Weighted Low Rank Approximations with Provable Guarantees

Ilya Razenshteyn
CSAIL
MIT
Cambridge, MA 02139, USA
ilyaraz@mit.edu

Zhao Song
CS Department
UT-Austin
Austin, TX 78712, USA
zhaos@utexas.edu

David P. Woodruff *
IBM Almaden Research
Center
San Jose, CA 95120, USA
dpwoodru@us.ibm.com

ABSTRACT

The classical low rank approximation problem is: given a matrix A, find a rank-k matrix B such that the Frobenius norm of $A - B$ is minimized. It can be solved efficiently using, for instance, the Singular Value Decomposition (SVD). If one allows randomization and approximation, it can be solved in time proportional to the number of non-zero entries of A with high probability.

Inspired by practical applications, we consider a *weighted* version of low rank approximation: for a non-negative weight matrix W we seek to minimize $\sum_{i,j}(W_{i,j}\cdot(A_{i,j}-B_{i,j}))^2$. The classical problem is a special case of this problem when all weights are 1. Weighted low rank approximation is known to be NP-hard, so we are interested in a meaningful parametrization that would allow efficient algorithms.

In this paper we present several efficient algorithms for the case of small k and under the assumption that the weight matrix W is of low rank, or has a small number of distinct columns. An important feature of our algorithms is that they do not assume anything about the matrix A. We also obtain lower bounds that show that our algorithms are nearly optimal in these parameters. We give several applications in which these parameters are small. To the best of our knowledge, the present paper is the first to provide algorithms for the weighted low rank approximation problem with provable guarantees.

Perhaps even more importantly, our algorithms proceed via a new technique, which we call "guess the sketch". The technique turns out to be general enough to give solutions to several other fundamental problems: adversarial matrix completion, weighted non-negative matrix factorization and tensor completion.

*Supported in part by the XDATA program of the Defense Advanced Research Projects Agency (DARPA), administered through Air Force Research Laboratory contract FA8750-12-C-0323.

STOC '16, June 18-21, 2016, Cambridge, MA, USA

Categories and Subject Descriptors

F.2.1 [**Analyis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems—*Computations on matrices*

General Terms

Theory, Algorithms

Keywords

linear algebra, sketching, semi-algebraic sets, optimization

1. INTRODUCTION

Low rank approximation is arguably one of the most well-studied problems in randomized numerical linear algebra, with diverse applications to clustering [19, 23, 38, 18], data mining [5], distance matrix completion [13], information retrieval [45], learning mixtures of distributions [2, 32], recommendation systems [20], and web search [1, 34]. In practice one often has a low rank matrix which has been corrupted with noise of bounded norm, and low rank approximation allows one to approximately recover the original matrix. Low rank approximation may also help explain a dataset, revealing low dimensional structure in high dimensional data. Given a low rank approximation, one can store a matrix and compute a matrix-vector product much more efficiently by storing the corresponding factorization. It can also be used as a preprocessing step in applications, that is, by first projecting data onto a lower-dimensional subspace one preserves important properties of the input, but can now run subsequent algorithms in the lower-dimensional space. For example, it has been proposed to reduce the data dimension in Non-Negative Matrix Factorization [35] (more on this below), and Latent Dirichlet Allocation (LDA) [10].

The basic low rank approximation problem is: given an $n \times n$ matrix A, find a matrix $\hat{A}$ of rank at most k for which $\|A - \hat{A}\|_F$ is minimized, where for a matrix B, $\|B\|_F = \left(\sum_{i,j} B_{i,j}^2\right)^{1/2}$ is its Frobenius norm. This formulation intuitively corresponds to the matrix $\hat{A}$ capturing as much of the variance of A as possible. It is well-known that the optimal solution is given by A_k, which if $U\Sigma V^\top$ is the singular value decomposition (SVD) of A, where U and V are orthogonal matrices and Σ is a non-negative diagonal matrix with $\Sigma_{1,1} \geq \Sigma_{2,2} \geq \cdots \geq \Sigma_{n,n}$, then $A_k = U\Sigma_k V^\top$, where Σ_k agrees with Σ on its first k diagonal entries and is 0 otherwise. Although the SVD is computable in polynomial

time, it is often acceptable to output a matrix $\hat{A}$ for which $\|A - \hat{A}\|_F \leq (1+\epsilon)\|A - A_k\|_F$ with high probability. In the latter case, much more efficient algorithms are known, and it is possible to compute such an $\hat{A}$ in $\mathrm{nnz}(A) + n \cdot \mathrm{poly}(k/\epsilon)$ time, where $\mathrm{nnz}(A)$ denotes the number of non-zero entries of A [16, 42, 44]. We note that for typical applications k and $1/\epsilon$ are assumed to be much smaller than n, e.g., in [28] they are treated as absolute constants.

Despite the large body of work on low rank approximation, the *weighted* case is not well understood. In this case one is given an $n \times n$ matrix A and an $n \times n$ matrix W with $W_{i,j} \geq 0$, and one seeks to solve:

$$\min_{\text{rank-}k \text{ matrices } \hat{A}} \|W \circ (A - \hat{A})\|_F^2$$
$$= \min_{\text{rank-}k \text{ matrices } \hat{A}} \sum_{i,j} W_{i,j}^2 (A_{i,j} - \hat{A}_{i,j})^2.$$

The classical low rank approximation is a special case in which $W_{i,j} = 1$ for all i and j. However, in general there may not be a good reason to weight all elements of the approximation error $A - \hat{A}$ equally, especially if one is given prior knowledge about the distribution of the errors. For example, suppose the columns of A each come from a low-dimensional subspace but one of the columns is then shifted by a fixed large vector so that its mean is different. One may first want to recenter the data by subtracting off the mean from each of the columns. While this is possible without weighted low rank approximation, suppose instead that each of the columns of A comes from a perturbation of columns in a low dimensional subspace but one of the columns has a much larger variance. Then if all weights were equal, it would be enough for $\hat{A}$ to fit this one single large variance column, which fails to capture the entire low-dimensional subspace. One way of fixing this is to reweight each entry of A by the inverse of its variance. This is a common technique used in gene expression analysis, where the error model for microarray measurements provides entry-specific noise estimates, or when entries of A represent aggregates of many samples such as in word co-occurrence matrices and non-uniform weights are needed to appropriately capture any differences in the sample sizes; see [52] for a discussion, and also the Wikipedia entry on weighted low rank approximation for a brief introduction[1].

While the extension of low rank approximation to the weighted case goes back to work of Young in 1940 [55], its complexity is not well-understood, partly because the weighted case does not admit a solution via the SVD and may have many local minima [52]. Early work by Shpak [51] looked at gradient-based approaches while Lu et al. [40, 39] looked at alternating minimization methods. These were significantly sped up in practice by the work of Srebro and Jaakkola [52], with success in various applications such as color image restoration [41], though there are no provable time bounds and in the worst case the running times could be exponential or worse. In fact, weighted low rank approximation is known to be NP-hard to approximate up to a $(1 \pm 1/\mathrm{poly}(n))$ factor [24]. We note that this also follows from the fact that matrix completion, arguably one of the most important special cases of weighted low rank approximation in which case all weights are 0 or 1, which we discuss more below, is also known to be NP-hard [46, 29]. Typically, though, assumptions such as incoherence and randomly sampled entries allow one to circumvent this hardness [12, 37]. There is some debate as to whether these assumptions are valid, for instance in [50] an argument is made why randomly missing entries may not hold for real-world datasets.

Many natural questions are left open from previous work. In particular the main question as we see it is the following:

- *for which weight matrices W is the problem tractable? More generally, is it possible to identify a natural parameter of W and to obtain parameterized complexity bounds in terms of that parameter?*

- There are many variants of weighted low rank approximation, such as weighted non-negative matrix factorization, matrix and tensor completion, bicriteria approximations, etc. *Can one obtain similar parameterized bounds for these problems?*

1.1 Our Results for Weighted Low Rank Approximation

In this paper we provide an answer to the above questions by parameterizing the complexity of the problem in terms of the *rank of the weight matrix W* and the *number of distinct columns of the weight matrix.* Note that we make *no assumptions* about the input matrix A. Let OPT denote the quantity $\min_{\text{rank-}k \text{ matrices } \hat{A}} \|W \circ (\hat{A} - A)\|_F^2$. Our main theorems are the following. For a function f, define $\tilde{f} = f \cdot \mathrm{poly}(\log(f))$.

THEOREM 1.1. *(Algorithm for Weighted Low Rank Approximation) Let r be the rank of W. There is an algorithm running in time $n^{O(k^2 r/\varepsilon)}$ which outputs a factorization (into an $n \times k$ and a $k \times n$ matrix) of a rank-k matrix $\hat{A}$ for which*

$$\|W \circ (A - \hat{A})\|_F^2 \leq (1+\epsilon)OPT,$$

with probability at least 9/10.

We also have the following theorem.

THEOREM 1.2. *(Algorithm for Weighted Distinct Columns) Let r be the number of distinct columns of W. For every $\varepsilon > 0$ and for an arbitrarily small constant $\gamma > 0$, there is an algorithm running in time $O((\mathrm{nnz}(A) + \mathrm{nnz}(W)) \cdot n^{\gamma}) + n \cdot 2^{\widetilde{O}(k^2 r^2/\varepsilon)}$ which outputs a factorization (into an $n \times k$ and a $k \times n$ matrix) of a rank-k matrix $\hat{A}$ for which*

$$\|W \circ (A - \hat{A})\|_F^2 \leq (1+\epsilon)OPT,$$

with probability at least 9/10.

Note that Theorem 1.2 does not make any assumptions on the number of distinct rows, which is important for the applications described below.

Let us point out two things here:

- Before only the case of W having rank 1 was known to be provably solvable in polynomial time by a direct reduction to the SVD[2];

[1] `https://en.wikipedia.org/wiki/Low-rank_approximation#Weighted_low-rank_approximation_problems`

[2] Namely, in that case, we can rewrite the problem as $\min_{\text{rank-}k \text{ matrices } \hat{A}} \|D(\hat{A} - A)E\|_F^2$ for diagonal matrices D and E, from which we can replace A with DAE and solve the un-

- The result for at most r distinct columns implies that, with respect to this parametrization, the weighted low rank approximation problem is fixed-parameter tractable.

We complement the above positive results by proving a lower bound, which assumes the Exponential Time Hypothesis for the average case hardness of random 4-SAT. We state the assumption as follows.

ASSUMPTION 1.3. *("Random Exponential Time Hypothesis") Let $c > \ln 2$ be a constant. Consider a random 4-SAT formula on n variables in which each clause has 4 literals, and in which each of the $16n^4$ clauses is picked independently with probability c/n^3. Then any algorithm which always outputs 1 when the random formula is satisfiable, and outputs 0 with probability at least $1/2$ when the random formula is unsatisfiable, must run in $2^{c'n}$ time on some input, where $c' > 0$ is an absolute constant.*

We are not aware of prior work using this form of the Exponential Time Hypothesis, though both the Exponential Time Hypothesis and Feige's original assumption [22] that there is no polynomial time algorithm for the problem in Assumption 1.3 are commonly used. We do not know of any better algorithm for the problem in Assumption 1.3 and have consulted several experts[3] about the assumption who do not know a counterexample to it.

