) as a metric, which is given by

$$E_{\text{test, NRMSE}} \triangleq \begin{cases} \sqrt{\frac{1}{|\mathcal{T}|}\sum_{(u,i)\in\mathcal{T}}(p_{u,i} - \boldsymbol{\theta}_u^{\star T}\boldsymbol{\psi}_i^{\star})^2}, \\ \qquad\qquad \text{for KM and NNM} \\ \frac{1}{r_{\max}-r_{\min}}\sqrt{\frac{1}{|\mathcal{T}|}\sum_{(u,i)\in\mathcal{T}}(r_{u,i} - \hat{r}_{u,i})^2}, \\ \qquad\qquad \text{for MF and SVD++} \end{cases},$$

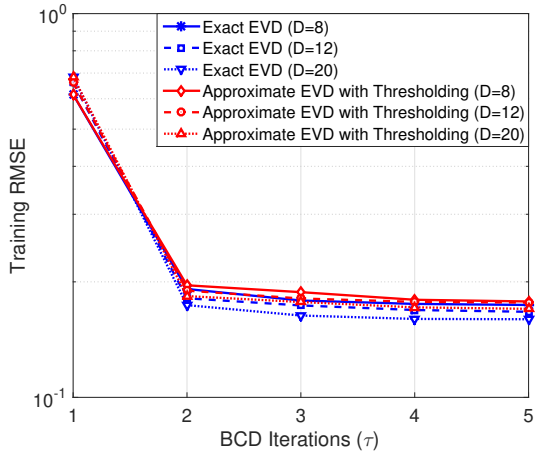
where $r_{\max} - r_{\min} = 5 - 1 = 4$ and $\hat{r}_{u,i}$ is the predicted rating score via MF or SVD++. The above normalization, which scales by $1/(r_{\max} - r_{\min})$ ensuring that the predicted values of all the different methods are contained in $[0, 1]$, is widely used in ML [14]. The NRMSE results of prediction on (D2) and (D3) are provided in Table V. In Table V, $\lambda_u$ and $\mu_i$ are two hyperparameters to mitigate overfitting by using cross validation [14]. Specifically, the value of $(\lambda_u, \mu_i)$ in each entry indicates the best parameter pair associated with corresponding method and $D$. To ensure a reasonable comparison, the size of factorization for MF and SVD++, i.e., $k$, is chosen to be as close as possible to $D$. It reveals that the KM with enhanced GD shows significantly better prediction performance compared to the benchmarks and the predication error gap between this method and the benchmarks improves with increasing $D$. This is attributed to the advantageous nature of KM that being an accurate model in a mathematical sense and rooted in probability theory, while other benchmarks are based on intuition.




(a)




(b)




(c)

Fig. 7. Training RMSE vs. BCD iterations of the enhanced GD with exact EVD and approximate EVD with thresholding: (a) the artificial dataset (D1), (b) ML100K (D2), (c) ML1M (D3).

### C. Interpretability via Logical Relation Mining

In Section II-B3, we have briefly introduced the interpretability of KM. In order to exploit the logical relations between random variables $X_{u,i}$ and $X_{u,j}$ ($(u,i) \in \mathcal{K}$ and $(u,j) \in \mathcal{K}$) based on the optimized KM parameters $\boldsymbol{\psi}_i^\star$ and $\boldsymbol{\psi}_j^\star$, $\forall i,j \in \mathcal{I}_\mathcal{K}$, an indicator matrix of the logical relations, $\mathbf{N} \in \mathbb{B}^{|\mathcal{I}_\mathcal{K}|\times|\mathcal{I}_\mathcal{K}|}$, also known as an adjacency matrix [14], can be built as

$$N(i,j) = \begin{cases} 1, & \text{if } \mathrm{supp}(\boldsymbol{\psi}_j^\star) \subseteq \mathrm{supp}(\boldsymbol{\psi}_i^\star) \\ 0, & \text{otherwise} \end{cases},$$

where the nonzero entry $N(i,j) = 1$ shows that $X_{u,i}$ and $X_{u,j}$ are coupled and mutually influencial. Constructing $\mathbf{N}$ allows us to further evaluate how much $X_{u,i}$ influences or is influenced by $X_{u,j}$, $\forall j \in \mathcal{I}_\mathcal{K}$, via introducing the normalized influence score [14] as

$$\varsigma_i = \frac{1}{|\mathcal{I}_\mathcal{K}|} \sum_{j \in \mathcal{I}_\mathcal{K}} N(i,j), \ \forall i \in \mathcal{I}_\mathcal{K}. \tag{21}$$

