informs
https://pubsonline.informs.org/journal/ijoc

INFORMS JOURNAL ON COMPUTING

Vol. 00, No. 0, Xxxxx 0000, pp. 000–000

Submitted to *INFORMS Journal on Computing*

ISSN 0899-1499, EISSN 1526-5528

Disjunctive Branch-And-Bound for Certifiably Optimal Low-Rank Matrix Completion

Dimitris Bertsimas
Sloan School of Management and Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, USA.
ORCID: 0000-0002-1985-1003
dbertsim@mit.edu

Ryan Cory-Wright
Department of Analytics, Marketing and Operations, Imperial Business School, London, UK
ORCID: 0000-0002-4485-0619
r.cory-wright@imperial.ac.uk

Sean Lo
Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA, USA.
ORCID: 0000-0001-8456-6471
seanlo@mit.edu

Jean Pauphilet
Management Science & Operations, London Business School, London, UK.
ORCID: 0000-0001-6352-0984
jpauphilet@london.edu

Abstract. Low-rank matrix completion consists of computing a matrix of minimal complexity that recovers a given set of observations as accurately as possible. Unfortunately, existing methods for matrix completion are heuristics that, while highly scalable and often identifying high-quality solutions, do not possess any optimality guarantees. We reexamine matrix completion with an optimality-oriented eye. We reformulate low-rank matrix completion problems as convex problems over the non-convex set of projection matrices and implement a disjunctive branch-and-bound scheme that solves them to certifiable optimality. Further, we derive a novel and often near-exact class of convex relaxations by decomposing a low-rank matrix as a sum of rank-one matrices and incentivizing that two-by-two minors in each rank-one matrix have determinant zero. In numerical experiments, our new convex relaxations decrease the optimality gap by two orders of magnitude compared to existing attempts, and our disjunctive branch-and-bound scheme solves $n \times m$ rank-r matrix completion problems to certifiable optimality or near optimality in hours for $\max\{m,n\} \leq 2500$ and $r \leq 5$. Moreover, this improvement in the training error translates into an average 2%–50% improvement in the test set error.

Key words: Low-rank matrix completion; branch-and-bound; eigenvector disjunctive cuts; matrix perspective relaxation; semidefinite programming.

1. Introduction

This paper proposes a branch-and-bound scheme for solving low-rank matrix completion to provable optimality. Given observations $A_{i,j} : (i,j) \in \mathcal{I} \subseteq [n] \times [m]$ from a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$, we seek a low-

rank matrix $\boldsymbol{X} \in \mathbb{R}^{n\times m}$ which approximates the observed entries of $\boldsymbol{A}$. This admits the formulation:

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times m}} \quad \frac{1}{2\gamma}\|\boldsymbol{X}\|_F^2 + \frac{1}{2}\sum_{(i,j)\in\mathcal{I}} \left(X_{i,j} - A_{i,j}\right)^2 \quad \text{s.t.} \quad \text{Rank}\left(\boldsymbol{X}\right) \leq k, \tag{1}$$

where the hyperparameter k bounds the rank of $\boldsymbol{X}$ and γ controls its sensitivity to noise.

Problem (1) has received a great deal of attention since the Netflix Competition (Bell and Koren 2007) and heuristics methods now find high-quality solutions for very large-scale instances. Unfortunately, low-rank problems cannot be modeled as mixed-integer conic optimization problems (see Lubin et al. 2022) and thus are out of scope for traditional combinatorial optimization techniques and algorithms. For this reason, developing a certifiably optimal method for low-rank matrix completion would be theoretically meaningful. Moreover, from a practical perspective, Bertsimas et al. (2022) have shown that solving (1) via a provably optimal method gives a smaller out-of-sample mean squared error (MSE) than the MSE obtained via a heuristic. However, to the best of our knowledge, no method solves Problem (1) to provable optimality for n, m beyond 50 or r beyond 1 (Naldi 2016, Bertsimas et al. 2022).

In this paper, we revisit Problem (1) and design a custom branch-and-bound method that solves it to provable optimality using eigenvector disjunctive cuts as branching regions. Numerically, we assess the scalability of our approach for instances with $\max\{m, n\}$ up to 2500 and k up to 5. We also show numerically that this approach obtains more accurate solutions than state-of-the-art methods like alternating minimization implementations of the method of Burer and Monteiro (2003), by up to 50% for some small-scale instances with many local optima.

1.1. Literature Review

We propose a branch-and-bound algorithm that solves Problem (1) to provable optimality at scale. To develop this algorithm, we require three ingredients: a strategy for recursively partitioning the solution space; a technique for generating high-quality convex relaxations on each partition; a local optimization strategy for quickly obtaining good solutions in a given partition. To put our contribution into context, we now review all three aspects of the relevant literature and refer to Udell et al. (2016), Bertsimas et al. (2022) for overviews of low-rank optimization.

Branching A variety of spatial branch-and-bound schemes have been proposed for generic non-convex quadratically constrained quadratic optimization problems such as (1), since the work of McCormick (1976). Unfortunately, these solvers currently cannot obtain high-quality solutions for problems with more than fifty variables (see Kronqvist et al. 2019, for a benchmark and

references). Recently, Bertsimas et al. (2022) reformulated (1) via projection matrices and solved problems with 50×50 matrices and $r = 1$ using off-the-shelf solvers that use McCormick branching regions. Problem-specific branching strategies have been investigated for other non-convex QCQPs. Anstreicher (2022) integrated the disjunctive cuts of Saxena et al. (2010) within a branch-and-bound scheme, and solved some numerically challenging two-trust-region problems. In a parallel direction, Kocuk et al. (2018) proposed a branch-and-cut algorithm for optimal power flow problems.

Relaxations A number of works have proposed solving Problem (1) via their convex relaxations, originating with Fazel (2002), who proposed replacing the rank minimization objective with a trace term for positive semidefinite factor analysis and matrix completion problems (see also Candès and Recht 2009, for an extension to the asymmetric case). More recently, Bertsimas et al. (2023) proposed a general procedure for obtaining strong bounds to low-rank problems (see also Kim et al. 2022, Li and Xie 2023, for related attempts). Namely, they combined the projection matrix reformulation of Bertsimas et al. (2022) with a matrix analog of perspective functions (c.f. Ebadian et al. 2011) and obtained a new class of convex relaxations for low-rank problems.

High-quality heuristics The most popular approach to find good solutions to Problem (1) is to enforce the rank constraint via the decomposition $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^\top$, where the matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ have only k columns, and to iteratively optimize with respect to $\boldsymbol{U}$ and $\boldsymbol{V}$ (Burer and Monteiro 2003). However, the Burer-Monteiro approach is only guaranteed to converge to a local solution and several works have investigated conditions under which convergence to a global optimum is guaranteed for matrix completion (Jain et al. 2013, Hardt 2014, Ge et al. 2016, Ma and Fattahi 2023) or generic semidefinite optimization problems (Boumal et al. 2016, Bhojanapalli et al. 2016, Cifuentes 2019).

1.2. Contributions

We develop a spatial branch-and-bound scheme which solves medium-sized instances of Problem (1) to certifiable (near) optimality.

The key contributions of the paper are threefold. First, we derive eigenvector disjunctive cuts that can be used to recursively partition the feasible region and strengthen the matrix perspective relaxations of Problem (1) more effectively than via McCormick relaxations. Although eigenvector disjunctive cuts were proposed decades ago in the optimization literature (Saxena et al. 2008), they are a "sleeping beauty" in the sense of Ke et al. (2015) as they were forgotten soon after they were proposed due to the lack of appropriate algorithmic and computational machinery in those days. Thus, the discovery that they are computationally useful for solving low-rank problems is a contribution of the paper. Second, by combining an old characterization of rank via determinant

minors, which is new to the matrix completion literature, with the Shor relaxation, we derive new and tighter convex relaxations for matrix completion problems, from which we derive valid inequalities. Third, we design a branch-and-bound algorithm that partitions the feasible region using disjunctive cuts, computes valid lower bounds via the matrix perspective relaxation, and obtains high-quality feasible solutions by running alternating minimization at the root and child nodes.

We remark that our approach improves the scalability of certifiably optimal methods for Problem (1) compared to the state-of-the-art. In particular, we solve instances of Problem (1) with $\max\{m, n\}$ in the thousands and k up to 5 to provable optimality in minutes or hours, while previous attempts Bertsimas et al. (2022) solves problems where $n = m = 50$ and $r = 1$ to provable optimality in hours, but returns optimality gaps larger than 100% after hours when $n \geq 60$ or $r > 1$. This is because our approach involves a custom branching scheme that supports imposing semidefinite constraints at the root node and refining relaxations via eigenvector disjunctions, while Bertsimas et al. (2022) used a commercial solver which does not support semidefinite relaxations at the root node, and refines relaxations using weaker McCormick disjunctions. Indeed, as we establish in this work (Proposition 1), using McCormick relaxations, it is often impossible to improve upon the root node relaxation of a low-rank problem without expanding at least $O(2^{k(n-k)/2})$ nodes, which is very large for $n \gg k$. On the other hand, eigenvector disjunctions improve after expanding $O(2^k)$ nodes.

1.3. Structure

The rest of the paper is structured as follows:

In Section 2, we derive Problem (1)'s matrix perspective relaxation and propose disjunctive cuts for improving it. With our disjunctive inequalities, we separate an optimal solution to a relaxation from its feasible region via a single eigenvalue cut. In contrast, using McCormick disjunctions, we prove in Section 2.2 that disjuncting on fewer than $2^{(n-k)/2}$ variables never separates a solution from a McCormick relaxation.

In Section 3, we leverage a characterization of the rank of a matrix via its determinant minors to develop a novel convex relaxation for low-rank problems, and derive new valid inequalities.

In Section 4, we combine the analysis in Sections 2–3 to design a spatial branch-and-bound algorithm that converges to a certifiably optimal solution of Problem (1). We also discuss different aspects of our algorithmic implementation, including node selection, branching rule, and an alternating minimization strategy to obtain high-quality feasible solutions.

In Section 5, we investigate the performance of our branch-and-bound scheme via a suite of numerical experiments. We identify that our new convex relaxation and preprocessing techniques

reduce (sometimes substantially) the optimality gap at the root node and that our branch and bound scheme solves instances of Problem (1) to certifiable (near) optimality when $\max\{m,n\} = 2500$ in minutes or hours. We also verify that running our branch-and-bound method with a time limit of minutes or hours generates matrices with an out-of-sample MSE up to 50% lower than via state-of-the-art heuristics such as the method of Burer and Monteiro (2003) and its recent refinements.

1.4. Notations

We let non-boldface characters, lowercase bold-faced characters, and uppercase bold-faced characters (e.g., $b, \boldsymbol{x}, \boldsymbol{A}$) denote scalars, vectors, and matrices respectively. Calligraphic uppercase characters such as $\mathcal{Z}$ denote sets. We let $[n]$ denote the running set of indices $\{1,\ldots,n\}$. We let $\boldsymbol{e}$ denote the vector of ones, $\mathbf{0}$ the vector of all zeros, $\boldsymbol{e}_j$ the j-th vector of the canonical basis, and $\mathbb{I}$ the identity matrix. We let $\mathcal{S}^n$ denote the set of $n \times n$ symmetric matrices, and $\mathcal{S}^n_+$ denote $n \times n$ positive semidefinite matrices. We let $\mathcal{Y}^k_n := \left\{ \boldsymbol{Y} \in \mathcal{S}^n : \boldsymbol{Y}^2 = \boldsymbol{Y},\ \mathrm{tr}\,(\boldsymbol{Y}) \leq k \right\}$ denote the set of $n \times n$ orthogonal projection matrices with rank at most k. The convex hull $\mathcal{Y}^k_n$ is $\mathrm{Conv}\left(\mathcal{Y}^k_n\right) = \{ \boldsymbol{Y} \in S^n : \mathbf{0} \preceq \boldsymbol{Y} \preceq \mathbb{I},\ \mathrm{tr}\,(\boldsymbol{Y}) \leq k \}$.

2. Mixed-Projection Formulations, Relaxations, and Disjunctions

In this section, we derive Problem (1)'s semidefinite relaxation in Section 2.1, and refine its relaxation via eigenvector disjunctive cuts. This is significant because while eigenvector disjunctive cuts have been studied in the non-convex quadratically constrained optimization literature, they have not yet been shown to be practically useful at solving large-scale problems, and our refinement technique sets the stage to do so. We also establish that our proposed approach allows us to separate an optimal solution to the original semidefinite relaxation with a single disjunctive cut. In comparison, we justify the poor performance of traditional disjunctions based on McCormick inequalities, as benchmarked against in our numerics (Section 5), by proving in Section 2.2 that some McCormick disjunctions over $2^{k(n-k)/2}$ regions cannot improve the semidefinite relaxation.

2.1. Mixed-Projection Formulations and Relaxations

Motivated by Bertsimas et al. (2022, 2023), we introduce a projection matrix $\boldsymbol{Y} \in \mathcal{Y}_n$ to model the rank of $\boldsymbol{X}$ via the bilinear constraint $\boldsymbol{X} = \boldsymbol{Y}\boldsymbol{X}$. Hence, we can replace the rank constraint on $\boldsymbol{X}$ by a

linear constraint on $\boldsymbol{Y}$: $\text{tr}(\boldsymbol{Y}) \leq k$. By Bertsimas et al. (2023, Theorem 1), we enforce the bilinear constraint implicitly via the domain of a matrix perspective function and write (1) as:

$$\min_{\boldsymbol{Y} \in \mathcal{Y}_n^k} \min_{\substack{\boldsymbol{X} \in \mathbb{R}^{n \times m} \\ \boldsymbol{\Theta} \in \mathcal{S}^m}} \quad \frac{1}{2\gamma} \text{tr}\left(\boldsymbol{\Theta}\right) + \frac{1}{2} \sum_{(i,j) \in \mathcal{I}} \left(X_{i,j} - A_{i,j}\right)^2 \quad \text{s.t.} \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0}. \tag{2}$$

By relaxing $\mathcal{Y}_n^k$ to its convex hull, $\text{Conv}\left(\mathcal{Y}_n^k\right) = \{\boldsymbol{Y} \in \mathcal{S}_+^n : \boldsymbol{Y} \preceq \mathbb{I}\}$, we immediately have the following semidefinite relaxation (see also Bertsimas et al. 2022, Lemma 4):

$$\min_{\substack{\boldsymbol{Y} \in \text{Conv}\left(\mathcal{Y}_n^k\right) \\ \boldsymbol{U} \in \mathbb{R}^{n \times k}}} \min_{\substack{\boldsymbol{X} \in \mathbb{R}^{n \times m} \\ \boldsymbol{\Theta} \in \mathcal{S}^m}} \quad \frac{1}{2\gamma} \text{tr}(\boldsymbol{\Theta}) + \frac{1}{2} \sum_{(i,j) \in \mathcal{I}} \left(X_{i,j} - A_{i,j}\right)^2 \quad \text{s.t.} \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0}, \; \boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top, \tag{3}$$

where we introduce the redundant matrix variable $\boldsymbol{U} \in \mathbb{R}^{n \times k}$ in order to obtain a tractable set of disjunctions in the next section. Note that if $\boldsymbol{Y}$ is a rank-k projection matrix then we have that $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top$ for some $\boldsymbol{U} \in \mathbb{R}^{n \times k}$. In our relaxation, we impose the semidefinite constraint $\boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top$, with the idea to enforce the reverse inequality via the disjunctions we are about to derive.

When we solve (3), one of two situations occur. Either we obtain a solution $\hat{\boldsymbol{Y}}$ with binary eigenvalues and the relaxation is tight, or $\hat{\boldsymbol{Y}}$ has strictly fractional eigenvalues. In the latter case, we improve the relaxation by separating $\hat{\boldsymbol{Y}}$ from the feasible region using disjunctive programming techniques, as we propose in the rest of this section.