THEOREM 1.4. *(Weighted Low Rank Approximation Hardness) Let r be an upper bound on the number of distinct columns of W. Under Assumption 1.3, there is an absolute constant $\varepsilon_0 \in (0,1)$ for which any algorithm for solving the weighted low rank approximation algorithm with $\epsilon \leq \varepsilon_0$ and for any $k \geq 1$, with constant probability, requires $2^{\widetilde{\Omega}(r)}$ time. Further, this holds even if W also only has r distinct rows.*

Note that for constant k and ϵ, and $r \geq C \log n$ for a constant $C > 0$, our upper bound assuming at most r distinct columns is $2^{\tilde{O}(r^2)}$, which nearly matches our lower bound in Theorem 1.4 of $2^{\widetilde{\Omega}(r)}$.

There are naturally arising applications in which the rank of the weight matrix or the number of distinct columns is small. Consider a matrix in which the rows correspond to users and the columns correspond to ratings of a movie, such as in the Netflix matrix. Further, suppose for each movie, there are r columns, indicating different aspects of the movie to be rated, such as acting, plot, sound effects, visual effects, etc. For a given user, one can look at the distribution of scores across movies along one of these aspects. These distributions may have different variances for that user and one can renormalize the scores by the reciprocal of their variance. In this case, the weight matrix consists of r distinct columns, one for each aspect, each copied n/r times, where n/r is the total number of movies. Each entry of a column is a variance for a certain user for that aspect. This naturally generalizes to other applications for which the columns can be clustered into r groups, such as in stochastic block models or more general latent space models [13]. Also, in some of these applications, one would want a low rank nonnegative factorization, and we remark that we can achieve this below.

Suppose now that A has constant rank k. A consequence of Theorem 1.1, which we elaborate on more below, is that even if an adversary deletes up to $O(1)$ entries in each column of A in an arbitrary way, one can still recover a matrix $\widehat{A}$ of rank at most k which agrees with A on all of the remaining entries in $\text{poly}(n)$ time. This is a form of *adversarial* matrix completion, which was studied in [30, 50]. In these works the authors had to make an incoherence assumption on the entries of A, which may not always hold, e.g., in the presence of outliers. Without this assumption the prior results would not be able to recover such an $\widehat{A}$ even if a single adversarially chosen entry was deleted in each column.

1.2 Other Results

Our techniques can be applied to weighted non-negative factorization, a problem that has been extensively studied both in the unweighted (see, e.g., [4, 43]) and weighted cases (see, e.g., [27, 26, 33, 56, 54]). In this problem, one seeks *non-negative* matrices $U \in \mathbb{R}^{n \times k}$ and $V \in \mathbb{R}^{k \times n}$ for which

$$\forall i, j \text{ such that } W_{i,j} > 0,\ A_{i,j} = (U \cdot V)_{i,j},$$

where the unweighted case corresponds to $W_{i,j} = 1$ for all i and j. This problem naturally arises in applications such as topic modeling in which negative entries do not make sense. In a beautiful line of work [4, 43], an upper bound of $n^{O(k^2)}$ was established for the unweighted case. By combining their techniques with ours, we obtain the first results for the weighted case when the weight matrix W has rank at most r. We defer the details to the full version of the paper.

Our results also apply to adversarial tensor completion. Here we use the standard idea of "flattening" a tensor to a matrix. Our algorithm follows that in [50] and flattens the tensor multiple times, and each time solves an adversarial matrix completion problem. We then introduce additional polynomial constraints to put these solutions together to recover a tensor agreeing with the original tensor on the entries for which the corresponding weights are positive. We defer the details to the full version of the paper.

Finally, there is a simple bi-criteria approximation algorithm for weighted low rank approximation when the weight matrix has entries that are all either 0 or 1. In this case it is possible to obtain a rank-rk approximation to A in $\text{nnz}(A) + \text{nnz}(W) + n \cdot \text{poly}(rk/\epsilon)$ time with cost at most $(1+\epsilon)$ times the cost of the best rank-k approximation. Here r denotes the rank of the weight matrix. This has a better dependence on r and could be useful when one does not need to output a matrix of rank exactly k. This is given in the full version of the paper.

1.3 Our Techniques

To prove Theorem 1.1, we first prove a structural theorem about regression. Then our main new theme is what we call "guessing a sketched matrix".

Structural Regression Theorem .

Given any number t of multiple response regression instances, i.e., instances of the form $\min_{X^1, X^2, \cdots, X^t} \|A^i X^i - B^i\|_F^2$, where $A^1, \ldots, A^t$ and $B^1, \ldots, B^t$ are matrices and each A^i has rank at most k, if one is interested in simultaneously

weighted low rank approximation, obtaining an $\hat{A}$ for which we can then output $D^{-1}\hat{A}E^{-1}$ if the diagonal entries of D and E are non-zero. If they are zero we can first remove rows and columns from A

[3]Personal communication with Russell Impagliazzo and Ryan Williams.

minimizing the sum of their costs, i.e., the objective function is $\min_{X^1,X^2,\cdots,X^t} \sum_{i=1}^{t} \|A^iX^i - B^i\|_F^2$, then one can choose a Gaussian matrix S with $O(k/\epsilon)$ rows and if one solves the problem $\min_{Y^1,Y^2,\cdots,Y^t} \sum_{i=1}^{t} \|SA^iY^i - SB^i\|_F^2$, then the minimizers $Y^1, Y^2, \cdots, Y^t$ for this latter problem satisfy $\sum_{i=1}^{t} \|A^iY^i - B^i\|_F^2 \leq (1+\epsilon) \min_{X^1,X^2,\cdots,X^t} \sum_{i=1}^{t} \|A^iX^i - B^i\|_F^2$ with high constant probability. Interestingly, the number of rows of S does not depend on the number t of regression instances, and is optimal already for $t = 1$ and when B^1 only has a single column [15]. This also generalizes affine embeddings in [16] in which $t = 1$ and B^1 may have multiple columns; we stress that the design matrices A^i may be different. The proof uses a novel observation that a Gaussian S is a *subspace embedding on average over i*. This part uses a tail bound from [14] for the condition number for a Gaussian matrix. Having this, we bootstrap the approximation to $(1 \pm \varepsilon)$ using the approximate matrix product property of a Gaussian matrix.

Guessing a Sketched Matrix .

Given our structural result, the main new theme of this paper is to "guess a sketched matrix". Suppose one is given an optimization problem of the form

$$p_1(x_1, \ldots, x_v) \geq 0, p_2(x_1, \ldots, x_v) \geq 0, \ldots, p_m(x_1, \ldots, x_v) \geq 0,$$

where each p_i is a polynomial of degree at most d, the $x_1, \ldots, x_v$ are indeterminates over the reals, and we are are interested in an assignment to $x_1, \ldots, x_v$ which simultaneously satisfies these c polynomial inequalities. Then this can be solved in time $(md)^{O(v)}$ using generic solvers [49, 48, 47, 8], and such techniques have been used in the context of database theory [21], non-negative matrix factorization [4, 43], learning mixtures of Gaussians [36], computing approximate PSD factorizations [6], and solving small-scale mixed-norm low rank approximation instances [17]. There are two kinds of algorithms for semi-algebraic sets: the ones from [47, 48, 8] are able to determine if a given sem-algebraic set is empty or not. The one from [49] is able to return a δ-approximate solution to a given semi-algebraic formulae by paying an extra factor of $\log(\frac{1}{\delta})$ in the running time. For weighted low-rank approximation, there are two ways to output the matrices. One is using the algorithm from [49] directly. Another option is to perform binary search using the algorithm from [47, 48, 8] for the entries of $\widehat{A}_{ij}$ one by one.

Our main idea here is to use polynomial optimization for large-scale non-convex optimization by combining it directly with sketching. For example, suppose one is given the multiple response regression problem $\min_V \|UV - A\|_F^2$ in which the number of columns of U is small. The twist though, is that *both U and V are unknown*! Then UV is just a low rank approximation to A, and if we knew U we could solve the sketched optimization problem $\min_X \|SUV - SA\|_F^2$, for a random oblivious sketching matrix S, and our solution V would be a good solution to the original problem. Since we do not know U, we instead choose a random S and create variables for $S \cdot U$, which is small, and also compute $S \cdot A$, which we know. We then solve a regression problem for V in terms of the variables that we created for $S \cdot U$. Given V, we can then plug it into the original regression problem and solve for U in terms of V which is in turn in terms of our variables for $S \cdot U$. Finally we can verify the solution by requiring that $\|UV - A\|_F^2$ is small, which is now a system in a small number of variables. This verification step is essential because our S only has a probabilistic guarantee that it works for a fixed U with good probability, but crucially, we know there *exists* a U for which it works, and so by doing the verification step we will find such a U. We note there are several issues with this approach which we discuss below.

While this may seem like an unnecessarily complicated way of doing standard low rank approximation, this idea proves crucial for weighted low rank approximation. In this case, using say, the rank constraint on the weight matrix W, and using our structural result on multiple instances of regression, we are able to choose a single sketching matrix S and create variables for only r regression problems, $SD_{W_1}U, \ldots, SD_{W_r}U$, where $D_{W_1}, \ldots, D_{W_r}$ are diagonal matrices with independent columns of W on each diagonal, and U is a fixed optimal solution. We can then try to express all regression solutions $\min_{X^i} \|D_{W_i}UX^i - D_{W_i}A^i\|_2^2$, for $i = 1, \ldots, n$ and where X^i and A^i are the i-th columns of X and A respectively, in terms of these variables, and hope to carry out the procedure above.

Dealing with Linear Dependencies .

At this point another obstacle arises which is that when the columns of a given $D_{W_i}U$ are not linearly independent, there is no way to write down the pseudoinverse of $D_{W_i}U$ in terms of the variables we have created. We also cannot afford to create more than $r \cdot \text{poly}(k/\epsilon)$ variables, since our optimization procedure is exponential in this quantity, and so we cannot create new variables for each $i = 1, \ldots, n$. To get around this, we observe that there is a solution $U \cdot V$ for which for all i, the first $\min(|\text{supp}(D_{W_i})|, k)$ columns of U are linearly independent, where $\text{supp}(D_{W_i})$ denotes the set of non-zero entries of W_i, and further the solution cost of $U \cdot V$ is an arbitrarily small amount larger than that of the optimal cost. This follows by a simple perturbation argument applied to the optimal solution. This immediately gives an algorithm with additive error when we parameterize W by its rank.

In order to turn it into a relative error algorithm, we need a lower bound on the cost assuming that the cost is non-zero. The main idea here is that if we correctly guess which subsets of columns are linearly independent for the different matrices D_{W_i}, then we can set up a non-negative polynomial system and provided this system has non-zero cost, we can apply known lower bounds on the cost of polynomial optimazation problems as a function of the degrees, number of variables, number of constraints, and coefficient sizes [7]. While this is not an algorithmic procedure, since we cannot afford to make guess for each D_{W_i} without spending exponential in n time, it suffices for lower bounding the cost. Given such a lower bound, the above perturbation argument can then be used to argue that we achieve relative error.