Stated differently, $\varsigma_i$ counts the (normalized) number of relations that $X_{u,i}$ is logically connected to. In particular, $\varsigma_i = 1$ denotes a maximally supported random variable, i.e., $\boldsymbol{\psi}_i^\star = \mathbf{1}$ and $\mathrm{supp}(\boldsymbol{\psi}_j^\star) \subseteq \mathrm{supp}(\boldsymbol{\psi}_i^\star)$ holds $\forall j \in \mathcal{I}_\mathcal{K}$.

We display the normalized influence score, i.e., $\varsigma_i$ in (21), mined by two KM learning algorithms including the proposed KM with enhanced GD (Algorithm 4) and previous KM with SDRwR [14, Algorithm 1], for the ML100K dataset (D2). In Fig. 8, we find that the results of logical relation mining are quite similar for the above two algorithms. However, the proposed KM with enhanced GD offers an order of magnitude reduction in the computational complexity, compared to the KM with SDRwR [57]. Furthermore, we confirm the efficacy of logical relation mining of Algorithm 4 by identifying the set of items corresponding to $\varsigma_i = 1$, as in Table VI. Theoretically, if a user likes one of these items, then the user likes all other items in the training set. In Table VI, the first column shows the item index with $\varsigma_i = 1$, while the second column lists the user index in the training set who have rated the corresponding item. The total number of items rated by the user is shown in the third column. We calculate the accuracy of logical relation mining by setting a threshold to the empirical probability of the training set, i.e., $p_{u,i} \geq 50\%$, indicating that the user $u$ likes the item $i$. For instance, the item of index 1201 has been rated by the user of index 90 and this user has rated 164 items in total. By checking the empirical probabilities of the ML100K dataset, we find that there are 153 items with $p_{u,i} \geq 50\%$. As observed from Table VI, the accuracy of logical relation mining by using the KM with enhanced GD is above $80\%$.

TABLE VI
ACCURACY OF LOGICAL RELATIONS MINING FOR THE ML100K DATASET (D2).

| item index with $\varsigma_i = 1$ | user index | # of items rated | accuracy |
|---|---|---|---|
| 1201 | 90 | 164 | 93.29% |
| 1293 | 146 | 19 | 84.21% |
| | 489 | 109 | 80.73% |
| | 519 | 49 | 81.63% |
| 1467 | 244 | 117 | 86.32% |
| | 886 | 240 | 80.00% |
| 1599 | 437 | 238 | 82.77% |

## VI. CONCLUSION

In this paper, we presented a novel KM learning algorithm by using an enhanced GD approach based on dual

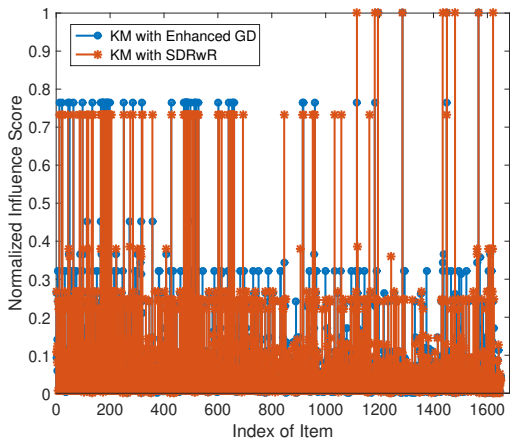



Fig. 8. Normalized influence score for two different algorithms on the ML100K dataset (D2) when $D = 8$.

