Optimal Low-Rank Matrix Completion: Semidefinite Relaxations and Eigenvector Disjunctions

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 College Business School, London, UK
IBM Thomas J. Watson Research Center, USA
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

Low-rank matrix completion consists of computing a matrix of minimal complexity that recovers a given set of observations as accurately as possible, and has numerous applications such as product recommendation. Unfortunately, existing methods for solving low-rank 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, by reformulating low-rank problems as convex problems over the non-convex set of projection matrices and implementing a disjunctive branch-and-bound scheme that solves them to certifiable optimality. Further, we derive a novel and often tight class of convex relaxations by decomposing a low-rank matrix as a sum of rank-one matrices and incentivizing, via a Shor relaxation, that each two-by-two minor in each rank-one matrix has determinant zero. In numerical experiments, our new convex relaxations decrease the optimality gap by two orders of magnitude compared to existing attempts. Moreover, we showcase the performance of our disjunctive branch-and-bound scheme and demonstrate that it solves matrix completion problems over 150×150 matrices to certifiable optimality in hours, constituting an order of magnitude improvement on the state-of-the-art for certifiably optimal methods.

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

1. Introduction

The presence of missing or noisy entries is a significant challenge faced by data-driven approaches in many real-world contexts. Indeed, incomplete data problems arise in settings as diverse as product recommendation (Bell and Koren 2007), sensor location (Biswas and Ye 2004), and statistical inference (Little and Rubin 2019). For instance, in product recommendation, observed entries correspond to past product-customer 'interactions' (e.g., purchases, views) and are a small fraction of all possible pairs, since most customers only interact with a small proportion of all products. The most common approach for managing missing data when the number of observed entries is small compared to the size of the dataset is to make structural assumptions about the data generation process and solve a data imputation problem to calibrate the model. Perhaps the most popular imputation assumption is that the underlying data structure can be modeled by a low-rank matrix.

Structural low-rank assumptions are relevant because most datasets are approximately low-rank, and we can recover a low-rank matrix after observing a small subset of its entries. Indeed, Udell and Townsend (2019) prove that for a generic matrix $\boldsymbol{A}$ generated by a latent variable model which produces full-rank matrices, the optimal value of

$$\min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}} \ \mathrm{Rank}(\boldsymbol{X}) \text{ s.t. } \|\boldsymbol{X} - \boldsymbol{A}\|_\infty \leq \epsilon \tag{1}$$

grows as $\mathcal{O}(\log(m+n)/\epsilon^2)$. Moreover, a rank-$r$ $n \times m$ matrix has $(n+m)r$ degrees of freedom in a singular value decomposition (SVD). Therefore, one can recover a rank-r matrix after observing a small subset of its entries, of order $mr\log m$ (Candès and Recht 2009).

Formally, the low-rank matrix completion problem with or without noise can be stated as follows:

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 $\boldsymbol{A}$. In the noiseless case (later referred to as the basis pursuit case), this problem can be formulated as:

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

while in the noisy case (later referred to as the matrix completion case) we have:

$$\min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}} \quad \frac{1}{2\gamma}\|\boldsymbol{X}\|_F^2 + \frac{1}{2}\sum_{(i,j) \in \mathcal{I}} \left(X_{i,j} - A_{i,j}\right)^2 \quad \text{s.t.} \quad \mathrm{Rank}\left(\boldsymbol{X}\right) \leq k, \tag{3}$$

where k, γ are hyperparameters which respectively bound the rank of $\boldsymbol{X}$ and regularize $\boldsymbol{X}$ to control its sensitivity to noise, and are to be selected in practice by minimizing a cross-validation error metric such as leave-one-out (c.f. Ding and Chen 2020).

From a generative model perspective, Problems (2)-(3) are consistent with a low-rank model (see, e.g., Candès and Recht 2009) where the matrix of observations $\boldsymbol{A}$ can be decomposed as

$$A_{i,j} = A_{i,j}^{\text{true}} + Z_{i,j}, \quad \forall \ (i,j) \in \mathcal{I}, \tag{4}$$

for an underlying low-rank matrix $\boldsymbol{A}^{\text{true}}$ and a noise matrix $\boldsymbol{Z}$. We have $\boldsymbol{Z} = \boldsymbol{0}$ in the noiseless case, which arises in Euclidean Distance Embedding problems (see, e.g., So and Ye 2007). We refer to Udell et al. (2016) for a review of stochastic generative models addressed via rank constraints.

Problems (2)-(3) have received a great deal of attention since the Netflix Competition (c.f. Bell and Koren 2007) and heuristics methods now find high-quality solutions for very large-scale instances. Unfortunately, to our knowledge, no method currently solves Problems (2)-(3) to provable optimality for m, n beyond 50 and $r > 1$ (Naldi 2016, Bertsimas et al. 2022). Improving the scalability of exact schemes would be both statistically and practically meaningful. Indeed, in many statistical estimation settings, there exists an information-theoretic gap between the minimum number of observations required to impute a matrix via certifiably optimal methods and the number required via any polynomial time method, a phenomenon referred to as the overlap gap conjecture (Gamarnik 2021). Moreover, from a more practical perspective, Bertsimas et al. (2022) have shown that, on average, solving (2) via a provably optimal method gives a low-rank matrix $\boldsymbol{X}$ with a smaller out-of-sample mean squared error (MSE) than the MSE obtained via a heuristic.

In this paper, we revisit (2)-(3) and design a custom branch-and-bound method that solves both problems to provable optimality at a scale of $n = 100$s via a combination of the matrix perspective relaxation (Bertsimas et al. 2023), eigenvector disjunctive cuts (Saxena et al. 2010) we use to define branching regions, and new convex relaxations and presolving techniques. We also show numerically that this approach obtains more accurate solutions than the state-of-the-art method of Burer and Monteiro (2003), in terms of out-of-sample performance.

1.1. Literature Review

We propose a branch-and-bound algorithm that solves Problems (2)-(3) to provable optimality at scale. To develop such an algorithm, we require three ingredients. First, a strategy for recursively partitioning the solution space. Second, a technique for generating high-quality convex relaxations which provide strong lower bounds on each partition. Third, a local optimization strategy for quickly obtaining locally optimal 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), Nguyen et al. (2019), Bertsimas et al. (2022) for general overviews of low-rank optimization:

Exact Methods for Non-Convex Quadratically Constrained Problems A variety of spatial branch-and-bound schemes have been proposed for non-convex quadratically constrained quadratic optimization (QCQO) problems such as (2)-(3) since the work of McCormick (1976), who observed that quadratic products xy can be bounded within a rectangle and used this observation to design a spatial branch-and-bound scheme which recursively partitions the feasible region into rectangles (see also Bao et al. 2009, Misener and Floudas 2012, Speakman and Lee 2017, for related improvements).

Eventually, this progress led to the development of mature global solvers like `BARON` (Sahinidis 1996), `Couenne` (Belotti et al. 2009), `ANTIGONE` (Misener and Floudas 2014), and `Gurobi` (Bixby 2012). Unfortunately, these solvers currently cannot obtain high-quality solutions to QCQOs with more than fifty variables (see Kronqvist et al. 2019, for a benchmark).

Generalizing techniques from mixed-integer optimization, Saxena et al. (2010, 2011) proposed a disjunctive cut scheme that begins with a semidefinite relaxation and iteratively imposes disjunctive cuts, and solved some QCQO problems with up to 50 variables. Their approach was subsequently refined by Anstreicher (2022), who integrated their disjunctive cuts within a branch-and-bound scheme (where the disjunctions are used as branching regions), and solved some numerically challenging two-trust-region problems. In a parallel direction, Kocuk et al. (2018) proposed a branch-and-cut algorithm for rank-one optimal power flow problems and strengthened their relaxations by (a) deriving valid inequalities from the 2×2 minors of a rank-one matrix and (b) taking convex envelopes of appropriate substructures of their problem. In a different context, Das Gupta et al. (2023) reformulated the problem of designing an optimal first-order method as a QCQO problem and obtained orders of magnitude improvement over off-the-shelf solvers by exploiting the problem structure to design strong convex relaxations.

More recently, Bertsimas et al. (2022) proposed introducing orthogonal projection matrices to model the rank of a matrix and demonstrated that this gives rise to strong relaxations which allow off-the-shelf branch-and-cut solvers to solve matrix completion problems over up to 50×50 matrices to provable optimality when $r = 1$. This paper falls within this tradition and provides evidence that although QCQO is intractable in the worst-case, many instances of (2)-(3) which arise in practice can nonetheless be tractably solved to (near) optimality.

Strong Convex Relaxations for Low-Rank Matrix Completion A number of works have proposed solving Problems (2)-(3) via their convex relaxations, originating with Shapiro (1982), Fazel (2002), who proposed replacing a rank minimization objective $\min_{\boldsymbol{X} \in \mathcal{S}^n_+} \operatorname{Rank}(\boldsymbol{X})$ with a trace term $\min_{\boldsymbol{X} \in \mathcal{S}^n_+} \operatorname{tr}(\boldsymbol{X})$ for positive semidefinite factor analysis and matrix completion problems (see also Candès and Recht 2009, Candès and Plan 2010, Recht et al. 2010, for extensions to the asymmetric case). This approach is justified under the (strong) assumption that the eigenvalues of $\boldsymbol{X}$ are bounded by a small constant relative to the magnitude of the problem data, since

$$\operatorname{Conv}\left\{ \boldsymbol{X} \in \mathcal{S}^n_+ : \operatorname{Rank}(\boldsymbol{X}) \leq k,\ \boldsymbol{X} \preceq M\mathbb{I} \right\} = \left\{ \boldsymbol{X} \in \mathcal{S}^n_+ : \operatorname{tr}(\boldsymbol{X}) \leq kM,\ \boldsymbol{X} \preceq M\mathbb{I} \right\}, \qquad (5)$$

and thus replacing the rank of a matrix with its trace is related to taking the convex envelope of a low-rank objective. Indeed, the eigenvalue assumption is equivalent to assuming that a restricted isometry property (RIP; see, e.g., Bhojanapalli et al. 2018) holds. When RIP holds, it is well-documented that trace minimization performs well in practice. Unfortunately, verifying whether the RIP holds is computationally challenging (Bandeira et al. 2013), and when it does not hold, it is unclear how well the trace minimization approach performs. Indeed, Fazel (2002)'s trace minimization approach is ultimately a generalization of the big-M approach from mixed-integer optimization, which can generate weak convex relaxations (Bienstock 2010).

Bertsimas et al. (2023) recently proposed a general procedure for obtaining strong bounds to low-rank problems (see also Wang and Kılınç-Karzan 2022, Li and Xie 2022, 2023, for related attempts). Namely, they combined the orthogonal projection matrix reformulation of Bertsimas et al. (2022) with the matrix analog of perspective functions (c.f. Ebadian et al. 2011) and obtained a new class of convex relaxations for low-rank problems.

Local Optimization Strategies for Matrix Completion and Recovery: Owing to both Problem (2)-(3)'s significant modeling power and the substantial difficulty inherent in solving (2)-(3) to provable optimality, several authors have proposed local optimization algorithms that tend to provide high-quality locally optimal solutions in practice. Perhaps the most significant work in this tradition is Burer and Monteiro (2003, 2005), who proposed implicitly modeling a rank-k constraint via the nonlinear reformulation $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{n \times k}, \boldsymbol{V} \in \mathbb{R}^{m \times k}$, and solving for $\boldsymbol{U}, \boldsymbol{V}$ via an augmented Lagrangian method; recent refinements (Jain et al. 2013, Zheng and Lafferty 2015) use more scalable methods like alternating minimization or gradient descent to converge towards a locally optimal solution.

Interestingly, several authors have shown in different contexts (Boumal et al. 2016, Bhojanapalli et al. 2016, Cifuentes 2019) that if one makes stronger assumptions on the amount of data available to complete $\boldsymbol{X}$, then the Burer-Monteiro approach generically obtains a second-order critical point which is globally optimal, although this is not guaranteed to occur (see Bhojanapalli et al. 2018, Zhang et al. 2018, for counterexamples). This suggests that partitioning the feasible region via branch-and-bound and running a Burer-Monteiro approach on subregions with strong lower bounds is potentially useful for identifying high-quality solutions to matrix completion problems.

We remark that all heuristics reviewed here can be applied to Problem (3) but, in general, rely upon a feasible initialization point and thus may not even provide a feasible solution to Problem (2). This is perhaps unsurprising, since even finding a feasible solution to (2) is $\exists\mathbb{R}$-complete, i.e., as hard as solving a system of polynomial equalities and inequalities (Bertsimas et al. 2022).

1.2. Contributions and Structure

We propose a spatial branch-and-bound scheme which solves medium-sized instances of Problems (2) and (3) to certifiable (near) optimality. Our spatial branch-and-bound solver is open-source and does not rely on any commercial spatial branch-and-bound technology. We hope this code helps other researchers further improve the scalability of certifiably optimal low-rank methods.

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 Problems (2)-(3). Second, by combining an old characterization of rank via determinant minors, which is new to the matrix completion literature, with the well-known Shor semidefinite relaxation, we derive tighter convex relaxations for matrix completion problems, from which we derive presolving strategies and valid inequalities. Third, we design a spatial branch-and-bound algorithm that partitions the feasible region using disjunctive cuts, computes valid lower bounds via the strengthened matrix perspective relaxation, and obtains high-quality feasible solutions by running an alternating minimization heuristic at the root and child nodes. We open-source our implementation as the `OptimalMatrixCompletion.jl` package at `https://github.com/sean-lo/OptimalMatrixCompletion.jl`.

We remark that our approach improves the scalability of certifiably optimal methods for Problems (2)–(3) compared to the state-of-the-art. In particular, we solve instances of Problems (2)-(3) with 100s of variables to provable optimality in minutes or hours, while our prior work Bertsimas et al. (2022) solves problems where $n = 20$ and $r = 1$ to provable optimality in hours, but return 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 our previous attempt 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.

We also improve in terms of obtaining provably near-optimal solutions to Problems (2)–(3). To our knowledge, the state-of-the-art is currently (a) using the matrix perspective relaxation developed in our prior work Bertsimas et al. (2023) as a lower bound, and (b) invoking a strong heuristic such as the method of Burer and Monteiro (2003) as an upper bound. On the other hand, our branch-and-bound scheme bounds at the root node by design, and iteratively improves them as the search tree expands. In numerical experiments (Section 5), we observe that stronger upper and lower bounds translate to an improved out-of-sample MSE compared to the method of Burer and Monteiro (2003) alone. Namely, we observe an out-of-sample MSE improvement from our approach with a runtime budget of one hour of up to 40% for $n \leq 50$ and a ground truth rank of two, and 1–10% for $n \in \{100, 125, 150\}$ and a ground truth rank up to five.

However, we should point out that there exist (much) more scalable heuristics than our method that do not possess optimality guarantees. For instance, the randomized sketching conditional gradient method of Yurtsever et al. (2017) scales to problems where $n = 10^8$; bridging the gap between the scalability of certifiably optimal methods and heuristics remains an ongoing challenge.

The rest of the paper is structured as follows:

In Section 2, we derive Problems (2)-(3)'s matrix perspective relaxations and propose disjunctive cuts for improving the relaxation. With our disjunctive inequalities, we separate an optimal solution to the relaxation from the feasible region using a single eigenvalue separation oracle. On the contrary, we prove in Section 2.5 that one McCormick disjunction alone—the most widely used disjunctions for non-convex QCQO—never separates a relaxed solution from the feasible region.

In Section 3, we leverage a characterization of the rank of a matrix via its determinant minors to develop a novel convex relaxation for strongly convex low-rank problems. Although not necessarily computationally tractable for problem sizes we are interested in, this new relaxation gives rise to two efficient techniques for strengthening our original relaxation. First, a presolving technique that partially completes the input matrix $\boldsymbol{A}$ and generates valid additional equality constraints for Problem (2). Second, a technique for generating valid inequalities for Problems (2)-(3), which we can impose in a column-and-constraint-generation fashion for computational tractability.

In Section 4, we combine the analysis of Sections 2 and 3 to design a spatial branch-and-bound algorithms that converges to a certifiably optimal solution of Problem (2)-(3). 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 spatial branch-and-bound scheme and find that it solves instances of Problems (2)-(3) to certifiable (near) optimality when $n = 250$ 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 MSE up to 50% lower than state-of-the-art heuristics.

1.3. Notation

We let non-boldface characters such as b denote scalars, lowercase bold-faced characters such as $\boldsymbol{x}$ denote vectors, uppercase bold-faced characters such as $\boldsymbol{A}$ denote matrices, and calligraphic uppercase characters such as $\mathcal{Z}$ denote sets. We let $[n]$ denote the running set of indices $\{1, \dots, n\}$. We let $\boldsymbol{e}$ denote the vector of ones, $\mathbf{0}$ denote the vector of all zeros, and $\mathbb{I}$ denote the identity matrix. We let $\mathcal{S}^n$ denote the cone of $n \times n$ symmetric matrices, and $\mathcal{S}^n_+$ denote the cone of $n \times n$ positive semidefinite matrices. We let $\mathcal{Y}_n := \{ \boldsymbol{P} \in \mathcal{S}^n : \boldsymbol{P}^2 = \boldsymbol{P} \}$ denote the set of $n \times n$ orthogonal projection matrices and $\mathcal{Y}^k_n := \{ \boldsymbol{P} \in \mathcal{S}^n : \boldsymbol{P}^2 = \boldsymbol{P},\ \mathrm{tr}(\boldsymbol{P}) \le k \}$ denote the projection matrices with rank at most k. The convex hulls of these sets are well-studied in the optimization literature. In particular, we have $\mathrm{Conv}(\mathcal{Y}_n) = \{ \boldsymbol{P} \in S^n : \mathbf{0} \preceq \boldsymbol{P} \preceq \mathbb{I} \}$ and $\mathrm{Conv}(\mathcal{Y}^k_n) = \{ \boldsymbol{P} \in S^n : \mathbf{0} \preceq \boldsymbol{P} \preceq \mathbb{I},\ \mathrm{tr}(\boldsymbol{P}) \le k \}$ (Overton and Womersley 1992, Theorem 3).

2. Mixed-Projection Formulations, Their Relaxations, and Valid Disjunctions

In this section, we derive Problem (2)-(3)'s semidefinite relaxations in Section 2.1, and refine these relaxations by using eigenvector disjunctive cuts inspired by Saxena et al. (2010) in Sections 2.2-2.4. We show 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 relative poor performance of traditional disjunctions based on McCormick inequalities by showing in Section 2.5 that some McCormick disjunctions over 2^k regions do not improve the semidefinite relaxation.

2.1. Mixed-Projection Formulations and Their Matrix Perspective Relaxations

First, motivated by our prior work (Bertsimas et al. 2022, 2023), we introduce an orthogonal 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 $\boldsymbol{Y}$: $\mathrm{tr}(\boldsymbol{Y}) \leq k$.

By Bertsimas et al. (2023, Theorem 1), we enforce the bilinear constraint implicitly (and in a convex manner) via the domain of a matrix perspective function and can rewrite (2)-(3) as:

$$\min_{\boldsymbol{Y} \in \mathcal{Y}_n^k} \min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}, \boldsymbol{\Theta} \in \mathcal{S}^m} \frac{1}{2\gamma} \mathrm{tr}\left(\boldsymbol{\Theta}\right) + g(\boldsymbol{X}) \text{ s.t. } \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \mathbf{0}, \tag{6}$$

where $g(\boldsymbol{X})$ is a convex function that satisfies the following assumption:

Assumption 1. *The function $g(\boldsymbol{X})$ is one of the following:*

- *In the noiseless case* (2),

$$g(\boldsymbol{X}) = \begin{cases} 0 & \text{if } X_{i,j} = A_{i,j} \ \forall \ (i,j) \in \mathcal{I}, \\ +\infty & \text{otherwise,} \end{cases} \tag{7}$$

and we set $\gamma = 1$ without loss of generality since the optimal solutions are identical for any $\gamma > 0$.

- *In the noisy case* (3),

$$g(\boldsymbol{X}) = \frac{1}{2} \sum_{(i,j) \in \mathcal{I}} \left(X_{i,j} - A_{i,j}\right)^2, \tag{8}$$

and $\gamma > 0$ is a hyperparameter.

By introducing an orthogonal projection matrix $\boldsymbol{Y}$ and invoking Assumption 1, the non-convexity in the rank constraint in Problem (2)-(3) has been isolated within the feasible set of Problem (6) and the fact that $\boldsymbol{Y}$ is a projection matrix—i.e., in the non-convex quadratic constraint $\boldsymbol{Y}^2 = \boldsymbol{Y}$.