2.2. Failure of McCormick Disjunctions

To further justify the need for a new type of disjunctions, we emphasize that most commercial spatial branch-and-bound code for non-convex quadratic optimization problems rely on McCormick disjunctions (McCormick 1976). However, these disjunctions can be ineffective, as shown in a different context by Khajavirad (2023). In this section, we support theoretically the ineffectiveness of McCormick disjunctions: For our matrix completion problem, we show that, even after $O(2^{(n-k)/2})$ McCormick disjunctions, the value of the relaxation (3) can be left unimproved. Our theoretical result echoes the poor numerical performance of McCormick disjunctions we observe in Section 5.

Formally, for each entry of $\boldsymbol{U}$, $U_{i,j}$, we start with box constraints $U_{i,j} \in [-1, 1]$ and recursively partition the region into smaller boxes $U_{i,j} \in [\underline{U}_{i,j}, \overline{U}_{i,j}]$. We also introduce variables

V_{i,j_1,j_2} to model the product $U_{i,j_1}U_{i,j_2}$. It is well documented that the convex hull of the set $\left\{ (v,x,y) \,\middle|\, v = xy, (x,y) \in [\underline{x},\overline{x}] \times [\underline{y},\overline{y}] \right\}$, denoted $\mathcal{M}(\underline{x},\overline{x},\underline{y},\overline{y})$, is given by

$$\left\{ (v,x,y) \,\middle|\, \max\{\underline{x}y + \underline{y}x - \underline{x}\underline{y}, \overline{x}y + \overline{y}x - \overline{x}\overline{y}\} \leq v \leq \min\{\underline{y}x + \overline{x}y - \overline{x}\underline{y}, \overline{y}x + \underline{x}y - \underline{x}\overline{y}\} \right\}.$$

For our matrix completion problem, this yields a disjunction over relaxations of the form:

$$\begin{aligned} \min_{\substack{\boldsymbol{Y}\in \mathrm{Conv}(\mathcal{Y}_n^k),\\ \boldsymbol{U}\in\mathbb{R}^{n\times k},\\ \boldsymbol{V}\in\mathbb{R}^{n\times k\times k}}} \min_{\boldsymbol{X}\in\mathbb{R}^{n\times m},\boldsymbol{\Theta}\in\mathcal{S}^m} & \frac{1}{2\gamma}\mathrm{tr}\left(\boldsymbol{\Theta}\right) + \frac{1}{2}\sum_{(i,j)\in\mathcal{I}} \left(X_{i,j} - A_{i,j}\right)^2 \quad \text{s.t.} \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0},\ \boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top, \\ & \sum_{i=1}^n V_{i,j,j} = 1\ \forall\ j \in [k],\ \sum_{i=1}^n V_{i,j_1,j_2} = 0\ \forall\ j_1, j_2 \in [k] :\ j_1 \neq j_2, \\ & (V_{i,j_1,j_2}, U_{i,j_1}, U_{i,j_2}) \in \mathcal{M}(\underline{U}_{i,j_1}, \overline{U}_{i,j_2}, \underline{U}_{i,j_2}, \overline{U}_{i,j_2}),\ \forall\ i \in [n],\ \forall j_1, j_2 \in [k], \end{aligned} \tag{4}$$

where the last constraint links $\boldsymbol{U}$ and $\boldsymbol{V}$.

We now prove that, for any ρ, disjuncting on at most $\rho \leq (n-k)/2$ variables $U_{i,j}$ in each column $j \in [k]$ of $\boldsymbol{U}$ cannot improve on our root node relaxation (proof deferred to Appendix EC.1.1):

PROPOSITION 1. *Consider Problem* (4) *and suppose that for every column j in $\boldsymbol{U}$ we have a disjunction $\cup_t [\underline{U}_{i,j}^t, \overline{U}_{i,j}^t]$ for each i in some index set $\mathcal{I}(j)$ of cardinality at most $\rho < (n-k)/2$, but $[\underline{U}_{i',j}, \overline{U}_{i',j}] = [-1,1]$ for all other indices $i' \notin \mathcal{I}(j)$. Then, Problem* (4) *possesses the same optimal objective value as Problem* (3).

Proposition 1 reveals that we must branch on more than $(n-k)/2$ variables in a given column of $\boldsymbol{U}$ before we can hope to improve upon the root node relaxation. Moreover, the structure of the proof reveals that we must branch on either $U_{t,i}$ or $U_{t,j}$ for each $t \in [n]$ to have any hope of improving on the root node relaxation. This behavior will contrast with our eigenvector disjunction, which separates the solution to the root node relaxation with a single cut and often improves (numerically) our root node relaxation immediately. This observation complements a body of work demonstrating that McCormick relaxations are often dominated by other relaxations in both theory and practice (Kocuk et al. 2018, Khajavirad 2023).

Moreover, it challenges the ongoing practice in commercial non-convex solvers, which use McCormick disjunctions. Indeed, some branching strategies like best-first search rely upon estimates of the improvement in solution quality after branching and hence struggle when no branching decision at a root node improves the lower bound.

2.3. Improving Relaxations via Eigenvector Disjunctions

A challenging predicament in separating an optimal solution to Problem (3) from (2)'s feasible region is that branching on the eigenvalues of $\boldsymbol{Y}$ directly—which would be the most natural extension of the branching scheme in binary optimization—is not, to our knowledge, possible. Accordingly, we generalize the lifted approach proposed by Saxena et al. (2010) for mixed-integer QCQO to optimization over rank-k projection matrices where we may have $k > 1$.

For $\boldsymbol{Y}$ to be a rank-k projection matrix, it should satisfy $\hat{\boldsymbol{Y}} \preceq \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top$ in an optimal solution $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ of the semidefinite relaxation (3). Owing to the constraint $\boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top$, this would imply that $\hat{\boldsymbol{Y}} = \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top$, and thus $\hat{\boldsymbol{Y}}$ is a rank-k matrix. Since $\mathrm{Span}(\hat{\boldsymbol{X}}) \subseteq \mathrm{Span}(\hat{\boldsymbol{Y}})$ (Bertsimas et al. 2023), we can conclude that $\hat{\boldsymbol{X}}$ solves the original rank-constrained problem (1). Otherwise, we have $\hat{\boldsymbol{Y}} \not\preceq \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top$, i.e., there exists an $\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x}\|_2 = 1$ such that $\|\hat{\boldsymbol{U}}\boldsymbol{x}\|_2^2 < \boldsymbol{x}^\top\hat{\boldsymbol{Y}}\boldsymbol{x}$. Therefore, we would like to impose the (non-convex) inequality $\boldsymbol{x}^\top\boldsymbol{Y}\boldsymbol{x} \leq \|\boldsymbol{U}^\top\boldsymbol{x}\|_2^2$ via a disjunction.

Observe that, for any feasible $\boldsymbol{U}$, $\|\boldsymbol{U}\boldsymbol{x}\|_2^2 \leq 1$. First, we decompose the square-norm term $\|\boldsymbol{U}\boldsymbol{x}\|_2^2 = \sum_{j\in[n]}(\boldsymbol{e}_j^\top\boldsymbol{U}\boldsymbol{x})^2$. Then, we bound each term by a piece-wise linear upper approximation. For any $u_0 \in (-1, 1)$, the function $u \in [-1, 1] \mapsto u^2$ has the following piecewise upper approximation with breakpoint u_0:

$$u^2 \leq \begin{cases} u + uu_0 - u_0 & \text{if } u \in [u_0, 1], \\ -u + uu_0 + u_0 & \text{if } u \in [-1, u_0]. \end{cases} \tag{5}$$

Using $u_0 = \boldsymbol{e}_j\hat{\boldsymbol{U}}^\top\boldsymbol{x}$ yields the following upper bound:

$$(\boldsymbol{e}_j^\top\boldsymbol{U}\boldsymbol{x})^2 \leq \begin{cases} \boldsymbol{x}^\top\hat{\boldsymbol{U}}\boldsymbol{U}\boldsymbol{x} + \boldsymbol{e}_j^\top(\hat{\boldsymbol{U}} - \boldsymbol{U})\boldsymbol{x} & \text{if } \boldsymbol{e}_j^\top\boldsymbol{U}\boldsymbol{x} \in [-1, \boldsymbol{e}_j^\top\hat{\boldsymbol{U}}\boldsymbol{x}], \\ \boldsymbol{x}^\top\hat{\boldsymbol{U}}\boldsymbol{U}\boldsymbol{x} + \boldsymbol{e}_j^\top(\boldsymbol{U} - \hat{\boldsymbol{U}})\boldsymbol{x} & \text{if } \boldsymbol{e}_j^\top\boldsymbol{U}\boldsymbol{x} \in [\boldsymbol{e}_j^\top\hat{\boldsymbol{U}}\boldsymbol{x}, 1]. \end{cases}$$

and the following disjunction over 2^k regions:

$$\bigvee_{L\subseteq[n]} \left\{ \boldsymbol{Y} \;\middle|\; \begin{array}{ll} \boldsymbol{e}_j^\top\boldsymbol{U}\boldsymbol{x} \in [-1, \boldsymbol{e}_j^\top\hat{\boldsymbol{U}}\boldsymbol{x}] & \forall\, j \in L, \\ \boldsymbol{e}_j^\top\boldsymbol{U}\boldsymbol{x} \in [\boldsymbol{e}_j^\top\hat{\boldsymbol{U}}\boldsymbol{x}, 1] & \forall\, j \in [k] \setminus L, \\ \boldsymbol{x}^\top\boldsymbol{Y}\boldsymbol{x} \leq \boldsymbol{x}^\top\hat{\boldsymbol{U}}\boldsymbol{U}\boldsymbol{x} + \displaystyle\sum_{j\in L} \boldsymbol{e}_j^\top(\hat{\boldsymbol{U}} - \boldsymbol{U})\boldsymbol{x} + \sum_{j\in[n]\setminus L} \boldsymbol{e}_j^\top(\boldsymbol{U} - \hat{\boldsymbol{U}})\boldsymbol{x} & \end{array} \right\}. \tag{6}$$

As a result, we can strengthen our convex relaxation (3) by imposing this disjunction and optimizing over the 2^k resulting convex problems. We formalize this result in the following proposition:

PROPOSITION 2. *Let* $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ *be an optimal solution to* (3) *so that* $\|\hat{\boldsymbol{U}}\boldsymbol{x}\|_2^2 < \boldsymbol{x}^\top \hat{\boldsymbol{Y}}\boldsymbol{x}$ *for some vector* $\boldsymbol{x} \in \mathbb{R}^n$ *such that* $\|\boldsymbol{x}\|_2 = 1$. *Then, any solution to* (3) *with* $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top$ *satisfies* (6)*, but* $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ *does not satisfy* (6)*.*

Proof of Proposition 2 Let $\boldsymbol{U}, \boldsymbol{Y}$ be matrices so that $\boldsymbol{U}\boldsymbol{U}^\top = \boldsymbol{Y}$. We have $\boldsymbol{x}^\top (\boldsymbol{Y} - \boldsymbol{U}\boldsymbol{U}^\top)\boldsymbol{x} = 0$ for any vector $\boldsymbol{x}$, which implies the disjunction is equivalent to requiring that, for each j, either $\boldsymbol{e}_j^\top \boldsymbol{U}^\top \boldsymbol{x} \in [-1, \boldsymbol{e}_j^\top \hat{\boldsymbol{U}}^\top \boldsymbol{x}]$ or $\boldsymbol{e}_j^\top \boldsymbol{U}^\top \boldsymbol{x} \in [\boldsymbol{e}_j^\top \hat{\boldsymbol{U}}^\top \boldsymbol{x}, 1]$, which is trivially true.

On the other hand, at the point $\boldsymbol{U} = \hat{\boldsymbol{U}}$, our approximation of the constraint $\boldsymbol{x}^\top \boldsymbol{Y}\boldsymbol{x} \leq \|\boldsymbol{U}^\top \boldsymbol{x}\|_2^2$ is tight so our disjunction requires $\boldsymbol{x}^\top (\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}})\boldsymbol{x} \geq 0$, which contradicts $\boldsymbol{x}^\top (\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}})\boldsymbol{x} < 0$. □

Proposition 2 reveals that the disjunction (6) in Problem (3) separates the optimal solution to (3) whenever it is infeasible for the original problem (1).

REMARK 1. For tractability considerations, we work on a lifted version of the semidefinite relaxation (3), which involves additional variables $\boldsymbol{U} \in \mathbb{R}^{n \times k}$ such that $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top$. Instead, we could derive a set of disjunctions on $\boldsymbol{Y}$ alone by manipulating the constraint $\boldsymbol{Y}^2 = \boldsymbol{Y}$. Unfortunately, while simpler to derive, this results in a set of 2^n, rather than 2^k, disjunctions, which is problematic as $k \ll n$ in practical low-rank matrix completion problems. That being said, working on a lifted formulation may create symmetry issues: if $(\boldsymbol{Y}, \boldsymbol{U}, \boldsymbol{X}, \boldsymbol{\Theta})$ is a solution to the relaxation, then so is $(\boldsymbol{Y}, \boldsymbol{U}\boldsymbol{\Pi}, \boldsymbol{X}, \boldsymbol{\Theta})$ for any k-by-k permutation matrix $\boldsymbol{\Pi}$. Hence, Problem (3)'s lower bound may not actually improve until we apply a disjunction for each permutation of the columns of $\boldsymbol{U}$, which is likely computationally prohibitive in practice. We alleviate these issues by exploring branching strategies and symmetry-breaking constraints in Section 4.

REMARK 2. We obtain our 2^k disjunction by constructing a piecewise linear upper approximation of $u \in [-1, 1] \mapsto u^2$ with two pieces, i.e., a single breakpoint u_0. We can construct tighter approximations by considering more pieces/breakpoints. For example, with the two breakpoints $\{-|u_0|, +|u_0|, \}$, we obtain the following 3-piece linear approximation

$$u^2 \leq \begin{cases} -u - u|u_0| - |u_0| & \text{if } u \in [-1, -|u_0|], \\ u_0^2 & \text{if } u \in [-|u_0|, |u_0|], \\ u + u|u_0| - |u_0| & \text{if } u \in [|u_0|, 1], \end{cases} \tag{7}$$

and with the three breakpoints $\{-|u_0|, 0, |u_0|\}$,

$$u^2 \leq \begin{cases} -u - u|u_0| - |u_0| & \text{if } u \in [-1, -|u_0|], \\ -u|u_0| & \text{if } u \in [-|u_0|, 0], \\ u|u_0| & \text{if } u \in [0, |u_0|], \\ u + u|u_0| - |u_0| & \text{if } u \in [|u_0|, 1]. \end{cases} \tag{8}$$

In general, introducing more pieces results in a stronger but more expensive to compute disjunctive bound. Therefore, there is a trade-off between the quality of the piecewise linear upper approximation and the number of convex optimization problems that need to be solved to compute the corresponding bound, which we explore numerically in Section 5.

3. Convex Relaxations and Valid Inequalities

In this section, we use a different characterization of low-rank matrices in terms of their determinant minors to derive a new convex relaxation for low-rank matrix completion problems, which improves upon (3). We also show how to use this relaxation to derive valid inequalities that strengthen (3).

Our starting point is the observation that the rank of $\boldsymbol{X}$ is fully characterized by the following well-known lemma (as proven in, e.g., Horn and Johnson 1985):

LEMMA 1. *Let $\boldsymbol{X}$ be a matrix. Then, the rank of $\boldsymbol{X}$ is at most k if and only if all $(k+1) \times (k+1)$ minors of $\boldsymbol{X}$ have determinant* 0.

Lemma 1 provides a characterization of low-rank matrices that is complementary to the projection-matrix characterization studied in the previous section. For our low-rank matrix completion problems, we decompose $\boldsymbol{X}$ into a sum of rank-one matrices, apply Lemma 1's characterization to each rank-one matrix individually, and combine this characterization with our matrix perspective relaxation to obtain stronger relaxations.