optimization. To be specific, the BQP subproblem of KM learning was reformulated as a regularized dual optimization problem of strong convexity, which can be solved by GD. Considering the demand of scalability and the drawback of traditional GD due to a high reliance on the computation of EVD, we proposed an efficient enhanced GD with EVD elimination. Furthermore, a numerical approximate EVD was adopted to extract the spectra of symmetric matrices with low computational complexity. Inspired by the approximation error analysis, we explored the tractable bound which depends only on the traces and the normalized Minkowski $\ell_1$-norm, and then proposed a thresholding scheme for the approximate EVD. The proposed methods were applied to different datasets and numerical results demonstrated their superiority compared to other benchmarks in terms of computational cost, training/prediction performance, and interpretability.

## ACKNOWLEDGEMENT

We are deeply indebted to the reviewers and editor, whose consistent comments greatly improved the manuscript.

## APPENDIX A
## PROOF OF LEMMA 1

**Proof** The Lagrangian of the primal problem in (11) is given by

$$\mathcal{L}(\mathbf{X},\mathbf{u},\mathbf{D}) = \langle \mathbf{X},\mathbf{A}\rangle + \frac{1}{2\gamma}\|\mathbf{X}\|_F^2 - \langle \mathbf{X},\mathbf{D}\rangle + \sum_{i=1}^{D+1} u_i(\langle \mathbf{X},\mathbf{B}_i\rangle - 1), \quad (22)$$

where $\mathbf{u} \in \mathbb{R}^{D+1}$ and $\mathbf{D} \succeq \mathbf{0}$ are Lagrangian multipliers. Since the problems in (11) and (22) are feasible, strong duality holds and $\nabla_{\mathbf{X}}\mathcal{L}(\mathbf{X}^\star,\mathbf{u}^\star,\mathbf{D}^\star) = 0$, where $\mathbf{X}^\star$, $\mathbf{u}^\star$, and $\mathbf{D}^\star$ are optimal solutions to (22). Then we have

$$\mathbf{X}^\star = \gamma\Big(\mathbf{D}^\star - \mathbf{A} - \sum_{i=1}^{D+1} u_i^\star \mathbf{B}_i\Big) = \gamma(\mathbf{D}^\star + \mathbf{C}(\mathbf{u}^\star)), \quad (23)$$

where $\mathbf{C}(\mathbf{u}^\star) = -\mathbf{A} - \sum_{i=1}^{D+1} u_i^\star \mathbf{B}_i$. Substituting $\mathbf{X}^\star$ in (22), we obtain the dual formulation

$$\max_{\mathbf{u}\in\mathbb{R}^{D+1},\mathbf{D}\succeq\mathbf{0}} -\mathbf{u}^T\mathbf{1} - \frac{\gamma}{2}\|\mathbf{D}+\mathbf{C}(\mathbf{u})\|_F^2. \quad (24)$$

For a given $\mathbf{u}$, the dual problem in (24) is equivalent to

$$\min_{\mathbf{D}\succeq\mathbf{0}} \quad \frac{\gamma}{2}\|\mathbf{D}+\mathbf{C}(\mathbf{u})\|_F^2. \quad (25)$$

The solution to (25) is $\mathbf{D}^\star = \Pi_+(-\mathbf{C}(\mathbf{u}))$. Due to the fact that $\mathbf{C}(\mathbf{u}) = \Pi_+(\mathbf{C}(\mathbf{u})) - \Pi_+(-\mathbf{C}(\mathbf{u}))$, it follows

$$\mathbf{D}^\star + \mathbf{C}(\mathbf{u}) = \Pi_+(\mathbf{C}(\mathbf{u})). \quad (26)$$

Thus the dual formulation in (24) can be simplified to (12).