While this leads to our time bound in the case in which we parameterize the weight matrix by its rank, in the case in which we parameterize by the number of distinct columns of W, this is too slow. The issue is that we have at least n constraints to enforce, namely, that the first $\min(\text{supp}(W_i), k)$ columns of $SD_{W_i}U$ are linearly independent. This would

lead to a running time of $n^{\text{poly}(rk/\epsilon)}$ as opposed to the $\text{poly}(n) \cdot 2^{\text{poly}(kr/\epsilon)}$ that we desire. We notice though that when we have r distinct columns, there are only r constraints to enforce, one for each distinct column. These are "not equal" constraints but can be transformed to a single equality constraint of degree $\text{poly}(rk/\epsilon)$ by introducing a single auxiliary variable. While this ultimately enables us to write down all the entries of V using only a $\text{poly}(rk/\epsilon)$ number of variables, we are then faced with the task of writing down U in terms of such a V. Here a priori we could have many distinct *rows* in W, and may have n constraints to enforce. We observe though that since the entries of W are integers in $\{1, 2, \ldots, \text{poly}(n)\}$, if we round them to the nearest power of $1+\epsilon$, the solution cost changes only by a $(1+\epsilon)$-factor. Moreover, given that we have r distinct columns, for any row it is entirely specified on these r columns and after rounding, there are only $O((\log n)/\epsilon)^r$ choices for entries on these r columns, which upper bounds the number of distinct rows. This ultimately allows us to write down U in terms of V with only $O((\log n)/\epsilon)^r$ not equal constraints. This ultimately yields our improved running time when parameterizing W by its number of distinct columns.

Dealing with Rational Functions .

There is a subtle problem with the above arguments. When one solves for the i-th column V^i of V in terms of $SD_{W_i}U$, the entries of V^i are *rational functions* rather than polynomials, and we cannot afford to clear the denominators of V^i for every i without blowing up the degree of the polynomials to $\Omega(n)$, which would give a running time of $n^{\text{poly}(kr/\epsilon)}$. While this is not a problem when we parameterize the problem by the rank of W, since we anyway spend this amount of time, this is a problem when we parameterize by the number of distinct columns of W, in which we seek polynomial time even for super-constant r (and constant k/ϵ). Instead, we write V as $V' \cdot D$, where V' has entries which are polynomials, and D is a diagonal matrix whose entries are $\frac{1}{\det(U^\top D_{W_i} S^\top S D_{W_i} U)}$, given by Cramer's rule. The entries of D are rational functions, and we would like to make them polynomials. Since W has at most r distinct columns, D has at most r distinct entries and we can create r new variables for the entries of D. However, when we try to solve for U in terms of V we face the same problem again: we can write U as $E \cdot U'$, where U' only has polynomial entries, but since we do not assume a small number of distinct rows of W, it follows that E could have n distinct entries $\frac{1}{\det(U^\top D_{W^i} S^\top S D_{W^i} U)}$ for $i = 1, \ldots, n$, where D_{W^i} is the diagonal matrix with the i-th row of W on the diagonal. As mentioned above, we fix this problem by rounding the entries of W to powers of $1+\epsilon$, and then observing that the number of distinct rows of W can only be $O((\log n)/\epsilon)^r$ after rounding.

Other Methods .

Our techniques for weighted non-negative matrix factorization and tensor completion largely follow these ideas as well, where we combine our "guessing a sketched matrix" approach with techniques of Arora-Ge-Kannan-Moitra [4] and Moitra [43]. Our bi-criteria solution directly follows from known sketching results.

1.4 Empirical Results

We perform some preliminary experiments in Section 10, which show our algorithm may perform better than our theory predicts.

2. PRELIMINARIES

Notation .

Let $\mathbb{R}$ denote the real numbers, and $\mathbb{R}_{\geq 0}$ denote the nonnegative real numbers. Let $\|A\|$ (and sometimes $\|A\|_2$) denote the spectral norm of matrix A. Let $\|A\|_F^2 = \sum_{i,j} A_{i,j}^2$ denote the Frobenius norm of A. Let $W \circ A$ denote the entry-wise product of matrices W and A. Let $\|A\|_W^2 = \sum_{i,j} W_{i,j}^2 A_{i,j}^2$ denote the weighted Frobenius norm of A. Let $\text{nnz}(A)$ denote the number of nonzero entries of A. Let $\det(A)$ denote the determinant of a square matrix A. Let $A^\top$ denote the transpose of A. Let $A^\dagger$ denote the Moore-Penrose pseudoinverse of A. Let A^{-1} denote the inverse of a full rank square matrix A.

For the weight matrix W, we always use W_j to denote the j-th column vector of W, and W^i to denote the i-th row of W. Let D_{W_j} denote the diagonal matrix with entries from the column vector W_j and D_{W^i} denote the diagonal matrix with entries from the row vector W^i.

The following real algebraic geometry definitions are needed when proving a lower bound for the minimum nonzero cost of our problem. For a full discussion, we refer the reader to Bochnak *et al.* [11]. Here we use the brief summary by Basu *et al.*[9].

DEFINITION 2.1 ([9]). *Let R be a real closed field. Given $x = (x_1, \cdots, x_v) \in R^v, r \in R, r > 0$, we denote*

$$B_v(x,r) = \{y \in R^v \,|\, \|y-x\|^2 < r^2\} \qquad \text{(open ball)},$$
$$\bar{B}_v(x,r) = \{y \in R^v \,|\, \|y-x\|^2 \leq r^2\} \qquad \text{(closed ball)}.$$

A set $S \subset R^v$ is open if it is the union of open balls, i.e., if every point of U is contained in an open ball contained in U.

A set $S \subset R^v$ is closed if its complement is open. Clearly, the arbitrary union of open sets is open and the arbitrary intersection of closed sets is closed.

Semi-algebraic sets are defined by a finite number of polynomial inequalities and equalities.

A semi-algebraic set has a finite number of connected components, each of which is semi-algebraic. Here, we use the topological definition of a connected component, which is a maximal connected subset (ordered by inclusion), where connected means it cannot be divided into two disjoint nonempty closed sets.

A closed and bounded semi-algebraic set is compact.

A semi-algebraic set $S \subset R^v$ is semi-algebraically connected if S is not the disjoint union of two non-empty semi-algebraic sets that are both closed in S. Or, equivalently, S does not contain a non-empty semi-algebraic strict subset which is both open and closed in S.

A semi-algebraically connected component of a semi-algebraic set S is a maximal semi-algebraically connected subset of S.

Renegar [47, 48] and Basu *et al.* [8] independently provided an algorithm for the decision problem for the existential theory of the reals is to decide the truth or falsity of a sentence $(x_1, \cdots, x_v)F(f_1, \cdots, f_m)$ where F is a quantifier-free Boolean formula with atoms of the form $\text{sign}(f_i) = \sigma$ with $\sigma \in \{0, 1, -1\}$. Note that this problem is equivalent to deciding if a given semi-algebraic set is empty or not.

Here we formally state that theorem. For a full discussion of algorithms in real algebraic geometry, we refer reader to [9] and [7].

THEOREM 2.2 (DECISION PROBLEM [47, 48, 8]). *Given a real polynomial system $P(x_1, x_2, \cdots, x_v)$ having v variables and m polynomial constraints $f_i(x_1, x_2, \cdots, x_v)\Delta_i 0, \forall i \in [m]$, where Δ_i is any of the "standard relations": $\{>, \geq, =, \neq, \leq, <\}$, let d denote the maximum degree of all the polynomial constraints and let H denote the maximum bitsize of the coefficients of all the polynomial constraints. Then in*

$$(md)^{O(v)}\text{poly}(H)$$

time one can determine if there exists a solution to the polynomial system P.

The key result we used for proving lower bound is the following bound on the minimum value attained by an integer polynomial restricted to a compact connected component of a basic closed semi-algebraic subset of $\mathbb{R}^v$ defined by polynomials with integer coefficients in terms of the degrees and the bitsizes of the coefficients of the polynomials invovled.

THEOREM 2.3 ([31]). *Let $T = \{x \in \mathbb{R}^v | f_1(x) \geq 0, \cdots, f_\ell(x) \geq 0, f_{\ell+1}(x) = 0, \cdots, f_m(x) = 0\}$ be defined by polynomials $f_1, \cdots, f_m \in \mathbb{Z}[x_1, \cdots, x_v]$ with $n \geq 2$, degrees bounded by an even integer d and coefficients of absolute value at most H, and let C be a compact connected component of T. Let $g \in \mathbb{Z}[x_1, \cdots, x_v]$ be a polynomial of degree at most d and coefficients of absolute value bounded by H. Then, the minimum value that g takes over C is satisfies that if it is not zero, then its absolute value is greater than or equal to*

$$(2^{4-v/2}\widetilde{H}d^v)^{-v2^v d^v},$$

where $\widetilde{H} = \max\{H, 2v + 2m\}$.

While the above theorem involves notions from topology, we shall apply it in an elementary way. Namely, in our setting T will be bounded and so every connected component, which is by definition closed, will also be bounded and therefore compact. As the connected components partition T the theorem will just be applied to give a global minimum value of g on T provided that it is non-zero.

3. MULTIPLE REGRESSION SKETCH

THEOREM 3.1. *Let $A^1, \ldots, A^m \in \mathbb{R}^{n\times k}$ be m matrices of size $n \times k$. Let $b^1, \ldots, b^m \in \mathbb{R}^{n\times 1}$ be m column vectors of dimension n.*

For $1 \leq i \leq m$ denote:

$$x^i = \underset{x\in\mathbb{R}^{k\times 1}}{\text{argmin}} \left\| A^i x - b^i \right\|_2^2$$

the solution of the i-th regression problem.

Let $S \in \mathbb{R}^{t\times n}$ be a random matrix with i.i.d. Gaussian entries with zero mean and standard deviation $1/\sqrt{t}$. For $1 \leq i \leq m$ denote:

$$y^i = \underset{y\in\mathbb{R}^{k\times 1}}{\text{argmin}} \left\| SA^i y - Sb^i \right\|_2^2$$

the solution of the i-th regression problem in the sketch space.

We claim that for every $0 < \varepsilon < 1/2$ one can set $t = O(k/\varepsilon)$ such that:

$$\sum_{i=1}^m \left\| A^i y^i - b^i \right\|_2^2 \leq (1+\varepsilon) \cdot \sum_{i=1}^m \left\| A^i x^i - b^i \right\|_2^2.$$

The rest of this section is devoted to the proof of this theorem.