3.1. Convex Relaxations

We construct our new convex relaxation via the following steps: First, we decompose $\boldsymbol{X} = \sum_{t=1}^{k} \boldsymbol{X}^t$, as a sum of rank-one matrices $\boldsymbol{X}^t$. Second, for each $t \in [k]$, we introduce a matrix to model the outer product of each vectorized two-by-two minor of $\boldsymbol{X}$ with itself, in particular letting $W^t_{i,j}$ model $(X^t_{i,j})^2$ in all moment matrices. Third, we use that each two-by-two minor should have zero determinant to eliminate some moment variables —by introducing the same variable $V^{3,t}_{(i_1,i_2),(j_1,j_2)}$ on each entry of the anti-diagonal of the moment matrix. Fourth, we model the term

$X_{i,j}^2 = \left(\sum_{t=1}^k X_{i,j}^t\right)^2 = \sum_{t\in[k]}(X_{i,j}^t)^2 + 2\sum_{t'<t} X_{i,j}^t X_{i,j}^{t'}$ in the objective by also performing a relaxation on the moment matrix generated by the vector $\begin{pmatrix}1 & X_{i,j}^1 & X_{i,j}^2 & \dots & X_{i,j}^k\end{pmatrix}$ and selecting the appropriate off-diagonal terms to model each product $X_{i,j}^{t_1}X_{i,j}^{t_2}$. Finally, we replace $X_{i,j}^2$ with $W_{i,j}$ in the objective where applicable and link $\Theta_{i,j}$ with the appropriate terms in $\boldsymbol{W}$ to impose appropriate objective pressure in our relaxation.

We have the following result:

PROPOSITION 3. *The following relaxation is at least as strong as the matrix perspective relaxation for matrix completion* (3) *where* $k \in [\min(n,m)]$*:*

$$\min_{\substack{\boldsymbol{X}^t,\boldsymbol{W},\boldsymbol{W}^t\in\mathbb{R}^{n\times m}\ \forall t\in[k],\\ \boldsymbol{Y}\in\mathrm{Conv}(\mathcal{Y}_n^k),\ \boldsymbol{\Theta}\in\mathcal{S}_+^m,\\ \boldsymbol{V}^{\cdot,t},\boldsymbol{H}^{t,t'},\forall t,t'\in[k]}} \frac{1}{2\gamma}\mathrm{tr}(\boldsymbol{\Theta}) + \frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\sum_{t\in[k]}\left(A_{i,j}^2 - 2X_{i,j}^t A_{i,j} + W_{i,j}\right) \tag{9}$$

$$s.t. \quad \begin{pmatrix} \boldsymbol{Y} & \sum_{t\in[k]}\boldsymbol{X}^t \\ \sum_{t\in[k]}\boldsymbol{X}^{t\top} & \boldsymbol{\Theta}\end{pmatrix} \succeq \boldsymbol{0},$$

$$\begin{pmatrix} 1 & X_{i_1,j_1}^t & X_{i_1,j_2}^t & X_{i_2,j_1}^t & X_{i_2,j_2}^t \\ X_{i_1,j_1}^t & W_{i_1,j_1}^t & V_{i_1,(j_1,j_2)}^{1,t} & V_{(i_1,i_2),j_1}^{2,t} & V_{(i_1,i_2),(j_1,j_2)}^{3,t} \\ X_{i_1,j_2}^t & V_{i_1,(j_1,j_2)}^{1,t} & W_{i_1,j_2}^t & V_{(i_1,i_2),(j_1,j_2)}^{3,t} & V_{(i_1,i_2),j_2}^{2,t} \\ X_{i_2,j_1}^t & V_{(i_1,i_2),j_1}^{2,t} & V_{(i_1,i_2),(j_1,j_2)}^{3,t} & W_{i_2,j_1}^t & V_{i_2,(j_1,j_2)}^{1,t} \\ X_{i_2,j_2}^t & V_{(i_1,i_2),(j_1,j_2)}^{3,t} & V_{(i_1,i_2),j_2}^{2,t} & V_{i_2,(j_1,j_2)}^{1,t} & W_{i_2,j_2}^t \end{pmatrix} \succeq \boldsymbol{0}, \quad \begin{array}{l}\forall\, i_1 < i_2 \in [n],\\ \forall\, j_1 < j_2 \in [m],\\ \forall\, t \in [k],\end{array}$$

$$\begin{pmatrix} 1 & X_{i,j}^1 & X_{i,j}^2 & \dots & X_{i,j}^k \\ X_{i,j}^1 & W_{i,j}^1 & H_{i,j}^{1,2} & \dots & H_{i,j}^{1,k} \\ X_{i,j}^2 & H_{i,j}^{1,2} & W_{i,j}^2 & \dots & H_{i,j}^{2,k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ X_{i,j}^k & H_{i,j}^{1,k} & H_{i,j}^{2,k} & \dots & W_{i,j}^k \end{pmatrix} \succeq \boldsymbol{0}, \quad \forall\, i \in [n],\ j \in [m],$$

$$\Theta_{j,j} = \sum_{i\in[n]} W_{i,j},\ W_{i,j} = \sum_{t\in[k]}\left(W_{i,j}^t + \sum_{t'\in[k]:t'\neq t} H_{i,j}^{t',t}\right) \forall\, i \in [n],\ j \in [m].$$

Proof of Proposition 3 This follows immediately from the fact that any feasible solution to (9) is feasible in (3) with the same objective value. □

Section 5 augments Proposition 3 by providing many examples where (9) provides a strictly stronger lower bound than (3), sometimes to the extent that, when combined with a feasible solution from a heuristic, the bound gap from (9) is an order of magnitude smaller than that from (3).

3.2. Partial Convex Relaxations

Despite its strength, our new relaxation is also computationally prohibitive to solve at scale. Accordingly, we use it to derive valid inequalities that we further impose to (3) to develop more scalable (yet less tight) semidefinite relaxations.

To develop our relaxation, we begin with the following problem, which is equivalent to (3) but also includes each slice $\boldsymbol{X}^t$ and variables $W_{i,j}$ to model $X^2_{i,j}$ for each (i,j):

$$\begin{aligned}\min_{\substack{\boldsymbol{X}^t,\boldsymbol{W},\boldsymbol{W}^t\in\mathbb{R}^{n\times m}\ \forall t\in[k],\\ \boldsymbol{Y}\in\text{Conv}(\mathcal{Y}^k_n),\ \boldsymbol{\Theta}\in\mathcal{S}^m_+,\boldsymbol{V}}} \quad & \frac{1}{2\gamma}\text{tr}(\boldsymbol{\Theta})+\frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\sum_{t\in[k]}\left(A^2_{i,j}-2X^t_{i,j}A_{i,j}+W_{i,j}\right)\\ \text{s.t.} \quad & \begin{pmatrix}\boldsymbol{Y} & \sum_{t\in[k]}\boldsymbol{X}^t\\ \sum_{t\in[k]}\boldsymbol{X}^{t\top} & \boldsymbol{\Theta}\end{pmatrix}\succeq\boldsymbol{0},\\ & \Theta_{j,j}=\sum_{i\in[n]}W_{i,j},\ \forall j\in[m],\ \left(\sum_{t\in[k]}X^t_{i,j}\right)^2\le W_{i,j},\ \forall\, i\in[n],\ j\in[m].\end{aligned} \tag{10}$$

Given a solution to this relaxation, we select a subset of the 2×2 minors of $\boldsymbol{X}$ according to a pre-specified criteria, e.g., all 2×2 minors where all four entries (i_1,i_2,j_1,j_2) are observed, and a random sample of 2×2 minors where at least three entries are observed. Next, for each such minor (i_1,i_2,j_1,j_2), we introduce appropriately indexed variables $\boldsymbol{V},\boldsymbol{W}^t,\boldsymbol{H}$, omit the constraint linking $X^t_{i,j}$ and $W_{i,j}$, and impose the following constraints:

$$\begin{pmatrix}
1 & X^t_{i_1,j_1} & X^t_{i_1,j_2} & X^t_{i_2,j_1} & X^t_{i_2,j_2}\\
X^t_{i_1,j_1} & W^t_{i_1,j_1} & V^{1,t}_{i_1,(j_1,j_2)} & V^{2,t}_{(i_1,i_2),j_1} & V^{3,t}_{(i_1,i_2),(j_1,j_2)}\\
X^t_{i_1,j_2} & V^{1,t}_{i_1,(j_1,j_2)} & W^t_{i_1,j_2} & V^{3,t}_{(i_1,i_2),(j_1,j_2)} & V^{2,t}_{(i_1,i_2),j_2}\\
X^t_{i_2,j_1} & V^{2,t}_{(i_1,i_2),j_1} & V^{3,t}_{(i_1,i_2),(j_1,j_2)} & W^t_{i_2,j_1} & V^{1,t}_{i_2,(j_1,j_2)}\\
X^t_{i_2,j_2} & V^{3,t}_{(i_1,i_2),(j_1,j_2)} & V^{2,t}_{(i_1,i_2),j_2} & V^{1,t}_{i_2,(j_1,j_2)} & W^t_{i_2,j_2}
\end{pmatrix}\succeq\boldsymbol{0},\quad\forall\, t\in[k],$$

$$\begin{pmatrix} 1 & X^1_{i,j} & X^2_{i,j} & \dots & X^k_{i,j} \\ X^1_{i,j} & W^1_{i,j} & H^{1,2}_{i,j} & \dots & H^{1,k}_{i,j} \\ X^2_{i,j} & H^{1,2}_{i,j} & W^2_{i,j} & \dots & H^{2,k}_{i,j} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ X^k_{i,j} & H^{1,k}_{i,j} & H^{2,k}_{i,j} & \dots & W^k_{i,j} \end{pmatrix} \succeq \mathbf{0}, \quad \forall\, i \in \{i_1, i_2\},\ j \in \{j_1, j_2\},$$

$$W_{i,j} = \sum_{t \in [k]} \left(W^t_{i,j} + \sum_{t' \in [k]: t' \neq t} H^{t',t}_{i,j} \right), \quad \forall\, i \in \{i_1, i_2\},\ j \in \{j_1, j_2\}.$$

All in all, we impose k 5×5 PSD constraint and at most 4 $(k+1) \times (k+1)$ PSD constraint for each 2×2 submatrix we aim to model, which allows us to control the computational complexity of our relaxation via the number of cuts at each problem size. In particular, the computational complexity of modeling a Shor relaxation of a given 2×2 minor scales independently of n, m.

4. A Custom Branch-and-Bound Algorithm

In this section, we propose a nonlinear branch-and-bound framework to obtain ϵ-optimal and ϵ-feasible solutions to Problem (2) by recursively solving its semidefinite relaxation (3), strengthened by the techniques described in Section 3, and partitioning the feasible region via the eigenvector disjunctions derived in Section 2. We first provide a high-level description of our approach and relevant pseudocode before describing our branching and incumbent generation strategies in detail.

4.1. Pseudocode and Convergence Results

At the root node, we solve the matrix perspective relaxation (3), possibly strengthened with some of the valid inequalities from Section 3.2.

After solving the relaxation, we obtain a solution $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}})$ and compute the smallest eigenvector of $\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}}$, denoted $\boldsymbol{x}$. If $\boldsymbol{x}^\top \left(\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}} \right) \boldsymbol{x} \geq -\epsilon$ then, recalling $\hat{\boldsymbol{Y}} \succeq \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top$, $\hat{\boldsymbol{Y}}$ is, up to a tolerance of ϵ, a projection matrix and solves (2) to optimality and ϵ-feasibility. Otherwise, we have $\boldsymbol{x}^\top \left(\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}} \right) \boldsymbol{x} < -\epsilon$ and apply a disjunction, e.g., (6), as in Section 2. This generates a new node or subproblem for each piece of the disjunction, which is appended to the list of open nodes and eventually selected for branching.

We proceed recursively, by selecting a subproblem from the list of open nodes, solving the corresponding convex relaxation, and imposing a new disjunction which creates more nodes, until

we identify an ϵ-feasible solution $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}})$. We obtain a search tree where each node models a semidefinite relaxation and each edge corresponds to selecting one region of a disjunction.

To avoid retaining each node in memory and maintain a search tree of a practical size, we also perform a prune or fathom step (i.e., do not impose an additional disjunction) in two different situations. First, if $\lambda_{\min}(\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}}) \geq -\epsilon$ then we have an ϵ-feasible solution which provides a global upper bound on (2)'s optimal value and we need not impose additional disjunctions at this node. As we establish in Theorem 1, this is guaranteed to occur at a sufficiently large tree depth for any $\epsilon > 0$. Second, if the value of the relaxation at a given node is within ϵ of our global upper bound, then no solution with the constraints inherited from this node can improve upon an incumbent solution by more than ϵ and we do not branch.

We describe our branch-and-bound scheme in Algorithm 1; implementation details will be made clear throughout this section. In Algorithm 1, we describe a subproblem as a set of constraints $\mathcal{Q}$ and a depth t. The constraints in $\mathcal{Q}$ stem from the eigenvector cuts described in Section 2 and are parameterized by the solution at which they were obtained, $(\hat{\boldsymbol{U}}, \hat{\boldsymbol{Y}})$, the most negative eigenvector $\boldsymbol{x}$, and a vector $\boldsymbol{z} \in [q]^k$ encoding which one of the q^k regions we are considering. For concision, we denote $\overline{f}_{i;q}(\cdot;\, u_0)$ the ith piece of the piece-wise linear upper-approximation with q pieces of $u \mapsto u^2$, obtained from the breakpoint u_0.

To verify that Algorithm 1 convergences, we prove in Theorem 1 that a breadth-first node selection strategy which iteratively minimizes (3) and imposes a disjunctive cut of the form (6), according to a most negative eigenvector of $\boldsymbol{U}\boldsymbol{U} - \boldsymbol{Y}^\top$, an optimal solution to (3) at the current iterate, eventually identifies an ϵ-feasible solution (where $\lambda_{\min}(\boldsymbol{U}\boldsymbol{U}^\top - \boldsymbol{Y}) > -\epsilon$) within a finite number of iterations for any $\epsilon > 0$. The proof of Theorem 1 reveals that any node sufficiently deep in our search tree which contains a feasible solution to the semidefinite relaxation (intersected with appropriate constraints from our disjunctions) gives an ϵ-feasible solution to the original problem. Therefore, due to the enumerative nature of branch-and-bound, this result verifies that our approach converges for any search and node selection strategy, possibly with a different number of pieces in each disjunction.

THEOREM 1. *Let $(\boldsymbol{Y}_\ell, \boldsymbol{U}_\ell)$ denote a solution generated by the ℓth iterate of the procedure:*

- *For each $t \in \mathbb{N}$, set $(\boldsymbol{Y}_t, \boldsymbol{U}_t)$ according to the optimal solution of* (3)*, possibly with disjunctive cuts of the form* (6)*.*
- *If $\lambda_{\min}(\boldsymbol{U}_t\boldsymbol{U}_t - \boldsymbol{Y}_t) \geq -\epsilon$ then terminate.*
- *Else, impose the disjunctive cut* (6) *in* (3) *with the eigenvector $\boldsymbol{x}_t$, where*

$$\boldsymbol{x}_t \in \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathbb{R}^n:\ \|\boldsymbol{x}\|_2 = 1} \langle \boldsymbol{x}\boldsymbol{x}^\top, \boldsymbol{U}_t\boldsymbol{U}_t^\top - \boldsymbol{Y}_t \rangle.$$

For any $\epsilon > 0$ there exists an $\ell \in \mathbb{N}$ so that $\lambda_{\min}(\boldsymbol{U}_\ell \boldsymbol{U}_\ell - \boldsymbol{Y}_\ell) \geq -\epsilon$ and $\boldsymbol{Y}_\ell$ is an optimal, ϵ-feasible solution to (2)*. Moreover, suppose we set $\epsilon \to 0$. Then, any limit point of $\{\boldsymbol{Y}_t\}_{t=1}^{\infty}$ solves* (2)*.*

Proof of Theorem 1 Deferred to Appendix EC.1.2. □

Note that an "optimal, ϵ-feasible" solution refers to a solution with objective value at least as good as any solution to (2), which is ϵ-feasible. This is also known as a "superoptimal" solution. In our branch-and-bound code, we also impose symmetry-breaking constraints on $\boldsymbol{U}$ throughout the tree, by noting that $\boldsymbol{U}$ represents a set of k orthonormal vectors in $\mathbb{R}^n$. We do so via:

$$U_{ij} \geq 0, \quad \forall\, j \in [k],\ \forall\, i \in \{n-k+j, n-k+j+1, \ldots, n\} \tag{12}$$

We now expound on implementation details of Algorithm 1 pertaining to node selection (Section 4.2), the branching strategy used (Section 4.3), and incumbent selection (Section 4.4).