We take the first-order derivative of $d_\gamma(\mathbf{u})$ in (12) with respect to $\mathbf{u}$ and obtain

$$\begin{aligned}\nabla_{\mathbf{u}} d_\gamma(\mathbf{u}) &= -\mathbf{1} - \gamma\nabla_{\mathbf{u}}\Big(\frac{1}{2}\|\Pi_+(\mathbf{C}(\mathbf{u}))\|_F^2\Big)\\ &= -\mathbf{1} + \gamma\Phi[\Pi_+(\mathbf{C}(\mathbf{u}))],\end{aligned}$$

where the last equality is due to $\nabla_{\mathbf{U}}(\frac{1}{2}\|\Pi_+(\mathbf{U})\|_F^2) = \nabla_{\mathbf{U}}(\frac{1}{2}\sum_{i=1}^N(\max(0,\lambda_{\mathbf{U},i}))^2) = \Pi_+(\mathbf{U})$, where $\lambda_{\mathbf{U},i}$ is the $i$th eigenvalue of $\mathbf{U} \in \mathbb{R}^{N\times N}$. This concludes the proof.

## APPENDIX B
## PROOF OF LEMMA 2

**Proof** First, suppose the following decomposition

$$\begin{aligned}\mathbf{H} &= \mathbf{P}^T\mathbf{C}(\mathbf{u}_i)\mathbf{P} = [\mathbf{P}_m,\mathbf{P}_n]^T\mathbf{C}(\mathbf{u}_i)[\mathbf{P}_m,\mathbf{P}_n]\\ &= \begin{bmatrix}\mathbf{P}_m^T\mathbf{C}(\mathbf{u}_i)\mathbf{P}_m & \mathbf{P}_m^T\mathbf{C}(\mathbf{u}_i)\mathbf{P}_n\\ \mathbf{P}_n^T\mathbf{C}(\mathbf{u}_i)\mathbf{P}_m & \mathbf{P}_n^T\mathbf{C}(\mathbf{u}_i)\mathbf{P}_n\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{H}_m & \mathbf{H}_{mn}^T\\ \mathbf{H}_{mn} & \mathbf{H}_n\end{bmatrix}, \quad (27)\end{aligned}$$

where $\mathbf{P} \in \mathbb{R}^{(D+1)\times(D+1)}$ is an orthonormal matrix, $\mathbf{P}_m \in \mathbb{R}^{(D+1)\times m}$ and $\mathbf{P}_n \in \mathbb{R}^{(D+1)\times(D+1-m)}$ are two sub-matrices of $\mathbf{P}$, and $\mathbf{H}_{mn} \in \mathbb{R}^{(D+1-m)\times m}$ has only one nonzero entry on its top-right corner, i.e., $\mathbf{H}_{mn}(1,m) = \beta_{m+1}$. Given the above decomposition, we show the proofs of the upper bound on $r_e(\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i, \hat{\lambda}_i\hat{\mathbf{v}}_i)$, $\max_i|\lambda_i - \hat{\lambda}_i|$, and $\min_i|\lambda_i - \hat{\lambda}_i|$, respectively.

i) We compute

$$\begin{aligned}\|\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i - \hat{\lambda}_i\hat{\mathbf{v}}_i\|_2 &= \|\mathbf{C}(\mathbf{u}_i)\mathbf{P}_m\mathbf{q}_i - \vartheta_i\mathbf{P}_m\mathbf{q}_i\|_2\\ &= \|\mathbf{P}^T\mathbf{C}(\mathbf{u}_i)\mathbf{P}_m\mathbf{q}_i - \vartheta_i\mathbf{P}^T\mathbf{P}_m\mathbf{q}_i\|_2\\ &= \left\|\begin{bmatrix}\mathbf{H}_m\mathbf{q}_i\\ \mathbf{H}_{mn}\mathbf{q}_i\end{bmatrix} - \begin{bmatrix}\vartheta_i\mathbf{q}_i\\ \mathbf{0}\end{bmatrix}\right\|_2\\ &\overset{(a)}{=} \|\mathbf{H}_{mn}\mathbf{q}_i\|_2\\ &\overset{(b)}{=} \beta_{m+1}|q_i(m)| \overset{(c)}{\leq} \beta_{m+1}, \quad (28)\end{aligned}$$

where $(a)$ follows from the fact that $\mathbf{H}_m\mathbf{q}_i = \vartheta_i\mathbf{q}_i$, $(b)$ holds because of the special structure of $\mathbf{H}_{mn}$ in (27), and $(c)$ is due to the fact that $\mathbf{q}_i$ is unit norm.