For $1 \leq i \leq m$ we let $D^i \geq 1$ denote the smallest number such that for every $x \in \mathbb{R}^{k\times 1}$ and $\lambda \in \mathbb{R}$ one has:

$$\left\| S\left(A^i x + \lambda b^i\right) \right\|_2^2 \in \left[\frac{1}{D^i}; D^i\right] \cdot \left\| A^i x + \lambda b^i \right\|_2^2.$$

CLAIM 3.2. *For every i*

$$\left\| A^i y^i - b^i \right\|_2^2 \leq \left(D^i\right)^2 \cdot \left\| A^i x^i - b^i \right\|_2^2.$$

PROOF. This follows from the definition of D^i. □

CLAIM 3.3. *One can set $t = O(k/\varepsilon)$ such that for every i*

$$\Pr_S\left[D^i \geq 1.01\right] \leq 2^{-\Omega(1/\varepsilon)}.$$

PROOF. This follows from Theorem 2.1 from [53]. □

CLAIM 3.4. *One can set $t = O(k/\varepsilon)$ such that for every i*

$$\mathop{\mathrm{E}}_S\left[\left\| A^i y^i - b^i \right\|_2^2 - \left\| A^i x^i - b^i \right\|_2^2 \;\middle|\; D^i \leq 1.01\right] \leq \varepsilon \cdot \left\| A^i x^i - b^i \right\|_2^2.$$

PROOF. This follows from the proofs of Theorem 2.8 and Theorem 2.16 from [53] (adapted to Gaussian matrices). □

CLAIM 3.5. *One can set $t = O(k/\varepsilon)$ such that for every i*

$$\mathrm{E}_S\left[\left(D^i\right)^2 \;\middle|\; D^i \geq 1.01\right] = O(1).$$

PROOF. One can see that D^i is polynomially related to the condition number of a random $O(k/\varepsilon) \times (k+1)$ matrix with i.i.d. Gaussian entries; indeed it corresponds to the maximum distortion of S applied to the vectors in the column span of an $n \times (k+1)$ orthonormal matrix U whose columns span the space spanned by the columns of A^i together with b^i. By rotational invariance, $S \cdot U$ also has i.i.d. Gaussian entries. To understand the condition number one can invoke the main result from [14] which gives for all sufficiently large t: $\Pr_S\left[D^i \geq t\right] = \frac{1}{t^{\Theta(k/\varepsilon)}}$. Thus,

$$\mathrm{E}_S\left[\left(D^i\right)^2 \;\middle|\; D_i \geq 1.01\right] \leq O(1) + \int_{1.01}^{\infty} \frac{t^2}{t^{\Theta(k/\varepsilon)}}\, dt = O(1).$$

□

Having these Claims, let us complete the proof. We have for every $1 \leq i \leq m$:

$$\begin{aligned}
& \mathrm{E}_S\left[\left\|A^i y^i - b^i\right\|_2^2 - \left\|A^i x^i - b^i\right\|_2^2\right] \\
= \; & \Pr_S\left[D^i \geq 1.01\right] \\
\cdot \; & \mathrm{E}_S\left[\left\|A^i y^i - b^i\right\|_2^2 - \left\|A^i x^i - b^i\right\|_2^2 \;\middle|\; D^i \geq 1.01\right] \\
+ \; & \Pr_S\left[D^i \leq 1.01\right] \\
\cdot \; & \mathrm{E}_S\left[\left\|A^i y^i - b^i\right\|_2^2 - \left\|A^i x^i - b^i\right\|_2^2 \;\middle|\; D^i \leq 1.01\right] \\
\leq \; & 2^{-\Omega(1/\varepsilon)} \cdot \mathrm{E}_S\left[\left(D^i\right)^2 - 1 \;\middle|\; D^i \geq 1.01\right] \cdot \left\|A^i x^i - b^i\right\|_2^2 \\
+ \; & \Pr_S\left[D^i \leq 1.01\right] \\
\cdot \; & \mathrm{E}_S\left[\left\|A^i y^i - b^i\right\|_2^2 - \left\|A^i x^i - b^i\right\|_2^2 \;\middle|\; D^i \leq 1.01\right] \\
\leq \; & 2^{-\Omega(1/\varepsilon)} \cdot \mathrm{E}_S\left[\left(D^i\right)^2 - 1 \;\middle|\; D^i \geq 1.01\right] \cdot \left\|A^i x^i - b^i\right\|_2^2 \\
+ \; & \varepsilon \cdot \left\|A^i x^i - b^i\right\|_2^2 \\
\leq \; & O(\varepsilon) \cdot \left\|A^i x^i - b^i\right\|_2^2,
\end{aligned}$$

where the second step is by Claim 3.2 and Claim 3.3, the third step is by Claim 3.4, and the fourth step is by Claim 3.5.

Summing over i and applying the Markov's inequality, we are done.

While the above result is for Gaussian sketching matrices, one can also combine a Gaussian random matrix with a Count-Sketch matrix [16]. This way we are still getting $O(k/\varepsilon)$ rows, but now one can perform a matrix-vector multiplication in time proportional to the sparsity of the vector plus $\mathrm{poly}(k\widetilde{r}/\varepsilon)$, where $\widetilde{r}$ is the dimension of the union of column spaces of A^i (which is at most km in the worst case, but is much smaller for our applications).

4. ADDITIVE APPROXIMATION

In this Section, to demonstrate the new technique, we prove the following theorem.

THEOREM 4.1. *Given $A, W \in \mathbb{R}^n$, $1 \leq k \leq n$ and $0 < \varepsilon, \tau < 0.1$ such that:*

- $\mathrm{rank}(W) = r$;
- *all the non-zero entries of A and W are multiples of $\delta > 0$;*
- *all the entries of A and W are at most $\Delta > 0$ in absolute value,*

one can output a number Λ in time $n^{O(k^2 r/\varepsilon)} \cdot \log^{O(1)} \frac{\Delta}{\delta\tau}$ such that $\mathrm{OPT} \leq \Lambda \leq (1+\varepsilon)OPT + \tau$, where

$$\mathrm{OPT} = \min_{U \in \mathbb{R}^{n\times k}, V \in \mathbb{R}^{k\times n}} \|(UV - A) \circ W\|_F^2.$$

We first assume that W has no zero entries; later, we will remove this assumption by being slightly more careful.

LEMMA 4.2.

$$\mathrm{OPT} \leq \mathrm{poly}(n, \Delta)$$

PROOF. Set U and V to be the zero matrices. □

Let us expand the objective function in two ways. On the one hand:

$$\left\|(UV - A) \circ W\right\|_F^2 = \sum_{i=1}^{n} \left\|U^i V D_{W^i} - A^i D_{W^i}\right\|_2^2. \quad (1)$$

On the other hand:

$$\left\|(UV - A) \circ W\right\|_F^2 = \sum_{j=1}^{n} \left\|D_{W_j} U V_j - D_{W_j} A_j\right\|_2^2. \quad (2)$$

We can sketch (1) and (2) using Gaussian matrices $S' \in \mathbb{R}^{n\times t}$ and $S'' \in \mathbb{R}^{t\times n}$ as follows:

$$\sum_{i=1}^{n} \left\|U^i V D_{W^i} S' - A^i D_{W^i} S'\right\|_2^2.$$

and

$$\sum_{j=1}^{n} \left\|S'' D_{W_j} U V_j - S'' D_{W_j} A_j\right\|_2^2.$$

Denote, for $1 \leq i \leq n$, $P^i := V D_{W^i} S'$ and for $1 \leq j \leq n$ denote $Q_j := S'' D_{W_j} U$.

The crucial observation is that we can encode all P^i's and Q_j's using linear functions of $2krt$ variables, since W has rank r, and we can represent its rows/columns as linear combinations of r fixed rows/columns.

For fixed P^i's we define:

$$\widehat{U} := \operatorname*{argmin}_{U \in \mathbb{R}^{n\times k}} \sum_{i=1}^{n} \left\|U^i P^i - A^i D_{W^i} S'\right\|_2^2.$$

Similarly, for fixed Q_j's we define:

$$\widehat{V} := \operatorname*{argmin}_{V \in \mathbb{R}^{k\times n}} \sum_{j=1}^{n} \left\|Q_j V_j - S'' D_{W_j} A_j\right\|_2^2.$$

Let us use $\left\|\left(\widehat{U}\widehat{V} - A\right) \circ W\right\|_F^2$ as a proxy for the objective function. We need to argue that we can, in fact, minimize the new objective function efficiently, and that it gives a good approximation to the original objective function, if $t = \Theta(k/\varepsilon)$, with high probability.

Optimization .

We assume that all P^i's and Q_j's have the maximum rank k. Later we will show that, since W has no zero entries, this does not affect the quality of the solution found under this non-degeneracy constraint.

Assuming non-degeneracy of P^i's and Q_j's, we can express $\widehat{U}$ and $\widehat{V}$ as follows:

$$\widehat{U}^i = A^i D_{W^i} S' (P^i)^\top \left(P^i (P^i)^\top\right)^{-1}$$

and

$$\widehat{V}_j = \left(Q_j^\top Q_j\right)^{-1} Q_j^\top S'' D_{W_j} A_j. \quad (3)$$

Since the entries of P^i's and Q_j's are linear functions of $2krt$ variables, we can represent the entries of $\widehat{U}$ and $\widehat{V}$ as rational functions over $2krt$ variables and of degree $O(k)$ (we use Cramer's formula for that).

Finally, we can represent $\left\|\left(\widehat{U}\widehat{V}-A\right)\circ W\right\|_F^2$ as a rational function over $2krt$ variables of degree $O(kn)$.

We can minimize the objective function using the algorithm for checking the feasibility of a system of polynomial inequalities from Theorem 2.2, together with a binary search over the value of the objective function. Each iteration of the binary search takes time

$$(\#\text{degree of the polynomials})^{O(\#\text{variables})}\cdot\text{poly(input size)}.$$

Since the degree is $O(nk)$, the number of variables is $O(rkt)=O(k^2r/\varepsilon)$, and the input size is $\text{poly}(n,\log(\Delta/\delta))$, the running time of a single iteration of the binary search is

$$n^{O(k^2r/\varepsilon)}\cdot\log^{O(1)}(\Delta/\delta).$$

Finally, to perform the binary search, we need $O(\log(n\Delta/\tau)/\varepsilon)$ iterations to check for the existence of a solution with cost at most $(1+\varepsilon)\text{OPT}+\tau$ (assuming that sketching and non-degeneracy constraints on P^i's and Q_j's increase the cost to at most $(1+\varepsilon)\text{OPT}+\tau$, which we will show later), since, by Lemma 4.2, $\text{OPT}\le\text{poly}(n,\Delta)$. The overall running time is thus: $n^{O(k^2r/\varepsilon)}\cdot\log^{O(1)}\frac{\Delta}{\delta\tau}$.

Near-optimality .

Here we show that one can set $t=O(k/\varepsilon)$ so that, with high probability,

$$\min_{P^i,Q_j}\left\|\left(\widehat{U}\widehat{V}-A\right)\circ W\right\|_F^2\le(1+\varepsilon)\cdot\min_{U,V}\left\|(UV-A)\circ W\right\|_F^2+\tau'. \tag{4}$$

for every $\tau'>0$. This, together with the above discussion about the optimization procedure, concludes the analysis of the algorithm.

As such, (4) follows from Theorem 3.1. Indeed, if the optimal solution is U^*V^*, then set $Q_j:=S''D_{W_j}U^*$. Q_j may be degenerate, but, since W has no zero entries, we can perturb U^* by an arbitrarily small amount to make Q_j non-degenerate (with probability one over S''). Then, with high probability over S'', we have

$$\left\|\left(U^*\widehat{V}-A\right)\circ W\right\|_F^2\le(1+\varepsilon)\cdot\left\|(U^*V^*-A)\circ W\right\|_F^2+\tau' \tag{5}$$

for an arbitrarily small $\tau'>0$. Similarly, we can set $P^i=\widehat{V}D_{W^i}S'$ (again, it can be degenerate, but the same argument as above for Q_j applies), which gives, with high probability over S',

$$\left\|\left(\widehat{U}\widehat{V}-A\right)\circ W\right\|_F^2\le(1+\varepsilon)\cdot\left\|\left(U^*\widehat{V}-A\right)\circ W\right\|_F^2+\tau' \tag{6}$$

for an arbitrarily small $\tau'>0$.