4.2. Node Selection: Depth-First vs Breadth-First vs Best-First Search

One of the most significant design decisions in a branch-and-bound scheme is the node selection strategy employed. The three node selection heuristics that we consider in this work are depth-first search (where nodes are selected in a last-in-first-out manner), breadth-first search (where nodes are selected in a first-in-first-out manner), and best-first search (where the node with the lowest remaining lower bound is selected at each iteration).

While we assumed a breadth-first strategy to establish the convergence of Algorithm 1 in Theorem 1, we observe in Section 5 that best-first search is usually faster than the other strategies considered here. Accordingly, except where explicitly stated otherwise, we use best-first search.

4.3. Branching Strategy

When running our branch-and-bound algorithm on a rank-k matrix, each node that is not fathomed generates q^k, $q \geq 2$, child nodes corresponding to the q^k regions of the disjunction. For example, the disjunction (6) generates 2^k child nodes, but, as explained for the rank-one case in Section 2.3, more fine-grained upper approximations of the ℓ_2^2 norm lead to disjunctions over 3^k or 4^k regions.

Accordingly, another algorithmic design decision is selecting the number of pieces $q \geq 2$ which should be used in our disjunctive cuts. Indeed, increasing the number of child nodes generated at each iteration, q^k, improves the tightness of the bound at the expense of additional computational time for solving all q^k subproblems. We investigate this tradeoff numerically in Section 5.

Algorithm 1 Branch-and-bound for low-rank matrix completion (3)

Initialize $\mathcal{D} = \emptyset$, $t = 0$, $Z_{\text{lower}} = -\infty$, $Z_{\text{upper}} = +\infty$;

Initialize a queue of problems derived from Problem (3), $\mathcal{Q} = \{ (\mathcal{D}, t) \}$;

while $\mathcal{Q}$ is non-empty and $Z_{\text{upper}} - Z_{\text{lower}} > \epsilon$ **do**

Retrieve a problem $(\mathcal{D}, t)$ from the queue: $\mathcal{Q} \leftarrow \mathcal{Q} \setminus (\mathcal{D}, t)$;

Solve following problem, yielding $(\hat{\boldsymbol{\Theta}}_{(t+1)}, \hat{\boldsymbol{Y}}_{(t+1)}, \hat{\boldsymbol{X}}_{(t+1)}, \hat{\boldsymbol{U}}_{(t+1)})$ with objective Z:

$$
\begin{array}{ll}
\min\limits_{\substack{\boldsymbol{\Theta}\in\mathcal{S}^m \\ \boldsymbol{Y}\in\mathcal{S}^n \\ \boldsymbol{X}\in\mathbb{R}^{n\times m} \\ \boldsymbol{U}\in\mathbb{R}^{n\times k}}} & \dfrac{1}{2}\displaystyle\sum_{(i,j)\in\mathcal{I}} (X_{i,j} - A_{i,j})^2 + \dfrac{1}{2\gamma}\mathrm{tr}\,(\boldsymbol{\Theta}) \\
\text{s.t.} & \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0}, \quad \boldsymbol{0} \preceq \boldsymbol{Y} \preceq \mathbb{I}, \quad \mathrm{tr}\,(\boldsymbol{Y}) \le k, \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{U} \\ \boldsymbol{U}^\top & \mathbb{I} \end{pmatrix} \succeq \boldsymbol{0}, \\
& \left.\begin{array}{l} \boldsymbol{U}_j^\top \boldsymbol{x} \in [b_{z_j}, b_{z_j+1}] \quad \forall\, j \in [k] \\ \langle \boldsymbol{Y}, \boldsymbol{x}\boldsymbol{x}^\top \rangle \le \displaystyle\sum_{j=1}^{k} \overline{f}_{z_j;q}(\boldsymbol{U}_j^\top \boldsymbol{x};\ \hat{\boldsymbol{U}}_j^\top \boldsymbol{x}) \end{array}\right\} \forall\, (\hat{\boldsymbol{U}}, \hat{\boldsymbol{Y}}, \boldsymbol{x}, z) \in \mathcal{D}
\end{array}
\tag{11}
$$

if $\hat{\boldsymbol{U}}_{(t+1)}\hat{\boldsymbol{U}}_{(t+1)}^\top - \hat{\boldsymbol{Y}}_{(t+1)} \succeq \boldsymbol{0}$ **then**

if $Z < Z_{\text{upper}}$ **then**

$Z_{\text{upper}} \leftarrow Z$

$(\boldsymbol{\Theta}_{\text{opt}}, \boldsymbol{Y}_{\text{opt}}, \boldsymbol{X}_{\text{opt}}, \boldsymbol{U}_{\text{opt}}) \leftarrow (\hat{\boldsymbol{\Theta}}_{(t+1)}, \hat{\boldsymbol{Y}}_{(t+1)}, \hat{\boldsymbol{X}}_{(t+1)}, \hat{\boldsymbol{U}}_{(t+1)})$

end if

else if $Z < Z_{\text{upper}}$ **then**

Compute (unit-length) $\boldsymbol{x}_{(t+1)}$ s.t. $\boldsymbol{x}_{(t+1)}^\top \left(\hat{\boldsymbol{U}}_{(t+1)}\hat{\boldsymbol{U}}_{(t+1)}^\top - \hat{\boldsymbol{Y}}_{(t+1)}\right) \boldsymbol{x}_{(t+1)} < 0$;

Generate q^k subproblems and add them to the queue:

for each $\boldsymbol{z} \in [q]^k$ **do**

$\mathcal{D}_z := \mathcal{D} \cup \{(\hat{\boldsymbol{U}}_{(t+1)}, \hat{\boldsymbol{Y}}_{(t+1)}, \boldsymbol{x}_{(t+1)}, \boldsymbol{z})\}$

$\mathcal{Q} \leftarrow \mathcal{Q} \cup \{ (\mathcal{D}_z, t+1) \}$

end for

Update Z_{lower} as the minimum of the lower bounds of all unexplored nodes.

end if

end while

return $(\boldsymbol{\Theta}_{\text{opt}}, \boldsymbol{Y}_{\text{opt}}, \boldsymbol{X}_{\text{opt}}, \boldsymbol{U}_{\text{opt}})$

4.4. Incumbent Selection via Alternating Minimization

High-quality feasible solutions dramatically accelerate the convergence of branch-and-bound algorithms such as Algorithm 1, by providing an initial upper bound which allows Algorithm 1's search tree to be aggressively pruned. Accordingly, in our implementation of Algorithm 1 on matrix completion problems with noise, we supply an initial incumbent solution derived from an alternating minimization heuristic. However, we do not provide an initial solution at the root node for noiseless matrix completion problems. Indeed, identifying feasible solutions to noiseless problems is as hard as solving them to optimality.

To implement the alternating minimization heuristic, we follow Burer and Monteiro (2003) in applying the reformulation $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}$. Starting with a solution $\hat{\boldsymbol{U}}^0$, we have:

$$\begin{aligned} \hat{\boldsymbol{V}}^{t+1} &= \underset{\boldsymbol{V}\in\mathbb{R}^{k\times m}}{\arg\min} \quad \frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\left((\hat{\boldsymbol{U}}^t\boldsymbol{V})_{i,j} - A_{i,j}\right)^2 + \frac{1}{2\gamma}\|\hat{\boldsymbol{U}}^t\boldsymbol{V}\|_F^2 \\ \hat{\boldsymbol{U}}^{t+1} &= \underset{\boldsymbol{U}\in\mathbb{R}^{n\times k}}{\arg\min} \quad \frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\left((\boldsymbol{U}\hat{\boldsymbol{V}}^{t+1})_{i,j} - A_{i,j}\right)^2 + \frac{1}{2\gamma}\|\boldsymbol{U}\hat{\boldsymbol{V}}^{t+1}\|_F^2 \end{aligned} \tag{13}$$

and we iteratively solve for $(\hat{\boldsymbol{U}}^t, \hat{\boldsymbol{V}}^t)$ until we either converge to a local minimum or exceed a limit on the number of iterations. To initialize the method, we obtain an initial $\boldsymbol{X}$ by setting $(X_{\text{initial}})_{i,j} = A_{i,j}$ if $(i,j) \in \mathcal{I}$, and 0 otherwise, and recover an initial $\hat{\boldsymbol{U}}^0 \in \mathbb{R}^{n\times k}$ from $\boldsymbol{X}_{\text{initial}}$ as the $\boldsymbol{U}$-factor of the compact SVD of $\boldsymbol{X}_{\text{initial}}$. On convergence, we let $\boldsymbol{X}_{\text{altmin}} := \hat{\boldsymbol{U}}_{\text{end}}\hat{\boldsymbol{V}}_{\text{end}}$ and compute $\boldsymbol{U}_{\text{altmin}}$ via a compact SVD of $\boldsymbol{X}_{\text{altmin}}$, so $\boldsymbol{U}_{\text{altmin}}$ has orthogonal columns.

Since the alternating minimization approach is only guaranteed to converge to a stationary (or locally optimal) solution, the solution obtained at the root node might not be globally optimal. Accordingly, we consider running alternating minimization on subregions defined by nodes of our search tree by requiring that $\boldsymbol{U}$ remains (approximately) within the feasible region defined by the node. After solving the relaxation (3) at a child node and obtaining a relaxed solution $(\boldsymbol{Y}_R, \boldsymbol{U}_R, \boldsymbol{X}_R, \boldsymbol{\Theta}_R)$, we consider rounding $\boldsymbol{Y}_R$ to be the projection matrix $\boldsymbol{Y}_{\text{round}} = \boldsymbol{U}_{\text{round}}\boldsymbol{U}_{\text{round}}^\top$, where $\boldsymbol{U}_{\text{round}}$ is the $\boldsymbol{U}$-factor of a compact SVD of $\boldsymbol{Y}_R$. We initialize an alternating minimization process by $\boldsymbol{U}_{\text{round}}$ and replace (13) with $\hat{\boldsymbol{U}}^{t+1}$ set equal to

$$\underset{\substack{\boldsymbol{U}\in\mathbb{R}^{n\times k},\ \boldsymbol{U}\in\mathcal{P}_{\boldsymbol{U}},\\ \|\boldsymbol{U}_i\|_2^2\le 1,\ \|\boldsymbol{U}_i \pm \boldsymbol{U}_j\|_2^2 \le 2 \quad \forall i,j\in[k].}}{\arg\min} \quad \frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\left((\boldsymbol{U}\hat{\boldsymbol{V}}^{t+1})_{i,j} - A_{i,j}\right)^2 + \frac{1}{2\gamma}\|\boldsymbol{U}\hat{\boldsymbol{V}}^{t+1}\|_F^2, \tag{14}$$

where $\mathcal{P}_{\boldsymbol{U}}$ is a polyhedron composed of linear constraints on $\boldsymbol{U}$ which are derived from disjunctions applied from the root node to this child node, and we impose a second-order cone relaxation of

the constraint $\boldsymbol{U}^\top \boldsymbol{U} \preceq \mathbb{I}$, to approximately restrict alternating minimization within the set of feasible $\boldsymbol{U}$'s defined by the current node without incurring the computational expense of repeatedly solving SDOs, as suggested by Bertsimas et al. (2022).

After convergence, we obtain a feasible solution via $\boldsymbol{X}_{\text{altmin}} := \hat{\boldsymbol{U}}_{\text{end}} \hat{\boldsymbol{V}}_{\text{end}}$ and taking a compact SVD to obtain $\boldsymbol{U}_{\text{altmin}}$. Since we want the subregions which we run this alternating minimization procedure on to be diverse, we select the nodes on which we run alternating minimization randomly, with a probability of selection that degrades with the tree depth. In Section 5, we investigate the trade-off between the improved upper bounds from implementing alternating minimization at certain leaf nodes against its time cost.

5. Numerical Experiments

In this section, we evaluate the numerical performance of our branch-and-bound scheme, implemented in `Julia` version 1.9 using `Mosek` version 10.0 to solve all semidefinite optimization problems. All experiments were conducted on MIT's supercloud cluster (Reuther et al. 2018), which hosts Intel Xeon Platinum 8260 processors. To perform our experiments, we generate synthetic instances of matrix completion and basis pursuit problems, as described in Section 5.1.

We evaluate the effectiveness of the valid inequalities in Section 3 in strengthening the root node relaxation, in Section 5.2. Next, in Section 5.3, we benchmark the performance of different node selection, branching, breakpoint, and incumbent generation strategies for our branch-and-bound algorithm. Finally, we evaluate the scalability of our branch-and-bound algorithm in Section 5.4, in terms of its ability to identify an optimal solution, and find feasible solutions that outperform a naive alternating minimization strategy. To bridge the gap between theory and practice, we make our branch-and-bound implementation available at `github.com/sean-lo/OptimalMatrixCompletion.jl` and our experiments at `github.com/sean-lo/OLRMC_experiments`.

5.1. Generation of Synthetic Instances

We compute a matrix of observations, $\boldsymbol{A}_{\text{full}} \in \mathbb{R}^{n \times m}$, from a low-rank model: $\boldsymbol{A}_{\text{full}} = \boldsymbol{U}\boldsymbol{V} + \epsilon \boldsymbol{Z}$, where the entries of $\boldsymbol{U} \in \mathbb{R}^{n \times k}$, $\boldsymbol{V} \in \mathbb{R}^{k \times m}$, and $\boldsymbol{Z} \in \mathbb{R}^{n \times m}$ are drawn independently from a standard normal distribution, and $\epsilon \geq 0$ models the degree of noise. We fix $\epsilon = 0.1$. We then sample a random subset $\mathcal{I} \subseteq [n] \times [m]$, of predefined size, which contains at least one entry in each row and column of the matrix (see also Candès and Recht 2009, Section 1.1.2). To do so, if the target size $|\mathcal{I}|$ is large enough, we iteratively add random entries until this property is satisfied, which will happen with

less than $|\mathcal{I}|$ draws with high probability. In regimes where $|\mathcal{I}|$ is small, i.e., close to $k(n+m)$, we use a random permutation matrix to sample $k(n+m)$ entries, one in each row and column, directly. We then independently sample the remaining entries of $\mathcal{I}$ to reach the desired size.

For simplicity, in our experiments, we set $m = n$ and only vary the dimension of the problem, n, and the number of observed entries, $|\mathcal{I}|$. Also, to allow for a better comparison across instances of varying size n, we generate one large $N \times N$ matrix $\boldsymbol{A}_{\text{full}}$ and consider its top-left n-by-n submatrices for various values of n, which creates correlations between the instances generated for different values of n—the instances generated for a given size n, however, remain independent since they come from different matrices $\boldsymbol{A}_{\text{full}}$ obtained with different random seeds.

We remark that this data generation process is standard in the literature, and generates matrices from the same distribution as those considered by many other authors (e.g., Candès and Recht 2009, Bertsimas et al. 2022).