Therefore, we now explore strategies for optimizing over this constraint, by first relaxing $\{\boldsymbol{Y} \in \mathcal{S}^n : \boldsymbol{Y}^2 = \boldsymbol{Y}\}$ to its convex hull $\{\boldsymbol{Y} \in \mathcal{S}^n_+ : \boldsymbol{Y} \preceq \mathbb{I}\}$, and subsequently refining this relaxation via disjunctive cuts. We immediately have the following semidefinite relaxation, which is often very tight in practice (see also Bertsimas et al. 2022, Lemma 4):

$$\min_{\boldsymbol{Y} \in \mathrm{Conv}\left(\mathcal{Y}_n^k\right)} \min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}, \ \boldsymbol{\Theta} \in \mathcal{S}^m} \quad \frac{1}{2\gamma} \mathrm{tr}(\boldsymbol{\Theta}) + g(\boldsymbol{X}) \quad \text{s.t.} \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \mathbf{0}. \tag{9}$$

When we solve (9), 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 and, since $\mathcal{Y}_n^k$ is a bounded feasible region, we can improve the relaxation by separating $\hat{\boldsymbol{Y}}$ from the feasible region using disjunctive programming techniques.

2.2. Improving Matrix Perspective Relaxations via Eigenvector Disjunctions

To obtain a computationally tractable set of branching directions, we work on an equivalent lifted version of the semidefinite relaxation (9). Recall that in an optimal solution to Problem (9) where $\boldsymbol{Y}$ is an orthogonal projection matrix with binary eigenvalues, we can decompose $\boldsymbol{Y}$ as $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top$ for an orthogonal matrix $\boldsymbol{U} \in \mathbb{R}^{n \times k}$ with $\boldsymbol{U}^\top \boldsymbol{U} = \mathbb{I}_k$. Therefore, introducing $\boldsymbol{U}$ and relaxing $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top$ to $\boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top$ yields the following equivalent relaxation to (9):

$$\min_{\substack{\boldsymbol{Y} \in \mathrm{Conv}(\mathcal{Y}_n^k) \\ \boldsymbol{U} \in \mathbb{R}^{n \times k}}} \quad \min_{\substack{\boldsymbol{X} \in \mathbb{R}^{n \times m} \\ \boldsymbol{\Theta} \in \mathcal{S}^m}} \quad \frac{1}{2\gamma} \mathrm{tr}\left(\boldsymbol{\Theta}\right) + g(\boldsymbol{X}) \quad \text{s.t.} \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0}, \ \boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top. \tag{10}$$

This lifted formulation involves a semidefinite constraint $\boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top$, which, in its rank-one version, is prevalent in the mixed-integer quadratic optimization literature.

A challenging predicament in separating an optimal solution to Problem (9) from (6)'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. To avoid this predicament, we adapt the lifted approach proposed by Saxena et al. (2010) for general mixed-integer QCQO (see also Dong and Luo 2018) to our lifted formulation (10).

If $\mathrm{Rank}(\hat{\boldsymbol{X}}) \leq k$ in an optimal solution $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ to (10) then $\hat{\boldsymbol{X}}$ also solves the original rank-constrained problem, (2) or (3). Otherwise, the non-convex constraint $\hat{\boldsymbol{Y}} = \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top$ does not hold, which implies that $\hat{\boldsymbol{Y}} \not\preceq \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top$. Equivalently, in this case, there exists a vector $\boldsymbol{x} \in \mathbb{R}^n$ (e.g. a negative eigenvector of $\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}}$) such that $\boldsymbol{x}^\top(\hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}})\boldsymbol{x} < 0, \ \|\boldsymbol{x}\|_2 = 1$. Therefore, we would like to impose the (non-convex) inequality

$$\boldsymbol{x}^\top \boldsymbol{Y} \boldsymbol{x} \leq \boldsymbol{x}^\top \boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{x} = \|\boldsymbol{U}^\top \boldsymbol{x}\|_2^2 \tag{11}$$

via a disjunction. We describe this construction for the rank-one and $k > 1$ cases respectively, in the following two sections. We note that our initial development is similar to Saxena et al. (2010), as well as more recent developments by Fampa and Lee (2021), Rahimian and Mehrotra (2020), Anstreicher (2022), although our extension to projection matrices and where the rank potentially exceeds one is, to our knowledge, new.

2.3. Eigenvector Disjunctions for the Rank-one Case

In the rank-one case, the matrix $\boldsymbol{U}$ is a single column vector and $\boldsymbol{U}^\top \boldsymbol{x}$ is a scalar. Recalling that $\|\boldsymbol{U}^\top \boldsymbol{x}\|_2^2 \leq \|\boldsymbol{x}\|_2^2 = 1$, imposing such a disjunction is equivalent to bounding the function $u \mapsto u^2$ from above over the interval $[-1,1]$. For any $u_0 \in (-1,1)$, we have the following piecewise upper approximation with breakpoints $\{-1, u_0, 1\}$:

$$u^2 \leq \begin{cases} \frac{1-u}{1-u_0} \cdot u_0^2 + \frac{u-u_0}{1-u_0} & \text{if } u \in [u_0, 1] \\ \frac{u+1}{u_0+1} \cdot u_0^2 + \frac{u_0-u}{u_0+1} & \text{if } u \in [-1, u_0] \end{cases} = \begin{cases} u + uu_0 - u_0 & \text{if } u \in [u_0, 1], \\ -u + uu_0 + u_0 & \text{if } u \in [-1, u_0]. \end{cases} \tag{12}$$

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

$$\|\boldsymbol{U}^\top \boldsymbol{x}\|_2^2 \leq \begin{cases} \boldsymbol{x}^\top \hat{\boldsymbol{U}} \boldsymbol{U}^\top \boldsymbol{x} + (\hat{\boldsymbol{U}} - \boldsymbol{U})^\top \boldsymbol{x} & \text{if } \boldsymbol{U}^\top \boldsymbol{x} \in [-1, \hat{\boldsymbol{U}}^\top \boldsymbol{x}], \\ \boldsymbol{x}^\top \hat{\boldsymbol{U}} \boldsymbol{U}^\top \boldsymbol{x} + (\boldsymbol{U} - \hat{\boldsymbol{U}})^\top \boldsymbol{x} & \text{if } \boldsymbol{U}^\top \boldsymbol{x} \in [\hat{\boldsymbol{U}}^\top \boldsymbol{x}, 1]. \end{cases}$$

Accordingly, for $\boldsymbol{U}$ such that $\boldsymbol{U}^\top \boldsymbol{x} \in [-1, \hat{\boldsymbol{U}}^\top \boldsymbol{x}]$ (or, alternatively, $\boldsymbol{U}^\top \boldsymbol{x} \in [\hat{\boldsymbol{U}}^\top \boldsymbol{x}, 1]$), we can safely approximate the inequality (11) by a linear constraint in $(\boldsymbol{U}, \boldsymbol{Y})$. Formally, we have to the following two-term disjunction over an extended formulation of the set of rank-one projection matrices:

$$\left\{ (\boldsymbol{U}, \boldsymbol{Y}) \;\middle|\; \begin{array}{l} -1 \leq \boldsymbol{U}^\top \boldsymbol{x} \leq \hat{\boldsymbol{U}}^\top \boldsymbol{x}, \\ \boldsymbol{x}^\top \hat{\boldsymbol{U}} \boldsymbol{U}^\top \boldsymbol{x} + (\hat{\boldsymbol{U}} - \boldsymbol{U})^\top \boldsymbol{x} \geq \boldsymbol{x}^\top \boldsymbol{Y} \boldsymbol{x} \end{array} \right\} \bigvee \left\{ (\boldsymbol{U}, \boldsymbol{Y}) \;\middle|\; \begin{array}{l} \hat{\boldsymbol{U}}^\top \boldsymbol{x} \leq \boldsymbol{U}^\top \boldsymbol{x} \leq 1, \\ \boldsymbol{x}^\top \hat{\boldsymbol{U}} \boldsymbol{U}^\top \boldsymbol{x} + (\boldsymbol{U} - \hat{\boldsymbol{U}})^\top \boldsymbol{x} \geq \boldsymbol{x}^\top \boldsymbol{Y} \boldsymbol{x} \end{array} \right\}. \tag{13}$$

Moreover, we can strengthen our convex relaxation (10) by imposing this disjunction and optimizing over the two resulting convex problems. We formalize this result in the following proposition (see also Saxena et al. 2010, Section 3):

PROPOSITION 1. *Set $k = 1$ and let $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ be an optimal solution to* (10) *such that $\boldsymbol{x}^\top (\hat{\boldsymbol{U}} \hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}}) \boldsymbol{x} < 0$, $\|\boldsymbol{x}\|_2 = 1$ for some vector $\boldsymbol{x} \in \mathbb{R}^n$. Then, any solution to* (10) *with $\boldsymbol{Y} = \boldsymbol{U} \boldsymbol{U}^\top$ satisfies* (13)*, but $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ does not satisfy* (13)*.*

Proof of Proposition 1 Let $\boldsymbol{U}, \boldsymbol{Y}$ be matrices such that $\boldsymbol{U} \boldsymbol{U}^\top = \boldsymbol{Y}$. Then, we certainly have $\boldsymbol{x}^\top \left(\boldsymbol{Y} - \boldsymbol{U} \boldsymbol{U}^\top \right) \boldsymbol{x} = 0$ for any vector $\boldsymbol{x}$, which implies the disjunction is equivalent to requiring that either $\boldsymbol{U}^\top \boldsymbol{x} \in [-1, \hat{\boldsymbol{U}}^\top \boldsymbol{x}]$ or $\boldsymbol{U}^\top \boldsymbol{x} \in [\hat{\boldsymbol{U}}^\top \boldsymbol{x}, 1]$, which is trivially true.

On the other hand, at the point $\boldsymbol{U} = \hat{\boldsymbol{U}}$ both sides of the disjunction require that $\boldsymbol{x}^\top (\hat{\boldsymbol{U}} \hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}}) \boldsymbol{x} \geq 0$, which contradicts $\boldsymbol{x}^\top (\hat{\boldsymbol{U}} \hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}}) \boldsymbol{x} < 0$. □

REMARK 1. Observe that imposing (13) actually helps enforce $\boldsymbol{U}^\top \boldsymbol{U} = \mathbb{I}$ as well as $\boldsymbol{Y} = \boldsymbol{U} \boldsymbol{U}^\top$. Indeed, for any matrix $\boldsymbol{U}$ and $\boldsymbol{Y} := \boldsymbol{U} \boldsymbol{U}^\top$ such that $\boldsymbol{0} \preceq \boldsymbol{Y} \preceq \mathbb{I}$ and $\mathrm{tr}(\boldsymbol{Y}) = k$, $\boldsymbol{Y}$ has binary eigenvalues, thus is an orthogonal projection matrix, and thus $\boldsymbol{U}^\top \boldsymbol{U} = \mathbb{I}$. In other words, $\boldsymbol{Y} = \boldsymbol{U} \boldsymbol{U}^\top$, $\boldsymbol{0} \preceq \boldsymbol{Y} \preceq \mathbb{I}$ and $\mathrm{tr}(\boldsymbol{Y}) = k$ collectively imply $\boldsymbol{U}^\top \boldsymbol{U} = \mathbb{I}$.

We obtain our two-term disjunction by constructing a piecewise linear upper approximation of $\|\boldsymbol{U}^\top \boldsymbol{x}\|_2^2$ with two pieces (i.e., a single breakpoint). Our disjunction separates the previous solution of the relaxation $(\hat{\boldsymbol{U}}, \hat{\boldsymbol{Y}})$ because the upper approximation is tight at $\hat{\boldsymbol{U}}^\top \boldsymbol{x}$. We can construct tighter approximations by considering multiple breakpoints. For example, we can consider the four breakpoints $\{-1, -|u_0|, +|u_0|, 1\}$, which yields the following three-piece linear upper approximation:

$$u^2 \leq \begin{cases} -u - u|u_0| - |u_0| & \text{if } u \in [-1, -|u_0|], \\ u_0^2 & \text{if } u \in [-|u_0|, |u_0|], \\ u + u|u_0| - |u_0| & \text{if } u \in [|u_0|, 1], \end{cases} \tag{14}$$

while the five breakpoints $\{-1, -|u_0|, 0, |u_0|, 1\}$ lead to:

$$u^2 \leq \begin{cases} -u - u|u_0| - |u_0| & \text{if } u \in [-1, -|u_0|], \\ -u|u_0| & \text{if } u \in [-|u_0|, 0], \\ u|u_0| & \text{if } u \in [0, |u_0|], \\ u + u|u_0| - |u_0| & \text{if } u \in [|u_0|, 1]. \end{cases} \tag{15}$$

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.

2.4. Eigenvector Disjunctions for the Rank $k > 1$ Case

Let us generalize our disjunctions to the case where $k > 1$. In this case, $\|\boldsymbol{U}^\top \boldsymbol{x}\|_2^2 = \sum_{j \in [k]} \|\boldsymbol{U}_j^\top \boldsymbol{x}\|_2^2$, where $\boldsymbol{U}_j$, $j \in [k]$, is the jth column of $\boldsymbol{U}$. Then, we apply our disjunctions to each term $\|\boldsymbol{U}_j^\top \boldsymbol{x}\|_2^2$ separately. For example, applying a piecewise linear upper bound with two pieces and breakpoints $\{-1, \hat{\boldsymbol{U}}_j^\top \boldsymbol{x}, 1\}$ to each term leads to the following disjunction over 2^k regions:

$$\bigvee_{L \subseteq [k]} \left\{ (\boldsymbol{U}, \boldsymbol{Y}) \;\middle|\; \begin{aligned} & \boldsymbol{U}_j^\top \boldsymbol{x} \in [-1, \hat{\boldsymbol{U}}_j^\top \boldsymbol{x}] && \forall\, j \in L, \\ & \boldsymbol{U}_j^\top \boldsymbol{x} \in (\hat{\boldsymbol{U}}_j^\top \boldsymbol{x}, 1] && \forall\, j \in [k] \setminus L, \\ & \boldsymbol{x}^\top \boldsymbol{Y} \boldsymbol{x} \leq \sum_{j \in L} \left(\boldsymbol{x}^\top \boldsymbol{U}_j \hat{\boldsymbol{U}}_j^\top \boldsymbol{x} + (\hat{\boldsymbol{U}}_j - \boldsymbol{U}_j)^\top \boldsymbol{x} \right) \\ & \qquad + \sum_{j \in [k] \setminus L} \left(\boldsymbol{x}^\top \boldsymbol{U}_j \hat{\boldsymbol{U}}_j^\top \boldsymbol{x} + (\boldsymbol{U}_j - \hat{\boldsymbol{U}}_j)^\top \boldsymbol{x} \right) \end{aligned} \right\}. \tag{16}$$

Moreover, we have the following corollary to Proposition 1:

Corollary 1. *Let* $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ *be an optimal solution to* (10) *so that* $\boldsymbol{x}^\top (\hat{\boldsymbol{U}} \hat{\boldsymbol{U}}^\top - \hat{\boldsymbol{Y}}) \boldsymbol{x} < 0$ *for some vector* $\boldsymbol{x} \in \mathbb{R}^n$ *such that* $\|\boldsymbol{x}\|_2 = 1$*. Then, any solution to* (10) *with* $\boldsymbol{Y} = \boldsymbol{U}\boldsymbol{U}^\top$ *satisfies* (16)*, but* $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}}, \hat{\boldsymbol{X}}, \hat{\boldsymbol{\Theta}})$ *does not satisfy* (16)*.*

Corollary 1 reveals that solving (10) and imposing the disjunction (16) in Problem (10) separates the optimal solution to (10) whenever it is infeasible for the original problem (2) or (3).

We close this section by remarking that in extending the disjunction in (13) beyond the rank-one case, we encounter symmetry issues; if $(\boldsymbol{Y}, \boldsymbol{U}, \boldsymbol{X}, \boldsymbol{\Theta})$ is a solution to the relaxation, then so is $(\boldsymbol{Y}, \boldsymbol{U\Pi}, \boldsymbol{X}, \boldsymbol{\Theta})$ for any k-by-k permutation matrix $\boldsymbol{\Pi}$. Hence, Problem (10)'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 partially alleviate these issues by exploring different branching strategies and symmetry-breaking constraints (discussed in Section 4).

2.5. Comparison With McCormick Disjunctions

An alternative approach for strengthening our convex relaxation (10) is to perform a McCormick disjunction on the variables $U_{i,j}$ (McCormick 1976). Formally, we start with box constraints $U_{i,j} \in [-1,1]$ and recursively partition the feasible 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 $\{\, (v,x,y) : v = xy, (x,y) \in [\underline{x}, \overline{x}] \times [\underline{y}, \overline{y}] \,\}$, denoted $\mathcal{M}(\underline{x}, \overline{x}, \underline{y}, \overline{y})$, is given by

$$\left\{\, (v,x,y) : \max\left\{\underline{x}y + \underline{y}x - \underline{x}\underline{y}, \overline{x}y + \overline{y}x - \overline{x}\overline{y}\right\} \le v \le \min\left\{\underline{y}x + \overline{x}y - \overline{x}\underline{y}, \overline{y}x + \underline{x}y - \underline{x}\overline{y}\right\} \,\right\}.$$

Therefore, instead of our disjunctive approach with eigenvector cuts, we can use the aforementioned McCormick relaxation to link $\boldsymbol{U}$ and $\boldsymbol{V}$ within each box. This eventually yields a disjunction over relaxations of the following form:

$$\begin{aligned} \min_{\substack{\boldsymbol{Y} \in \mathrm{Conv}(\mathcal{Y}_n^k),\\ \boldsymbol{U} \in \mathbb{R}^{n \times k}, \boldsymbol{V} \in \mathbb{R}^{n \times k \times k}}} \quad \min_{\boldsymbol{X} \in \mathbb{R}^{n \times m}, \boldsymbol{\Theta} \in \mathcal{S}^m} \quad & \frac{1}{2\gamma} \mathrm{tr}(\boldsymbol{\Theta}) + g(\boldsymbol{X}) \qquad (17) \\ \text{s.t.} \quad & \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0}, \ \boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top, \\ & \sum_{i=1}^n V_{i,j,j} = 1 \ \forall\ j \in [k], \ \sum_{i=1}^n V_{i,j_1,j_2} = 0 \ \forall\ j_1, j_2 \in [k] : \ j_1 \neq j_2, \\ & (V_{i,j_1,j_2}, U_{i,j_1}, U_{i,j_2}) \in \mathcal{M}(\underline{U}_{i,j_1}, \overline{U}_{i,j_2}, \underline{U}_{i,j_2}, \overline{U}_{i,j_2}), \quad \begin{matrix} \forall\ i \in [n], \\ \forall\ j_1, j_2 \in [k]. \end{matrix} \end{aligned}$$

where the last constraint denotes a McCormick envelope over $V_{i,j_1,j_2}, U_{i,j_1}, U_{i,j_2}$ with box bounds $[\underline{U}_{i,j_1}, \overline{U}_{i,j_1}] \times [\underline{U}_{i,j_2}, \overline{U}_{i,j_2}]$. Unfortunately, as we observe in Section 5, this approach performs poorly in practice and we often need to expand millions of nodes to improve the root node relaxation.

We provide insight into why this situation arises via the following result, which demonstrates that, when $n \ge k+2$, disjuncting on a single variable $U_{i,j}$ in each column $j \in [k]$ of $\boldsymbol{U}$ does not improve our root node relaxation (proof deferred to Section EC.1.1):

PROPOSITION 2. *Consider Problem* (17) *where* $n \ge k+2$ *and suppose that for every column* j *we have a disjunction* $\cup_t [\underline{U}^t_{i(j),j}, \overline{U}^t_{i(j),j}]$ *for some index* $(i(j), j)$*, but* $[\underline{U}_{i,j'}, \overline{U}_{i,j'}] = [-1,1]$ *for all other indices* $i' \neq i(j)$*. Then, Problem* (17) *possesses the same optimal objective value as Problem* (10)*.*

The McCormick disjunctions in Proposition 2 contain at least 2^k regions, like our eigenvector disjunctions, yet fail to improve upon the matrix perspective relaxation bound. Indeed, Proposition 2 (and its proof) reveals that we need to disjunct on at least two variables in a given column of $\boldsymbol{U}$ to even have a hope of tightening the lower bound. This multilinear behavior stands in direct contrast to our eigenvector disjunction, which separates the solution to the root node relaxation with a single cut and often improves 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 (Meyer and Floudas 2004, Kocuk et al. 2016, Khajavirad 2023).