ii) Defining $\hat{\mathbf{H}} \triangleq \begin{bmatrix}\mathbf{H}_m & \mathbf{0}\\ \mathbf{0} & \mathbf{H}_n\end{bmatrix}$ and $\tilde{\mathbf{H}} \triangleq \begin{bmatrix}\mathbf{0} & \mathbf{H}_{mn}^T\\ \mathbf{H}_{mn} & \mathbf{0}\end{bmatrix}$, we have $\mathbf{H} = \hat{\mathbf{H}} + \tilde{\mathbf{H}}$ and the eigenvalues of $\hat{\mathbf{H}}$ include

the eigenvalues of $\mathbf{H}_m$, i.e., $\vartheta_1, \ldots, \vartheta_m$. Then, based on the perturbation theory [49], we obtain

$$|\lambda_i - \hat{\lambda}_i| \leq \|\tilde{\mathbf{H}}\|_2 = \|\mathbf{H}_{mn}\|_2 = \beta_{m+1}.$$

iii) Since $\hat{\mathbf{v}}_i = (\mathbf{C}(\mathbf{u}_i) - \hat{\lambda}_i\mathbf{I})^{-1}(\mathbf{C}(\mathbf{u}_i) - \hat{\lambda}_i\mathbf{I})\hat{\mathbf{v}}_i$ when $\hat{\lambda}_i \neq \lambda_i$, $\forall i$, the following holds

$$1 = \|\mathbf{v}_i\|_2 \leq \|(\mathbf{C}(\mathbf{u}_i) - \hat{\lambda}_i\mathbf{I})^{-1}\|_2 \|\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i - \hat{\lambda}_i\hat{\mathbf{v}}_i\|_2. \quad (29)$$

By assuming that $\mathbf{C}(\mathbf{u}_i) = \mathbf{V}_\mathbf{C}\mathbf{\Lambda}_\mathbf{C}\mathbf{V}_\mathbf{C}^T$ where $\mathbf{\Lambda}_\mathbf{C} = \text{diag}([\lambda_1, \cdots, \lambda_{D+1}]^T)$, we have

$$\begin{aligned}\|(\mathbf{C}(\mathbf{u}_i) - \hat{\lambda}_i\mathbf{I})^{-1}\|_2 &= \|\mathbf{V}_\mathbf{C}(\mathbf{\Lambda}_\mathbf{C} - \hat{\lambda}_i\mathbf{I})^{-1}\mathbf{V}_\mathbf{C}^T\|_2 \\ &= \frac{1}{\min_i |\lambda_i - \hat{\lambda}_i|}. \end{aligned} \quad (30)$$

By substituting (30) into (29), we obtain

$$\begin{aligned}\min_i |\lambda_i - \hat{\lambda}_i| \leq r_e(\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i, \hat{\lambda}_i\hat{\mathbf{v}}_i) &= \|\mathbf{C}(\mathbf{u}_i)\hat{\mathbf{v}}_i - \hat{\lambda}_i\hat{\mathbf{v}}_i\|_2 \\ &= \beta_{m+1}|q_i(m)|,\end{aligned}$$

where the last equality is due to (28).

This concludes the proof.

## Appendix C
## Proof of Lemma 3

**Proof** According to Algorithm 5, we have

$$\begin{aligned}\beta_{j+1} &= \|\mathbf{C}(\mathbf{u}_i)\mathbf{p}_j - \beta_j\mathbf{p}_{j-1} - \alpha_j\mathbf{p}_j\|_2 \\ &\leq \|\mathbf{C}(\mathbf{u}_i)\mathbf{p}_j - \alpha_j\mathbf{p}_j\|_2 + \beta_j,\end{aligned}$$

where the inequality follows from the triangle inequality and the fact that $\mathbf{p}_{j-1}$ is unit norm. Then,