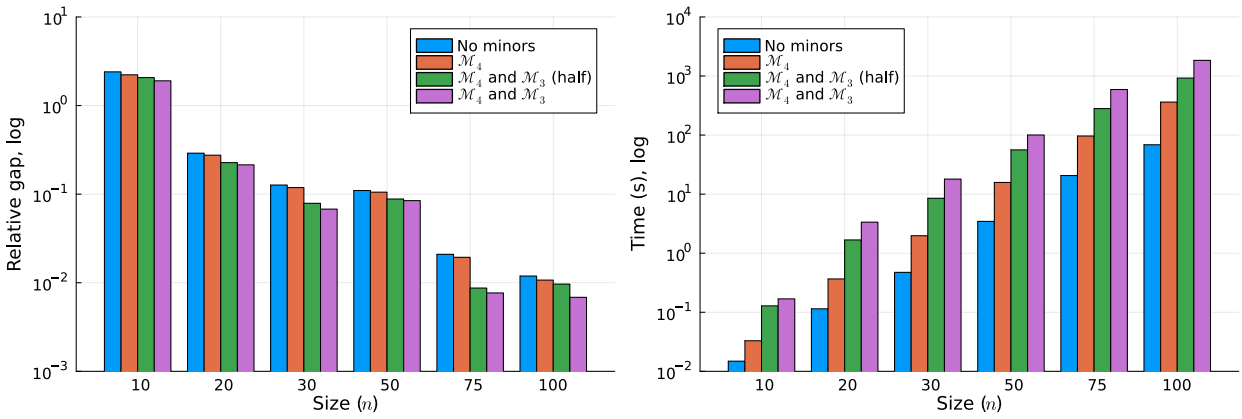
5.2. Root Node: Strengthened Relaxations

In this section, we evaluate the benefit of the valid inequalities presented in Section 3.

As argued in Section 3.2, we can strengthen the matrix perspective relaxation (3) by imposing additional semidefinite constraints (Shor LMIs) on all 2×2 minors of the slices of $\boldsymbol{X}$, $\boldsymbol{X}^t, t \in [k]$. To minimize the impact on tractability, however, we do not impose these constraints for all minors. Instead, we consider two-by-two minors with all four entries present in $\mathcal{I}$ (denoted $\mathcal{M}_4$), or with at least three entries in $\mathcal{I}$ ($\mathcal{M}_3$). We compare the original relaxation (3) with (10) strengthened with Shor LMIs for all minors in $\mathcal{M}_4$, all minors in $\mathcal{M}_4$ and a (random) half of the minors in $\mathcal{M}_3$, and all minors in $\mathcal{M}_4$ and $\mathcal{M}_3$, using two metrics: optimality gap at the root node (the upper bound is obtained via alternating minimization) and computational time for solving the relaxation.

We consider rank-one matrix completion where $n \in \{10, 20, 30, 50, 75, 100\}$, and $|\mathcal{I}| = 2n\log(n)$. Figures 1a and 1b represent the optimality gap at the root node and computational time respectively of using these strengthened relaxations. We observe that, while adding Shor LMIs for minors in $\mathcal{M}_4$ only has a moderate impact on the optimality gap, considering all minors in $\mathcal{M}_4$ and $\mathcal{M}_3$ reduces the optimality gap by one to two orders of magnitude, often solving the problem to optimality, but substantially increases the computational time. Alternatively, introducing Shor LMIs for only half the minors in $\mathcal{M}_4$ and $\mathcal{M}_3$ partially alleviates the computational burden while retaining some benefits in relaxation tightness. This provides us with greater flexibility to control the optimality-tractability trade-off. Figure EC.1 and Table EC.1 in EC.2.1 report the optimality gap and computational time of each method and each instance.

We observe that the relative strength of our relaxation crucially depends on the proportion of missing entries, especially as the rank k increases. Figures 2a–2b show the objective value of our relaxation with Shor LMIs for minors in $\mathcal{M}_4$, relative to the relaxation without any minors, for different values of $|\mathcal{I}|$. We consider $|\mathcal{I}| = Cn$ (resp. $|\mathcal{I}| = Cn^2$) for rank-1 (resp. rank-2) instances in Figure 2a (resp. Figure 2b). In both cases, we observe that the relative benefit from adding Shor LMIs to the root node relaxation increases as C increases. We also observe that the number of entries required to observe an improvement from the Shor LMIs is higher in the rank-2 than in the rank-1 case—notably in its scaling with respect to n. A larger number of observed entries, however, might be computationally more expensive (because there are more Shor LMIs to add) and not available in practice.



(a) Root node relative gap (rank-1)

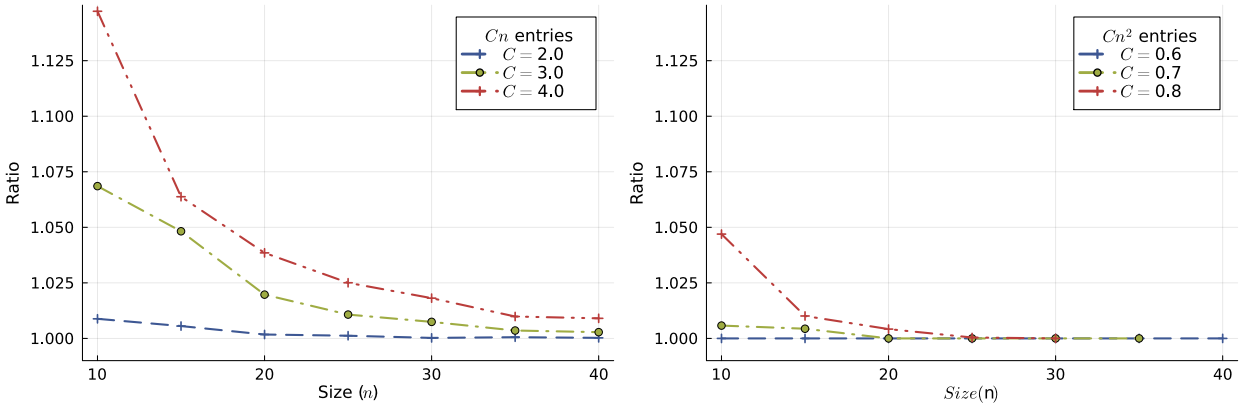
(b) Time taken (rank-1)

Figure 1 **Root node relative gap (a) and time taken (b) at the root node for rank-1 n-by-n matrix completion problems with $2kn\log_{10}(n)$ filled entries, in a regime with low regularization ($\gamma = 80.0$).**

5.3. Branch-and-bound Design Decisions

In this section, we benchmark the efficacy of different algorithmic design options for our branch-and-bound algorithm, including whether to use disjunctive cuts (as developed in Section 2) or a disjunctive scheme based on McCormick inequalities (as described in Section 2.2); the node expansion strategy (breadth-first, best-first, or depth-first); and whether alternating minimization, as described in Section 4.4, should be run at child nodes in the tree.

Tables 1–2 report the final optimality gaps and total computational time of our branch-and-bound scheme with different configurations as we vary n. We impose two termination criteria for these experiments: a relative optimality gap 10^{-4} and a time limit of one hour. Eigenvector disjunctions



(a) (Scaled) lower bounds of $\mathcal{M}_4$ (rank-1) (b) (Scaled) lower bounds of $\mathcal{M}_4$ (rank-2)

Figure 2 **(Scaled) lower bounds of $\mathcal{M}_4$ relaxation relative to the relaxation with no Shor LMIs, with Cn or Cn^2 entries, with low regularization ($\gamma = 80.0$).**

achieve optimality gaps about an order of magnitude smaller than McCormick disjunctions. Running alternating minimization at child nodes also improves the average optimality gap by around an order of magnitude, which provides evidence that the Burer-Monteiro alternating minimization method is not optimal in practice when run from the root node, and can be improved via branch-and-bound. Moreover, the best-first node selection strategy, which comprises selecting the unexpanded node with the smallest lower bound at each iteration, outperforms both breadth-first and depth-first search. This phenomenon is consistent across p and γ; see Tables EC.2–EC.7. Accordingly, we use best-first search with eigenvector disjunctions and alternating minimization run at child nodes throughout the rest of the paper, unless explicitly indicated otherwise.

We remark that in preliminary numerical experiments, we also considered solving matrix completion problems via the multi-tree branch-and-cut approach proposed in Bertsimas et al. (2022), and directly with `Gurobi`'s non-convex QCQP solver. Unfortunately, neither approach was competitive with either the McCormick disjunction or the eigenvector disjunction approach, likely because `Gurobi` does not allow semidefinite constraints to be imposed, and the root node relaxation without semidefinite constraints is often quite weak. Indeed, for instances where $n = 50$ and $r > 1$, neither approach produced a better lower bound than the matrix perspective relaxation.

We also investigate the effect of the number of pieces in our disjunction, q, on the time needed to achieve a 10^{-4} optimality gap for rank-one matrix completion problems, in Figure 3. We do not observe a significant difference between using $q = 2$ or $q = 3$ pieces. However, we find that implementing a disjunctive scheme with $q = 4$ pieces allows our branch-and-bound strategy to converge orders of magnitude faster, across all values of p and γ. We suspect this occurs because

n	Alternating minimization	With McCormick disjunctions			With eigenvector disjunctions		
		Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	6.63×10^{-1}	6.82×10^{-1}	6.84×10^{-1}	5.72×10^{-1}	5.61×10^{-1}	2.65×10^{-1}
10	✓	1.26×10^{-2}	2.18×10^{-2}	1.63×10^{-1}	3.57×10^{-3}	1.01×10^{-2}	5.29×10^{-2}
20	✗	8.24×10^{-2}	8.24×10^{-2}	8.24×10^{-2}	6.44×10^{-2}	6.75×10^{-2}	8.78×10^{-2}
20	✓	9.44×10^{-3}	1.19×10^{-2}	1.19×10^{-2}	1.71×10^{-3}	3.36×10^{-3}	9.52×10^{-3}
30	✗	4.88×10^{-2}	4.88×10^{-2}	4.88×10^{-2}	4.15×10^{-2}	4.23×10^{-2}	4.79×10^{-2}
30	✓	1.36×10^{-2}	1.71×10^{-2}	1.71×10^{-2}	1.71×10^{-3}	2.88×10^{-3}	1.12×10^{-2}
40	✗	4.92×10^{-2}	4.92×10^{-2}	4.92×10^{-2}	4.44×10^{-2}	4.49×10^{-2}	2.14×10^{-2}
40	✓	2.03×10^{-2}	2.03×10^{-2}	2.03×10^{-2}	6.53×10^{-4}	9.50×10^{-4}	1.10×10^{-2}
50	✗	1.50×10^{-2}	1.50×10^{-2}	1.50×10^{-2}	2.68×10^{-2}	2.81×10^{-2}	2.97×10^{-2}
50	✓	6.06×10^{-3}	6.06×10^{-3}	6.09×10^{-3}	2.22×10^{-4}	3.36×10^{-4}	8.83×10^{-3}

Table 1 **Final optimality gap across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 2.0$, $\gamma = 20.0$).**

n	Alternating minimization	With McCormick disjunctions			With eigenvector disjunctions		
		Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	3061.5	3078.2	3240.0	2448.9	2546.7	3240.0
10	✓	1453.9	1603.7	3061.8	1675.8	1974.4	2824.6
20	✗	3060.1	3060.1	3060.1	1939.4	2254.9	2894.2
20	✓	2540.1	2700.1	2700.1	1539.7	1862.5	2370.4
30	✗	3600.3	3600.4	3600.2	3084.1	3155.5	3600.3
30	✓	3060.4	3060.4	3060.3	2285.9	2363.7	3060.4
40	✗	2881.3	2881.1	2881.0	1818.1	1827.6	2881.4
40	✓	2701.1	2701.2	2701.2	1203.4	1290.2	2701.0
50	✗	2703.0	2702.9	2702.8	1749.5	2400.8	2703.7
50	✓	2343.4	2343.1	2343.1	959.32	1299.7	2437.2

Table 2 **Total computational time (s) across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 2.0$, $\gamma = 20.0$).**

four-piece disjunctions include zero as a breakpoint, which breaks some symmetry. However, we also observe this advantage vanishes as n increases; Figure EC.2 confirms this on larger instances.

5.4. Scalability Experiments

Table 2 reveals that the strongest implementation of our algorithm solves rank-one matrix completion problems where $m = n = 50$ in minutes. Accordingly, we now investigate the scalability of our algorithm with this configuration, and its ability to obtain higher-quality low-rank matrices than heuristics. We apply our algorithm to rank-k matrix completion problems on rectangular instances of size $50\times m$, with $km\log_{10}(m)$ observed entries, $k \in \{1,2,3,4,5\}$, and a time limit of three hours.

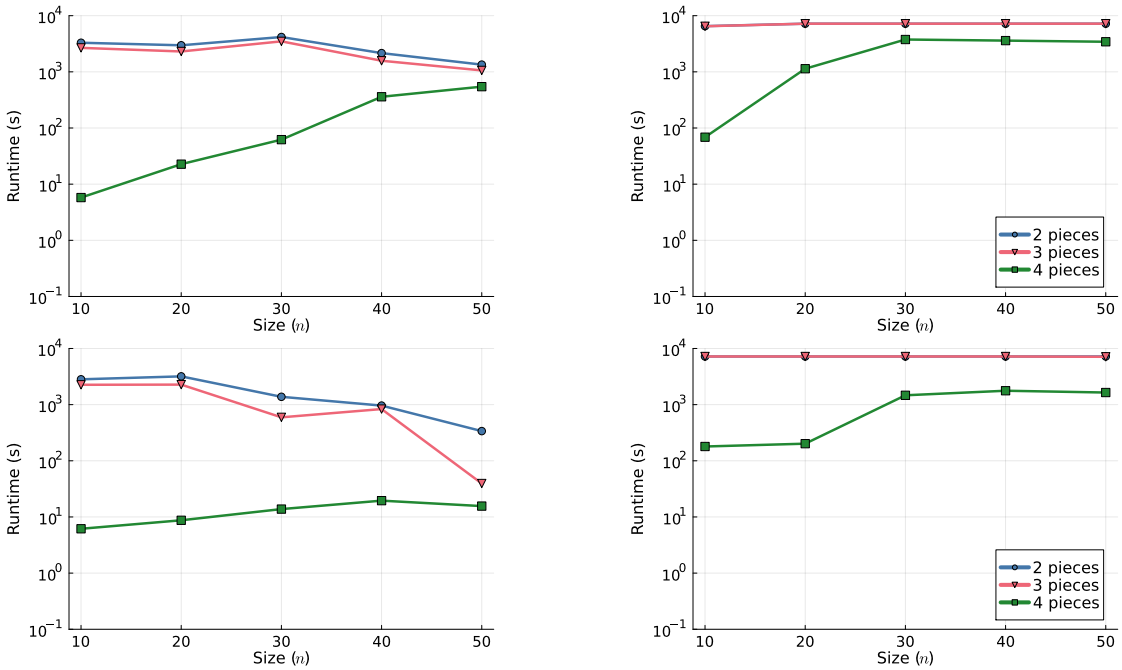


Figure 3 **Comparison of time taken to optimality (relative gap 10^{-4}) for rank-one matrix completion problems with $pn\log_{10}(n)$ filled entries, over different numbers of pieces $q \in \{2,3,4\}$ in upper-approximation. We set $p = 2.0$ (less entries) in the top plots, and $p = 3.0$ (more entries) in the bottom plots. We set $\gamma = 20.0$ (more regularization) in the left plots, and $\gamma = 80.0$ (less regularization) in the right plots.**

Figure 4 depicts the relative gap between the root node relaxation and the best incumbent solution at the root node (left panel), and after applying branch-and-bound for three hours (right panel).Comparing the left and right panels, we observe that our algorithm effectively reduces this gap across all instances, and is especially effective for $k = 1$.