Moreover, it challenges the ongoing practice in commercial non-convex solvers of using McCormick, rather than eigenvector, disjunctions *by default*. 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. Non-trivial improvements are only obtainable after branching on multiple variables simultaneously via more computationally intensive schemes such as strong branching (c.f. Achterberg et al. 2005). We return to this topic in Section 4, when we propose our overall branching scheme.

3. Convex Relaxations, Valid Inequalities, and Presolving Strategies

In this section, we invoke a different characterization of low-rank matrices in terms of their determinant minors to derive a new convex relaxation for our problems in Section 3.1, which improves upon the matrix perspective relaxation (9). Since it is computationally significantly more expensive, however, we do not use this improved relaxation in and of itself, but invoke it to design presolving strategies in Section 3.2 and a strong and computationally convex relaxation in Section 3.3. Hence, we tighten the matrix perspective relaxation (9) in a computationally affordable manner.

Our starting point is the observation that the rank of $\boldsymbol{X}$ is fully characterized by the following well-known lemma (see, 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. Indeed, one could design an entirely new approach to certifiably optimal low-rank matrix completion using Lemma 1. Kocuk et al. (2018) successfully applies this idea for optimal power flow problems where the rank $k=1$. This approach appears to be computationally prohibitive for $k>1$, however, since explicitly constraining the determinant of each $(k+1)\times(k+1)$ minor of $\boldsymbol{X}$ requires introducing a separate (semidefinite) moment matrix for each minor. Instead, we decompose $\boldsymbol{X}$ into a sum of rank-one matrices, apply

Lemma 1's characterization to each rank-one matrix's Shor relaxation individually, and combine this characterization with our matrix perspective relaxation to obtain strong convex relaxations.

Our perspective on Lemma 1 is similar to the mixed-integer community's perspective on pure cutting-plane methods in mixed-integer optimization (c.f. Gomory 1958, Balas et al. 1996) and their eventual integration within commercial branch-and-bound solvers. Indeed, initial attempts at developing pure cutting-plane solvers for MIO were not competitive with branch-and-bound schemes since they typically required exponentially many cuts to converge. Eventually, however, various authors proposed introducing a small number of cuts within a branch-and-bound procedure and demonstrated that this accelerated branch-and-bound's performance, often substantially (c.f. Bixby and Rothberg 2007). Similarly, while using Lemma 1 as the basis for a new approach to low-rank optimization is likely computationally prohibitive, we argue throughout this section that Lemma 1 enhances the projection matrix framework, often substantially.

3.1. Convex Relaxations

In this section, we develop a new convex relaxation for low-rank basis pursuit and matrix completion problems, first in the rank-one case for clarity of exposition, and subsequently in the rank-k case. Our approach exploits the characterization of a rank-one matrix in terms of its two-by-two minors and thus could be applied to other low-rank problems.

We proceed by taking the following four steps: First, we introduce a matrix to model the outer product of each vectorized two-by-two minor of $\boldsymbol{X}$ with itself, in particular letting $W_{i,j}$ model $X_{i,j}^2$ in all moment matrices. Second, we use that each two-by-two minor should have zero determinant to eliminate some moment variables. Third, we replace $X_{i,j}^2$ with $W_{i,j}$ in the objective where applicable. Finally, we link $\Theta_{i,j}$ (which, in spirit, models $\sum_{\ell=1}^m X_{i,\ell}X_{j,\ell}$) with the appropriate terms in the Shor matrix to impose appropriate objective pressure in our relaxation and avoid allowing any of the entries $W_{i,j}$ to be set to $+\infty$ without affecting the relaxation.

Formally, we have the following result:

Proposition 3. *The following relaxation is at least as strong as the matrix perspective relaxation* (9) *for basis pursuit and matrix completion problems where* $k=1$*:*

$$\min_{\substack{\boldsymbol{X},\boldsymbol{W}\in\mathbb{R}^{n\times m},\\ \boldsymbol{Y}\in\mathrm{Conv}(\mathcal{Y}_n^1),\\ \boldsymbol{\Theta}\in\mathcal{S}_+^m,\boldsymbol{V}}} \quad h(\boldsymbol{X},\boldsymbol{W},\boldsymbol{\Theta}) \tag{18}$$

$$\text{s.t.} \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0},$$

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

$$\Theta_{j_1,j_2}=\sum_{i\in[n]}V^1_{i,(j_1,j_2)}\ \forall\ j_1<j_2\in[m],\ \ \Theta_{j,j}=\sum_{i\in[n]}W_{i,j},\ \forall\ j\in[m],$$

where the function $h(\boldsymbol{X},\boldsymbol{W},\boldsymbol{\Theta})$ is defined as:

- *In the basis pursuit/noiseless case* (2),

$$h(\boldsymbol{X},\boldsymbol{W},\boldsymbol{\Theta}):=\begin{cases}\frac{1}{2\gamma}\mathrm{tr}(\boldsymbol{\Theta}) & \textit{if } X_{i,j}=A_{i,j}\ \forall(i,j)\in\mathcal{I},\\ +\infty & \textit{otherwise.}\end{cases}$$

- *In the matrix completion/noisy case* (3),

$$h(\boldsymbol{X},\boldsymbol{W},\boldsymbol{\Theta}):=\frac{1}{2\gamma}\mathrm{tr}(\boldsymbol{\Theta})+\frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\left(A^2_{i,j}-2X_{i,j}A_{i,j}+W_{i,j}\right).$$

Observe that Proposition 3 relies on the presence of a $\mathrm{tr}(\boldsymbol{\Theta})$ term, derived from the strongly convex $\|\boldsymbol{X}\|_F^2$ term, to apply objective pressure so that the relaxation is non-trivial. Indeed, without this term, we can set $W_{i,j}$ to be arbitrarily large for any $(i,j)\notin\mathcal{I}$, potentially weakening the relaxation substantially. Accordingly, we refer to (18) as a relaxation designed for strongly convex low-rank problems here and throughout the paper.

Remark 2. As can be deduced from Beck (2007) (see also Wang and Kılınç-Karzan 2022, Section 1.1.2), Problem (18)'s relaxation is tight for rank-one two-by-two matrix completion and basis pursuit problems of the form

$$\min_{X_{i,j},X_{i,\ell},X_{k,j},X_{k,\ell}} g(\boldsymbol{X}) \quad \text{s.t.} \quad X_{i,j}X_{k,\ell}=X_{i,\ell}X_{k,j}, \tag{19}$$

where $\mathcal{I}$, the set of observed entries of $\boldsymbol{A}$, is arbitrary. Therefore, we should expect that applying a Shor relaxation to each two-by-two minor separately in conjunction with the matrix perspective relaxation yields a strong overall relaxation in practice. Indeed, the same phenomena has been observed in several different contexts for mixed-integer nonlinear optimization problems (c.f. Frangioni et al. 2020, Wei et al. 2022).

Proof of Proposition 3 This follows immediately from the fact that any feasible solution to (18) is feasible in (9) with the same objective value. □

Section 5 augments Proposition 3 by providing many examples where (18) provides a strictly stronger lower bound than (9), sometimes to the extent that, when combined with a feasible solution from a heuristic, the bound gap from (18) is an order of magnitude smaller than that from (9).

We now generalize this relaxation to matrix completion and basis pursuit problems where $k>1$ by decomposing $\boldsymbol{X}=\sum_{t=1}^k\boldsymbol{X}^t$, replacing $\boldsymbol{X}$ with $\sum_{t=1}^k\boldsymbol{X}^t$ everywhere applicable, and requiring that each slice $\boldsymbol{X}^t$ (in spirit, a rank-one matrix) satisfies our aforementioned Shor relaxation of the 2×2 minor characterization. Moreover, we model the term $X_{i,j}^2=\left(\sum_{t=1}^k X^t_{i,j}\right)^2=\sum_{t\in[k]}(X^t_{i,j})^2+2\sum_{t'<t}X^t_{i,j}X^{t'}_{i,j}$ in the objective by also performing a Shor relaxation on the moment matrix generated by the vector $\begin{pmatrix}1 & X^1_{i,j} & X^2_{i,j} & \dots & X^k_{i,j}\end{pmatrix}$ and selecting the appropriate off-diagonal terms to model each product $X^{t_1}_{i,j}X^{t_2}_{i,j}$. All in all, we have the following result:

Corollary 2. *The following relaxation is at least as strong as the matrix perspective relaxation for matrix completion and basis pursuit problems where* $k \in [\min(n,m)]$*:*

$$\min_{\substack{\boldsymbol{X}^t, \boldsymbol{W}, \boldsymbol{W}^t \in \mathbb{R}^{n\times m} \ \forall t\in[k], \\ \boldsymbol{Y}\in \mathrm{Conv}(\mathcal{Y}^k_n), \\ \boldsymbol{\Theta}\in \mathcal{S}^m_+, \boldsymbol{V}}} \quad h\left(\sum_{t\in[k]} \boldsymbol{X}_t, \boldsymbol{W}, \boldsymbol{\Theta}\right) \tag{20}$$

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

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

$$\begin{pmatrix} 1 & X^1_{i,j} & X^2_{i,j} & \dots & X^k_{i,j} \\ X^1_{i,j} & W^1_{i,j} & H^{1,2}_{i,j} & \dots & H^{1,k}_{i,j} \\ X^2_{i,j} & H^{1,2}_{i,j} & W^2_{i,j} & \dots & H^{2,k}_{i,j} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ X^k_{i,j} & H^{1,k}_{i,j} & H^{2,k}_{i,j} & \dots & W^k_{i,j} \end{pmatrix} \succeq \boldsymbol{0}, \quad \forall\ i\in[n],\ j\in[m],$$

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

As we observe numerically in Section 5, this relaxation is extremely powerful in practice for $k = 1$, often yielding bound gaps of less than 1% when contrasted against the best solution found via an alternating minimization heuristic. Unfortunately, it is also computationally prohibitive to solve this relaxation at scale. Accordingly, we use information from this relaxation to strengthen our matrix perspective relaxation (9). This is the topic of the next two subsections.

3.2. Presolving Techniques for Basis Pursuit

In this section, we develop a suite of presolving techniques for low-rank basis pursuit problems which exploits Lemma 1's characterization to eliminate variables and impose additional constraints. We remark that the idea of presolving basis pursuit problems by filling in entries fully prescribed by a determinant minor appeared in Nan (2009), and the idea of solving mixed-integer optimization problems more efficiently by preprocessing them is widely known; see Gamrath et al. (2015) for a review. However, to our knowledge, the more general presolving technique for basis pursuit and matrix completion problems we present here is new.

Rank-One Basis Pursuit: We consider the two-by-two minors in rank-one basis pursuit problems, before generalizing our analysis to matrix completion problems. In such a two-by-two minor:

$$\begin{pmatrix} X_{i,j} & X_{i,\ell} \\ X_{k,j} & X_{k,\ell} \end{pmatrix},$$

we know between zero and four entries of the minor, and by Lemma 1 have the governing equation $X_{i,j}X_{k,\ell} = X_{i,\ell}X_{k,j}$. If we know all or none of the entries in the minor then the minor is uninteresting, since we either have no entries to complete or no information with which to complete entries.

If there is one missing entry (e.g., $X_{k,\ell}$) in 2×2 minor, then by the determinant minor equation we can compute its value, $X_{k,\ell} = \frac{A_{i,\ell}A_{k,j}}{A_{i,j}}$—assuming $A_{i,j} \neq 0$; if $A_{i,j} = 0$ then this minor gives us no information about $X_{k,\ell}$ and we cannot presolve it—without even solving an optimization problem. Accordingly, for rank-one basis pursuit problems, we run a presolving step to inspect all 2×2 minors and fill the entries that can be computed in this manner.

We can also derive valid equality constraints when there are exactly two missing entries that belong to the same row or column. Assume that the two missing entries belong to the same column and are $X_{k,\ell}$ and $X_{i,\ell}$ (by transposition, we can treat the case where they belong to the same row in a similar fashion). Since they appear in different terms of the determinant constraint, they must satisfy the following linear relationship:

$$X_{k,\ell} = X_{i,\ell}\,\frac{A_{k,j}}{A_{i,j}}.$$

In our numerical algorithm, we consider imposing these additional linear constraints at the root node of our branch-and-bound scheme, after having completed the minors with one missing entry. A subtlety in our implementation is that if an equality constraint links $X_{k,\ell}$ and $X_{i,\ell}$ and a second constraint links $X_{i,\ell}$ and $X_{t,\ell}$ then $X_{k,\ell}$ and $X_{t,\ell}$ are already linked, and imposing an additional equality constraint is redundant. So, we only impose constraints not implied by other constraints.

If there are two missing entries that belong to a different row and a different column or if three entries are missing, we cannot presolve one of the missing entries or derive any additional valid convex constraint. We now extend our presolving strategy to rank-two problems and suggest how this approach can be further extended to $k > 2$.

Rank-Two Basis Pursuit: In rank-two basis pursuit problems, we are interested in deriving valid inequalities from 3×3 minors of the form

$$\begin{pmatrix} X_{i_1,j_1} & X_{i_1,j_2} & X_{i_1,j_3} \\ X_{i_2,j_1} & X_{i_2,j_2} & X_{i_2,j_3} \\ X_{i_3,j_1} & X_{i_3,j_2} & X_{i_3,j_3} \end{pmatrix},$$

which must have determinant zero in any rank-two solution, by Lemma 1. Therefore, we now have the following governing equation

$$\begin{aligned} &X_{i_1,j_1}X_{i_2,j_2}X_{i_3,j_3}+X_{i_1,j_2}X_{i_2,j_3}X_{i_3,j_1}+X_{i_1,j_3}X_{i_2,j_1}X_{i_3,j_2} \\ &\qquad -X_{i_1,j_1}X_{i_2,j_3}X_{i_3,j_2}-X_{i_1,j_2}X_{i_2,j_1}X_{i_3,j_3}-X_{i_1,j_3}X_{i_2,j_2}X_{i_3,j_1}=0. \end{aligned} \tag{21}$$

We now consider some special cases of this constraint that lend themselves readily to presolving.

Analogously to the rank-one case, if there is exactly one missing entry in any 3×3 minor then we can simply compute its value by solving (21). As in the rank-one case, we can loop over all 3×3 minors and apply this presolving procedure wherever possible before running branch-and-bound. We remark that the same procedure can be applied in the rank-k case, although the probability that a given minor has exactly one missing entry degrades with k; see also Section EC.2 for a more detailed description and analysis of our presolving strategy.

Alternatively, if there are at most three missing entries in the same row or column of a 3×3 minor, it is possible to impose valid equalities in the problem's original space without resorting to a Shor relaxation. For example, if we have a sparsity pattern of the form

$$\begin{pmatrix} * & * & * \\ \checkmark & \checkmark & \checkmark \\ \checkmark & \checkmark & \checkmark \end{pmatrix},$$

where "$*$" denotes a missing entry and "$\checkmark$" denotes a known entry, then (21) becomes a linear equality. More generally, if at most one row or column in a $(k+1)\times(k+1)$ minor is missing in a rank-k problem then we can apply the same approach to derive valid equality constraints.

3.3. Partial Convex Relaxations

In this section, we develop more scalable yet less tight semidefinite relaxations than those proposed in Section 3.1, by imposing a subset of the semidefinite constraints in the computationally intensive relaxation (20) and augmenting the relaxation with quadratic constraints. The primary motivation for this development is that, at scale, the computationally intensive relaxation (20) is stronger than the matrix perspective relaxation, but is also computationally intractable. Accordingly, a tractable version of this relaxation is desirable in practice. We remark that similar ideas of partial Lasserre relaxations have already been successfully implemented in the context of mixed-integer optimization (c.f. Kronqvist et al. 2021).

To develop our relaxation, we begin with the following problem, which is equivalent to (9) but also includes each slice $\boldsymbol{X}^t$ and variables $W_{i,j}$ to model $X_{i,j}^2$ for each (i,j):

$$\min_{\substack{\boldsymbol{X}^t\in\mathbb{R}^{n\times m}\ \forall\ t\in[k],\ \boldsymbol{W}\in\mathbb{R}^{n\times m},\\ \boldsymbol{Y}\in\mathrm{Conv}(\mathcal{Y}_n^1),\ \boldsymbol{\Theta}\in\mathcal{S}_+^m}} \quad h\left(\sum_{t\in[k]}\boldsymbol{X}_t,\boldsymbol{W},\boldsymbol{\Theta}\right) \tag{22}$$

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

$$\Theta_{j,j} = \sum_{i\in[n]} W_{i,j}, \quad \left(\sum_{t\in[k]} X^t_{i,j}\right)^2 \leq W_{i,j}, \quad \forall\, i\in[n],\ j\in[m].$$

Given a solution to this relaxation, we proceed in the following way: 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:

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

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

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

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 (6) by recursively solving its semidefinite relaxation (10), strengthened by the techniques described in Section 3, and partitioning the feasible region via the eigenvector disjunctions derived in Section 2. Our approach relates to the approach proposed in Anstreicher (2022) for trust-region problems, although we address a different optimization problem using a different convex relaxation at the root node, and consider letting the rank exceed one. We first provide a high-level description of our approach and relevant pseudocode before describing our branching and incumbent generation strategies in more detail; see also Belotti et al. (2013) and references therein for a general theory of nonlinear branch-and-bound, and our associated `GitHub` repository for our full implementation.

4.1. Pseudocode and Convergence Results

At the root node, we solve the matrix perspective relaxation (10), possibly strengthened with the presolving techniques from Section 3.2 and some of the valid inequalities from Section 3.3.

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 that $\hat{\boldsymbol{Y}} \succeq \hat{\boldsymbol{U}}\hat{\boldsymbol{U}}^\top$, we have that $\hat{\boldsymbol{Y}}$ is, up to a tolerance of ϵ, a projection matrix and solves (6) 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 we apply a disjunction, e.g., (16), as in Section 2. This disjunction 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 $(\hat{\boldsymbol{Y}}, \hat{\boldsymbol{U}})$. All in all, 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 three 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 (6)'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 our continuous relaxation is infeasible, as may occur in basis pursuit problems. Third, 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 therefore do not branch.

We describe our overall branch-and-bound scheme in Algorithm 1; implementation details will be made clear throughout the rest of 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 (10) and imposes a disjunctive cut of the form (16), according to a most negative eigenvector of

$\boldsymbol{U}\boldsymbol{U} - \boldsymbol{Y}^\top$, an optimal solution to (10) 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. Our proof technique is fairly standard and bears resemblance to the proof of convergence of other cutting-plane methods (see, e.g., Mutapcic and Boyd 2009)—proof deferred to Section EC.1.2:

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

- *For each $t \in \mathbb{N}$, set $(\boldsymbol{Y}_t, \boldsymbol{U}_t)$ according to the optimal solution of* (10)*, possibly with disjunctive cuts of the form* (16)*.*
- *If $\lambda_{\min}(\boldsymbol{U}_t\boldsymbol{U}_t - \boldsymbol{Y}_t) \geq -\epsilon$ then terminate.*
- *Else, impose the disjunctive cut* (16) *in* (10) *with the eigenvector $\boldsymbol{x}_t$, where*

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

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 (6)*. Moreover, suppose we set $\epsilon \to 0$. Then, any limit point of $\{\boldsymbol{Y}_t\}_{t=1}^\infty$ solves* (6)*.*

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 (Wolsey and Nemhauser 1999). The three node selection heuristics which 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). One could also consider other node selection rules, such as selecting the node with disjunctions such that the solution to its parent node violates its disjunctions maximally (most infeasible node); see Belotti et al. (2013) for a review. However, in practice, such branching rules are usually less efficient than the aforementioned rules (Achterberg et al. 2005).