Combining (5) and (6), we are done.

4.1 Handling Weight Matrices With Zero Entries

Here we prove the version of Theorem 4.1 for the case when W is allowed to have zero entries. Let us first see what breaks in the previous argument.

What does not work anymore is that (after a small perturbation) $Q_j=S''D_{W_j}U$ and $P^i=VD_{W^i}S'$ can be assumed to have the maximum possible rank k. Nevertheless, we can assume that every Q_j has rank equal to

$$t_j=\min(k,\text{the number of non-zero entries of } W_j).$$

Moreover, we can assume that the first t_j columns of Q_j are linearly-independent. A similar argument applies to P^i's as well.

The above argument allows us to express $\widehat{U}$ and $\widehat{V}$ as before, but instead of Q_j we use the first t_j columns of Q_j (and, similarly for the P^i's).

5. MULTIPLICATIVE APPROXIMATION

In order to get a genuine multiplicative $(1+\varepsilon)$-approximation, we need to lower bound OPT–provided that it is not equal to zero–which would allow us to set $\tau\le\varepsilon\cdot\text{OPT}$ in the algorithm from the previous section.

We do this for the following optimization problem:

$$\min_{\substack{U,V:\\ \|UV\|_F^2\le(\Delta/\delta)^{\text{poly}(n)}}}\|(UV-A)\circ W\|_F^2.$$

Note that we assume $\|UV\|_F$ has an upper bound, as otherwise we cannot write down U and V using $\text{poly}(n)$ bits.

Using the approach outlined above, one can write down a rational function $p(x_1,\ldots,x_l)/q(x_1,\ldots,x_l)$ such that:

- $l=O(k^2r/\varepsilon)$;
- for every x such that $q(x)\ne0$, one has $p(x)/q(x)\ge\text{OPT}$;
- for every $\tau'>0$, there exists x^* such that $p(x^*)/q(x^*)\le(1+\varepsilon)\text{OPT}+\tau'$;
- both p and q are homogeneous, and their degrees are $O(kn)$;
- the coefficients of p and q are integers with absolute values at most $(\Delta/\delta)^{\text{poly}(n)}$;
- $q(x)=\prod_{i=1}^{2n}g_i^2(x)$, where every $g_i(x)$ is the determinant polynomial.

Only the fifth item needs an explanation. If sketch matrices S' and S'' were integer, then the last item would hold automatically (by scaling up all the coefficients). But, in reality, S' and S'' are Gaussian matrices. Fortunately, one can show that it is possible to discretize them up to $\pm1/\text{poly}(n)$ so that the multiple regression theorem (Theorem 3.1) still goes through. Indeed, this just follows from the fact that discretization to $\pm1/\text{poly}(n)$ preserves condition number and subspace embeddings (since one argues about preserving lengths of unit vectors), and approximate matrix product properties used in that theorem.

Lower Bound .

We use the same way explained in section 4 to create variables, write down the system in a small number of variables, and also create some "$\ne0$" constraints. It will generate $2n$ determinant polynomials, which are defined in the following way,

$$\begin{aligned}g_i(x)&=\det((SD_{W_i}U)_{P_i}^\top(SD_{W_i}U)_{P_i}),\forall i\in[n]\\ g_{i+n}(x)&=\det((VD_{W^i}S)^{Q_i}((VD_{W^i}S)^{Q_i}))^\top),\forall i\in[n]\end{aligned}$$

where P_i and Q_i are maximal linearly indepdent subsets. Then we can write down the following optimization problem,

$$\begin{aligned} \min_{x\in\mathbb{R}^l} \quad & p(x)/q(x) \\ s.t. \quad & g_i^2(x) \neq 0, \forall i \in [2n], \\ & q(x) = \prod_{i=1}^{2n} g_i^2(x) \end{aligned}$$

To lower bound $p(x)/q(x)$, let us introduce a new variable y. Lower bounding $p(x)/q(x)$ is equivalent to lower bounding $p(x)y$ subject to $q(x)y - 1 = 0$. By assumption $\|U\|_F^2, \|V\|_F^2 \leq (\Delta/\delta)^{\mathrm{poly}(n)}$[4], we know the upper bound of all the variables we created except for y. In order to give an upper bound for y, it suffices to show a lower bound for $q(x)$. By definition, $q(x)$ is the square of the product of all the determinant polynomials. Thus, for every determinant polynomial $g_i(x)$, we need to show that if $g_i^2(x)$ is nonzero, then it is at least something.

Define B to be an ℓ_2-ball with bounded radius, e.g., $B = \{x \in \mathbb{R}^l | \sum_{i=1}^{l} x_i^2 \leq (\Delta/\delta)^{\mathrm{poly}(n)}\}$. By [9], we know that B is a closed and bounded semi-algebraic set. Thus B is also compact. Let x^* denote the optimal solution of the original problem $\min_{x\in\mathbb{R}^l} g_i^2(x)$ when all variables are bounded. Because the radius of the ball B is large enough, $x^* \in B$. Define $T_1 = \{x \in \mathbb{R}^l | g_i(x) \geq 0\}$ and let $T = T_1 \cap B$. By definition (see [9]), T_1 is a basic closed semi-algebraic set. Thus, the intersection of T_1 and B is a semi-algebraic set with a finite number of connected components. Because B is compact and T_1 is closed, each of these connected components is compact. There must exist one compact connected component C which contains the optimal soltuion x^*. Applying Theorem 2.3 on system $\{T, C, g_i^2(x)\}$, we conclude that if $g_i^2(x)$ is not zero, then it is at least

$$\begin{aligned} \left((\Delta/\delta)^{\mathrm{poly}(n)}\right)^{-k^{O(l)}} &= (\Delta/\delta)^{-\mathrm{poly}(n)2^{\widetilde{O}(l)}} \\ &= (\Delta/\delta)^{-\mathrm{poly}(n)} \end{aligned}$$

which immediately gives us an upper bound for y,

$$y \leq ((\Delta/\delta)^{\mathrm{poly}(n)})^n = (\Delta/\delta)^{\mathrm{poly}(n)}.$$

Now, we are able to show a lower bound for $p(x)y$. Define

$$T_1 = \{x \in \mathbb{R}^l, y \in \mathbb{R} | \prod_{i=1}^{m} g_i^2(x)y - 1 = 0\}$$

Define B to be a bounded ball over $l+1$ variables,

$$B = \{(x, y) \in \mathbb{R}^{l+1} | \sum_{i=1}^{l} x_i^2 + y^2 \leq (\Delta/\delta)^{\mathrm{poly}(n)}\}$$

By [9], B is a closed and bounded semi-algebraic set. Thus B is also compact. Define $T = T_1 \cap B$. Let (x^*, y^*) denote the optimal solution of $\min_{(x,y)\in T_1} p(x)y$, then (x^*, y^*) is also the optimal solution of $\min_{(x,y)\in T} p(x)y$, because all variables are bounded and the radius of the ball is large enough.

[4]A priori, we know only that $\|UV\|_F^2$ is bounded. But we can get that each of the matrices is bounded by taking an optimal solution and orthonormalizing one of the matrices.

By definition (see [9]), T_1 is a basic closed semi-algebraic set. Thus, the intersection of T_1 and B is a semi-algebraic set with a finite number of connected components. Because B is compact and T_1 is closed, each of the connected components is compact.

There must exist a compact connected component that contains the optimal solution (x^*, y^*). Let C denote that component. Applying Theorem 2.3 on system $\{T, C, p(x)y\}$, where the number of constraints is bounded by $O(1)$, the maximum coefficient of absolute value is bounded by $(\Delta/\delta)^{\mathrm{poly}(n)}$, the maximum degree is bounded by $O(nk)$, the number of variables is bounded by $l = O(rk^2/\varepsilon)$, we conclude that if the minimum cost is not zero, then it is at least

$$\begin{aligned} & ((\Delta/\delta)^{\mathrm{poly}(n)})^{-n^{O(rk^2/\varepsilon)}} \\ = \quad & \exp\left(-n^{O(k^2 r/\varepsilon)} \log^{O(1)}\left(\frac{\Delta}{\delta}\right)\right). \end{aligned} \tag{7}$$

Hence, OPT is at least (7) as well.

Plugging $\tau \ll \varepsilon \cdot$ (7) $\leq \varepsilon$OPT into the algorithm from the previous section with an additional constraint $\|\widehat{U}\widehat{V}\|_F^2 \leq \left(\frac{\Delta}{\delta}\right)^{\mathrm{poly}(n)}$, we do binary search to narrow down the range of $[\Lambda^-, \Lambda^+]$ until we reach Λ. During the jth step of binary search, we use Theorem 2.2 to check if the following semi-algebraic set is empty or not,

$$S = \{x \in \mathbb{R}^l | p(x) \leq \Lambda_j^+ q(x), p(x) \geq \Lambda_j^- q(x), q(x) \neq 0\}$$

where Λ_1^- is initialized to be τ and Λ_1^+ is initialized to be $\mathrm{poly}(n, \Delta)$. We obtain the following theorem.

THEOREM 5.1. *Given $A, W \in \mathbb{R}^n$, $1 \leq k \leq n$ and $0 < \varepsilon, \tau < 0.1$ such that:*

- $\mathrm{rank}(W) = r$;
- *all the non-zero entries of A and W are multiples of $\delta > 0$;*
- *all the entries of A and W are at most $\Delta > 0$ in absolute value,*

one can output a number Λ in time $n^{O(k^2 r/\varepsilon)} \cdot \log^{O(1)} \frac{\Delta}{\delta\tau}$ such that $\mathrm{OPT} \leq \Lambda \leq (1+\varepsilon)OPT$, where

$$\mathrm{OPT} = \min_{\substack{U\in\mathbb{R}^{n\times k}, V\in\mathbb{R}^{k\times n} \\ \|UV\|_F^2 \leq \left(\frac{\Delta}{\delta}\right)^{\mathrm{poly}(n)}}} \|(UV - A) \circ W\|_F^2.$$

6. RECOVERING THE SOLUTION ITSELF

Here we show how to recover an approximate solution, not only the value of OPT.

The idea is to recover the entries of U and V one by one and use the algorithm from the previous section for the corresponding decision problem. We initialize the semi-algebraic set to be

$$S = \{x \in \mathbb{R}^l | q(x) \neq 0, p(x) \leq \Lambda q(x)\}$$

We start by recovering the first entry of U. We perform the binary search to localize the entry within an interval of length δ', which takes $\mathrm{poly}(n) \cdot \log\left(\frac{\Delta}{\delta\delta'}\right)$ invocations of the decision algorithm. For each step of binary search, we use 2.2 to determine if the following semi-algebraic set S is empty or not,

$$S \cap \{U_{1,1}(x) \geq \widehat{U}_{1,1}^-, U_{1,1}(x) \leq \widehat{U}_{1,1}^+\}$$

After that, we declare the first entry of U to be any point in this interval. This changes the cost of the solution by at most $\delta' \cdot \left(\frac{\Delta}{\delta}\right)^{\mathrm{poly}(n)}$. Then, we add an equality constraint that fixes the entry of $\widehat{U}$ to this value, and add a new constraint into S permanently, e.g. $S \leftarrow S \cap \{U_{1,1}(x) = \widehat{U}_{1,1}\}$. Next, we repeat the same with the second entry of U and so on.