$$\begin{aligned}\beta_{m+1} &\leq \sum_{j=1}^{m} \|(\mathbf{C}(\mathbf{u}_i) - \alpha_j\mathbf{I})\mathbf{p}_j\|_2 \\ &\leq \sum_{j=1}^{m} \sigma_{\max}(\mathbf{C}(\mathbf{u}_i) - \alpha_j\mathbf{I}) \\ &\leq \sum_{j=1}^{m} (\sigma_{\max}(\mathbf{C}(\mathbf{u}_i)) + \sigma_{\max}(\alpha_j\mathbf{I})) \\ &= m\sigma_{\max}(\mathbf{C}(\mathbf{u}_i)) + \sum_{j=1}^{m} |\alpha_j| \\ &\leq 2m\sigma_{\max}(\mathbf{C}(\mathbf{u}_i)), \end{aligned} \quad (31)$$

where the last inequality is due to $|\alpha_j| = |\mathbf{p}_j^T\mathbf{C}(\mathbf{u}_i)\mathbf{p}_j| \leq \sigma_{\max}(\mathbf{C}(\mathbf{u}_i))$, $j = 1, \ldots, m$.

By introducing a normalized Minkowski $\ell_1$-norm $\hat{\sigma}_{\text{max,Minkowski}} \triangleq \frac{1}{D+1}\sum_{\ell=1}^{D+1}\sum_{j=1}^{D+1}|C(\ell, j)|$ [55], [58], which is an approximation of $\sigma_{\max}(\mathbf{C}(\mathbf{u}_i))$, we obtain

$$\begin{aligned}\beta_{m+1} &\leq 2m(\sigma_{\max}(\mathbf{C}(\mathbf{u}_i)) - \hat{\sigma}_{\text{max,Minkowski}}) + 2m\hat{\sigma}_{\text{max,Minkowski}} \\ &\leq 2m\big((\sigma_{\text{max,UB}} - \sigma_{\text{max,LB}}) + \hat{\sigma}_{\text{max,Minkowski}}\big),\end{aligned}$$

where the last inequality stems from the fact that $\sigma_{\text{max,LB}} \leq \sigma_{\max}(\mathbf{C}(\mathbf{u}_i)) \leq \sigma_{\text{max,UB}}$ and $\sigma_{\text{max,LB}} \leq \hat{\sigma}_{\text{max,Minkowski}} \leq \sigma_{\text{max,UB}}$ [55], [59]. This concludes the proof.

## References


[1] F. Zhang, W. Li, Y. Zhang, and Z. Feng, "Data driven feature selection for machine learning algorithms in computer vision," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4262–4272, 2018.

[2] W. Saad, "Tutorial machine learning for AI-driven wireless networks: Challenges and opportunities," in *2019 IEEE Symposium on Computers and Communications (ISCC)*, 2019, pp. 1–1.

[3] A. Faryal, A. Tauqir, A. M. Martinez-Enriquez, and M. Aslam, "Data mining based recommendation system using social websites," in *2015 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, vol. 1, 2015, pp. 365–368.

[4] M. Narayanan and A. K. Cherukuri, "A study and analysis of recommendation systems for location-based social network (LBSN) with big data," *IIMB Management Review*, vol. 28, no. 1, pp. 25–30, 2016.

[5] N. Yi, C. Li, X. Feng, and M. Shi, "Design and implementation of movie recommender system based on graph database," in *2017 14th Web Information Systems and Applications Conference (WISA)*, 2017, pp. 132–135.

[6] A. Nawrocka, A. Kot, and M. Nawrocki, "Application of machine learning in recommendation systems," in *2018 19th International Carpathian Control Conference (ICCC)*, 2018, pp. 328–331.

[7] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[8] B. Ren, L. Pueyo, G. Zhu, J. Debes, and G. Duchêne, "Non-negative matrix factorization: Robust extraction of extended structures," *The Astrophysical Journal*, vol. 852, no. 2, p. 104, Jan 2018.

[9] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008, pp. 426—-434.