Breadth-first and best-first search ensure that the overall lower bound increases at most iterations, provided no ties exist. As argued by Lawler and Wood (1966), best-first search is potentially

Algorithm 1 Branch-and-bound scheme for low-rank matrix completion problems (10)

```
Initialize $\mathcal{D} = \varnothing$, $t = 0$, $Z_{\text{lower}} = -\infty$, $Z_{\text{upper}} = +\infty$;
Initialize a queue of problems derived from Problem (10), $\mathcal{Q} = \{\, (\mathcal{D}, t) \,\}$;
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 the 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 $Z$:
        $\min_{\substack{\boldsymbol{\Theta} \in \mathcal{S}^m \\ \boldsymbol{Y} \in \mathcal{S}^n \\ \boldsymbol{X} \in \mathbb{R}^{n \times m} \\ \boldsymbol{U} \in \mathbb{R}^{n \times k}}} \quad \frac{1}{2} \sum_{(i,j) \in \mathcal{I}} (X_{i,j} - A_{i,j})^2 + \frac{1}{2\gamma} \mathrm{tr}(\boldsymbol{\Theta})$
        such that $\begin{pmatrix} \boldsymbol{Y} & \boldsymbol{X} \\ \boldsymbol{X}^\top & \boldsymbol{\Theta} \end{pmatrix} \succeq \boldsymbol{0}, \quad \boldsymbol{0} \preceq \boldsymbol{Y} \preceq \mathbb{I}, \quad \mathrm{tr}(\boldsymbol{Y}) \leq k, \quad \begin{pmatrix} \boldsymbol{Y} & \boldsymbol{U} \\ \boldsymbol{U}^\top & \mathbb{I} \end{pmatrix} \succeq \boldsymbol{0},$
        $\left. \begin{aligned} & \boldsymbol{U}_{\boldsymbol{j}}^\top \boldsymbol{x} \in [b_{z_j}, b_{z_j+1}] \quad \forall\, j \in [k] \\ & \langle \boldsymbol{Y}, \boldsymbol{x}\boldsymbol{x}^\top \rangle \leq \sum_{j=1}^{k} \overline{f}_{z_j;q}(\boldsymbol{U}_{\boldsymbol{j}}^\top \boldsymbol{x};\ \hat{\boldsymbol{U}}_{\boldsymbol{j}}^\top \boldsymbol{x}) \end{aligned} \right\} \forall\, (\hat{\boldsymbol{U}}, \hat{\boldsymbol{Y}}, \boldsymbol{x}, \boldsymbol{z}) \in \mathcal{D}$
    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) $\boldsymbol{x}_{(t+1)}$ such that $\boldsymbol{x}_{(t+1)}^\top \left( \hat{\boldsymbol{U}}_{(t+1)} \hat{\boldsymbol{U}}_{(t+1)}^\top - \hat{\boldsymbol{Y}}_{(t+1)} \right) \boldsymbol{x}_{(t+1)} < 0$;
        Generate $q^k$ subproblems and add them to the queue:
        for each $\boldsymbol{z} \in [q]^k$ do
            $\mathcal{D}_{\boldsymbol{z}} := \mathcal{D} \cup \{(\hat{\boldsymbol{U}}_{(t+1)}, \hat{\boldsymbol{Y}}_{(t+1)}, \boldsymbol{x}_{(t+1)}, \boldsymbol{z})\}$
            $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\, (\mathcal{D}_{\boldsymbol{z}}, t+1) \,\}$
        end for
        Update $Z_{\text{lower}}$ as the minimum of the lower bounds of all unexplored nodes.
    end if
end while
return $(\boldsymbol{\Theta}_{\text{opt}}, \boldsymbol{Y}_{\text{opt}}, \boldsymbol{X}_{\text{opt}}, \boldsymbol{U}_{\text{opt}})$
```

advantageous, because if the set of branching directions is fixed, then any nodes expanded under this strategy must also be expanded under any other strategy. However, both strategies incur a high memory cost from maintaining many unexplored nodes in the queue. On the other hand, depth-first search maintains a queue size which is a linear function of the problem size throughout the entire search process (Ibaraki 1976), although at the price of spending less time tightening the upper bound and, therefore, often needed to expand more nodes overall.

While we assumed a breadth-first strategy to establish the convergence of Algorithm 1 in Theorem 1, we observe numerically in Section 5 that best-first search outperforms the other strategies considered here in terms of runtime. Accordingly, except where explicitly stated otherwise, we use best-first search in our numerical experiments.

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 (16) 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.

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. Note however that we do not provide an initial solution at the root node for noiseless matrix completion problems. Indeed, identifying feasible solutions to noiseless problems is about as hard as solving them to optimality.

To implement the alternating minimization heuristic, we follow Burer and Monteiro (2003, 2005) in applying the nonlinear reformulation $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}$. Starting with a solution $\hat{\boldsymbol{U}}^0$, we have:

$$\hat{\boldsymbol{V}}^{t+1} = \underset{\boldsymbol{V} \in \mathbb{R}^{k \times m}}{\arg\min} \quad \frac{1}{2} \sum_{(i,j) \in \mathcal{I}} \left((\hat{\boldsymbol{U}}^t \boldsymbol{V})_{i,j} - A_{i,j} \right)^2 + \frac{1}{2\gamma} \|\hat{\boldsymbol{U}}^t \boldsymbol{V}\|_F^2 \tag{23}$$

$$\hat{\boldsymbol{U}}^{t+1} = \underset{\boldsymbol{U} \in \mathbb{R}^{n \times k}}{\arg\min} \quad \frac{1}{2} \sum_{(i,j) \in \mathcal{I}} \left((\boldsymbol{U} \hat{\boldsymbol{V}}^{t+1})_{i,j} - A_{i,j} \right)^2 + \frac{1}{2\gamma} \|\boldsymbol{U} \hat{\boldsymbol{V}}^{t+1}\|_F^2 \tag{24}$$

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:

$$(X_{\text{initial}})_{i,j} = \begin{cases} A_{i,j} & (i,j) \in \mathcal{I} \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

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 $\boldsymbol{X}_{\text{initial}}$. Upon 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 $\boldsymbol{X}_{\text{altmin}}$, to ensure $\boldsymbol{U}_{\text{altmin}}$ 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 now develop a strategy for improving this solution which is very much inspired by the work of Danna et al. (2005), namely, computing high-quality solutions at various nodes of our branch-and-bound strategy. 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 (10) 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 (24) with

$$\hat{\boldsymbol{U}}^{t+1} = \underset{\substack{\boldsymbol{U}\in\mathbb{R}^{n\times k},\ \boldsymbol{U}\in\mathcal{P}_{\boldsymbol{U}},\\ \|\boldsymbol{U}_i\|_2^2\le 1,\ \|\boldsymbol{U}_i\pm\boldsymbol{U}_j\|_2^2\le 2\quad \forall i,j\in[k].}}{\arg\min} \quad \frac{1}{2}\sum_{(i,j)\in\mathcal{I}}\left((\boldsymbol{U}\hat{\boldsymbol{V}}^{t+1})_{i,j} - A_{i,j}\right)^2 + \frac{1}{2\gamma}\|\boldsymbol{U}\hat{\boldsymbol{V}}^{t+1}\|_F^2, \tag{26}$$

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 Atamtürk and Gómez (2019), 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 1.7.3 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 first evaluate the effectiveness of the valid inequalities and presolving strategies presented 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, both in terms of its ability to identify an optimal solution and find feasible solutions that outperform a naive alternating minimization strategy. Our numerical experiments can be found at `https://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 $\epsilon = 0$ for basis pursuit and $\epsilon = 0.1$ for matrix completion instances. 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.

5.2. Root Node: Strengthened Relaxations and Presolving

In this section, we evaluate the benefit of the valid inequalities and presolving technique presented in Section 3, for matrix completion and basis pursuit problems, respectively.

5.2.1. Valid Inequalities for Matrix Completion Problems

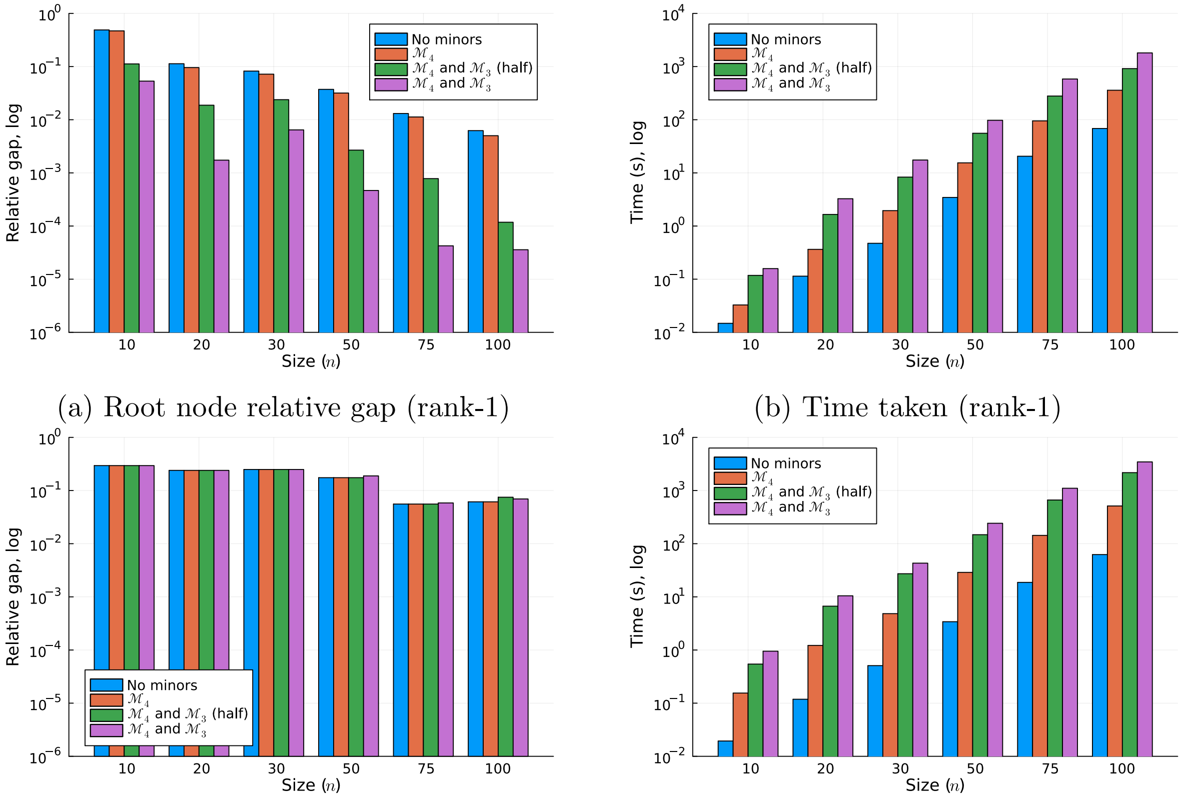
As argued in Section 3.3, we can strengthen the matrix perspective relaxation (9) by imposing additional semidefinite constraints (Shor LMIs) on all 2×2 minors of the slices of $\boldsymbol{X}$, $\boldsymbol{X}^t, t \in [k]$. To minimize the impact on computational tractability, however, we do not impose 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 (9) with (22) 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 first consider rank-one matrix completion problems with $n \in \{10, 20, 30, 50, 75, 100\}$, 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 of the 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.3.1 report the optimality gap and computational time of each method and each instance.

Figures 1c–1d display the same results for rank-k ($k = 2$) matrix completion problems with $n \in \{10, 20, 30, 50, 75, 100\}$, and $|\mathcal{I}| = 2kn\log(n)$. Unfortunately, we do not observe any improvement in the optimality gaps in these instances. Accordingly, future work could involve investigating technqiques for improving the matrix perspective relaxation when $k > 1$.

5.2.2. Presolving for Basis Pursuit We now evaluate the efficacy of our presolving strategy for basis pursuit problems, as laid out in Section 3.2. First, we investigate the proportion of entries of an $n \times n$ matrix that our approach fully prescribes. We consider different regimes for $|\mathcal{I}|$. From an information-theoretic perspective, on the order of $kn\log(n)$ entries are necessary for any method to successfully complete a low-rank matrix (Candès and Recht 2009). Accordingly, we also consider slower and faster rates: $|\mathcal{I}| = p\,k\,r(n)$ with $p = 2$ and $r(n) \propto \{n, n\log_{10}(n), n^{6/5}\log_{10}(n), n^{1.5}, n^2\}$. Figure 2 displays the proportion of random instances that are fully presolved, as n increases. When $|\mathcal{I}|$ grows strictly faster than $n\log_{10}(n)$, more than 90% of instances can be fully presolved when $n \geq 40$, both in the rank-one and rank-two cases, thus demonstrating the effectiveness of our presolving strategy. Moreover, when $|\mathcal{I}|$ is on the order of $n\log_{10}(n)$, the presolving step successfully solves the instance when n exceeds 20, without requiring any optimization.

In addition to reducing the number of variables in our optimization problem, presolving also improves the quality of the matrix perspective relaxation. We now compare the optimality gap at the root node (where we obtain an upper bound by running our branch-and-bound method to optimality on each instance) obtained via different convex relaxations: (i) the original matrix perspective relaxation (9); (ii) the relaxation (i) after presolving the entries belonging to two-by-two minors with three observed entries, $\mathcal{M}_3$; (iii) the approach (ii) strengthened by linear equality constraints for the minors with two observed entries, $\mathcal{M}_2$; and (iv) the relaxation (iii) further



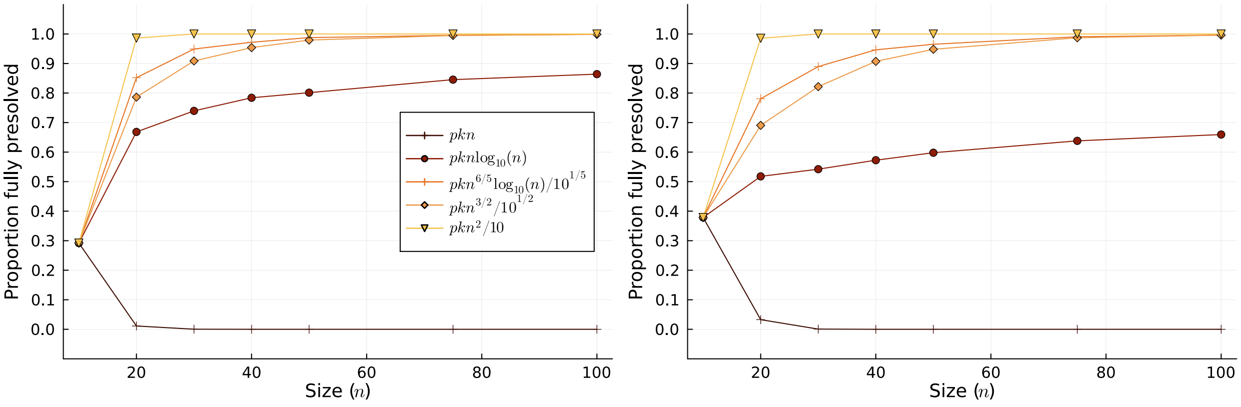
(a) Root node relative gap (rank-1)

(b) Time taken (rank-1)

(c) Root node relative gap (rank-2)

(d) Time taken (rank-2)

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



(a) Rank-1

(b) Rank-2

Figure 2 Comparison of proportion of fully presolved instances against problem size n for rank-k ($k \in \{1,2\}$) n-by-n basis pursuit problems, varying the number of entries observed initially, where $p = 2.0$. Each point is the average of 10,000 random problem instances with the specified parameters.

strengthened by Shor LMIs on minors from $\mathcal{M}_1$. Results on rank-one basis pursuit problem with $|\mathcal{I}| = pn\log_{10}(n)$ are presented in Figure 3. We observe that presolving alone completely closes the optimality gap, so further strengthening is unnecessary. For example, presolve closes the optimality gap at the root node from 33% for $n = 10$, $p = 2.0$ to zero.

For rank-two basis pursuit problems, for the problems which we cannot presolve fully, approaches (iii) and (iv) yield instances that are too large for Mosek to solve, even at the root node. Furthermore, approach (ii) followed by our branch-and-bound algorithm runs into out-of-memory errors so, unfortunately, we cannot properly assess the benefits of presolve in yielding tighter relaxations. Overall, for basis pursuit, we observed that presolve yields benefits in relaxation tightness but there is no incremental benefit in relaxation tightness after implementing (iii) and (iv).

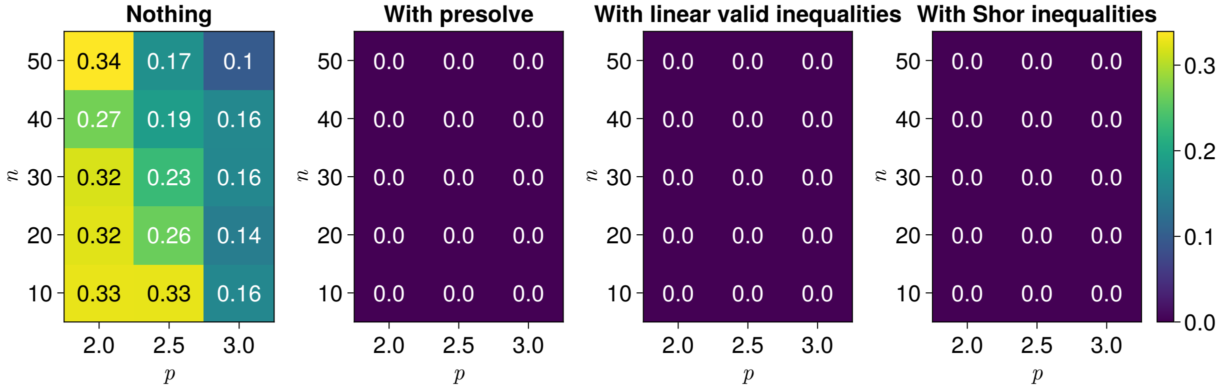


Figure 3 Comparison of relative optimality gap between root-node lower bound and optimal objective for rank-1 basis pursuit problems with $pn\log_{10}(n)$ entries, (i) after implementing presolve, (ii) after also including valid linear inequalities for basis pursuit, (iii) after also including valid Shor inequalities. Each cell is the average (geometric mean) over 20 random instances.

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 generic disjunctive scheme based on McCormick inequalities (as described in Section 2.5); 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 branch-and-bound tree.

Tables 1 and 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 achieve optimality gaps about an order of magnitude smaller on average 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.3–EC.8. 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 our prior work 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 of these approaches produced a lower bound better than the matrix perspective relaxation, which is attained at the root node of the present approach.

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	2.37×10^{-2}	3.06×10^{-2}	5.02×10^{-2}	5.28×10^{-3}	1.10×10^{-2}	2.60×10^{-2}
10	✓	3.29×10^{-4}	4.90×10^{-4}	7.92×10^{-3}	2.93×10^{-4}	4.91×10^{-4}	5.22×10^{-3}
20	✗	4.78×10^{-3}	4.78×10^{-3}	4.78×10^{-3}	2.61×10^{-4}	4.03×10^{-4}	4.03×10^{-3}
20	✓	5.51×10^{-4}	8.01×10^{-4}	8.01×10^{-4}	1.32×10^{-4}	1.92×10^{-4}	6.37×10^{-4}
30	✗	1.77×10^{-2}	1.77×10^{-2}	1.77×10^{-2}	2.00×10^{-3}	4.16×10^{-3}	1.35×10^{-2}
30	✓	2.01×10^{-3}	3.13×10^{-3}	3.13×10^{-3}	2.82×10^{-4}	4.53×10^{-4}	1.98×10^{-3}
40	✗	1.32×10^{-3}	1.32×10^{-3}	1.32×10^{-3}	3.28×10^{-4}	7.12×10^{-4}	6.11×10^{-4}
40	✓	1.12×10^{-4}	1.12×10^{-4}	1.12×10^{-4}	1.57×10^{-5}	1.94×10^{-5}	8.25×10^{-5}
50	✗	6.18×10^{-4}	6.18×10^{-4}	6.18×10^{-4}	8.11×10^{-5}	3.99×10^{-4}	8.11×10^{-4}
50	✓	6.37×10^{-5}	6.37×10^{-5}	6.40×10^{-5}	9.99×10^{-6}	1.13×10^{-5}	7.57×10^{-5}

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 ($p = 2.0$, $\gamma = 20.0$).