This allows us to recover a *solution* of cost at most $(1+\varepsilon)\mathrm{OPT} + \tau$ in time

$$n^{O(k^2 r/\varepsilon)} \cdot \log^{O(1)}\left(\frac{\Delta}{\delta\tau}\right).$$

7. ADVERSARIAL MATRIX COMPLETION

Here we prove the following theorem.

THEOREM 7.1. *Let $B \in \mathbb{R}^{n\times n}$ be a rank-k matrix with entries that are multiples of $\delta > 0$ bounded by $\Delta > 0$.*

Let $1 \le r \le n$ be an integer parameter.

Let C be an $n \times n$ matrix, where in every column there are at most r question marks *and other entries are equal to the corresponding entries of B.*

Then, there is an algorithm that:

- *receives C as an input;*
- *outputs a rank-k matrix that is τ-close to C in Frobenius norm (restricted to the entries that are not replaced with a question mark);*
- *has running time $n^{O(k^2 r)} \cdot \log^{O(1)}\left(\frac{\Delta}{\delta\tau}\right)$.*

A naive attempt is to use the algorithm from the previous section with W equal to 0 for the missing entries and to 1 for the "surviving" entries. Unfortunately, setting W like this does not work, since W may not have small rank.

We fix it by proving the following lemma.

LEMMA 7.2. *For every set system $z_1, z_2, \ldots, z_n \subseteq [n]$ with $|z_j| \le r$ there exists a rank-$(2r+1)$ integer matrix W such that, for every $1 \le i \le n$, the entry in the i-th column and j-th row,*

$$W_i^j \begin{cases} = 0 & \text{if } j \in z_i \\ > 0 & \text{if } j \in [n]\backslash z_i \end{cases}$$

Moreover, all the entries of W are bounded by $n^{O(r)}$ in absolute value.

Recall that for the i-th column of W, z_i denotes the set of row indices that have zero entry. For the i-th column of W, define a polynomial $p_i(x)$ such that

$$p_i(x) = \prod_{\ell \in z_i} (x-j)^2 = \sum_{\ell=1}^{2|z_i|+1} a_{i\ell} x^{\ell-1}$$

where all $a_{i\ell}$ are integers. Since $2|z_i|+1$ is at most $2r+1$, we can also think of $p_i(x)$ as a degree $2r$ polynomial,

$$p_i(x) = \sum_{\ell=1}^{2r+1} a_{i\ell} x^{\ell-1}$$

Then, we can use $a_{i\ell}$ to create a basis $T \in \mathbb{R}^{n\times(2r+1)}$ which has rank at most $2r+1$. Let T_ℓ^i denote the entry of the i-th row and ℓ-th column, then set $T_\ell^i = a_{i\ell}, \forall i \in [n], \forall \ell \in [2r+1]$.

Let T_ℓ denote the ℓ-th column of matrix T, $\forall \ell \in [2r+1]$. To guarantee a linear combination of those $2r+1$ columns always outputs a nonnegative vector, we can just choose coefficients $1, x, x^2, \cdots, x^{2r}$. Let S denote a set of all possible vectors formed by the following linear combination,

$$T(x) = \sum_{\ell=1}^{2r+1} x^{\ell-1} T_\ell \in \mathbb{R}^{n\times 1}$$

Moreover, we have

$$\begin{aligned} T(x) &= \sum_{\ell=1}^{2r+1} x^{\ell-1} T_\ell = \sum_{\ell=1}^{2r+1} x^{\ell-1} \begin{bmatrix} a_{1\ell} & a_{2\ell} & \cdots & a_{n\ell} \end{bmatrix}^\top \\ &= \begin{bmatrix} \sum_{\ell=1}^{2r+1} x^{\ell-1} a_{1\ell} & \sum_{\ell=1}^{2r+1} x^{\ell-1} a_{2\ell} & \cdots & \sum_{\ell=1}^{2r+1} x^{\ell-1} a_{n\ell} \end{bmatrix}^\top \\ &= \begin{bmatrix} p_1(x) & p_2(x) & \cdots & p_n(x) \end{bmatrix}^\top, \end{aligned}$$

where the second equality follows by the definition of T_ℓ and the last equality follows by the definition of $p_i(x)$. Let $T^j(x)$ denote the jth entry of column vector $T(x) \in \mathbb{R}^{n\times 1}$. For any column vector W_i, we assign $T(i)$ to it,

$$W_i \leftarrow T(i)$$

which has the following property: for every $1 \le i \le n$, the entry at the i-th column and $j-$th row satisfies

$$W_i^j = \begin{cases} T^j(i) = p_i(j) = 0 & \text{if } j \in z_i, \\ T^j(i) = p_i(j) > 0 & \text{if } j \in [n]\backslash z_i. \end{cases}$$

Note that for any column vector W_i, we know that $W_i \in S$ and $\mathrm{rank}(S) = 2r+1$. Thus, $\mathrm{rank}(W) = 2r+1$.

8. FEW DISTINCT COLUMNS

In this section we show how to improve the running time from $n^{\mathrm{poly}(k,r,1/\varepsilon)}$ to $\mathrm{poly}(n)\cdot 2^{\mathrm{poly}(k,r,1/\varepsilon)}$ under the following assumptions: (1) $\Delta = \mathrm{poly}(n)\delta$; and (2) $\|UV\|_F^2 \le (\Delta/\delta)^{n^\gamma}$, for an arbitrarily small constant $\gamma > 0$. In Section 8.1, as a warmup we assume that W has r *distinct columns* and r *distinct rows*, while in Section 8.2 and 8.3 we give our main result assuming only that W has r *distinct columns*.

A crucial observation is that the term $n^{\mathrm{poly}(k,r,1/\varepsilon)}$ shows up in the "rank-r" algorithm due to the fact that the degree of polynomials we optimize is $\Omega(n)$. The reason for this is that entries of $\widehat{U}$ and $\widehat{V}$ are rational functions with $\Omega(n)$ potentially different denominators. When we combine them in a single rational function that corresponds to $\|(\widehat{U}\widehat{V} - A) \circ W\|_F^2$, we get a denominator of degree $\Omega(n)$.

8.1 r distinct rows and columns

In this subsection, as a warmup we assume that W has r distinct rows *and* r distinct columns. Then, we get rid of the dependence on n in the degree. Indeed, now we have only $2r$ distinct denominators (w.l.o.g., assume the first r columns are distinct and the first r rows are distinct),

$$\begin{aligned} g_i(x) &= \det((SD_{W_i}U)_{P_i}^\top (SD_{W_i}U)_{P_i}), \forall i \in [r] \\ f_i(x) &= \det((VD_{W^i}S)^{Q_i}((VD_{W^i}S)^{Q_i})^\top), \forall i \in [r] \end{aligned}$$

where P_i and Q_i are maximal linearly independent subsets.

Then we can write down the following optimization problem,

$$\begin{aligned} \min_{x\in\mathbb{R}^l} \quad & p(x)/q(x) \\ s.t. \quad & g_i^2(x) \neq 0, f_i^2(x) \neq 0, \forall i \in [r], \\ & q(x) = \prod_{i=1}^{r} g_i^2(x) f_i^2(x) \end{aligned}$$

where $q(x)$ has degree $O(rk)$, the maximum coefficient in absolute value is $(\Delta/\delta)^{O(rkn^\gamma)}$, and the number of variables $O(rk^2/\varepsilon)$. Using the same argument as in the rank-r case and applying Theorem 2.3, we can achieve the following minimum nonzero cost: $(\Delta/\delta)^{-n^\gamma 2^{\widetilde{O}(rk^2/\varepsilon)}}$. Now, using the approach described in section 6, we can find the solution in time

$$(\mathrm{nnz}(A)+\mathrm{nnz}(W))n^\gamma + n2^{\widetilde{O}(rk^2/\varepsilon)}\log^{O(1)}(\Delta/\delta\tau)$$

within a multiplicative factor of $1+\varepsilon$ and additive factor of τ.

One can adjust the lower bound on OPT accordingly, and conclude that an algorithm for approximating OPT within a multiplicative factor of $1+\varepsilon$ can be done in time

$$(\mathrm{nnz}(A)+\mathrm{nnz}(W))n^\gamma + n2^{\widetilde{O}(rk^2/\varepsilon)}\log^{O(1)}(\Delta/\delta)$$

where

$$\mathrm{OPT} = \min_{\substack{U,V \\ \|UV\|_F^2 \leq (\Delta/\delta)^{n^\gamma}}} \|(UV-A)\circ W\|_F^2.$$

8.2 r distinct columns, OPT $= 0$

This section, we explain how to find the solution to the weighted low rank approximation problem when W has at most r distinct columns and $OPT = 0$.

The key observation is that for any matrix A and $W \in \mathbb{R}_{\geq 0}^{n\times n}$, if there exists a solution of an $n\times k$ matrix U and a $k\times n$ matrix V such that,

$$\|W\circ(UV-A)\|_F^2 = OPT,$$

then there exists another matrix $W' \in \{0,1\}^{n\times n}$ such that

$$\|W'\circ(UV-A)\|_F^2 = OPT$$

where $W'_{i,j} = 0$ if $W_{i,j} = 0$ and $W'_{i,j} = 1$ if $W_{i,j} > 0$.

The above observation states that modifying the weight matrix to be Boolean does not change the optimal cost. Since W has r distinct columns, now that it is Boolean it has at most 2^r distinct rows. Indeed, each row of W is completely determined after fixing its values on the r distinct columns, and there are only 2^r possibly fixings. W.l.o.g. we assume that the first r columns are distinct. Instead of having at most $2r$ distinct denominators as in Section 8.1, we have at most $r+2^r$ distinct denominators. We create l variables for $\{SD_{W_1}U,\cdots,SD_{W_r}U\}$. Then we can write down $\widehat{V}$ in the following way,

$$\begin{aligned} & \widehat{V}_j = (SD_{W_j}U)^\dagger \cdot SD_{W_j}A_j \\ = \quad & \left(((SD_{W_j}U)_{P_i})^\top \cdot (SD_{W_j}U)_{P_i}\right)^{-1} \cdot ((SD_{W_j}U)_{P_i})^\top SD_{W_j}A_j \end{aligned}$$

where P_i denotes a subset of rows. For all D_{W_j}s in the group Z_i, they share the same P_i, where for any $j\in Z_i$, $D_{W_j} = D_{W_i}$.

Thus to express $\widehat{V}$, there are only r distinct denominators $g_i(x)$ which are the determinants of $((SD_{W_j}U)_{P_i})^\top \cdot (SD_{W_j}U)_{P_i}$. In order to remove these denominators, we can create a new variable x_{l+i} and add a new equality constaint $g_i(x)x_{l+i}-1=0$. Therefore, we do not have any denominators in $\widehat{V}$.