As the rank k increases (and to a lesser extent, as m increases), the difference between the gap at the root node (left panel) and at termination (right panel) is less acute. This behavior can be explained by two factors. First, the root node gap decreases with m due to the sparsity regime, providing less room for improvement; for example, from 38% when $m = 100$ to 9% when $m = 2500$ for rank-2 instances. Second, the time needed to solve each SDP relaxation increases with k (and, to a lesser extent, with m), leading to a smaller number of nodes explored within the time limit (see Figure EC.3). Nevertheless, we show through our results that implementing the alternating minimization in subregions defined by the branch-and-bound tree (from Section 4.4) finds high-quality solutions while requiring a very small number of nodes, and therefore yields significant improvements in the relative gap within the time limit, even for the most challenging instances.

Interestingly, the benefits in terms of optimality gaps are partially due to the fact that branching, combined with alternating minimization within each node, finds better solutions than a standalone

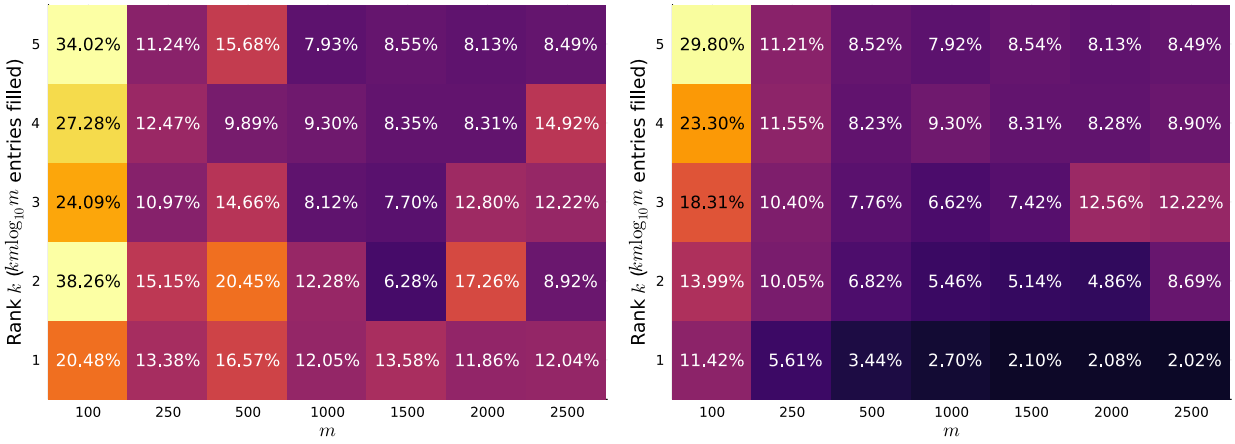


Figure 4 **Comparison of relative optimality gap at root node (left) and after running branch-and-bound for three hours (right) for rank-k matrix completion problems of dimension $50\times m$, with $km\log_{10}(m)$ filled entries, varying m and k, with $\gamma = 120.0$, averaged over 10 random instances.**

Burer-Monteiro method (i.e., the root node solution). To illustrate this, Figure 5 (and Table EC.8) compares the solution found via alternating minimization at the root node against the best solution found by our branch-and-bound scheme after branching for three hours, as measured by the percentage improvement in out-of-sample mean squared error (Figure EC.4 shows the before-and-after values of the out-of-sample MSE). We observe that the improved solutions found by branch-and-bound translate to improvements in MSE across the board. The improvements are consistently in the 1-10% range, and the largest improvements are when m is small and Burer-Monteiro at the root node yields poor-quality solutions. In short, branch-and-bound improves the test set performance of our imputed matrices.

5.5. Summary of Findings from Numerical Experiments

The main findings from our numerical experiments are as follows:

- Section 5.2 demonstrates that our valid inequalities for matrix completion problems strengthens, often significantly, the semidefinite relaxation (3), and indeed routinely improve the root node gap by an order of magnitude or more in the rank-1 case. However, their efficiency depends on the number of observed entries, especially at larger ranks. Future work could investigate more data-cheap relaxations for the rank $k > 1$ cases.
- Section 5.3 investigates the impact of design choices in our branch-and-bound scheme, and demonstrates that eigenvalue-based disjunctions obtain optimality gaps over an order of magnitude smaller than McCormick-based ones in the same amount of computational time.

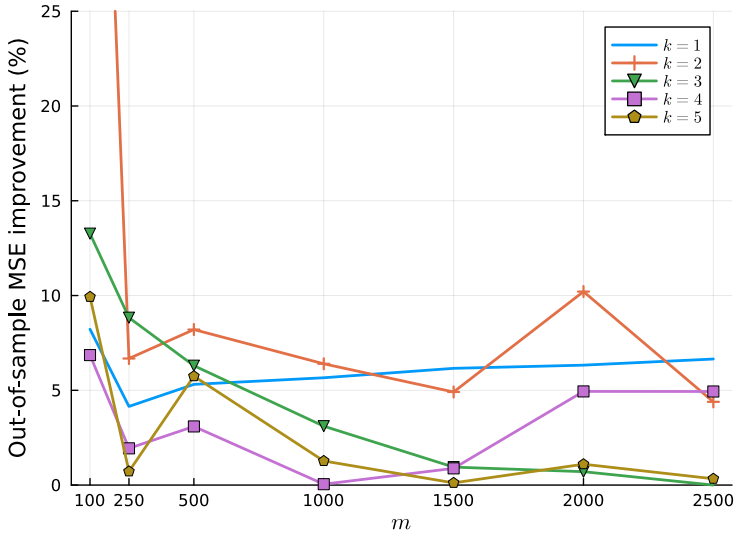


Figure 5 **Percentage improvement in out-of-sample MSE for rank-k $50 \times m$ matrix completion problems, with $km\log_{10}(m)$ filled entries, varying m and k, with $\gamma = 120.0$, averaged over 10 random instances.**

- Section 5.4 demonstrates the scalability of our branch-and-bound scheme for matrix completion problems. While scalability depends on the rank, data regime and extent of regularization, we show that we can solve rectangular $50 \times m$ instances with m as large as 2500 in hours. One necessary limitation of branch-and-bound is memory; our method requires $O(K \min\{m,n\}^2)$ memory, where K is the number of nodes explored.
- We observe that, at the information-theoretic threshold identified by Candès and Recht (2009) with $O(nk \log n)$ entries observed, our branch-and-bound scheme obtains low-rank matrices with an out-of-sample predictive power 1%–50% better than solutions obtained via a Burer-Monteiro heuristic, depending on the rank and the dimensionality of the problem.

6. Conclusion

In this paper, we propose a new branch-and-bound scheme for solving low-rank matrix completion problems to certifiable optimality. The framework considers matrix perspective relaxations and recursively partitions their feasible regions using eigenvector disjunctions. We provide theoretical and empirical evidence to justify the superiority of eigenvector disjunctions compared with the widely used McCormick disjunctions. We further strengthen the semidefinite relaxations via valid inequalities. On rank-1 matrix completion problems, we numerically find that these inequalities routinely improve the root node gap by an order of magnitude.

Altogether, our approach scales to matrix completion problems on $n \times m$ matrices with $\max\{m,n\} = 2500$ and rank up to 5 with optimality guarantees. Compared with the existing and

popular scalable heuristics (which, unfortunately, do not provide any optimality guarantees), our branch-and-bound scheme obtains low-rank matrices with an out-of-sample reconstruction error 1%–50% lower, depending on the rank and the dimensionality of the problem, illustrating the practical value of certifiably optimal methods for solving matrix completion problems.

Acknowledgments

Ryan Cory-Wright gratefully acknowledges the MIT-IBM Research Lab for hosting him while part of this work was conducted.

References

Anstreicher KM (2022) Solving two-trust-region subproblems using semidefinite optimization with eigenvector branching. *Journal of Optimization Theory and Applications* 1–17.

Bell RM, Koren Y (2007) Lessons from the Netflix prize challenge. Technical report, AT&T Bell Laboratories.

Bertsimas D, Cory-Wright R, Pauphilet J (2022) Mixed-projection conic optimization: A new paradigm for modeling rank constraints. *Operations Research* 70(6):3321–3344.

Bertsimas D, Cory-Wright R, Pauphilet J (2023) A new perspective on low-rank optimization. *Mathematical Programming* 202:47–92.

Bhojanapalli S, Neyshabur B, Srebro N (2016) Global optimality of local search for low rank matrix recovery. *Advances in Neural Information Processing Systems* 29.

Boumal N, Voroninski V, Bandeira A (2016) The non-convex Burer-Monteiro approach works on smooth semidefinite programs. *Advances in Neural Information Processing Systems* 29.

Burer S, Monteiro RD (2003) A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming* 95(2):329–357.

Candès EJ, Recht B (2009) Exact matrix completion via convex optimization. *Foundations of Computational Mathematics* 9(6):717–772.

Cifuentes D (2019) Burer-Monteiro guarantees for general semidefinite programs. *arXiv preprint arXiv:1904.07147* 8.

Ebadian A, Nikoufar I, Eshaghi Gordji M (2011) Perspectives of matrix convex functions. *Proceedings of the National Academy of Sciences* 108(18):7313–7314.

Fazel M (2002) *Matrix Rank Minimization with Applications*. Ph.D. thesis, PhD thesis, Stanford University.

Ge R, Lee JD, Ma T (2016) Matrix completion has no spurious local minimum. *Advances in neural information processing systems* 29.

Hardt M (2014) Understanding alternating minimization for matrix completion. *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, 651–660 (IEEE).

Horn RA, Johnson CR (1985) *Matrix Analysis* (Cambridge University Press, New York).

Jain P, Netrapalli P, Sanghavi S (2013) Low-rank matrix completion using alternating minimization. *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, 665–674.

Ke Q, Ferrara E, Radicchi F, Flammini A (2015) Defining and identifying sleeping beauties in science. *Proceedings of the National Academy of Sciences* 112(24):7426–7431.

Khajavirad A (2023) On the strength of recursive McCormick relaxations for binary polynomial optimization. *Operations Research Letters* 51(2):146–152.

Kim J, Tawarmalani M, Richard JPP (2022) Convexification of permutation-invariant sets and an application to sparse principal component analysis. *Mathematics of Operations Research* .

Kocuk B, Dey SS, Sun XA (2018) Matrix minor reformulation and SOCP-based spatial branch-and-cut method for the AC optimal power flow problem. *Mathematical Programming Computation* 10(4):557–596.

Kronqvist J, Bernal DE, Lundell A, Grossmann IE (2019) A review and comparison of solvers for convex MINLP. *Optimization and Engineering* 20(2):397–455.

Li Y, Xie W (2023) On the partial convexification for low-rank spectral optimization: Rank bounds and algorithms. *Optimization Online* .

Lubin M, Vielma JP, Zadik I (2022) Mixed-integer convex representability. *Mathematics of Operations Research* 47(1):720–749.

Ma J, Fattahi S (2023) On the optimization landscape of Burer-Monteiro factorization: When do global solutions correspond to ground truth? *arXiv preprint arXiv:2302.10963* .

McCormick GP (1976) Computability of global solutions to factorable nonconvex programs: Part i—convex underestimating problems. *Mathematical Programming* 10(1):147–175.

Naldi S (2016) Solving rank-constrained semidefinite programs in exact arithmetic. *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, 357–364.

Reuther A, Kepner J, Byun C, Samsi S, Arcand W, Bestor D, Bergeron B, Gadepally V, Houle M, Hubbell M, Jones M, Klein A, Milechin L, Mullen J, Prout A, Rosa A, Yee C, Michaleas P (2018) Interactive supercomputing on 40,000 cores for machine learning and data analysis. *2018 IEEE High Performance extreme Computing Conference (HPEC)*, 1–6 (IEEE).

Saxena A, Bonami P, Lee J (2008) Disjunctive cuts for non-convex mixed integer quadratically constrained programs. *International Conference on Integer Programming and Combinatorial Optimization*, 17–33 (Springer).

Saxena A, Bonami P, Lee J (2010) Convex relaxations of non-convex mixed integer quadratically constrained programs: extended formulations. *Mathematical Programming* 124(1):383–411.

Udell M, Horn C, Zadeh R, Boyd S (2016) Generalized low rank models. *Foundations and Trends® in Machine Learning* 9(1):1–118.

Electronic Companion

EC.1. Omitted Proofs

EC.1.1. Proof of Proposition 1

Proof of Proposition 1 It suffices to show that every feasible solution to Problem (3) can be mapped to a feasible solution to (4) with the same objective value. Therefore, let $(\boldsymbol{Y}, \boldsymbol{U})$ be a feasible solution to Problem (3), let $D = \{ (i,j) : j \in [k] \}$ denote the set of indices that have been disjuncted on, and select a branch of the disjunction such that $\boldsymbol{U}$ satisfies the constraints $U_{i,j} \in [\underline{U}_{i,j}, \overline{U}_{i,j}]$, and $U_{i,j} \in [-1,1]$ for all other indices. Then, we need only construct a $\boldsymbol{V}$ which satisfies the constraints in (4) to establish the result. To construct such a $\boldsymbol{V}$, we recall that $\rho < (n-k)/2$, and thus there are at least $(n-k)/2$ variables $V_{i,j,j}$ which have McCormick constraints reducing to $V_{i,j,j} \in [-1,1]$. Therefore, we can feasibly set $V_{i,j,j} = U_{i,j}^2$ for any $i \in \mathcal{I}(j)$ and $V_{i,j,j} = \frac{1-\sum_{i \in \mathcal{I}(j)} V_{i,j,j}}{n-|\mathcal{I}(j)|}$ for any $i \notin I(j)$, which is a feasible choice by the 2×2 minors of the inequality $\boldsymbol{I} \preceq \boldsymbol{Y} \preceq \boldsymbol{U}\boldsymbol{U}^\top$. Moreover, we have that $\sum_{i=1}^n V_{i,j,j} = 1$.