We also investigate the effect of the number of pieces in our disjunction, q, on the computational time needed to achieve a 10^{-4} optimality gap for rank-one matrix completion problems, in Figure 4. While we do not observe a significant difference between using $q = 2$ or $q = 3$ pieces, 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 four-piece disjunctions include zero as a breakpoint, which break some symmetry issues. However,

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	6.43×10^2	6.76×10^2	6.94×10^2	3.10×10^2	4.13×10^2	8.60×10^2
10	✓	1.40×10^2	1.36×10^2	5.70×10^2	6.37×10^1	1.04×10^2	3.98×10^2
20	✗	6.93×10^2	6.92×10^2	6.87×10^2	2.07×10^2	3.46×10^2	6.18×10^2
20	✓	2.06×10^2	2.28×10^2	2.37×10^2	5.88×10^1	9.17×10^1	2.63×10^2
30	✗	3.49×10^3	3.49×10^3	3.46×10^3	1.99×10^3	2.24×10^3	3.38×10^3
30	✓	9.21×10^2	9.04×10^2	9.28×10^2	3.07×10^2	3.35×10^2	8.86×10^2
40	✗	7.62×10^2	7.62×10^2	7.66×10^2	1.83×10^2	2.10×10^2	7.25×10^2
40	✓	5.14×10^2	5.08×10^2	5.19×10^2	8.19×10^1	9.53×10^1	4.99×10^2
50	✗	6.51×10^2	6.47×10^2	6.45×10^2	3.18×10^2	4.56×10^2	6.31×10^2
50	✓	3.22×10^2	3.26×10^2	3.26×10^2	1.08×10^2	1.47×10^2	4.35×10^2

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

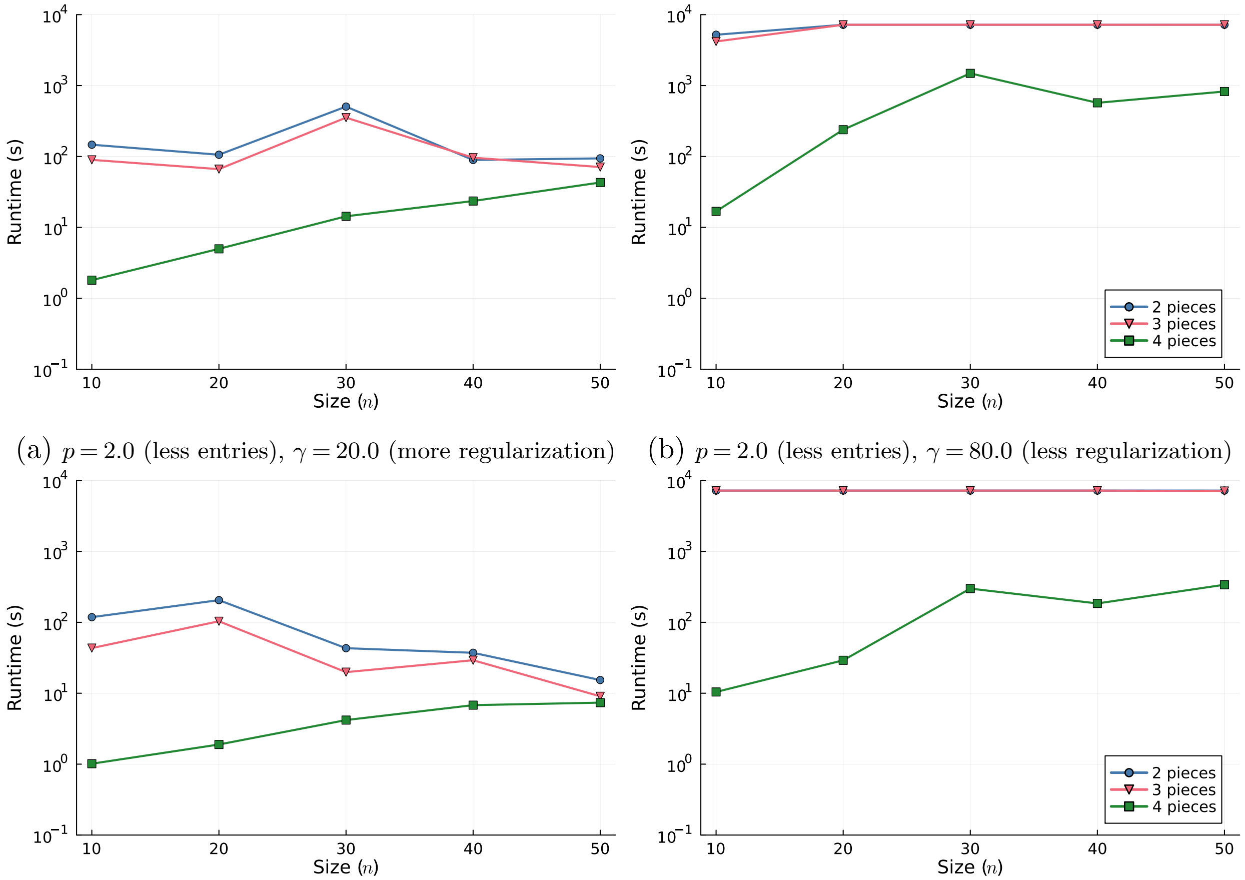
we also observe that this relative advantage vanishes as n increases, an observation which we confirm on larger instances in Figure EC.4.

5.4. Scalability Experiments

Table 2 reveals that the strongest performing implementation of our branch-and-bound algorithm solves rank-one matrix completion problems where $n = 50$ to optimality in minutes. Accordingly, we now investigate the scalability of our branch-and-bound algorithm, and its ability to obtain higher-quality low-rank matrices than popular heuristics. We apply our algorithm to rank-k matrix completion problems with $pkn\log_{10}(n)$ filled entries, $k \in \{1, 2, 3, 4, 5\}$, and a time limit of one hour. Figure 5a depicts the relative gap between the root node relaxation and the best incumbent solution at the root node, and after applying branch-and-bound for one hour, as we vary n.

We observe that the gaps at the root node are generally smaller for higher ranks, suggesting that a reasonable approach for large-scale instances could be to select the best solution found after running branch-and-bound with a time limit of minutes or hours. Furthermore, we observe that our branch-and-bound algorithm is highly effective in closing the bounds gap for $k = 1$ (when it is most needed) but less effective for higher ranks; for example, for $k = 1$ and $n = 10$, branch-and-bound reduces the bound gap from 49% at the root node to 1% after one hour of computational time.

Figure 5b contrasts the solution found via alternating minimization at the root node against the best solution found by our branch-and-bound scheme after one hour of computational time, as measured by the percentage improvement in mean-squared error on all entries (Figure EC.7c shows the absolute improvement in mean-squared error). We observe that branch-and-bound significantly improves the average relative MSE compared to alternating minimization, with an average relative improvement of more than 50% when $k = 2$ and $n \leq 50$, although branch-and-bound's



(a) $p = 2.0$ (less entries), $\gamma = 20.0$ (more regularization)

(b) $p = 2.0$ (less entries), $\gamma = 80.0$ (less regularization)

(c) $p = 3.0$ (more entries), $\gamma = 20.0$ (more regularization)

(d) $p = 3.0$ (more entries), $\gamma = 80.0$ (less regularization)

Figure 4 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.

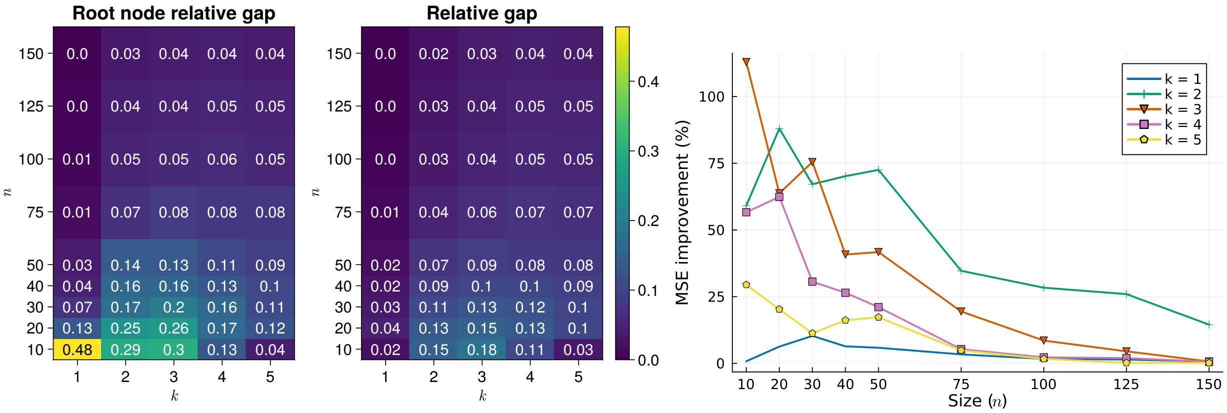
edge decreases as n increases. This could occur because each semidefinite nodal relaxation is more expensive to solve when n increases, or because there are fewer locally optimal solutions and the landscape of optimal solutions becomes "smoother" as n increases.

Section EC.3.3 augments our comparison of branch-and-bound with the method of Burer and Monteiro (2003), by generating the same plots in different sparsity settings, and comparing our branch-and-bound scheme with a matrix factorization approach using stochastic gradient descent (MFSGD) (Jin et al. 2016). We observe an even more significant MSE improvement over MFSGD than over the method of Burer and Monteiro (2003), which can be explained by the fact that MFSGD is designed for scalability rather than accuracy.

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 rank-one matrix completion problems significantly strengthen the semidefinite relaxation (9), and indeed routinely improve the root node



(a) Relative gap at root node and after branching

(b) MSE improvement (%)

Figure 5 Comparison of relative optimality gap at root node (a, left) and after running branch-and-bound for one hour (a, right), and percentage MSE improvement (b) for rank-k matrix completion problems with $2kn\log_{10}(n)$ filled entries, varying n and k, with $\gamma = 80.0$, averaged over 50 random instances per point.

gap by an order of magnitude or more. However, their efficiency wanes at larger ranks and future work could investigate stronger 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.
- Section 5.4 demonstrates that our branch-and-bound scheme solves matrix completion problems where $n = 150$ in minutes or hours, although scalability depends on the rank of the matrix, the amount of data observed and the degree of regularization. We also 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. This can be explained because Burer-Monteiro obtains a locally optimal solution which sometimes but not always is globally optimal.

6. Conclusion

In this paper, we propose a new branch-and-bound scheme for solving low-rank matrix completion and basis pursuit problems to certifiable optimality. The framework considers matrix perspective relaxations and recursively partitions their feasible regions using eigenvector disjunctions. Existing approaches are either scalable heuristics (but without optimality guarantees), or rely on weak McCormick relaxations and thus cannot scale to $n > 50$. On the other hand, our approach successfully scales to solve matrix completion problems over 150×150 matrices to certifiable optimality.

Future work could take three future directions: (1) using eigenvector disjunctions in conjunction with our presolving techniques to improve solution times for other non-convex quadratically constrained problems, such as ACOPF (Kocuk et al. 2016, 2018), (2) integrating eigenvector disjunctions within existing global solvers, either by default or as a viable and potentially more scalable alternative to McCormick branching, (3) developing stronger convex relaxations for problems where the rank exceeds one. Indeed, our numerical results reveal that our new convex relaxation can improve upon the matrix perspective relaxation by over an order of magnitude (in terms of the relative bound gap) in the rank-one case, but performs identically in the rank-two case.

Acknowledgement

We are grateful to Shuvomoy das Gupta for insightful comments on the manuscript.

References

Achterberg T, Koch T, Martin A (2005) Branching rules revisited. *Operations Research Letters* 33(1):42–54.

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

Atamtürk A, Gómez A (2019) Rank-one convexification for sparse regression. *arXiv preprint arXiv:1901.10334* .

Balas E, Ceria S, Cornuéjols G, Natraj N (1996) Gomory cuts revisited. *Operations Research Letters* 19(1):1–9.

Bandeira AS, Dobriban E, Mixon DG, Sawin WF (2013) Certifying the restricted isometry property is hard. *IEEE transactions on information theory* 59(6):3448–3450.

Bao X, Sahinidis NV, Tawarmalani M (2009) Multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic programs. *Optimization Methods & Software* 24(4-5):485–504.

Beck A (2007) Quadratic matrix programming. *SIAM Journal on Optimization* 17(4):1224–1238.

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

Belotti P, Kirches C, Leyffer S, Linderoth J, Luedtke J, Mahajan A (2013) Mixed-integer nonlinear optimization. *Acta Numerica* 22:1–131.

Belotti P, Lee J, Liberti L, Margot F, Wächter A (2009) Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods & Software* 24(4-5):597–634.

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, articles in advance* 1–46.

Bertsimas D, Li ML (2020) Fast exact matrix completion: A unified optimization framework for matrix completion. *J. Mach. Learn. Res.* 21(1), ISSN 1532-4435.

Bhojanapalli S, Boumal N, Jain P, Netrapalli P (2018) Smoothed analysis for low-rank solutions to semidefinite programs in quadratic penalty form. *Conference On Learning Theory*, 3243–3270 (PMLR).

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

Bienstock D (2010) Eigenvalue techniques for convex objective, nonconvex optimization problems. *International Conference on Integer Programming and Combinatorial Optimization*, 29–42 (Springer).

Biswas P, Ye Y (2004) Semidefinite programming for ad hoc wireless sensor network localization. *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, 46–54 (ACM).

Bixby R, Rothberg E (2007) Progress in computational mixed integer programming–a look back from the other side of the tipping point. *Annals of Operations Research* 149(1):37.

Bixby RE (2012) A brief history of linear and mixed-integer programming computation. *Documenta Mathematica* 2012:107–121.

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.

Burer S, Monteiro RD (2005) Local minima and convergence in low-rank semidefinite programming. *Mathematical programming* 103(3):427–444.

Candès EJ, Plan Y (2010) Matrix completion with noise. *Proceedings of the IEEE* 98(6):925–936.

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.

Danna E, Rothberg E, Pape CL (2005) Exploring relaxation induced neighborhoods to improve MIP solutions. *Mathematical Programming* 102(1):71–90.

Das Gupta S, Van Parys BP, Ryu EK (2023) Branch-and-bound performance estimation programming: A unified methodology for constructing optimal optimization methods. *Mathematical Programming, accepted* .

Ding L, Chen Y (2020) Leave-one-out approach for matrix completion: Primal and dual analysis. *IEEE Transactions on Information Theory* 66(11):7274–7301.

Dong H, Luo Y (2018) Compact disjunctive approximations to nonconvex quadratically constrained programs. *arXiv preprint arXiv:1811.08122* .

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

Fampa M, Lee J (2021) Convexification of bilinear forms through non-symmetric lifting. *Journal of Global Optimization* 80(2):287–305.

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

Frangioni A, Gentile C, Hungerford J (2020) Decompositions of semidefinite matrices and the perspective reformulation of nonseparable quadratic programs. *Mathematics of Operations Research* 45(1):15–33.

Gamarnik D (2021) The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences* 118(41).

Gamrath G, Koch T, Martin A, Miltenberger M, Weninger D (2015) Progress in presolving for mixed integer programming. *Mathematical Programming Computation* 7:367–398.

Gomory RE (1958) Outline of an algorithm for integer solutions to linear programs. *Bulletin Of the American Mathematical Society* 64:275–278.

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

Ibaraki T (1976) Theoretical comparisons of search strategies in branch-and-bound algorithms. *International Journal of Computer & Information Sciences* 5:315–344.

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.

Jin C, Kakade SM, Netrapalli P (2016) Provable efficient online matrix completion via non-convex stochastic gradient descent. *Advances in Neural Information Processing Systems* 29.

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

Kocuk B, Dey SS, Sun XA (2016) Strong SOCP relaxations for the optimal power flow problem. *Operations Research* 64(6):1177–1196.

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.

Kronqvist J, Misener R, Tsay C (2021) Between steps: Intermediate relaxations between big-M and convex hull formulations. *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, 299–314 (Springer).

Lawler EL, Wood DE (1966) Branch-and-bound methods: A survey. *Operations Research* 14(4):699–719.

Li Y, Xie W (2022) On the exactness of Dantzig-Wolfe relaxation for rank constrained optimization problems. *arXiv preprint arXiv:2210.16191* .

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

Little RJ, Rubin DB (2019) *Statistical Analysis with Missing Data*, volume 793 (John Wiley & Sons).

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

Meyer CA, Floudas CA (2004) Trilinear monomials with mixed sign domains: Facets of the convex and concave envelopes. *Journal of Global Optimization* 29:125–155.

Misener R, Floudas CA (2012) Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming* 136(1):155–182.

Misener R, Floudas CA (2014) ANTIGONE: algorithms for continuous/integer global optimization of nonlinear equations. *Journal of Global Optimization* 59(2-3):503–526.

Mutapcic A, Boyd S (2009) Cutting-set methods for robust convex optimization with pessimizing oracles. *Optimization Methods & Software* 24(3):381–406.

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.

Nan F (2009) *Low rank matrix completion.* Master's thesis, Massachusetts Institute of Technology.

Nguyen LT, Kim J, Shim B (2019) Low-rank matrix completion: A contemporary survey. *IEEE Access* 7:94215–94237.

Overton ML, Womersley RS (1992) On the sum of the largest eigenvalues of a symmetric matrix. *SIAM Journal on Matrix Analysis and Applications* 13(1):41–45.

Rahimian H, Mehrotra S (2020) Sequential convexification of a bilinear set. *Optimization Online* .

Recht B, Fazel M, Parrilo PA (2010) Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review* 52(3):471–501.

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).

Sahinidis NV (1996) BARON: A general purpose global optimization software package. *Journal of Global Optimization* 8(2):201–205.

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.

Saxena A, Bonami P, Lee J (2011) Convex relaxations of non-convex mixed integer quadratically constrained programs: projected formulations. *Mathematical Programming* 130(2):359–413.

Shapiro A (1982) Rank-reducibility of a symmetric matrix and sampling theory of minimum trace factor analysis. *Psychometrika* 47(2):187–199.

So AMC, Ye Y (2007) Theory of semidefinite programming for sensor network localization. *Mathematical Programming* 109(2-3):367–384.

Speakman E, Lee J (2017) Quantifying double McCormick. *Mathematics of Operations Research* 42(4):1230–1253.

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

Udell M, Townsend A (2019) Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science* 1(1):144–160.

Wang AL, Kılınç-Karzan F (2022) On the tightness of SDP relaxations of QCQPs. *Mathematical Programming* 193(1):33–73.

Wei L, Atamtürk A, Gómez A, Küçükyavuz S (2022) On the convex hull of convex quadratic optimization problems with indicators. *arXiv preprint arXiv:2201.00387* .

Wolsey LA, Nemhauser GL (1999) *Integer and Combinatorial Optimization*, volume 55 (John Wiley & Sons).

Yurtsever A, Udell M, Tropp J, Cevher V (2017) Sketchy decisions: Convex low-rank matrix optimization with optimal storage. *Artificial Intelligence and Statistics*, 1188–1196 (PMLR).

Zhang R, Josz C, Sojoudi S, Lavaei J (2018) How much restricted isometry is needed in nonconvex matrix recovery? *Advances in Neural Information Processing Systems* 31.

Zheng Q, Lafferty J (2015) A convergent gradient descent algorithm for rank minimization and semidefinite programming from random linear measurements. *Advances in Neural Information Processing Systems* 28.

Electronic Companion

EC.1. Omitted Proofs

EC.1.1. Proof of Proposition 2

Proof of Proposition 2 It suffices to show that every feasible solution to Problem (10) can be mapped to a feasible solution to (17) with the same objective value. Therefore, let $(\boldsymbol{Y}, \boldsymbol{U})$ be a feasible solution to Problem (10), let $D = \{ (i(j), j) : j \in [k] \}$ denote the set of indices that have been disjuncted on, and select the branch of the disjunction such that $\boldsymbol{U}$ satisfies the constraints $U_{i(j),j} \in [\underline{U}_{i(j),j}, \overline{U}_{i(j),j}]$ and $U_{i,j} \in [-1,1]$ for all other indices. Then, we need only construct a $\boldsymbol{V}$ which satisfies the constraints in (17) to establish the result. To construct such a $\boldsymbol{V}$, let us observe that for any V_{i,j_1,j_2}, we have constraints on V_{i,j_1,j_2} defined by one of the following four cases:

- If $j := j_1 = j_2$ and $i \neq i(j)$ then it is not too hard to see that the McCormick constraints reduce to $V_{i,j,j} \in [-1,1]$.
- If $j := j_1 = j_2$ and $i = i(j)$ then $V_{i,j,j}$ is a subset of $[0,1]$ which contains $U_{i,j}^2$.
- If $j_1 \neq j_2$ and $i(j_1), i(j_2) \neq i$ then the McCormick inequalities reduce to $V_{i,j_1,j_2} \in [-1,1]$.
- If $j_1 \neq j_2$ and $i(j_1) = i$ or $i(j_2) = i$ then V_{i,j_1,j_2} is a subset of $[-1,1]$ containing $U_{i,j_1} U_{i,j_2}$.