W.l.o.g., we assume that the first 2^r rows of W are distinct. Using $\widehat{V}$ we can write down $\widehat{U}$ in the following way,

$$\begin{aligned} \widehat{U}^j &= A^j D_{W^j}(\widehat{V}D_{W^j})^\dagger \\ &= A^j D_{W^j}((\widehat{V}D_{W^j})^{Q_i})^\top \left((\widehat{V}D_{W^j})^{Q_i}((\widehat{V}D_{W^j})^{Q_i})^\top\right)^{-1} \end{aligned}$$

where Q_i denotes a subset of columns, where all D_{W^j}s in the group Z'_i can share the same Q_i, and for any $j\in Z'_i$, $D_{W^j} = D_{W^i}$.

Thus, to express $\widehat{U}$, there are only 2^r distinct denominators $f_j(x)$ which are the determinants of the matrices $(\widehat{V}D_{W^j})^{Q_i}((\widehat{V}D_{W^j})^{Q_i})^\top$.

Finally, we can use a small number of variables to represent all the entries of $\widehat{U}$ and $\widehat{V}$. It allows us to write the following optmization problem,

$$\begin{aligned} \min_{x\in\mathbb{R}^{l+r}} \quad & p(x)/q(x) \\ s.t. \quad & g_i(x)x_{l+i}-1=0, \forall i\in[r] \\ & f_j^2(x)\neq 0, \forall j\in[2^r] \\ & q(x) = \prod_{j=1}^{2^r} f_j^2(x) \end{aligned}$$

where $q(x)$ has degree $O(2^r k^2)$, maximum coefficients bounded in absolute value by $(\Delta/\delta)^{O(2^r kn^\gamma)}$, $l = O(rk^2/\epsilon)$ and the number of variables $O(rk^2/\varepsilon)$. Using the same argument as for the rank-r case and applying Theorem 2.3, we have the following minimum nonzero cost: $(\Delta/\delta)^{-n^\gamma 2^{\widetilde{O}(r^2k^2/\varepsilon)}}$.

Using the approach described in section 6, we can find the solution achieving zero cost in time

$$(\mathrm{nnz}(A)+\mathrm{nnz}(W))n^\gamma + n2^{\widetilde{O}(r^2k^2/\varepsilon)}\log^{O(1)}(\Delta/\delta)$$

8.3 r distinct columns, OPT $\neq 0$

Lower Bound .

Let U,V denote the optimal solution A,W, which gives nonzero cost. We can modify W to a new matrix W' in the following sense, $W'_{i,j} = \delta$ if $W_{i,j}\neq 0$ and $W'_{i,j} = 0$ if $W_{i,j} = 0$. Then we know that

$$\|W'\circ(UV-A)\|_F^2 \leq \|W\circ(UV-A)\|_F^2 \neq 0$$

Note that if problem A,W' has a zero cost solution, then problem A,W also has a zero cost solution, which contradicts our assumption in this section. Thus problem A,W' does not have a zero cost solution. It follows from previous sections that the minimum nonzero cost of $\min_{U,V}\|W'\circ(UV-A)\|_F^2$ is at least

$$(\Delta/\delta)^{-n^\gamma 2^{\widetilde{O}(r^2k^2/\varepsilon)}}.$$

Let U',V' denote the optimal solution of problem A,W'. Thus we have

$$\|W\circ(UV-A)\|_F^2 \geq \|W'\circ(UV-A)\|_F^2 \geq \|W'\circ(U'V'-A)\|_F^2$$

which is at least $(\Delta/\delta)^{-n^\gamma 2^{\widetilde{O}(r^2k^2/\varepsilon)}}$.

Algorithm .

For notational convenience, let $\delta = 1$. Then each entry of the input weight matrix W' is in $\{0, 1, 2, \cdots, \text{poly}(n)\}$. For each entry $W'_{i,j}$, we round it to the smallest $(1+\varepsilon)^x$ such that $W'_{i,j} \leq (1+\varepsilon)^x$ where x is an integer. Because W' is bounded, the total number of choices for the power x is $O(\log(n)/\varepsilon)$. Define W to be the matrix after rounding. Define OPT to be $\min_{U,V} \|W' \circ (UV - A)\|_F^2$. Then W has the following properties

1. W has r distinct columns,
2. W has $R := (\log(n)/\varepsilon)^{O(r)}$ distinct rows,
3. $\text{OPT} \leq \min_{U,V} \|W \circ (UV - A)\|_F^2 \leq (1+\varepsilon)^2 \text{OPT}$.

We prove the above three properties one by one.

The rounding is a deterministic procedure: if two values are the same in W', then they are the same in W. Hence, Property 1 holds.

To prove Property 2, take the r distinct columns $i_1, ..., i_r$. Then every other column can be labeled j in $\{i_1, ..., i_r\}$. If you fix the values on entries $i_1, ..., i_r$ in a row, this fixes the values on every other column. So the number of distinct rows is the number of fixings to the values on $i_1, ..., i_r$. Each entry has $\log_{1+\varepsilon} \text{poly}(n) = O((\log n)/\varepsilon)$ possibilities, so there are $O((\log n)/\varepsilon)^r$ distinct rows.

Because of the rounding procedure, each $W'_{i,j}$ satisifies that $W'_{i,j} \leq W_{i,j} \leq (1+\varepsilon)W'_{i,j}$, which implies Property 3.

We use the same approach as in Section 8.2 to create variables, write down the polynomial systems and add not equal constraints. Instead of having $r + 2^r$ distinct denominators, we have $r + R$, where $R = O((\log n)/\epsilon)^r$. We create $l = O(rk^2/\varepsilon)$ variables for $\{SD_{W_1}U, \cdots, SD_{W_r}U\}$, then we can write down $\widehat{V}$ with r distinct denominators $g_i(x)$. Each $g_i(x)$ is non-zero in an optimal solution using the perturbation argument in Section 4. We create new variables x_{i+l} to remove the denominators $g_i(x)$. Then the entries of $\widehat{V}$ are polynomials as opposed to rational functions. Using $\widehat{V}$ we can express $\widehat{U}$ with $R = (\log(n)/\varepsilon)^{O(r)}$ distinct denominators $f_i(x)$, which are also non-zero by using the perturbation argument in 4, and using that W has at most this number of distinct rows. Finally we can write the following optimization problem,

$$\begin{aligned} \min_{x \in \mathbb{R}^{l+r}} \quad & p(x)/q(x) \\ s.t. \quad & g_i(x)x_{l+i} - 1 = 0, \forall i \in [r] \\ & f_j^2(x) \neq 0, \forall j \in [R] \\ & q(x) = \prod_{j=1}^{R} f_j^2(x) \end{aligned}$$

We then determine if there exists a solution to the above semi-algebraic set in time $(k^2R)^{O(rk^2/\varepsilon)} = (\log(n)/\varepsilon)^{O(r^2k^2/\varepsilon)}$. Combining the binary search explained in section 5 and 6 with the lower bound we obtained, we can find the solution for the original problem in time

$$(\text{nnz}(A) + \text{nnz}(W))n^\gamma + n2^{\widetilde{O}(r^2k^2/\varepsilon)} \log^{O(1)}(\Delta/\delta).$$

Note that there is no $\log\log n$ in the exponent $2^{\widetilde{O}(r^2k^2/\varepsilon)}$ since either $r^2k^2/\varepsilon = o(\log n/\log\log n)$, in which case this term is dominated by $n^{1+\gamma}$, or $\log(r^2k^2/\varepsilon) = \Omega(\log\log n)$.

9. HARDNESS

The Maximum Edge Biclique problem [3] is defined as:
Input: An n by n bipartite graph G.
Output: A k_1 by k_2 complete bipartite subgraph of G.
Objective Function: Maximize $k_1 \cdot k_2$.

We use the Maximum Edge Biclique problem under the R4SAT assumption in [25], which extends the previous work done by Feige [22] under the R3SAT assumption. That hardness result [25] shows under the R4SAT assumption there exist two constants $\epsilon_1 > \epsilon_2 > 0$ such that no efficient algorithm is able to distinguish between bipartite graphs $G(U, V, E)$ with $|U| = |V| = n$ which have a clique of size $\geq (n/16)^2(1+\epsilon_1)$ and those in which all bipartite cliques are of size $\leq (n/16)^2(1+\epsilon_2)$. Using the reduction of [25], one can show there exists a constant c such that for any instance of R4SAT with $\widetilde{n}$ variables and $c\widetilde{n}$ clauses, the corresponding bipartite graph G created in [25] has at least tn^2 edges with large probability, for a constant t, e.g. $t = 9/10$.

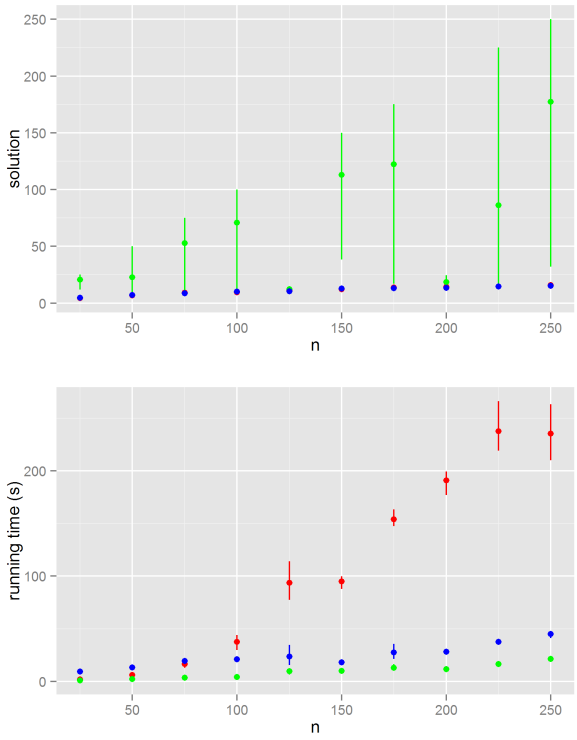
To construct a weighted low-rank appoximation problem from a given bipartite graph, for a given bipartite graph $G(U, V, E)$, we generate the matrix A and W as in [24]: $A_{ij} = 1$ if edge $(U_i, V_j) \in E$, $A_{ij} = 0$ if edge $(U_i, V_j) \notin E$. $W_{ij} = 1$ if edge $(U_i, V_j) \in E$, $W_{ij} = \text{poly}(n)$ if edge $(U_i, V_j) \notin E$. One can then show if there exists a biclique in G such the number of remaining edges is at most $tn^2 - (n/16)^2(1+\epsilon_1)$, then the solution to $\min \|W \circ \widehat{A} - W \circ A\|_F^2$ has cost at most $tn^2 - (n/16)^2(1+\epsilon_1)$. On the other hand, if there does not exist a biclique that has more than $(n/16)^2(1+\epsilon_2)$ edges, which leads to the number of remaining edges being at least $tn^2 - (n/16)^2(1+\epsilon_2)$, then any solution to $\min \|W \circ \widehat{A} - W \circ A\|_F^2$ has cost at least $tn^2 - (n/16)^2(1+\epsilon_2)$.

10. EXPERIMENTS

We experimentally evaluate the algorithm from Section 4. We stress that this evaluation is very preliminary and mostly serves to demonstrate that the algorithm is less impractical than it may look like. In particular, we only try the smallest non-trivial case $k = 1$ and $r = 2$. We implement the algorithm in Wolfram Mathematica using the built-in "industrial-quality" polynomial solver. The implementation is short enough to provide it fully below. We compare it with the naive algorithm which encodes the target matrices using $O(kn)$ variables and minimizes the objective function (which is a degree-2 polynomial) using the same built-in solver.