Similarly, set $V_{i,j_1,j_2} = U_{i,j_1} U_{i,j_2}$ for any $i \in \mathcal{I}(j_1) \cup \mathcal{I}(j_2)$. By assumption, there are at least $n - |\mathcal{I}(j_1) \cup \mathcal{I}(j_2)| \geq 1$ variables V_{i,j_1,j_2} that have not been fixed in this manner, for any $(j_1, j_2) \in [k] \times [k] : j_1 \neq j_2$, and thus we can set $V_{i,j_1,j_2} = \frac{-1}{n-|\mathcal{I}(j_1) \cup \mathcal{I}(j_2)|} \left(\sum_{i \in \mathcal{I}(j_1) \cup \mathcal{I}(j_2)}^n V_{i,j_1,j_2} \right)$ for these indices, and it follows from the fact that $|\boldsymbol{U}_{j_1}^\top \boldsymbol{U}_{j_2}| \leq 1$ that this is a feasible choice of $\boldsymbol{V}$ with $\sum_{i=1}^n V_{i,j_1,j_2} = 0$; observe that we need $\rho < (n-k)/2$ for $n - |\mathcal{I}(j_1) \cup \mathcal{I}(j_2)| \geq 1$. Therefore, $(\boldsymbol{Y}, \boldsymbol{U}, \boldsymbol{V})$ is feasible within one branch of our disjunction, and there is thus no hope of improving upon the root node relaxation without branching on $(n-k)/2 + 1$ variables in a given column at once. □

EC.1.2. Proof of Theorem 1

Proof of Theorem 1 Suppose that at the Lth iterate this procedure has not converged. Then, since $(\boldsymbol{Y}_L, \boldsymbol{U}_L)$ satisfies the disjunction (6) for any $\boldsymbol{x}_l$ such that $l < L$, we have that

$$\sum_{t=1}^k \langle \boldsymbol{x}_l^\top \boldsymbol{U}_l^t, \boldsymbol{x}_l^\top \boldsymbol{U}_L^t \rangle + |(\boldsymbol{U}_l^t - \boldsymbol{U}_L^t)^\top \boldsymbol{x}_l| \geq \langle \boldsymbol{x}_l \boldsymbol{x}_l^\top, \boldsymbol{Y}_L \rangle. \tag{EC.1}$$

But $(\boldsymbol{Y}_l, \boldsymbol{U}_l)$ was not ϵ-feasible, and thus we have that

$$\langle \boldsymbol{x}_l \boldsymbol{x}_l^\top, \boldsymbol{Y}_l \rangle - \sum_{t=1}^k \langle \boldsymbol{x}_l^\top \boldsymbol{U}_l^t, \boldsymbol{x}_l^\top \boldsymbol{U}_l^t \rangle > \epsilon. \tag{EC.2}$$

Adding these inequalities then reveals that

$$\langle \boldsymbol{x}_l \boldsymbol{x}_l^\top, \boldsymbol{Y}_l - \boldsymbol{Y}_L \rangle + \sum_{t=1}^k \langle \boldsymbol{x}_l^\top \boldsymbol{U}_l^t, \boldsymbol{x}_l^\top (\boldsymbol{U}_L^t - \boldsymbol{U}_l^t) \rangle + \sum_{t=1}^k |\boldsymbol{x}_l^\top (\boldsymbol{U}_l^t - \boldsymbol{U}_L^t)| > \epsilon. \tag{EC.3}$$

Next, since $|\boldsymbol{x}_l^\top \boldsymbol{U}_l^t| \leq 1$ by construction, using this identity and taking absolute values allows us to conclude that

$$|\langle \boldsymbol{x}_l \boldsymbol{x}_l^\top, \boldsymbol{Y}_l - \boldsymbol{Y}_L \rangle| + 2 \sum_{t=1}^{k} |\boldsymbol{x}_l^\top (\boldsymbol{U}_l^t - \boldsymbol{U}_L^t)| > \epsilon. \tag{EC.4}$$

Moreover, by applying the Cauchy-Schwarz inequality to both terms in this inequality we obtain $\langle \boldsymbol{x}_l \boldsymbol{x}_l^\top, \boldsymbol{Y}_l - \boldsymbol{Y}_L \rangle \leq \|\boldsymbol{Y}_L - \boldsymbol{Y}_l\|_F$ and $\sum_{t=1}^k |\boldsymbol{x}_l^\top (\boldsymbol{U}_l^t - \boldsymbol{U}_L^t)| \leq \sum_{t=1}^k \|\boldsymbol{U}_l^t - \boldsymbol{U}_L^t\|_2 = \|\boldsymbol{U}_l - \boldsymbol{U}_L\|_{2,1}$. Further, by norm equivalence we have $\|\boldsymbol{U}_l - \boldsymbol{U}_L\|_{2,1} \leq \sqrt{n}\|\boldsymbol{U}_l - \boldsymbol{U}_L\|_F$. Combining these results allows us to conclude

$$\|\boldsymbol{Y}_l - \boldsymbol{Y}_L\|_F + 2\sqrt{n}\|\boldsymbol{U}_l - \boldsymbol{U}_L\|_F > \epsilon. \tag{EC.5}$$

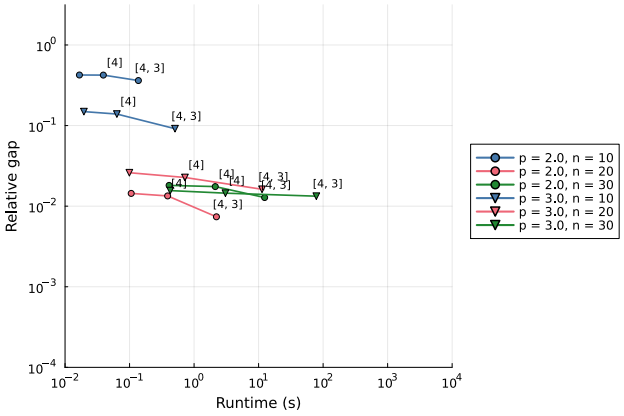
That is, with respect to the decision variables $(\boldsymbol{Y}, \boldsymbol{U})$, our procedure never visits any ball of radius $\frac{\epsilon}{2\sqrt{n}}$ twice. Moreover, the set of feasible $(\boldsymbol{Y}, \boldsymbol{U})$ is bounded via $\mathbb{I} \preceq \boldsymbol{Y} \preceq \boldsymbol{U}\boldsymbol{U}^\top, \mathrm{tr}(\boldsymbol{Y}) \leq \sqrt{k}$. Therefore, we have that $\|\boldsymbol{Y}\|_F \leq \sqrt{k}$ and $\|\boldsymbol{U}\|_F \leq \sqrt{k}$, and thus there are finitely many non-overlapping balls of radius $\frac{\epsilon}{2\sqrt{n}}$ which contain a point in the feasible region. Therefore, for any $\epsilon > 0$, our procedure converges within $L(\epsilon)$ iterations for some $L \in \mathbb{N}$. □

EC.2. Numerical results

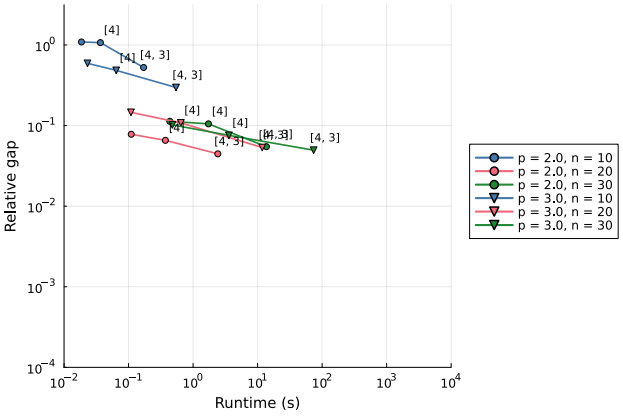
EC.2.1. Root Node: Strengthened Relaxations

Figure EC.1 and Table EC.1 illustrate the trade-off between optimality gap at the root node (the upper bound is obtained via alternating minimization) and computational time, for solving the strengthened relaxation in rank-1 matrix completion. Observe that imposing more Shor LMIs yields tighter but (potentially significantly) more computationally expensive semidefinite relaxations across all sizes, regularization parameters, and sparsity settings. Moreover, imposing constraints on all minors in $\mathcal{M}_4$ and $\mathcal{M}_3$ (as compared to just those in $\mathcal{M}_4$) leads to a larger reduction in optimality gap when $\gamma = 80.0$ (1–3 orders of magnitude) than when $\gamma = 20.0$ (0–2 orders of magnitude). Therefore, the benefits from our strengthened relaxation appear more salient in regimes with less regularization.

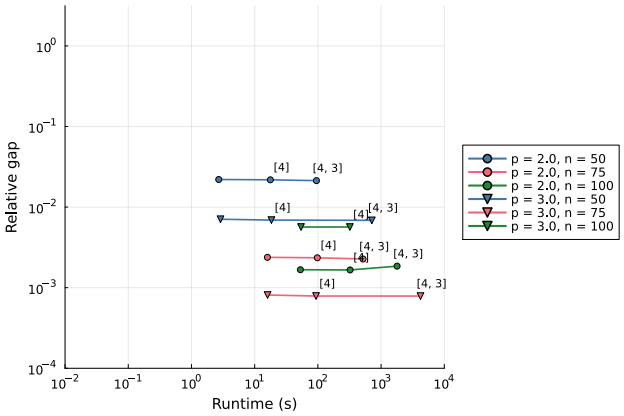
Notably, while including different types of minors ($\mathcal{M}_4$ and $\mathcal{M}_3$ instead of just $\mathcal{M}_4$) provides a way to control the trade-off between computation time and relaxation strength, this can also be done by including a subset of minors in $\mathcal{M}_4 \cup \mathcal{M}_3$. The merits of either approach differ based on sparsity (number of filled entries) and problem size.



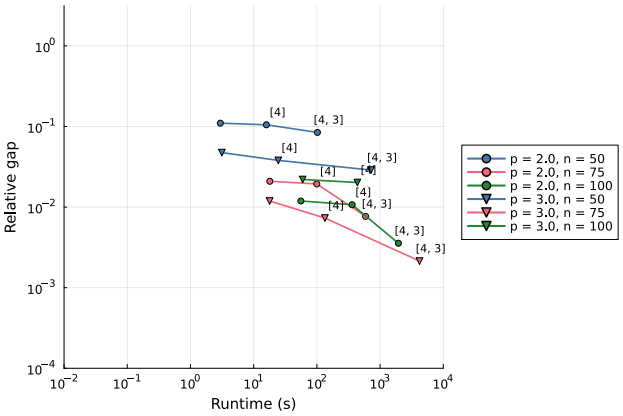
(a) small instances, $\gamma = 20.0$: more regularized



(b) small instances, $\gamma = 80.0$: less regularized



(c) larger instances, $\gamma = 20.0$: more regularized



(d) larger instances, $\gamma = 80.0$: less regularized

Figure EC.1 **Comparison of relative bound gap at the root node against time for the root node relaxations of rank-1 matrix completion problems, varying the number and kind of Shor LMIs added. Each point is the average (geometric mean) of 20 random problem instances with the specified parameters.**

			Time taken (s)			Root node gap		
n	p	γ	None	Shor(4)	Shor(4, 3)	None	Shor(4)	Shor(4, 3)
10	2.0	20.0	0.0167	0.0392	0.137	4.24×10^{-1}	4.23×10^{-1}	3.62×10^{-1}
		80.0	0.0186	0.0366	0.171	1.09×10^{0}	1.07×10^{0}	5.26×10^{-1}
	3.0	20.0	0.0196	0.0636	0.506	1.49×10^{-1}	1.39×10^{-1}	9.15×10^{-2}
		80.0	0.0231	0.0643	0.544	5.92×10^{-1}	4.84×10^{-1}	2.97×10^{-1}
20	2.0	20.0	0.106	0.389	2.22	1.44×10^{-2}	1.34×10^{-2}	7.38×10^{-3}
		80.0	0.110	0.372	2.42	7.80×10^{-2}	6.56×10^{-2}	4.46×10^{-2}
	3.0	20.0	0.0993	0.722	11.3	2.60×10^{-2}	2.27×10^{-2}	1.62×10^{-2}
		80.0	0.109	0.647	11.8	1.46×10^{-1}	1.08×10^{-1}	5.34×10^{-2}
30	2.0	20.0	0.410	2.13	12.4	1.81×10^{-2}	1.75×10^{-2}	1.28×10^{-2}
		80.0	0.436	1.73	13.7	1.13×10^{-1}	1.05×10^{-1}	5.47×10^{-2}
	3.0	20.0	0.427	3.06	78.5	1.56×10^{-2}	1.45×10^{-2}	1.33×10^{-2}
		80.0	0.477	3.59	73.9	1.02×10^{-1}	7.56×10^{-2}	4.95×10^{-2}
50	2.0	20.0	2.70	17.7	94.9	2.20×10^{-2}	2.18×10^{-2}	2.13×10^{-2}
		80.0	2.97	15.9	102.0	1.10×10^{-1}	1.05×10^{-1}	8.45×10^{-2}
	3.0	20.0	2.88	18.4	714.0	7.07×10^{-3}	6.90×10^{-3}	6.85×10^{-3}
		80.0	3.14	24.5	716.0	4.74×10^{-2}	3.80×10^{-2}	2.88×10^{-2}
75	2.0	20.0	15.9	98.0	518.0	2.38×10^{-3}	2.35×10^{-3}	2.27×10^{-3}
		80.0	18.0	99.6	590.0	2.09×10^{-2}	1.94×10^{-2}	7.67×10^{-3}
	3.0	20.0	16.0	93.6	4200.0	8.11×10^{-4}	7.88×10^{-4}	7.87×10^{-4}
		80.0	17.9	134.0	4210.0	1.19×10^{-2}	7.32×10^{-3}	2.14×10^{-3}
100	2.0	20.0	52.9	323.0	1790.0	1.67×10^{-3}	1.66×10^{-3}	1.85×10^{-3}
		80.0	55.8	361.0	1950.0	1.19×10^{-2}	1.07×10^{-2}	3.56×10^{-3}
	3.0	20.0	54.1	321.0	–	5.66×10^{-3}	5.66×10^{-3}	–
		80.0	59.6	437.0	–	2.20×10^{-2}	2.02×10^{-2}	–

Table EC.1 Time taken and relative gap at root node across rank-one matrix completion problems with $pkn\log_{10}(n)$ filled entries, with different Shor LMIs added (averaged over 20 instances per row).

EC.2.2. Branch-and-bound Design Decisions

In this section, we document the performance of our branch-and-bound scheme with various parameter settings: comparing our eigenvalue disjunctions to a naive McCormick-based approach, changing the order of the nodes explored, and including alternating minimization at child nodes of the search tree. We evaluate all combinations of parameter settings, and record their relative optimality gap and time taken (capped at 1 hour) for rank-1 matrix completion problems with regularization $\gamma \in \{20.0, 80.0\}$ and $pn\log_{10}(n)$ filled entries with $p \in \{2.0, 3.0\}$. The results are shown in Tables 1–2, EC.2–EC.3 (less entries $p = 2.0$ and less regularization $\gamma = 80.0$), EC.4–EC.5 (more entries $p = 3.0$ and more regularization $\gamma = 20.0$), and EC.6–EC.7 (more entries $p = 3.0$ and less regularization $\gamma = 80.0$) respectively.

The tables show that eigenvector disjunctions perform consistently better than McCormick disjunctions, and that best-first search on unexplored child nodes is usually a good node selection strategy. They also illustrate the power of performing alternating minimization at (some) child nodes, because better feasible solutions can be found with different initializations, which yield tight upper bounds and thereby accelerating the branch-and-bound procedure.

In Figure EC.2, we observe the effect of the number of pieces used q in our disjunctive scheme on the final relative gap after branch-and-bound for 1 hour on rank-1 matrix completion problems with $n \geq 50$. As the problem size n increases, keeping the number of observed entries at $2n\log_{10}(n)$, the relative advantage of 4-piece disjunctions vanishes – this is likely due to the fact that, as n increases, the time taken for a single semidefinite relaxation increases and using 4-piece disjunctions

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	1.42×10^{0}	1.50×10^{0}	1.50×10^{0}	2.21×10^{0}	2.40×10^{0}	1.91×10^{0}
10	✓	7.97×10^{-2}	1.14×10^{-1}	6.26×10^{-1}	5.39×10^{-2}	8.89×10^{-2}	3.18×10^{-1}
20	✗	3.38×10^{-1}	3.38×10^{-1}	3.38×10^{-1}	5.02×10^{-1}	5.27×10^{-1}	4.09×10^{-1}
20	✓	8.61×10^{-2}	8.77×10^{-2}	8.77×10^{-2}	5.11×10^{-2}	6.50×10^{-2}	2.25×10^{-1}
30	✗	2.34×10^{-1}	2.34×10^{-1}	2.34×10^{-1}	2.34×10^{-1}	2.42×10^{-1}	2.18×10^{-1}
30	✓	1.08×10^{-1}	1.13×10^{-1}	1.13×10^{-1}	4.71×10^{-2}	5.42×10^{-2}	1.04×10^{-1}
40	✗	2.12×10^{-1}	2.12×10^{-1}	2.12×10^{-1}	2.07×10^{-1}	2.11×10^{-1}	2.13×10^{-1}
40	✓	1.05×10^{-1}	1.05×10^{-1}	1.05×10^{-1}	3.08×10^{-2}	3.38×10^{-2}	8.51×10^{-2}
50	✗	7.45×10^{-2}	7.45×10^{-2}	7.45×10^{-2}	1.35×10^{-1}	1.37×10^{-1}	1.39×10^{-1}
50	✓	4.53×10^{-2}	4.53×10^{-2}	4.53×10^{-2}	2.15×10^{-2}	2.32×10^{-2}	6.64×10^{-2}

Table EC.2 Final optimality gap across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 2.0$, $\gamma = 80.0$).