Recall that $n \geq k+2$. Therefore, we can feasibly set $V_{i(j),j,j} = U_{i(j),j}^2$ and $V_{i,j,j} = \frac{1 - V_{i(j),j,j}}{n-1}$ for any $i \neq i(j)$. Then, it follows by construction that $\sum_{i=1}^n V_{i,j,j} = 1$. Moreover, set $V_{i(j_1),j_1,j_2} = U_{i(j_1),j_1} U_{i(j_1),j_2}$ and $V_{i(j_2),j_1,j_2} = U_{i(j_2),j_1} U_{i(j_2),j_2}$. Then, since it follows from the 2×2 minors of $\mathbb{I} \succeq \boldsymbol{Y} \succeq \boldsymbol{U}\boldsymbol{U}^\top$ that $|\boldsymbol{U}_i^\top \boldsymbol{U}_j| \leq 1$, we can set $V_{i,j_1,j_2} = \frac{-1}{n-k} \left(\sum_{j_1=1}^k V_{i(j_2),j_1,j_2} + \sum_{j_2=1}^k V_{i(j_1),j_1,j_2} \right)$, and it follows by construction that $\sum_{i=1}^n V_{i,j_1,j_2} = 0$. Therefore, $(\boldsymbol{Y}, \boldsymbol{U}, \boldsymbol{V})$ is feasible within at least one branch of our disjunction, and there is no single McCormick disjunction which can improve our root node relaxation. □

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 (16) for any $\boldsymbol{x}_l$ such that $l < L$, we have that

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

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

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

Adding these inequalities then reveals that

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

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

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

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 that

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

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. Presolving in Basis Pursuit: Theoretical Observations and Practical Implementation

In this section, we characterize the sparsity patterns of matrices that can arise after presolving in basis pursuit. Recalling that $\mathcal{I} \subset [n] \times [m]$ is the set of indices of observed entries, we define the following index set of presolved entries $\mathcal{I}^k$ corresponding to rank-k presolve:

Definition EC.1. The index set of presolved entries in rank-k presolve, $\mathcal{I}^k$, is constructed from $\mathcal{I}$ by the following procedure:

1. Initialize $\mathcal{I}^k \leftarrow \mathcal{I}$.

2. Find sets $R \subset [n]$, $C \subset [m]$, with $|R| = |C| = k+1$, such that $|\mathcal{I}^k \cap (R \times C)| = (k+1)^2 - 1$. If no such sets exist, terminate.

3. Update $\mathcal{I}^k \leftarrow \mathcal{I}^k \cup (R \times C)$ and return to step 2.

For convenience, we define the following sets:

$$\forall\, i \in [n], \quad \mathcal{I}^k(i,:) := \left\{ j \;\middle|\; (i,j) \in \mathcal{I}^k \right\} \tag{EC.1}$$

$$\forall\, j \in [m], \quad \mathcal{I}^k(:,j) := \left\{ i \;\middle|\; (i,j) \in \mathcal{I}^k \right\} \tag{EC.2}$$

$$\forall\, R \subset [n], \quad \mathcal{I}^k(R,:) := \left\{ j \;\middle|\; (i,j) \in \mathcal{I}^k \;\forall\, i \in R \right\} = \bigcap_{i \in R} \mathcal{I}^k(i,:) \tag{EC.3}$$

$$\forall\, C \subset [m], \quad \mathcal{I}^k(:,C) := \left\{ j \;\middle|\; (i,j) \in \mathcal{I}^k \;\forall\, j \in C \right\} = \bigcap_{j \in C} \mathcal{I}^k(:,j) \tag{EC.4}$$

In the rank-1 case, we have the following result:

Lemma EC.1. *$\mathcal{I}^1$ satisfies the following properties:*

- *For any $i_1, i_2 \in [n]$, either $\mathcal{I}^1(i_1,:) = \mathcal{I}^1(i_2,:)$ or $|\mathcal{I}^k(\{i_1,i_2\},:)| = 0$.*
- *For any $j_1, j_2 \in [m]$, either $\mathcal{I}^1(:,j_1) = \mathcal{I}^1(:,j_2)$ or $|\mathcal{I}^k(:,\{j_1,j_2\})| = 0$.*

This implies that $\mathcal{I}^1$ can be expressed as:

$$\mathcal{I}^1 = (R_1 \times C_1) \cup (R_2 \times C_2) \cup \dots, \tag{EC.5}$$

where $\{R_i\}$ are nonempty and form a partition of $[n]$, and C_j are nonempty and form a partition of $[m]$. Intuitively, there exists a permutation of the rows and columns of the matrix such that $\mathcal{I}^1$ has a "block-rectangular" structure.

In the general rank-k case, we have the following extension:

Lemma EC.2. *$\mathcal{I}^k$ satisfies the following properties:*

- *For $R_1, R_2 \subset [n]$ with $|R_1| = |R_2| = k$, either $\mathcal{I}^k(R_1,:) = \mathcal{I}^k(R_2,:)$ or $|\mathcal{I}^k(R_1 \cap R_2,:)| < k$.*
- *For $C_1, C_2 \subset [m]$ with $|C_1| = |C_2| = k$, either $\mathcal{I}^k(:,C_1) = \mathcal{I}^k(:,C_2)$ or $|\mathcal{I}^k(:,C_1 \cap C_2)| < k$.*

Proof of Lemma EC.2 We prove the first property (the second is symmetrical). Suppose $R_1 \neq R_2$ (which implies $|R_1 \cup R_2| \geq k+1$). (If $R_1 = R_2$ then $\mathcal{I}^1(R_1,:) = \mathcal{I}^1(R_2,:)$ and we are done.) Also suppose that $\mathcal{I}^1(R_1 \cap R_2,:) \geq k$. Then there exists k column indices $C = \{j_1, j_2, \ldots, j_k\}$ such that $C \subset \mathcal{I}^1(R_1 \cap R_2,:)$, i.e. $(R_1 \cup R_2) \times C \subset \mathcal{I}^k$.

Suppose for the sake of contradiction that there exists $j \in \mathcal{I}^1(R_1,:) \setminus \mathcal{I}^1(R_2,:)$ (the case where $|\mathcal{I}^1(R_2,:) \setminus \mathcal{I}^1(R_1,:)| > 0$ is identical). Select $i \in R_2 \setminus R_1$. Then, the k indices in $\{i\} \times C$ and the $k(k+1)$ indices in $R_1 \times (C \cup \{j\})$ all belong in $\mathcal{I}$, and therefore $(R_1 \cup \{i\}) \times (C \cup \{j\})$ defines a $(k+1)$-by-$(k+1)$ minor with $(k+1)^2 - 1$ observed entries. This means that (i,j) could still be added as a presolved entry to $\mathcal{I}^k$, which contradicts the definition of $\mathcal{I}^k$. □

In a practical implementation of the presolve stage for rank-k basis pursuit stage, we perform the procedure described in Definition EC.1. Instead of enumerating all $\binom{n}{k+1} \times \binom{m}{k+1}$ possibilities for R and C in step 2, which is prohibitively expensive, we use Lemma EC.2 to reduce the number of row-sets and column-sets to search over. The observation is that if $R \times C$ has $(k+1)^2 - 1$ entries in $\mathcal{I}^k$, then there must exist $C' \subset C$ ($R' \subset R$) with $|C'| = k$ ($|R'| = k$) respectively such that $R' \times C$ and $R \times C'$ are in $\mathcal{I}^k$. In particular, $\mathcal{I}^k(:, C') \geq (k+1)$, and we use this in line 5 of Algorithm 2. Also, $|R'| = k$ implies that the set $\overline{R} := \{r \in \mathcal{I}^k(:, C') : (r,j) \in \mathcal{I}^k\}$ must satisfy $|\overline{R}| \geq k$; we use this in line 8 of Algorithm 2 to accelerate the algorithm. These accelerations perform better in a sparser regime, because many row-sets and column-sets can be skipped over.

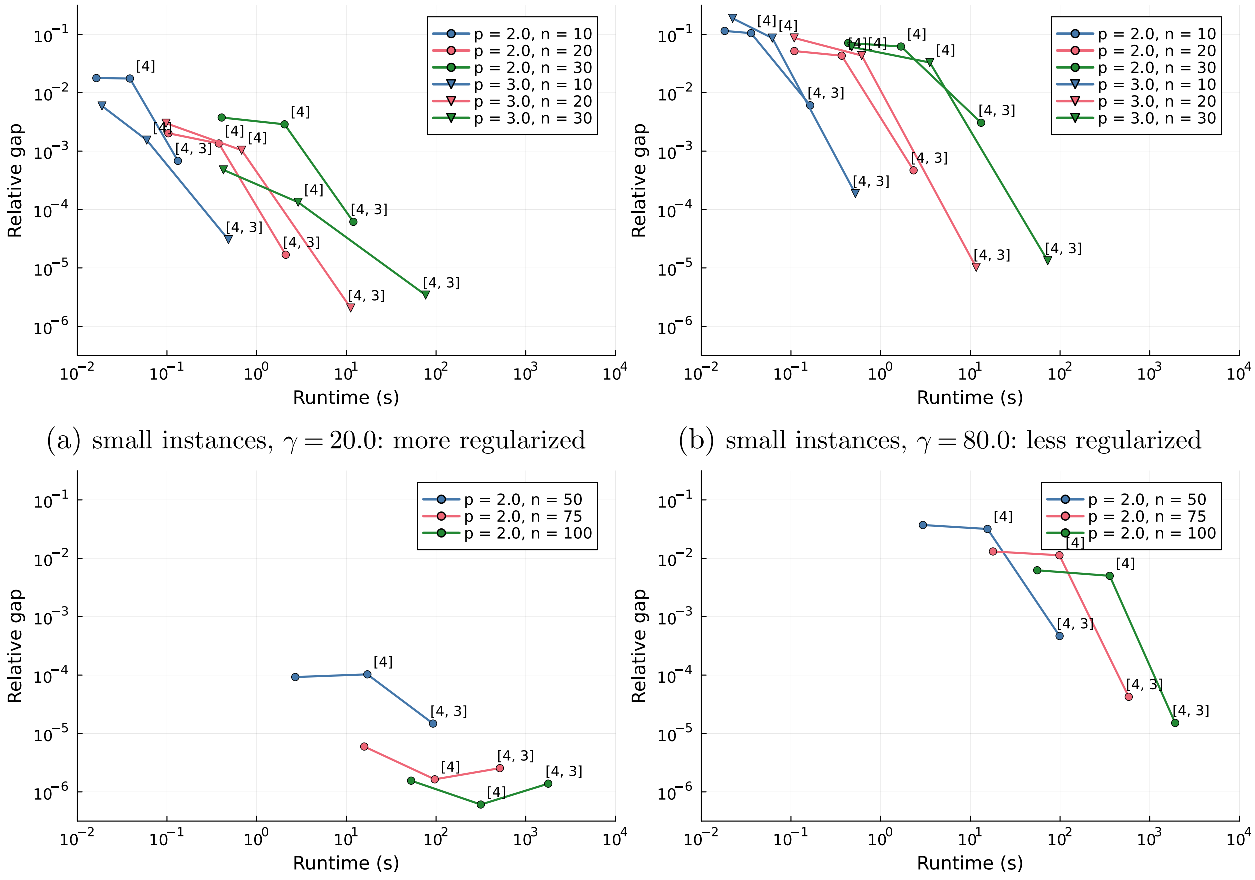
Algorithm 2 Presolve for rank-k basis pursuit problems

```
Initialize $\mathcal{I}^k = \mathcal{I}$;
while not converged do
    converged $\leftarrow$ true;
    for each $C' \subset [m]$ with $|C'| = k$ do
        if $|\mathcal{I}^k(:, C')| \leq k$, continue;
        for $j \in [m] \setminus C'$ do
            Define $C := C' \cup \{j\}$;
            Define $\overline{R} := \{r \in \mathcal{I}^k(:, C') : (r, j) \in \mathcal{I}^k\}$ (if $|\overline{R}| < k$, continue);
            Choose $R' \subset \overline{R}$ such that $|R'| = k$;
            for $i \in \mathcal{I}^k(:, C') \setminus \overline{R}$ do
                Define $R := R' \cup \{i\}$;
                Update $\mathcal{I}^k \leftarrow \mathcal{I}^k \cup (R \times C)$ and set converged $\leftarrow$ false;
                Presolve $(i, j)$ based on other entries in $R \times C$;
            end for
        end for
    end for
end while
return $\mathcal{I}^k$
```

EC.3. Numerical results

EC.3.1. Root Node: Strengthened Relaxations and Presolving

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. We observe that, across all sizes, regularization parameters, and sparsity settings, imposing more Shor LMIs yields tighter but (potentially significantly) more computationally expensive semidefinite relaxations. Moreover, we observe that 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). So, the benefits from our strengthened relaxation appear to be more salient in regimes with less regularization.



(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	0.0163	0.0386	0.132	1.78×10^{-2}	1.75×10^{-2}	6.81×10^{-4}
		80.0	0.0182	0.0359	0.164	1.14×10^{-1}	1.05×10^{-1}	6.09×10^{-3}
	3.0	20.0	0.0189	0.0596	0.484	5.93×10^{-3}	1.54×10^{-3}	3.04×10^{-5}
		80.0	0.0224	0.062	0.523	1.88×10^{-1}	8.62×10^{-2}	1.87×10^{-4}
20	2.0	20.0	0.104	0.379	2.11	2.02×10^{-3}	1.35×10^{-3}	1.68×10^{-5}
		80.0	0.109	0.368	2.33	5.17×10^{-2}	4.31×10^{-2}	4.69×10^{-4}
	3.0	20.0	0.0976	0.68	11.2	3.01×10^{-3}	1.04×10^{-3}	2.06×10^{-6}
		80.0	0.109	0.615	11.6	8.67×10^{-2}	4.33×10^{-2}	1.02×10^{-5}
30	2.0	20.0	0.408	2.05	12.0	3.76×10^{-3}	2.88×10^{-3}	6.15×10^{-5}
		80.0	0.435	1.69	13.2	7.10×10^{-2}	6.18×10^{-2}	3.06×10^{-3}
	3.0	20.0	0.426	2.9	76.6	4.77×10^{-4}	1.33×10^{-4}	3.42×10^{-6}
		80.0	0.475	3.54	72.9	6.08×10^{-2}	3.29×10^{-2}	1.32×10^{-5}

Table EC.1 Comparing time taken and relative gap at the 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)

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 $\mathcal{M}_4 \cup \mathcal{M}_3$. This is most clearly seen in Figure EC.2, where either half or all of the minors in $\mathcal{M}_4 \cup \mathcal{M}_3$ are imposed. The merits of either approach differ based on sparsity (number of filled entries) and problem size. Unfortunately, we do not observe a strengthened relaxation in the rank-2 case despite imposing constraints on more minors, as demonstrated in Figure EC.3 and EC.2.

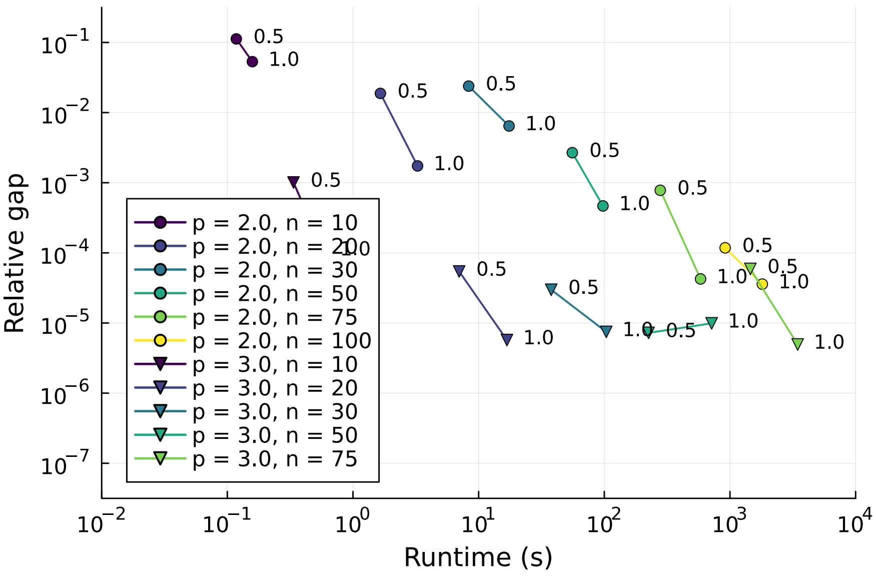
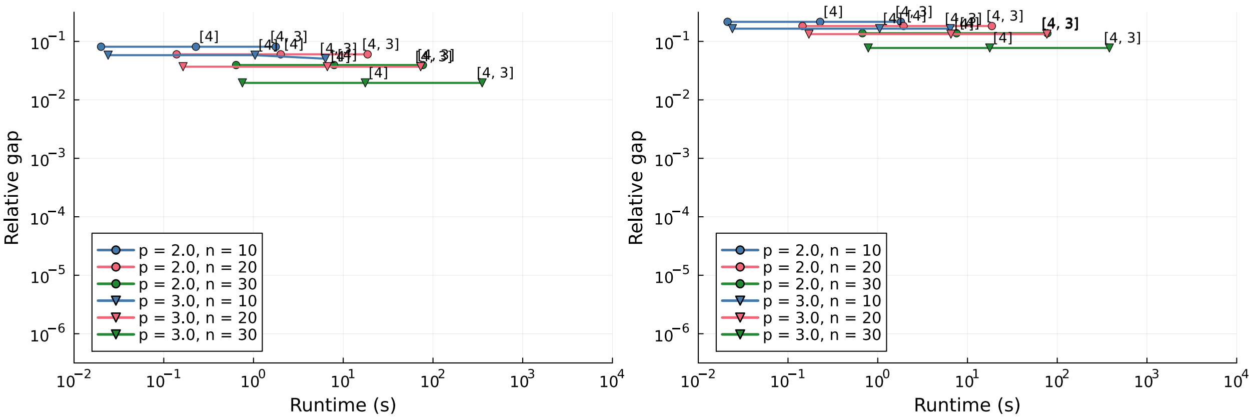


Figure EC.2 Comparison of relative bound gap at the root node against time for the root node relaxations of rank-1 matrix completion problems, varying the proportion of Shor LMIs added ($\gamma = 80.0$).