First, we set A to be the all-ones $n \times n$ matrix except the first column, which we sample from i.i.d. centered Gaussians with large (distinct) standard deviations. We set W to be the all-ones $n \times n$ matrix except the first column, which we set to be the inverses of the corresponding standard deviations. The best unweighted rank-1 approximation is essentially to take the first column of A and set everything else to zero, while the best *weighted* approximation with weights W is essentially the all-ones $n \times n$ matrix. See Figure 10 for the results for the naive algorithm, our algorithm with the sketch of size $t = 1$, and our algorithm with $t = 2$. Setting $t = 1$ gives sub-optimal results, while $t = 2$ consistenly gives near-optimal solutions. At the same time, our algorithm scales much better than the naive solution. Let us point out that the naive solution is able to exploit the structure of W indirectly by using the built-in Mathematica's solver (if one sets W to a Gaussian random matrix, the time blows up).

Next, we choose A to be a random rank-1 matrix plus



noise, and let W be a random non-negative rank-2 matrix. The magnitude of noise is inversely proportional to the corresponding entry of W. On this family of instances even if we set $t = 3$ the accuracy ends up being around 1.3. On the other hand, larger values of t blow-up the running time of our algorithm by quite a bit. Despite that, when n increases, our algorithm still scales better than the naive algorithm. Namely, when n changes from 10 to 100, the running time of the naive algorithm grows by three orders of magnitude, while the running time of our algorithm grows by a factor of two. And around $n = 100$ running times roughly match. Unfortunately, we were not able to run the algorithms for $n > 100$, since they require lots of RAM.

We conclude that our algorithms have potential to be practical, although that might require opening the polynomial solver and combining exact algorithm and heuristics.

```
GenGaussian[m_,n_] :=
    Table[RandomVariate[NormalDistribution[]], {m}, {n}];

FindBasis[W_] :=
    Module[{Wt},
       Wt = Select[Orthogonalize[W], Norm[#1] > 0.01 &];
       Transpose[Wt.Transpose[W]]];

FastSolve[A_,W_,k_,t_] :=
    Module[
    {decompCols, decompRows, r, Scols, Srows, xx, yy,
      U, V, res, n},
    n = Length[A];
    r = MatrixRank[W];
    decompCols = FindBasis[Transpose[W]];
    decompRows = FindBasis[W];
    Scols = GenGaussian[t, n];
    Srows = GenGaussian[n, t];
    xx = Array[x, {r, t, k}];
    yy = Array[y, {r, k, t}];
    allReal = Map[Element[#1, Reals] &,
      Flatten[{xx, yy}]];
    U=Assuming[allReal,
      Table[LeastSquares[
            Transpose[(decompRows.yy)[[i]]],
            ((A * W).Srows)[[i, All]]], {i, 1, n}]];
    V=Assuming[allReal,
      Transpose[Table[LeastSquares[
            (decompCols.xx)[[j]],
            (Scols.(A * W))[[All, j]]], {j, 1, n}]]];
    Minimize[Total[Flatten[((U.V - A) * W)^2]],
      Flatten[{xx, yy}]]];
```

Acknowledgments: The authors would to like thank Saugata Basu, Xue Chen, Uriel Feige, Mika Göös, Russell Impagliazzo, J. M. Landsberg, Ankur Moitra, Daniel Perrucci, Eric Price, James Renegar, Tselil Schramm, Elias Tsigaridas, Ryan Williams, and David Zuckerman for useful discussions.

11. REFERENCES

[1] D. Achlioptas, A. Fiat, A. R. Karlin, and F. McSherry. Web search via hub synthesis. In *FOCS*, 2001.

[2] D. Achlioptas and F. McSherry. On spectral learning of mixtures of distributions. In *COLT*, 2005.

[3] C. Ambühl, M. Mastrolilli, and O. Svensson. Inapproximability results for maximum edge biclique, minimum linear arrangement, and sparsest cut. *SIAM Journal on Computing*, 40(2):567–596, 2011.

[4] S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization - provably. In *STOC*, 2012.

[5] Y. Azar, A. Fiat, A. R. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *STOC*, 2001.

[6] A. Basu, M. Dinitz, and X. Li. Computing approximate PSD factorizations. *CoRR*, abs/1602.07351, 2016.

[7] S. Basu. Algorithms in real algebraic geometry: a survey. *arXiv preprint arXiv:1409.1534*, 2014.

[8] S. Basu, R. Pollack, and M. Roy. On the combinatorial and algebraic complexity of quantifier elimination. *J. ACM*, 43(6):1002–1045, 1996.

[9] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*, volume 20033. Springer, 2005.

[10] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. In *NIPS*, 2001.

[11] J. Bochnak, M. Coste, and M.-F. Roy. *Géométrie algébrique réelle*, volume 12. Springer Science & Business Media, 1987.

[12] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[13] S. Chatterjee. Matrix estimation by universal singular value thresholding. *pre-print*, 2012. `http://arxiv.org/abs/1212.1247`.

[14] Z. Chen and J. J. Dongarra. Condition numbers of gaussian random matrices. *SIAM Journal on Matrix Analysis and Applications*, 27(3):603–620, 2005.

[15] K. L. Clarkson and D. P. Woodruff. Numerical linear algebra in the streaming model. In *STOC*, 2009.

[16] K. L. Clarkson and D. P. Woodruff. Low rank approximation and regression in input sparsity time. In *STOC*, 2013.

[17] K. L. Clarkson and D. P. Woodruff. Input sparsity and hardness for robust subspace approximation. In *FOCS*, 2015.

[18] M. B. Cohen, S. Elder, C. Musco, C. Musco, and M. Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *STOC*, 2015.

[19] P. Drineas, A. M. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering large graphs via the singular value decomposition. *Machine Learning*, 56(1-3):9–33, 2004.

[20] P. Drineas, I. Kerenidis, and P. Raghavan. Competitive recommendation systems. In *STOC*, 2002.

[21] A. V. Evfimievski, R. Fagin, and D. P. Woodruff. Epistemic privacy. *J. ACM*, 58(1):2, 2010.

[22] U. Feige. Relations between average case complexity and approximation complexity. In *STOC*, 2002.

[23] D. Feldman, v Melanie Schmidt, and C. Sohler. Turning big data into tiny data: Constant-size coresets for k-means, PCA and projective clustering. In *SODA*, 2013.

[24] N. Gillis and F. Glineur. Low-rank matrix approximation with weights or missing data is np-hard. *SIMAX*, 32(4):1149–1165, 2011.

[25] A. Goerdt and A. Lanka. An approximation hardness result for bipartite clique. *ECCC*, 2004.

[26] Q. Gu, J. Zhou, and C. H. Q. Ding. Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. In *SDM*, 2010.

[27] D. Guillamet, J. Vitri, and B. Schiele. Introducing a weighted non-negative matrix factorization for image classification. *Pattern Recognition Letters*, 24(14):2447–2454, 2003.

[28] S. Har-Peled. Low rank matrix approximation in linear time. *CoRR*, abs/1410.8802, 2014.

[29] M. Hardt, R. Meka, P. Raghavendra, and B. Weitz. Computational limits for matrix completion. In *COLT*, 2014.

[30] D. Hsu, S. M. Kakade, and T. Zhang. Robust matrix decomposition with sparse corruptions. *ITIT*, 57(11):7221–7234, 2011.

[31] G. Jeronimo, D. Perrucci, and E. Tsigaridas. On the minimum of a polynomial function on a basic closed semialgebraic set and applications. *SIAM Journal on Optimization*, 23(1):241–255, 2013.

[32] R. Kannan, H. Salmasian, and S. Vempala. The spectral method for general mixture models. *SIAM J. Comput.*, 38(3):1141–1156, 2008.

[33] S. Kirbiz and B. Günsel. Perceptually weighted non-negative matrix factorization for blind single-channel music source separation. In *ICPR*, 2012.

[34] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[35] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, 2000.

[36] J. Z. Li and L. Schmidt. A nearly optimal and agnostic algorithm for properly learning a mixture of k gaussians, for any constant k. *CoRR*, abs/1506.01367, 2015.

[37] Y. Li, Y. Liang, and A. Risteski. Recovery guarantee of weighted low-rank approximation via alternating minimization. *CoRR*, abs/1602.02262, 2016.

[38] Y. Liang, M. Balcan, V. Kanchanapally, and D. P. Woodruff. Improved distributed principal component analysis. In *NIPS*, 2014.

[39] W. Lu and A. Antoniou. New method for weighted low-rank approximation of complex-valued matrices and its application for the design of 2-d digital filters. In *ISCAS*, 2003.

[40] W.-S. Lu, S.-C. Pei, and P.-H. Wang. Weighted low-rank approximation of general complex matrices and its application in the design of 2-d digital filters. In *IEEE Transactions on Circuits and Systems*, volume 44, pages 650–655, 1997.

[41] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *Image Processing, IEEE Transactions on*, 17(1):53–69, 2008.

[42] X. Meng and M. W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *STOC*, 2013.

[43] A. Moitra. An almost optimal algorithm for computing nonnegative rank. In *SODA*, 2013.

[44] J. Nelson and H. L. Nguyen. OSNAP: faster numerical linear algebra algorithms via sparser subspace embeddings. In *FOCS*, 2013.

[45] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala. Latent semantic indexing: A probabilistic analysis. *J. Comput. Syst. Sci.*, 61(2):217–235, 2000.

[46] R. Peeters. Orthogonal representations over finite fields and the chromatic number of graphs. *Combinatorica*, 16(3):417–431, 1996.

[47] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals, part I: introduction. preliminaries. the geometry of semi-algebraic sets. the decision problem for the existential theory of the reals. *J. Symb. Comput.*, 13(3):255–300, 1992.

[48] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals, part II: the general decision problem. preliminaries for quantifier elimination. *J. Symb. Comput.*, 13(3):301–328, 1992.

[49] J. Renegar. On the computational complexity of approximating solutions for real algebraic formulae. *SIAM J. Comput.*, 21(6):1008–1025, 1992.

[50] T. Schramm and B. Weitz. Low-rank matrix completion with adversarial missing entries. *CoRR*, abs/1506.03137, 2015.

[51] D. Shpak. A weighted-least-squares matrix decomposition method with applications to the design of two-dimensional digital filters. In *IEEE Thirty Third Midwest Symposium on Circuits and Systems*, 1990.

[52] N. Srebro and T. S. Jaakkola. Weighted low-rank approximations. In *ICML*, 2003.

[53] D. P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10:1–157, 2014.

[54] J. Yoo and S. Choi. Weighted nonnegative matrix co-tri-factorization for collaborative prediction. In *ACML*, 2009.

[55] G. Young. Maximum likelihood estimation and factor analysis. *Psychometrika*, 6(1):49–53, 1940.

[56] X. Zheng, S. Zhu, J. Gao, and H. Mamitsuka. Instance-wise weighted nonnegative matrix factorization for aggregating partitions with locally reliable clusters. In *IJCAI*, 2015.