[10] J. Cao, H. Hu, T. Luo, J. Wang, M. Huang, K. Wang, Z. Wu, and X. Zhang, "Distributed design and implementation of SVD++ algorithm for e-commerce personalized recommender system," in *Embedded System Technology*. Singapore: Springer Singapore, 2015, pp. 30–44.

[11] C. J. Stark, "Expressive recommender systems through normalized nonnegative models," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1081—-1087.

[12] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[13] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, pp. 206–215, 2019.

[14] H. Ghauch, M. Skoglund, H. Shokri-Ghadikolaei, C. Fischione, and A. H. Sayed, "Learning Kolmogorov models for binary random variables," in *ICML Workshop on Non-convex Optimization*, 2018.

[15] M. Jaggi, "Revisiting Frank-Wolfe: Projection-free sparse convex optimization," in *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, no. 1, 2013, pp. 427–435.

[16] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, "Map estimation via agreement on trees: message-passing and linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 11, pp. 3697–3717, 2005.

[17] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1568–1583, 2006.

[18] P. Ravikumar and J. Lafferty, "Quadratic programming relaxations for metric labeling and markov random field map estimation," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 737–744.

[19] M. P. Kumar, P. H. S. Torr, and A. Zisserman, "Solving markov random fields using second order cone programming relaxations," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1, 2006, pp. 1045–1052.

[20] B. Ghaddar, J. C. Vera, and M. F. Anjos, "Second-order cone relaxations for binary quadratic polynomial programs," *SIAM Journal on Optimization*, vol. 21, no. 1, pp. 391–414, 2011.

[21] S. Kim and M. Kojima, "Exact solutions of some nonconvex quadratic optimization problems via sdp and socp relaxations," *Computational Optimization and Applications*, vol. 26, pp. 143–154, 2003.

[22] M. Jordan and M. Wainwright, "Semidefinite relaxations for approximate inference on graphs with cycles," in *Advances in Neural Information Processing Systems*, vol. 16. MIT Press, 2004, pp. 369–376.

[23] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, 1996.

[24] H. Wolkowicz, R. Saigal, and L. Vandenberghe, *Handbook of Semidefinite Programming: Theory, Algorithms, and Applications*. Berlin, Germany: Springer US, 2000.

[25] M. Kisialiou and Z. Luo, "Probabilistic analysis of semidefinite relaxation for binary quadratic minimization," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1906–1922, 2010.

[26] T. Kim, D. J. Love, M. Skoglund, and Z. Jin, "An approach to sensor network throughput enhancement by PHY-aided MAC," *IEEE Transactions on Wireless Communications*, vol. 14, no. 2, pp. 670–684, 2015.

[27] Q. Duan, T. Kim, H. Ghauch, and E. W. M. Wong, "Enhanced beam alignment for millimeter wave MIMO systems: A Kolmogorov model," in *2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[28] G. Xiong, T. Kim, D. J. Love, and E. Perrins, "Optimality conditions of performance-guaranteed power minimization in MIMO networks: A distributed algorithm and its feasibility," *IEEE Transactions on Signal Processing*, vol. 69, pp. 119–135, 2021.

[29] E. L. Peterson, "An economic interpretation of duality in linear programming," *Journal of Mathematical Analysis and Applications*, vol. 30, pp. 172–196, 1970.

[30] T. Larsson and M. Rönnqvist, "A method for structural optimization which combines secondorder approximations and dual techniques," *Structural Optimization*, vol. 5, pp. 225–232, 1993.

[31] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[32] D. Kim and J. A. Fessler, "Optimized first-order methods for smooth convex minimization," *Math Program.*, vol. 159, no. 1, pp. 81–107, 2016.

[33] C. Broyden, "Quasi-Newton methods and their application to function minimisation," *Mathematics of Computation*, vol. 21, pp. 368–381, 1967.

[34] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. New York, NY, USA: Springer, 2006.

[35] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. The Johns Hopkins University Press, 1996.

[36] E. A. Houseman, W. P. Accomando, D. C. Koestler, and et al., "Dna methylation arrays as surrogate measures of cell mixture distribution," *BMC Bioinformatics*, vol. 13, no. 86, pp. 1–16, 2012.