(a) small instances, $\gamma = 20.0$: more regularized (b) small instances, $\gamma = 80.0$: less regularized

Figure EC.3 Comparison of relative bound gap at the root node against time for the root node relaxations of rank-2 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	$\mathcal{M}_4$	$\mathcal{M}_4 \cup \mathcal{M}_3$	None	$\mathcal{M}_4$	$\mathcal{M}_4 \cup \mathcal{M}_3$
10	2.0	20.0	0.0200	0.228	1.77	0.081	0.081	0.081
		80.0	0.0212	0.228	1.81	0.216	0.216	0.216
	3.0	20.0	0.0240	1.04	6.32	0.0581	0.0581	0.0506
		80.0	0.0240	1.05	6.43	0.165	0.165	0.165
20	2.0	20.0	0.139	2.01	18.7	0.0601	0.0601	0.0601
		80.0	0.145	1.94	18.7	0.183	0.183	0.183
	3.0	20.0	0.164	6.65	72.9	0.0369	0.0369	0.0369
		80.0	0.171	6.54	76.7	0.133	0.133	0.133
30	2.0	20.0	0.640	7.91	77.2	0.0394	0.0394	0.0394
		80.0	0.675	7.55	77.6	0.138	0.138	0.138
	3.0	20.0	0.752	17.6	353	0.0195	0.0195	0.0195
		80.0	0.784	17.7	381	0.0772	0.0772	0.0772

Table EC.2 Comparing time taken and relative gap at the root node across rank-two matrix completion problems with $pkn\log_{10}(n)$ filled entries, with different Shor LMIs added (averaged over 20 instances per row)

EC.3.2. 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

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	2.56×10^{-1}	3.21×10^{-1}	3.75×10^{-1}	3.19×10^{-1}	3.93×10^{-1}	4.73×10^{-1}
10	✓	1.44×10^{-2}	2.71×10^{-2}	1.12×10^{-1}	1.48×10^{-2}	2.63×10^{-2}	9.37×10^{-2}
20	✗	1.09×10^{-1}	1.09×10^{-1}	1.09×10^{-1}	1.12×10^{-1}	1.34×10^{-1}	1.39×10^{-1}
20	✓	5.88×10^{-2}	6.05×10^{-2}	6.06×10^{-2}	3.53×10^{-2}	4.64×10^{-2}	1.10×10^{-1}
30	✗	1.31×10^{-1}	1.31×10^{-1}	1.31×10^{-1}	1.34×10^{-1}	1.45×10^{-1}	1.48×10^{-1}
30	✓	6.94×10^{-2}	7.08×10^{-2}	7.08×10^{-2}	3.60×10^{-2}	4.27×10^{-2}	6.76×10^{-2}
40	✗	7.76×10^{-2}	7.76×10^{-2}	7.76×10^{-2}	6.77×10^{-2}	7.38×10^{-2}	7.93×10^{-2}
40	✓	4.27×10^{-2}	4.27×10^{-2}	4.27×10^{-2}	2.27×10^{-2}	2.58×10^{-2}	4.16×10^{-2}
50	✗	3.71×10^{-2}	3.71×10^{-2}	3.71×10^{-2}	4.85×10^{-2}	5.15×10^{-2}	5.41×10^{-2}
50	✓	2.55×10^{-2}	2.55×10^{-2}	2.55×10^{-2}	1.66×10^{-2}	1.83×10^{-2}	2.98×10^{-2}

Table EC.3 Comparing relative gap (of the branch-and-bound scheme, capped at 1 hour) across rank-one matrix completion problems with $pn\log_{10}(n)$ 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	✗	1.86×10^{3}	2.08×10^{3}	2.12×10^{3}	2.40×10^{3}	2.74×10^{3}	2.88×10^{3}
10	✓	1.40×10^{3}	1.55×10^{3}	2.18×10^{3}	1.68×10^{3}	2.10×10^{3}	2.73×10^{3}
20	✗	3.25×10^{3}	3.27×10^{3}	3.21×10^{3}	3.22×10^{3}	3.24×10^{3}	3.21×10^{3}
20	✓	2.89×10^{3}	2.78×10^{3}	3.16×10^{3}	2.64×10^{3}	2.49×10^{3}	3.07×10^{3}
30	✗	3.49×10^{3}	3.48×10^{3}	3.46×10^{3}	3.40×10^{3}	3.42×10^{3}	3.39×10^{3}
30	✓	3.38×10^{3}	3.26×10^{3}	3.45×10^{3}	2.97×10^{3}	2.89×10^{3}	3.28×10^{3}
40	✗	3.54×10^{3}	3.54×10^{3}	3.53×10^{3}	3.49×10^{3}	3.50×10^{3}	3.46×10^{3}
40	✓	3.48×10^{3}	3.41×10^{3}	3.53×10^{3}	3.22×10^{3}	3.18×10^{3}	3.42×10^{3}
50	✗	3.56×10^{3}	3.56×10^{3}	3.56×10^{3}	3.54×10^{3}	3.54×10^{3}	3.52×10^{3}
50	✓	3.50×10^{3}	3.46×10^{3}	3.55×10^{3}	3.36×10^{3}	3.34×10^{3}	3.48×10^{3}

Table EC.4 Comparing time taken (s) (of the branch-and-bound scheme, capped at one hour) across rank-one matrix completion problems with $pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 2.0$, $\gamma = 80.0$).

$\gamma \in \{20.0, 80.0\}$ and $pn\log_{10}(n)$ filled entries with $p \in \{2.0, 3.0\}$. The results are shown in Tables 1–2, EC.3–EC.4 (less entries $p = 2.0$ and less regularization $\gamma = 80.0$), EC.5–EC.6 (more entries $p = 3.0$ and more regularization $\gamma = 20.0$), and EC.7–EC.8 (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.

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	7.61×10^{-3}	8.77×10^{-3}	1.64×10^{-2}	9.09×10^{-4}	1.21×10^{-3}	1.42×10^{-2}
10	✓	6.27×10^{-4}	7.68×10^{-4}	6.26×10^{-3}	1.28×10^{-4}	2.08×10^{-4}	3.69×10^{-3}
20	✗	6.96×10^{-3}	6.96×10^{-3}	6.96×10^{-3}	6.23×10^{-5}	3.12×10^{-4}	8.24×10^{-4}
20	✓	6.22×10^{-4}	1.67×10^{-3}	1.67×10^{-3}	8.14×10^{-5}	1.40×10^{-4}	7.20×10^{-4}
30	✗	3.31×10^{-3}	3.31×10^{-3}	3.31×10^{-3}	9.04×10^{-5}	3.05×10^{-4}	1.04×10^{-3}
30	✓	1.05×10^{-4}	2.37×10^{-4}	2.37×10^{-4}	1.60×10^{-5}	2.25×10^{-5}	6.00×10^{-5}
40	✗	1.78×10^{-4}	1.78×10^{-4}	1.78×10^{-4}	2.16×10^{-5}	3.41×10^{-5}	1.39×10^{-4}
40	✓	5.86×10^{-5}	5.86×10^{-5}	5.86×10^{-5}	4.97×10^{-6}	5.30×10^{-6}	3.20×10^{-5}
50	✗	6.46×10^{-5}	6.46×10^{-5}	6.46×10^{-5}	3.86×10^{-7}	1.47×10^{-6}	2.58×10^{-6}
50	✓	2.24×10^{-6}	2.24×10^{-6}	2.24×10^{-6}	1.02×10^{-6}	1.04×10^{-6}	4.09×10^{-7}

Table EC.5 Comparing relative gap (of the branch-and-bound scheme, capped at one hour) across rank-one matrix completion problems with $pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 3.0$, $\gamma = 20.0$).

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	2.48×10^{2}	2.76×10^{2}	3.77×10^{2}	1.27×10^{2}	1.75×10^{2}	4.71×10^{2}
10	✓	1.26×10^{2}	1.46×10^{2}	3.90×10^{2}	5.65×10^{1}	1.10×10^{2}	3.87×10^{2}
20	✗	1.19×10^{3}	1.19×10^{3}	1.17×10^{3}	1.53×10^{2}	2.55×10^{2}	4.41×10^{2}
20	✓	4.35×10^{2}	6.66×10^{2}	6.85×10^{2}	1.30×10^{2}	2.01×10^{2}	4.15×10^{2}
30	✗	9.20×10^{2}	9.26×10^{2}	9.17×10^{2}	6.42×10^{1}	8.55×10^{1}	5.89×10^{2}
30	✓	3.48×10^{2}	3.74×10^{2}	3.75×10^{2}	3.31×10^{1}	5.51×10^{1}	1.60×10^{2}
40	✗	1.08×10^{2}	1.08×10^{2}	1.08×10^{2}	5.45×10^{1}	7.93×10^{1}	1.07×10^{2}
40	✓	1.10×10^{2}	1.07×10^{2}	1.08×10^{2}	3.19×10^{1}	4.79×10^{1}	1.08×10^{2}
50	✗	1.73×10^{2}	1.72×10^{2}	1.71×10^{2}	1.56×10^{1}	1.90×10^{1}	5.12×10^{1}
50	✓	6.23×10^{1}	6.20×10^{1}	6.21×10^{1}	1.33×10^{1}	1.50×10^{1}	3.68×10^{1}

Table EC.6 Comparing time taken (s) (of the branch-and-bound scheme, capped at one hour) across rank-one matrix completion problems with $pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 3.0$, $\gamma = 20.0$).

In Figure EC.4, 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 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	✗	1.44×10^{-1}	1.85×10^{-1}	3.45×10^{-1}	1.32×10^{-1}	1.96×10^{-1}	2.61×10^{-1}
10	✓	2.97×10^{-2}	5.86×10^{-2}	1.88×10^{-1}	3.16×10^{-2}	5.52×10^{-2}	1.61×10^{-1}
20	✗	1.29×10^{-1}	1.29×10^{-1}	1.29×10^{-1}	7.04×10^{-2}	9.22×10^{-2}	1.27×10^{-1}
20	✓	6.74×10^{-2}	9.05×10^{-2}	9.05×10^{-2}	2.69×10^{-2}	3.94×10^{-2}	8.27×10^{-2}
30	✗	1.03×10^{-1}	1.03×10^{-1}	1.03×10^{-1}	7.07×10^{-2}	8.38×10^{-2}	9.37×10^{-2}
30	✓	5.74×10^{-2}	6.08×10^{-2}	6.08×10^{-2}	2.39×10^{-2}	3.10×10^{-2}	5.28×10^{-2}
40	✗	5.15×10^{-2}	5.15×10^{-2}	5.15×10^{-2}	3.44×10^{-2}	4.02×10^{-2}	5.15×10^{-2}
40	✓	4.44×10^{-2}	4.44×10^{-2}	4.44×10^{-2}	2.16×10^{-2}	2.62×10^{-2}	4.02×10^{-2}
50	✗	2.62×10^{-2}	2.62×10^{-2}	2.62×10^{-2}	1.75×10^{-2}	2.04×10^{-2}	2.62×10^{-2}
50	✓	1.98×10^{-2}	1.98×10^{-2}	1.98×10^{-2}	7.78×10^{-3}	9.71×10^{-3}	1.44×10^{-2}

Table EC.7 Comparing relative gap (of the branch-and-bound scheme, capped at one hour) across rank-one matrix completion problems with $pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 3.0$, $\gamma = 80.0$).

		With McCormick disjunctions			With eigenvector disjunctions		
n	Alternating minimization	Best-first	Breadth-first	Depth-first	Best-first	Breadth-first	Depth-first
10	✗	1.89×10^{3}	2.16×10^{3}	2.17×10^{3}	2.83×10^{3}	2.82×10^{3}	2.88×10^{3}
10	✓	1.72×10^{3}	1.93×10^{3}	2.14×10^{3}	2.26×10^{3}	2.16×10^{3}	2.71×10^{3}
20	✗	3.30×10^{3}	3.31×10^{3}	3.24×10^{3}	3.26×10^{3}	3.27×10^{3}	3.23×10^{3}
20	✓	3.06×10^{3}	2.95×10^{3}	3.19×10^{3}	2.77×10^{3}	2.71×10^{3}	3.11×10^{3}
30	✗	3.49×10^{3}	3.48×10^{3}	3.47×10^{3}	3.42×10^{3}	3.42×10^{3}	3.40×10^{3}
30	✓	3.41×10^{3}	3.32×10^{3}	3.46×10^{3}	3.08×10^{3}	3.03×10^{3}	3.34×10^{3}
40	✗	3.54×10^{3}	3.54×10^{3}	3.54×10^{3}	3.50×10^{3}	3.50×10^{3}	3.49×10^{3}
40	✓	3.49×10^{3}	3.44×10^{3}	3.53×10^{3}	3.27×10^{3}	3.27×10^{3}	3.46×10^{3}
50	✗	3.57×10^{3}	3.56×10^{3}	3.56×10^{3}	3.54×10^{3}	3.54×10^{3}	3.54×10^{3}
50	✓	3.52×10^{3}	3.49×10^{3}	3.56×10^{3}	3.41×10^{3}	3.40×10^{3}	3.51×10^{3}

Table EC.8 Comparing time taken (s) (of the branch-and-bound scheme, capped at one hour) across rank-one matrix completion problems with $pn\log_{10}(n)$ filled entries, averaged over 20 instances per row ($p = 3.0$, $\gamma = 80.0$).

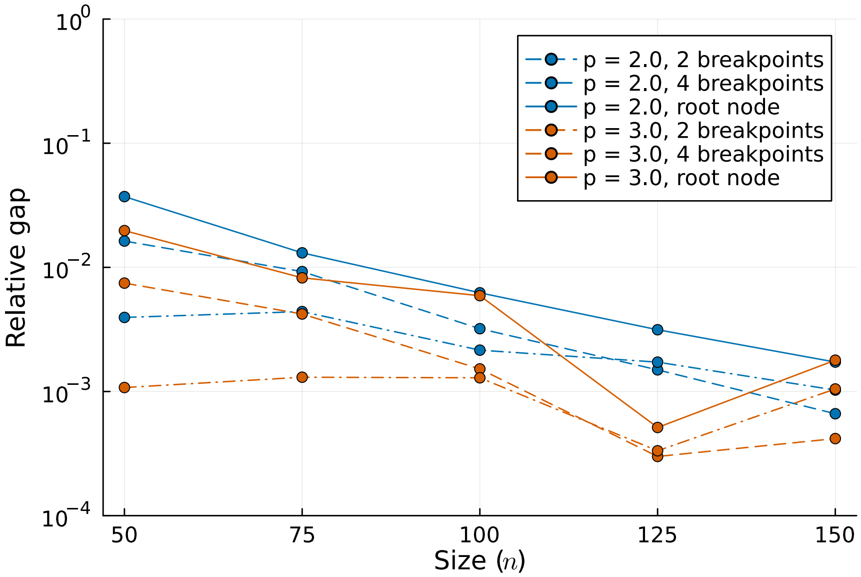


Figure EC.4 Comparison of relative gap (at root node, and after branching with 2-piece and 4-piece disjunctions) for rank-one matrix completion problems with $2n\log_{10}(n)$ filled entries, after 1 hour (averaged over 20 instances per point).

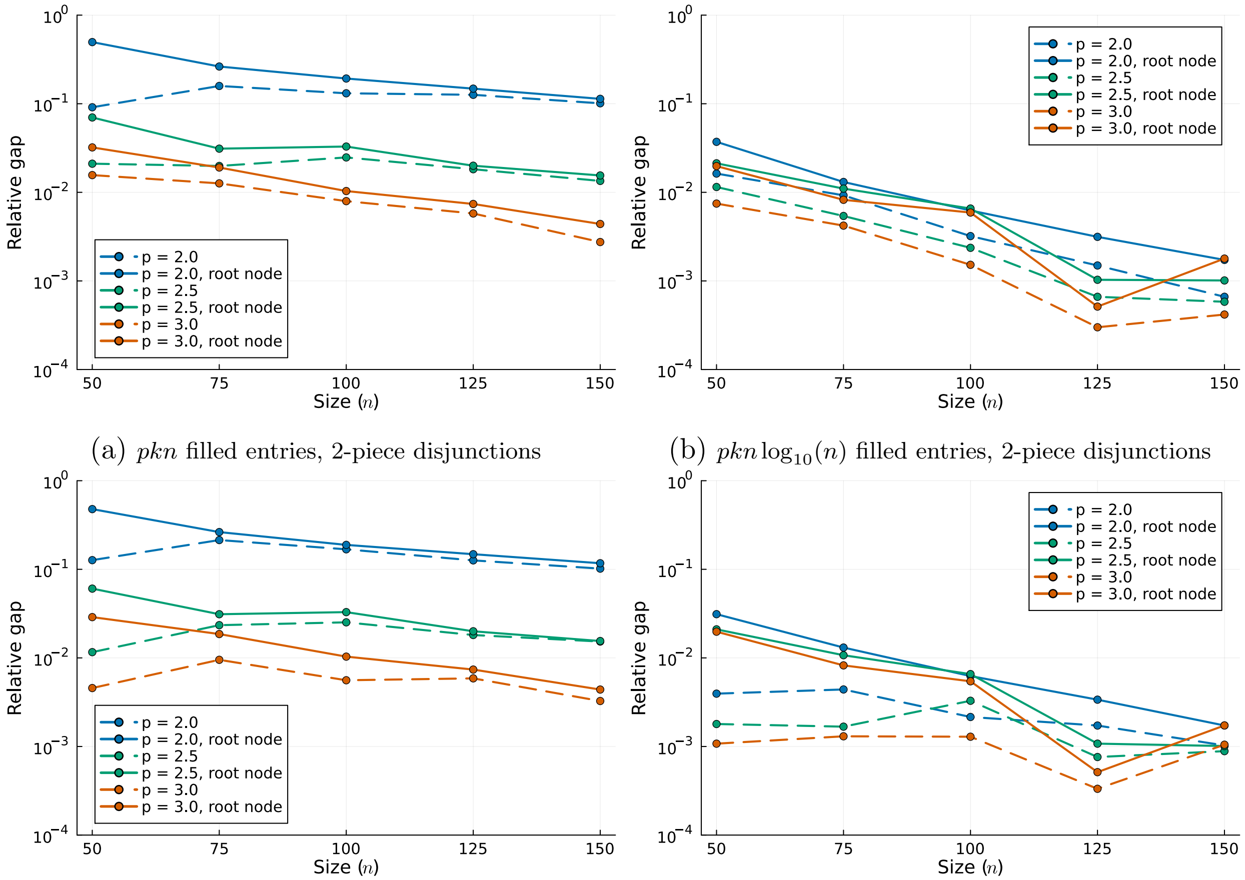
EC.3.3. Scalability Experiments

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.5 shows the relative optimality gap, at the root node and after branching, of 2-piece and 4-piece disjunctions applied to matrix completion problems of moderate to large sizes, with either pkn or $pkn\log_{10}(n)$ filled entries ($p \in \{2.0, 2.5, 3.0\}$). From Figure EC.5 we see that as n increases, the root node relative gap reduces, but the improvement in relative gap after branching also diminishes. We can also compare 2-piece and 4-piece disjunctions: 4-piece disjunctions perform better for smaller-sized problems, whereas 2-piece disjunctions yield a better relative gap after branching for larger-sized problems (this is the same trend illustrated in Figure EC.4). In the regime with $pkn\log_{10}(n)$ filled entries, the constant p has a smaller impact on relative optimality gaps compared to that for pkn filled entries.

Tables EC.9–EC.12 also demonstrate this but also show that as n increases, each individual SDP takes a longer time and so a smaller number of nodes are explored. This supports the hypothesis that as n increases, in an appropriate sparsity regime, the root node relaxation is tight and alternating minimization recovers the globally optimal solution. Indeed, in Tables EC.11–EC.12 we can solve 200×200 and 250×250 matrix completion problems by performing alternating minimization (in negligible time) to get a good solution, and certifying its optimality in an hour by solving the root node relaxation. Figure EC.6 illustrates the same trends but visualizes the variance over individual random instances by showing the empirical CDFs.

n	p	Nodes explored	Relaxation solve time (s)	Total time (s)	Relative gap (root node)	Relative gap
50	2.0	1035.35	3.227	>3600.0	4.96×10^{-1}	9.12×10^{-2}
50	2.5	1038.75	3.219	>3600.0	7.00×10^{-2}	2.11×10^{-2}
50	3.0	1053.75	3.174	>3600.0	3.21×10^{-2}	1.56×10^{-2}
75	2.0	191.55	18.3	>3600.0	2.63×10^{-1}	1.59×10^{-1}
75	2.5	199.3	17.54	>3600.0	3.11×10^{-2}	1.98×10^{-2}
75	3.0	197.15	17.77	>3600.0	1.91×10^{-2}	1.26×10^{-2}
100	2.0	59.5	59.95	>3600.0	1.93×10^{-1}	1.31×10^{-1}
100	2.5	59.85	59.72	>3600.0	3.29×10^{-2}	2.48×10^{-2}
100	3.0	59.35	60.36	>3600.0	1.03×10^{-2}	7.95×10^{-3}
125	2.0	23.85	152.6	>3600.0	1.48×10^{-1}	1.26×10^{-1}
125	2.5	24.4	148.7	>3600.0	2.00×10^{-2}	1.82×10^{-2}
125	3.0	24.2	150.2	>3600.0	7.39×10^{-3}	5.77×10^{-3}
150	2.0	11.0	340.2	>3600.0	1.14×10^{-1}	1.01×10^{-1}
150	2.5	11.35	334.3	>3600.0	1.55×10^{-2}	1.35×10^{-2}
150	3.0	11.1	337.5	>3600.0	4.39×10^{-3}	2.75×10^{-3}

Table EC.9 Results for rank-one matrix completion with 2-piece disjunctions starting with pkn filled entries over different problem instances (average of 20 instances per row).