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	3424.9	3425.3	3600.0	3430.1	3600.0	3600.0
10	✓	3162.1	3204.4	3600.0	3274.8	3600.0	3600.0
20	✗	3600.1	3600.1	3600.1	3600.1	3600.1	3600.1
20	✓	3600.1	3600.1	3600.1	3600.1	3600.1	3600.1
30	✗	3600.4	3600.3	3600.3	3600.3	3600.3	3600.4
30	✓	3600.4	3600.3	3600.2	3600.3	3600.3	3600.4
40	✗	3601.0	3601.1	3600.7	3600.8	3600.9	3600.9
40	✓	3601.2	3601.0	3600.8	3600.9	3601.1	3600.8
50	✗	3602.5	3602.1	3602.1	3602.0	3602.1	3602.7
50	✓	3602.4	3602.1	3602.1	3601.9	3602.5	3602.1

Table EC.3 Computational time (s) across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ entries, using McCormick disjunctions (top), eigenvector disjunctions (bottom), averaged over 20 instances ($p = 2.0$, $\gamma = 80.0$).

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	3.35×10^{-1}	3.55×10^{-1}	3.57×10^{-1}	1.73×10^{-1}	1.73×10^{-1}	3.13×10^{-1}
10	✓	3.40×10^{-2}	4.83×10^{-2}	1.49×10^{-1}	7.16×10^{-3}	1.29×10^{-2}	1.15×10^{-1}
20	✗	4.63×10^{-2}	4.63×10^{-2}	4.63×10^{-2}	1.08×10^{-2}	1.16×10^{-2}	1.20×10^{-2}
20	✓	9.58×10^{-3}	2.54×10^{-2}	2.54×10^{-2}	1.79×10^{-4}	9.03×10^{-4}	1.13×10^{-2}
30	✗	6.18×10^{-2}	6.18×10^{-2}	6.18×10^{-2}	9.70×10^{-3}	9.57×10^{-3}	4.65×10^{-2}
30	✓	3.99×10^{-3}	1.56×10^{-2}	1.56×10^{-2}	1.32×10^{-4}	2.22×10^{-4}	3.02×10^{-3}
40	✗	1.27×10^{-2}	1.27×10^{-2}	1.27×10^{-2}	4.43×10^{-3}	4.55×10^{-3}	8.10×10^{-3}
40	✓	9.62×10^{-3}	9.62×10^{-3}	9.62×10^{-3}	1.96×10^{-4}	2.89×10^{-4}	8.44×10^{-3}
50	✗	9.11×10^{-3}	9.11×10^{-3}	9.11×10^{-3}	1.01×10^{-4}	1.07×10^{-4}	2.31×10^{-3}
50	✓	7.13×10^{-3}	7.13×10^{-3}	7.13×10^{-3}	1.01×10^{-4}	1.06×10^{-4}	6.18×10^{-3}

Table EC.4 Final optimality gap across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 3.0$, $\gamma = 20.0$).

introduces more child nodes, leading to an overall increase in computational time. Hence, we only recommend using 4-piece disjunctions for rank-1 matrix completion problems of low to moderate size.

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	2547.4	2553.2	3060.0	2152.4	2118.6	3060.1
10	✓	1871.5	1885.8	3060.0	1524.9	1990.3	2809.7
20	✗	3240.1	3240.1	3240.1	1746.0	2260.7	2880.1
20	✓	2704.7	3060.1	3060.1	1743.3	2262.4	2880.1
30	✗	3060.4	3060.3	3060.3	1340.8	1517.6	2880.4
30	✓	2565.5	2700.5	2700.4	854.99	1429.6	2340.5
40	✗	1981.3	1981.3	1981.3	976.3	1546.2	1981.2
40	✓	1981.4	1981.5	1981.2	597.66	1141.3	1981.4
50	✗	1983.7	1983.8	1983.3	262.55	406.79	1100.1
50	✓	1443.7	1443.9	1443.7	253.22	381.37	959.29

Table EC.5 Computational time (s) across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ entries, using McCormick disjunctions (top), eigenvector disjunctions (bottom), averaged over 20 instances ($p = 3.0$, $\gamma = 20.0$).

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	5.26×10^{-1}	5.93×10^{-1}	5.96×10^{-1}	4.73×10^{-1}	5.40×10^{-1}	5.76×10^{-1}
10	✓	1.03×10^{-1}	1.48×10^{-1}	2.76×10^{-1}	7.03×10^{-2}	1.10×10^{-1}	2.52×10^{-1}
20	✗	4.63×10^{-2}	4.63×10^{-2}	4.63×10^{-2}	1.08×10^{-2}	1.16×10^{-2}	1.20×10^{-2}
20	✓	9.58×10^{-3}	2.54×10^{-2}	2.54×10^{-2}	1.79×10^{-4}	9.03×10^{-4}	1.13×10^{-2}
30	✗	6.18×10^{-2}	6.18×10^{-2}	6.18×10^{-2}	9.70×10^{-3}	9.57×10^{-3}	4.65×10^{-2}
30	✓	3.99×10^{-3}	1.56×10^{-2}	1.56×10^{-2}	1.32×10^{-4}	2.22×10^{-4}	3.02×10^{-3}
40	✗	1.27×10^{-2}	1.27×10^{-2}	1.27×10^{-2}	4.43×10^{-3}	4.55×10^{-3}	8.10×10^{-3}
40	✓	9.62×10^{-3}	9.62×10^{-3}	9.62×10^{-3}	1.96×10^{-4}	2.89×10^{-4}	8.44×10^{-3}
50	✗	9.11×10^{-3}	9.11×10^{-3}	9.11×10^{-3}	1.01×10^{-4}	1.07×10^{-4}	2.31×10^{-3}
50	✓	7.13×10^{-3}	7.13×10^{-3}	7.13×10^{-3}	1.01×10^{-4}	1.06×10^{-4}	6.18×10^{-3}

Table EC.6 Final optimality gap across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 3.0$, $\gamma = 80.0$).

n	Alternating minimization	With McCormick disjunctions			With eigenvector disjunctions		
		Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	3329.9	3346.5	3600.0	3600.0	3600.0	3600.0
10	✓	3228.0	3308.0	3600.0	3600.0	3600.0	3600.0
20	✗	3600.1	3600.1	3600.1	3600.1	3600.1	3600.1
20	✓	3600.1	3600.1	3600.1	3600.1	3600.1	3600.1
30	✗	3600.3	3600.4	3600.3	3600.4	3600.3	3600.3
30	✓	3600.3	3600.4	3600.3	3600.4	3600.2	3600.4
40	✗	3601.2	3600.9	3600.8	3600.9	3600.8	3601.1
40	✓	3601.0	3601.0	3600.8	3601.1	3600.9	3601.0
50	✗	3602.4	3602.2	3602.2	3602.5	3602.0	3602.4
50	✓	3602.3	3602.9	3602.0	3602.3	3602.1	3602.9

Table EC.7 Computational time (s) across rank-one matrix completion problems with $|\mathcal{I}| = pn\log_{10}(n)$ entries, using McCormick disjunctions (top), eigenvector disjunctions (bottom), averaged over 20 instances ($p = 3.0$, $\gamma = 80.0$).

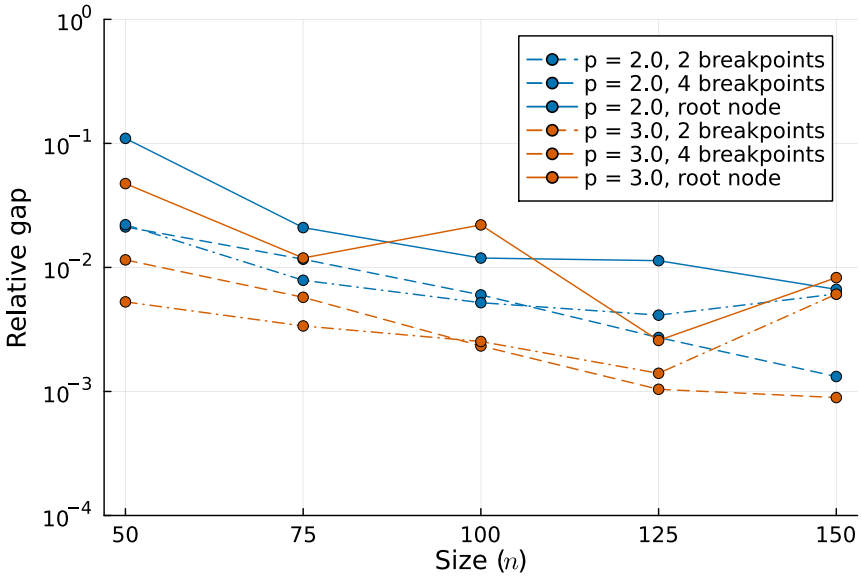


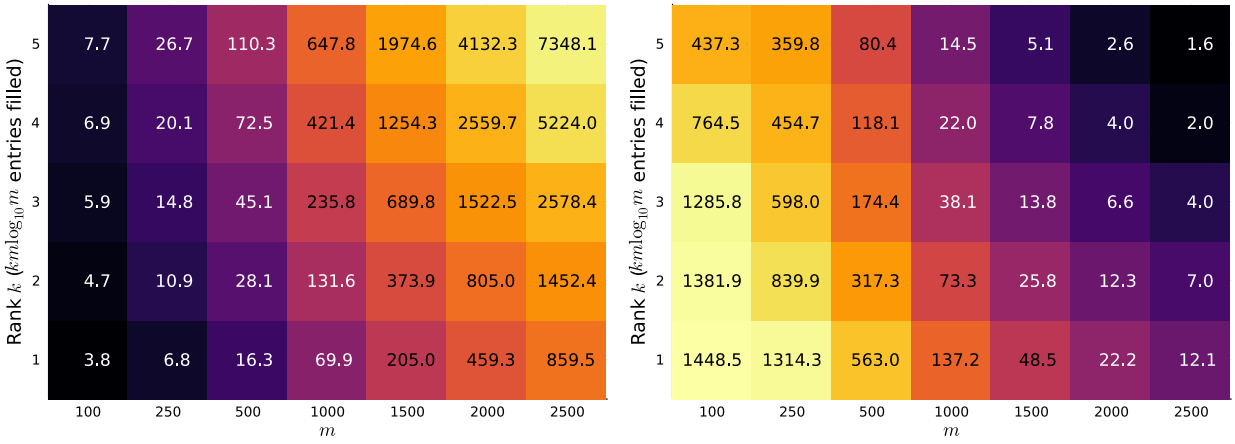
Figure EC.2 Comparison of relative gap (at root node, and after branching with 2-piece and 4-piece disjunctions) for rank-one matrix completion problems with $2n\log_{10}(n)$ filled entries, after 1 hour (averaged over 20 instances per point).

EC.2.3. Scalability Experiments

Here we delve into a more detailed investigation of the scalability of our approach for matrix completion problems to large problem sizes and larger ranks.

Figure EC.3 shows the number of nodes explored through branch-and-bound for three hours and the time taken to solve the semidefinite relaxation at each node. We see that as the time required for solving each SDP relaxation increases according to k and m, the number of nodes explored within 3 hours of branch-and-bound falls correspondingly.

We remark here that we terminate the branch-and-bound procedure for $m = 100$ possibly earlier than the three-hour time limit due to memory availability. Indeed, since we store $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}})$ for every node in the tree (as solutions from ancestors are required to impose disjunctive constraints), the memory requirements of branch-and-bound scale as n^2 multiplied by the number of total nodes in the tree.



(a) SDP relaxation solve time (in seconds) (b) Number of nodes explored

Figure EC.3 **SDP relaxation solve time in seconds (left) and number of branch-and-bound nodes explored (right) for rank-k matrix completion problems of dimension $50\times m$, with $km\log_{10}(m)$ filled entries, varying m and k, with $\gamma = 120.0$, averaged over 10 random instances.**

Figure EC.4 shows the out-of-sample mean-squared error (MSE), at the root node and after branching for three hours, on the large rectangular $50\times m$ instances. These plots show that branching improves the out-of-sample MSE for all values of m, with the most significant improvement in out-of-sample MSE when m is small.

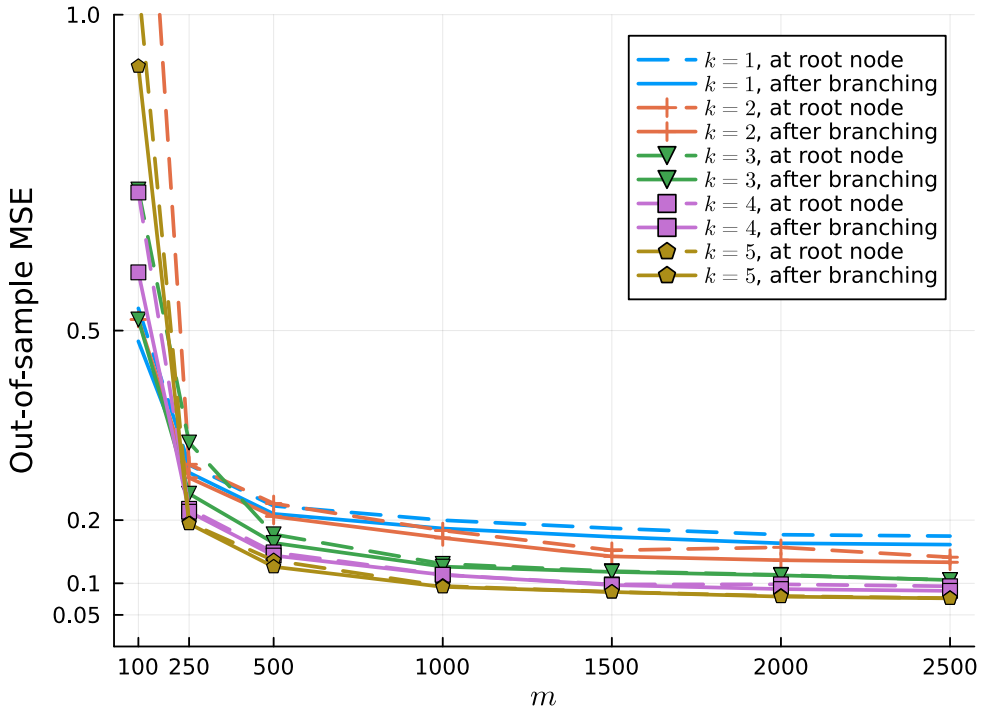


Figure EC.4 **Out-of-sample MSE at the root node and after branching, for rank-k matrix completion problems of dimension $50 \times m$, with $km\log_{10}(m)$ filled entries, varying m and k, with $\gamma = 120.0$, averaged over 10 random instances.**

	m						
k	100	250	500	1000	1500	2000	2500
1	8.22×10^{-2}	4.15×10^{-2}	5.31×10^{-2}	5.66×10^{-2}	6.16×10^{-2}	6.33×10^{-2}	6.65×10^{-2}
2	5.84×10^{-1}	6.68×10^{-2}	8.21×10^{-2}	6.40×10^{-2}	4.91×10^{-2}	1.02×10^{-1}	4.39×10^{-2}
3	1.33×10^{-1}	8.83×10^{-2}	6.30×10^{-2}	3.11×10^{-2}	9.48×10^{-3}	7.07×10^{-3}	-9.29×10^{-6}
4	6.86×10^{-2}	1.94×10^{-2}	3.09×10^{-2}	4.71×10^{-4}	8.82×10^{-3}	4.94×10^{-2}	4.94×10^{-2}
5	9.93×10^{-2}	7.21×10^{-3}	5.74×10^{-2}	1.27×10^{-2}	1.16×10^{-3}	1.10×10^{-2}	3.34×10^{-3}

Table EC.8 **Relative improvement in out-of-sample MSE after branching for 3 hours, for rank-k matrix completion problems of dimension $50 \times m$, with $km\log_{10}(m)$ entries, varying m and k, with $\gamma = 120.0$, averaged over 10 random instances.**