[37] S. Hur, T. Kim, D. J. Love, J. V. Krogmeier, T. A. Thomas, and A. Ghosh, "Millimeter wave beamforming for wireless backhaul and access in small cell networks," *IEEE Transactions on Communications*, vol. 61, no. 10, pp. 4391–4403, 2013.

[38] A. Alkhateeb, O. El Ayach, G. Leus, and R. W. Heath, "Channel estimation and hybrid precoding for millimeter wave cellular systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 5, pp. 831–846, 2014.

[39] R. M. Gray, *Probability, Random Process, and Ergodic Properties*. Springer US, 2009.

[40] W. M. Chan, H. Ghauch, T. Kim, E. De Carvalho, and G. Fodor, "Kolmogorov model for large millimeter-wave antenna arrays: Learning-based beam-alignment," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, 2019, pp. 411–415.

[41] K. Gomadam, V. R. Cadambe, and S. A. Jafar, "A distributed numerical approach to interference alignment and applications to wireless interference networks," *IEEE Transactions on Information Theory*, vol. 57, no. 6, pp. 3309–3322, 2011.

[42] Q. Shi, M. Razaviyayn, Z. Luo, and C. He, "An iteratively weighted mmse approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4331–4340, 2011.

[43] H. Ghauch, T. Kim, M. Bengtsson, and M. Skoglund, "Subspace estimation and decomposition for large millimeter-wave MIMO systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 3, pp. 528–542, 2016.

[44] W. Zhang, T. Kim, D. J. Love, and E. Perrins, "Leveraging the restricted isometry property: Improved low-rank subspace decomposition for hybrid millimeter-wave systems," *IEEE Transactions on Communications*, vol. 66, no. 11, pp. 5814–5827, 2018.

[45] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of Optimization Theory and Applications*, vol. 109, no. 3, pp. 475–494, 2001.

[46] P. Wang, C. Shen, A. v. d. Hengel, and P. H. S. Torr, "Large-scale binary quadratic optimization using semidefinite relaxation and applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 3, pp. 470–485, 2017.

[47] D. P. Bertsekas, *Nonlinear Programming*, third edition ed. Athena Scientific, 2016.

[48] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives." *Pacific J. Math.*, vol. 16, no. 1, pp. 1–3, 1966. [Online]. Available: https://projecteuclid.org:443/euclid.pjm/1102995080

[49] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*. Cambridge University Press, 1991.

[50] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *J. ACM*, vol. 42, no. 6, pp. 1115—-1145, 1995.

[51] Z. Luo, W. Ma, A. M. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 20–34, 2010.

[52] M. Kisialiou and Zhi-Quan Luo, "Performance analysis of quasi-maximum-likelihood detector based on semi-definite programming," in *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 3, 2005, pp. iii/433–iii/436 Vol. 3.

[53] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Natl. Bur. Stand. B*, vol. 45, pp. 255–282, 1950.

[54] W. E. Arnoldi, "The principle of minimized iterations in the solution of the matrix eigenvalue problem," *Quarterly of Applied Mathematics*, vol. 9, no. 1, pp. 17–29, 1951.

[55] S. Park, A. Alkhateeb, and R. W. Heath, "Dynamic subarrays for hybrid precoding in wideband mmwave MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 5, pp. 2907–2920, 2017.

[56] D. F. Shanno, "Conditioning of quasi-Newton methods for function minimization," *Mathematics of Computation*, vol. 24, no. 111, pp. 647–656, 1970.

[57] H. Ghauch, H. S. Ghadikolaei, M. Skoglund, and C. Fischione, "Learning Kolmogorov models for binary random variables," in *Asilomar*, Nov 2020.

[58] H. Lütkepohl, *Handbook of Matrices*, 1st ed. NJ, USA: Wiley, 1997.

[59] H. Wolkowicz and G. P. Styan, "Bounds for eigenvalues using traces," *Linear Algebra and its Applications*, vol. 29, pp. 471–506, 1980.