(a) pkn filled entries, 2-piece disjunctions

(b) $pkn\log_{10}(n)$ filled entries, 2-piece disjunctions

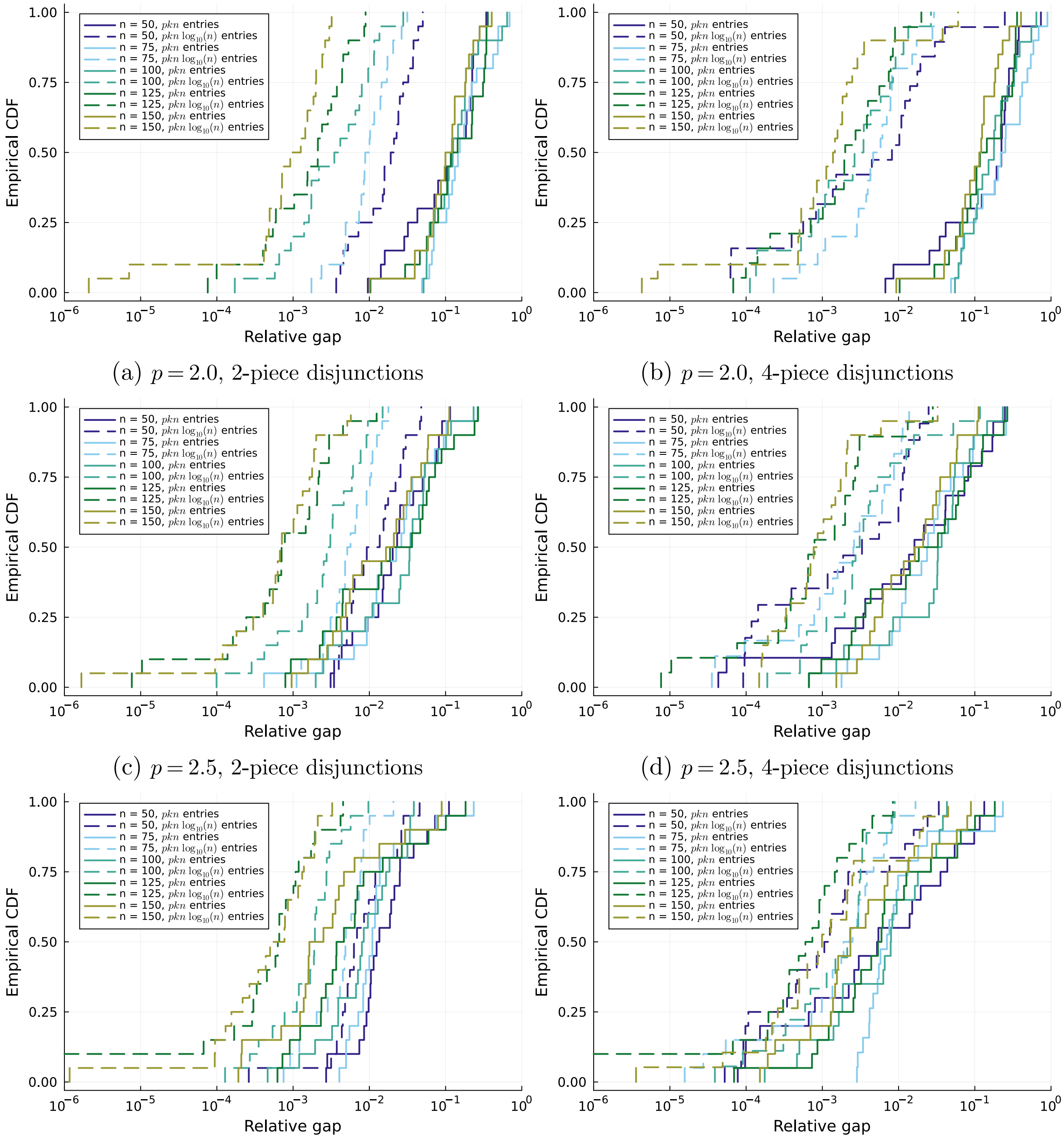
(c) pkn filled entries, 4-piece disjunctions

(d) $pkn\log_{10}(n)$ filled entries, 4-piece disjunctions

Figure EC.5 Comparison of relative gap at the root node (dashed lines), and after branch-and-bound with a 3600s time limit (solid lines), against n for rank-one matrix completion problems ($\gamma = 80.0$) using two-piece or four-piece disjunctions (averaged over 20 random instances).

Figure EC.7c shows the absolute improvement in mean-squared error (MSE) after branching for 1 hour, as compared to the alternating minimization solution found at the root node, across different sparsity settings ($|\mathcal{I}| \propto n$, $|\mathcal{I}| \propto n\log_{10}(n)$, and $|\mathcal{I}| \propto n^{6/5}\log_{10}(n)$). These plots illustrate the MSE improvement (in absolute terms) which is more pronounced in the sparser regime, and for low values of rank k other than $k = 1$.

In a similar vein, Figure EC.8 also shows the absolute improvement in MSE when comparing our approach to MFSGD (described in Jin et al. (2016), and found to be the best-performing benchmark of four methods studied in Bertsimas and Li (2020)). Across all parameter settings, our method yields a significant MSE improvement, with the improvement particularly stark in the sparser regime ($|\mathcal{I}| \propto n$). For more populous regimes, our approach yields a smaller improvement with larger ranks – this could be due to the time limit of 1 hour resulting in our methods not exploring many nodes and not improving over the alternating minimization solution found at the root node. Overall, these plots show the benefits of our optimality-focused approach.



(a) $p = 2.0$, 2-piece disjunctions

(b) $p = 2.0$, 4-piece disjunctions

(c) $p = 2.5$, 2-piece disjunctions

(d) $p = 2.5$, 4-piece disjunctions

(e) $p = 3.0$, 2-piece disjunctions

(f) $p = 3.0$, 4-piece disjunctions

Figure EC.6 Empirical CDF of relative gaps after branch-and-bound for 1 hour, for rank-one matrix completion problems (with $\gamma = 80.0$) (across 20 random instances).

n	p	Nodes explored	Relaxation solve time (s)	Total time (s)	Relative gap (root node)	Relative gap
50	2.0	654.6	5.235	>3600.0	4.78×10^{-1}	1.27×10^{-1}
50	2.5	580.32	5.588	3204.8	6.06×10^{-2}	1.16×10^{-2}
50	3.0	453.95	5.871	2183.8	2.89×10^{-2}	4.57×10^{-3}
75	2.0	137.25	25.89	>3600.0	2.63×10^{-1}	2.14×10^{-1}
75	2.5	133.1	26.71	>3600.0	3.11×10^{-2}	2.34×10^{-2}
75	3.0	123.79	29.09	>3600.0	1.86×10^{-2}	9.52×10^{-3}
100	2.0	47.0	76.66	>3600.0	1.89×10^{-1}	1.68×10^{-1}
100	2.5	44.6	80.94	>3600.0	3.29×10^{-2}	2.52×10^{-2}
100	3.0	34.05	84.9	2482.4	1.03×10^{-2}	5.59×10^{-3}
125	2.0	18.55	203.5	>3600.0	1.48×10^{-1}	1.26×10^{-1}
125	2.5	18.6	196.9	>3600.0	2.00×10^{-2}	1.81×10^{-2}
125	3.0	16.0	205.8	3082.3	7.39×10^{-3}	5.88×10^{-3}
150	2.0	8.3	456.7	>3600.0	1.17×10^{-1}	1.02×10^{-1}
150	2.5	8.1	479.2	>3600.0	1.55×10^{-2}	1.53×10^{-2}
150	3.0	7.95	460.5	>3600.0	4.39×10^{-3}	3.26×10^{-3}

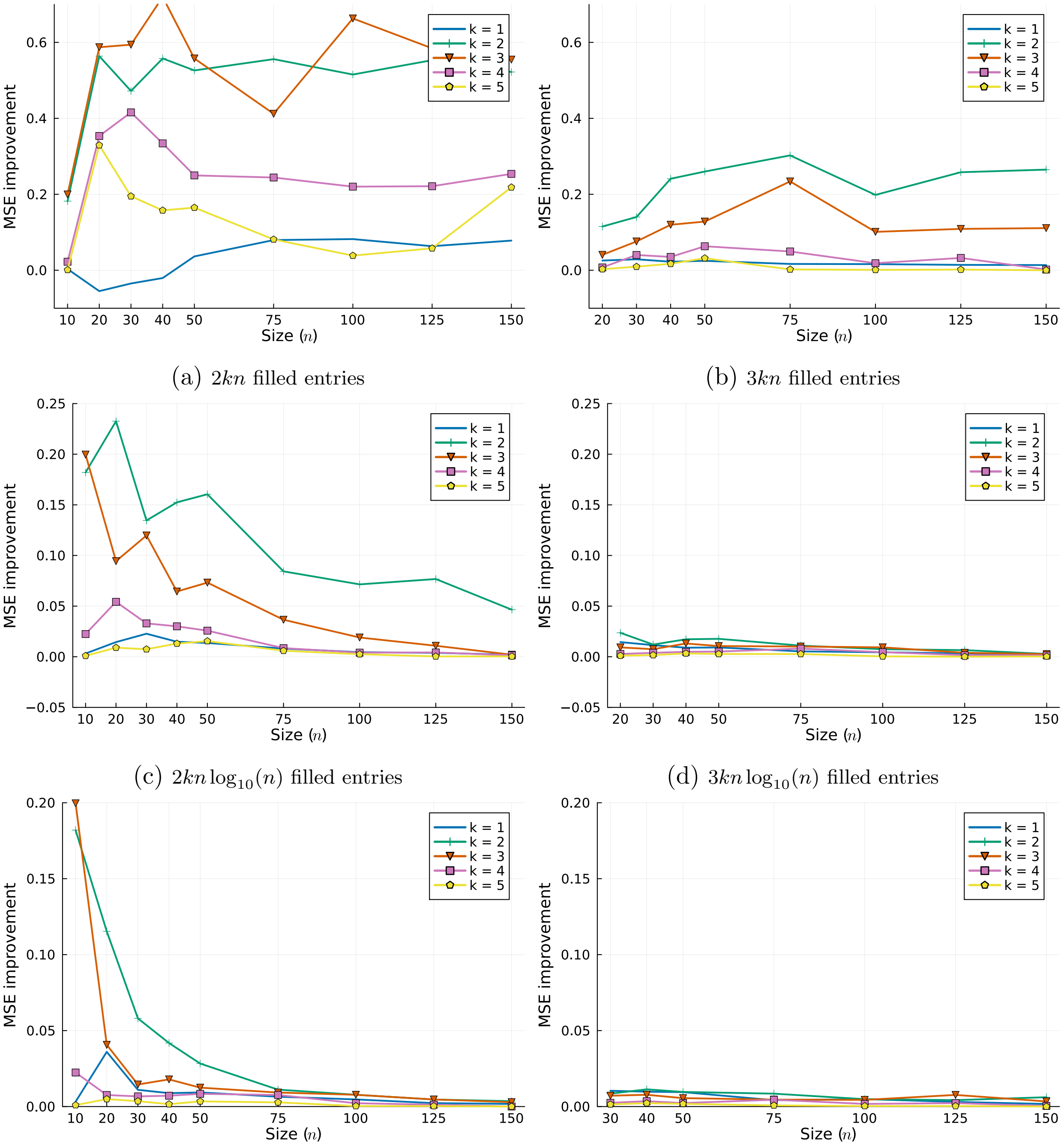
Table EC.10 Results for rank-one matrix completion with 4-piece disjunctions starting with pkn filled entries over different problem instances (average of 20 instances per row).

n	p	Nodes explored	Relaxation solve time (s)	Total time (s)	Relative gap (root node)	Relative gap
50	2.0	1051.2	3.175	>3600.0	3.72×10^{-2}	1.63×10^{-2}
50	2.5	1032.65	3.262	>3600.0	2.13×10^{-2}	1.15×10^{-2}
50	3.0	1018.8	3.332	>3600.0	1.98×10^{-2}	7.46×10^{-3}
75	2.0	191.3	18.35	>3600.0	1.31×10^{-2}	9.24×10^{-3}
75	2.5	191.2	18.38	>3600.0	1.10×10^{-2}	5.41×10^{-3}
75	3.0	187.9	18.79	>3600.0	8.23×10^{-3}	4.21×10^{-3}
100	2.0	60.65	59.09	>3600.0	6.24×10^{-3}	3.21×10^{-3}
100	2.5	58.4	61.4	>3600.0	6.56×10^{-3}	2.36×10^{-3}
100	3.0	58.1	61.8	>3600.0	5.91×10^{-3}	1.52×10^{-3}
125	2.0	21.7	154.1	2935.0	3.14×10^{-3}	1.49×10^{-3}
125	2.5	21.7	151.7	2669.8	1.03×10^{-3}	6.62×10^{-4}
125	3.0	20.1	155.3	2284.7	5.13×10^{-4}	3.00×10^{-4}
150	2.0	10.95	335.2	>3600.0	1.73×10^{-3}	6.62×10^{-4}
150	2.5	10.8	337.8	>3600.0	1.01×10^{-3}	5.82×10^{-4}
150	3.0	10.3	339.3	3404.0	1.79×10^{-3}	4.18×10^{-4}
200	2.0	1.45	1200.0	1563.0	6.06×10^{-7}	9.65×10^{-8}
200	2.5	1.4	1190.0	1496.6	5.04×10^{-7}	1.96×10^{-7}
200	3.0	1.4	1139.0	1462.1	4.69×10^{-7}	5.25×10^{-8}
250	2.0	1.2	3118.0	3581.6	5.54×10^{-7}	9.22×10^{-8}
250	2.5	1.19	3169.0	>3600.0	7.42×10^{-7}	2.14×10^{-7}
250	3.0	1.25	3175.0	>3600.0	5.96×10^{-7}	1.44×10^{-7}

Table EC.11 Results for rank-one matrix completion with 2-piece disjunctions starting with $pkn\log(n)$ filled entries over different problem instances (average of 20 instances per row).

n	p	Nodes explored	Relaxation solve time (s)	Total time (s)	Relative gap (root node)	Relative gap
50	2.0	439.42	5.499	1733.1	3.12×10^{-2}	3.95×10^{-3}
50	2.5	374.59	5.999	1483.6	2.10×10^{-2}	1.79×10^{-3}
50	3.0	248.4	5.233	429.4	1.98×10^{-2}	1.08×10^{-3}
75	2.0	86.95	28.58	1674.1	1.31×10^{-2}	4.40×10^{-3}
75	2.5	86.44	29.59	1862.2	1.07×10^{-2}	1.67×10^{-3}
75	3.0	74.3	28.1	1263.0	8.23×10^{-3}	1.30×10^{-3}
100	2.0	33.05	85.73	2350.6	6.24×10^{-3}	2.15×10^{-3}
100	2.5	30.7	88.56	2098.9	6.56×10^{-3}	3.28×10^{-3}
100	3.0	28.39	88.09	1858.3	5.44×10^{-3}	1.29×10^{-3}
125	2.0	13.84	212.4	2475.3	3.38×10^{-3}	1.72×10^{-3}
125	2.5	11.21	204.0	1725.8	1.08×10^{-3}	7.61×10^{-4}
125	3.0	10.65	205.1	1528.5	5.13×10^{-4}	3.33×10^{-4}
150	2.0	6.9	467.6	3135.6	1.73×10^{-3}	1.03×10^{-3}
150	2.5	7.15	456.1	3174.3	1.01×10^{-3}	8.86×10^{-4}
150	3.0	6.84	463.7	3092.4	1.73×10^{-3}	1.05×10^{-3}
200	2.0	1.4	1180.0	1506.3	6.06×10^{-7}	6.07×10^{-8}
200	2.5	1.5	1159.0	1506.9	5.04×10^{-7}	1.08×10^{-7}
200	3.0	1.55	1137.0	1533.1	4.69×10^{-7}	8.96×10^{-8}
250	2.0	1.27	3157.0	>3600.0	1.53×10^{-6}	7.21×10^{-7}
250	2.5	1.12	3112.0	3376.6	4.65×10^{-7}	1.24×10^{-7}
250	3.0	1.23	3077.0	>3600.0	8.79×10^{-7}	9.32×10^{-8}

Table EC.12 Results for rank-one matrix completion with 4-piece disjunctions starting with $pkn\log(n)$ filled entries over different problem instances (average of 20 instances per row).



(a) $2kn$ filled entries

(b) $3kn$ filled entries

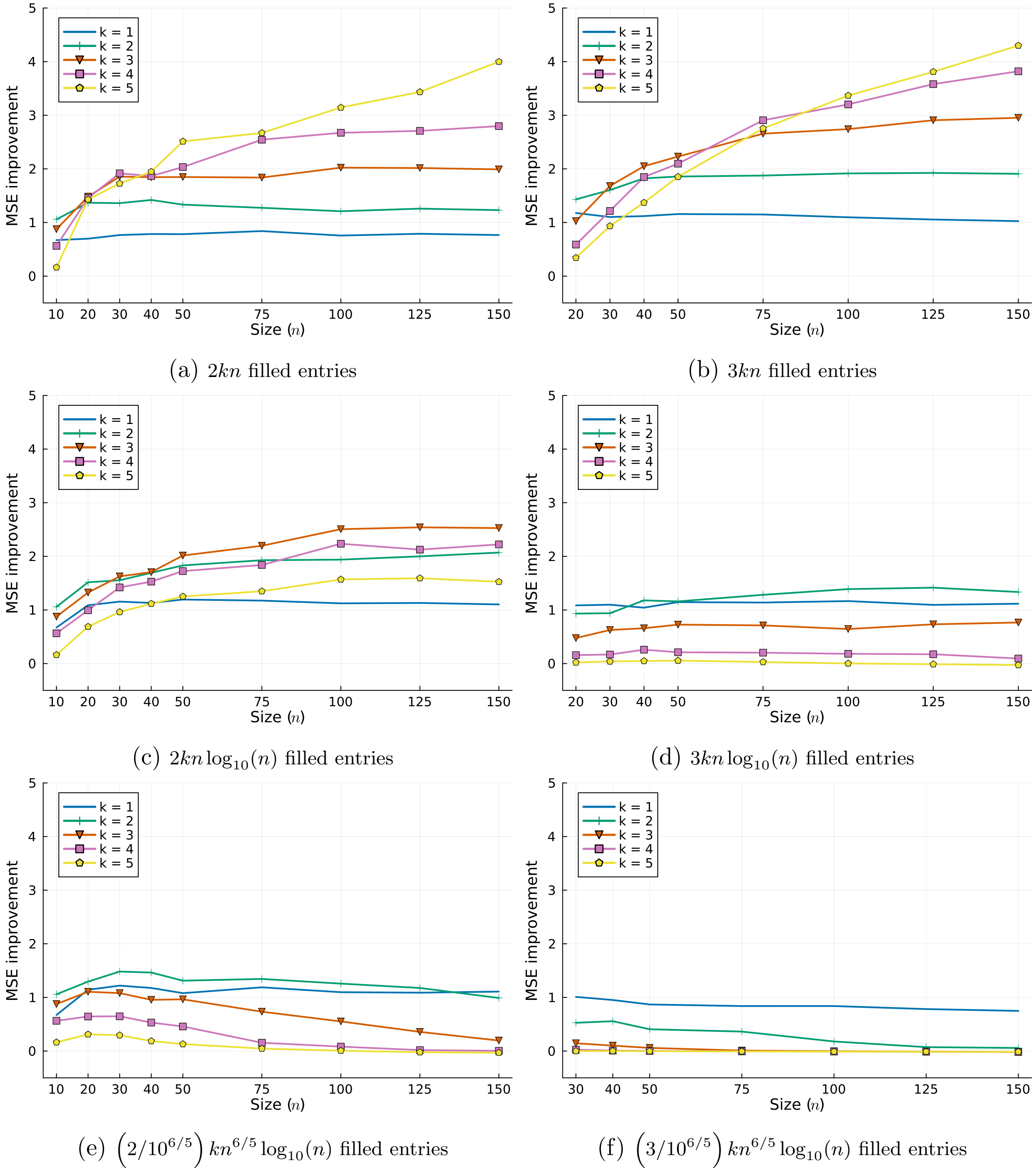
(c) $2kn\log_{10}(n)$ filled entries

(d) $3kn\log_{10}(n)$ filled entries

(e) $\left(2/10^{6/5}\right)kn^{6/5}\log_{10}(n)$ filled entries

(f) $\left(3/10^{6/5}\right)kn^{6/5}\log_{10}(n)$ filled entries

Figure EC.7 MSE improvement (in absolute terms) for rank-k matrix completion problems against MFSGD, varying n and k, with $\gamma = 80.0$, averaged over 50 random instances per point.



(a) $2kn$ filled entries

(b) $3kn$ filled entries

(c) $2kn\log_{10}(n)$ filled entries

(d) $3kn\log_{10}(n)$ filled entries

(e) $\left(2/10^{6/5}\right)kn^{6/5}\log_{10}(n)$ filled entries

(f) $\left(3/10^{6/5}\right)kn^{6/5}\log_{10}(n)$ filled entries

Figure EC.8 MSE improvement (in absolute terms) for rank-k matrix completion problems against alternating minimization at the root node, varying n and k, with $\gamma = 80.0$, averaged over 50 random instances per point.