Journal of Machine Learning Research 24 (2023) 1-51 Submitted 9/21; Published 10/23

Sparse Plus Low Rank Matrix Decomposition: A Discrete Optimization Approach

Dimitris Bertsimas DBERTSIM@MIT.EDU
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Ryan Cory-Wright R.CORY-WRIGHT@IMPERIAL.AC.UK
Imperial College Business School
London, SW7 2AZ, UK

Nicholas A. G. Johnson NAGJ@MIT.EDU
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Editor: Sathiya Keerthi

Abstract

We study the Sparse Plus Low-Rank decomposition problem (SLR), which is the problem of decomposing a corrupted data matrix into a sparse matrix of perturbations plus a low-rank matrix containing the ground truth. SLR is a fundamental problem in Operations Research and Machine Learning which arises in various applications, including data compression, latent semantic indexing, collaborative filtering, and medical imaging. We introduce a novel formulation for SLR that directly models its underlying discreteness. For this formulation, we develop an alternating minimization heuristic that computes high-quality solutions and a novel semidefinite relaxation that provides meaningful bounds for the solutions returned by our heuristic. We also develop a custom branch-and-bound algorithm that leverages our heuristic and convex relaxations to solve small instances of SLR to certifiable (near) optimality. Given an input n-by-n matrix, our heuristic scales to solve instances where $n = 10000$ in minutes, our relaxation scales to instances where $n = 200$ in hours, and our branch-and-bound algorithm scales to instances where $n = 25$ in minutes. Our numerical results demonstrate that our approach outperforms existing state-of-the-art approaches in terms of rank, sparsity, and mean-square error while maintaining a comparable runtime.

Keywords: Sparsity; Rank; Matrix Decomposition; Convex Relaxation; Branch-and-bound

1 Introduction

The *Sparse Plus Low Rank* (SLR) decomposition problem, or the problem of approximately decomposing a data matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$ into a sparse matrix $\boldsymbol{Y}$ plus a low-rank matrix $\boldsymbol{X}$, arises throughout many fundamental applications in Operations Research, Machine Learning, and Statistics, including collaborative filtering (Recht et al., 2010), medical resonance imaging (Chen et al., 2017), and economic modeling (Basu et al., 2019) among others.

Formally, given a target rank k_0 and a target sparsity k_1, we solve:

$$\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\lambda\|\boldsymbol{X}\|_F^2+\mu\|\boldsymbol{Y}\|_F^2 \text{ s.t. } \mathrm{Rank}(\boldsymbol{X})\leq k_0, \|\boldsymbol{Y}\|_0\leq k_1, \tag{1}$$

where $\lambda, \mu > 0$ are parameters that control sensitivity to noise and are to be cross-validated by minimizing a validation metric (see, e.g., Owen and Perry, 2009) to obtain strong out-of-sample performance in theory and practice (Bousquet and Elisseeff, 2002).

In SLR decomposition problems, the sparse matrix $\boldsymbol{Y}$ accounts for a small number of potentially large corruptions in $\boldsymbol{D}$, while $\boldsymbol{X}$ models the leading principal components of $\boldsymbol{D}$ after this corruption is removed. This is well justified, because SLR robustifies Principal Component Analysis (PCA), a leading technique for finding low-rank approximations of noiseless datasets (Pearson, 1901), which performs poorly in high-dimensional settings and in the presence of noise (Negahban and Wainwright, 2011). In an opposite direction, SLR robustly accounts for noise via the sparse matrix $\boldsymbol{Y}$, while $\boldsymbol{X}$ recovers the uncorrupted principal component directions of $\boldsymbol{D}$. Correspondingly, SLR decomposition schemes, which are also called Robust PCA since at least the work of Candès et al. (2011), are widely regarded as state-of-the-art approaches for high-dimensional matrix estimation problems (Chandrasekaran et al., 2011; Negahban and Wainwright, 2011).

Our formulation (1) is also well-justified from an information-theoretic perspective. Indeed, several authors (Arous et al., 2020; Gamarnik, 2021) have demonstrated for special cases of Problem (1) that when the ground truth is sparse and/or low-rank, exact sparse and/or low-rank formulations recover the ground truth at least as accurately as any polynomial time method, and indeed there is a gap between the amount of data required for an "exact" sparse plus low-rank formulation to recover the ground truth, and the amount of data required for a polynomial time approach (an Overlap Gap Property Gamarnik, 2021). A key characteristic of Problem (1) is that it directly employs a sparsity constraint on $\boldsymbol{Y}$ and a rank constraint on $\boldsymbol{X}$. These constraints are non-convex, which make (1) a difficult problem to solve exactly, both in practice—where the best-known exact algorithms cannot certify optimality beyond $n = 10$ (Lee and Zou, 2014)—and in theory, where the problem is NP-hard by reduction from low-rank matrix approximation (Gillis and Glineur, 2011).

In this work, we develop an alternating minimization heuristic and convex relaxation which collectively provide very small bound gaps for (1) and scale to high-dimensional settings. Our heuristic scales to $n = 10000$ in minutes and our convex relaxation scales to $n = 200$ in hours. A key feature of the approach is that it leverages the underlying discreteness of the problem to obtain tight yet computationally cheap lower bounds. We further demonstrate that the alternating minimization heuristic and convex relaxation can be embedded within a branch-and-bound tree to solve (1) to certifiable near-optimality for instances of size up to $n = 25$.

1.1 Contribution and Structure

The key contributions of the paper are threefold:

- First, from a methodological perspective, we introduce a novel formulation (1) for the SLR decomposition problem that directly exploits the underlying discreteness of the problem. Our formulation is inspired by incorporating robustness against adversarial perturbations in the input data in SLR, which is useful in noisy settings.

- Second, from an algorithmic perspective, we develop a heuristic that obtains high quality feasible solutions to Problem (1) in Section 3 and derive a convex relaxation of (1) that provides high-quality bounds for the solutions returned by our heuristic in Section 4. We also interpret the convex relaxation as a novel reverse Huber penalty which penalizes the sparse and low-rank matrices in a convex manner. Further, we present a branch-and-bound framework that solves (1) to certifiable near-optimality for small problem instances in Section 5.
- Third, from a computational perspective, we extensively benchmark our proposed approach. Across a suite of numerical experiments, we demonstrate in Section 6 that our approach outperforms state-of-the-art non-convex methods like AccAltProj, GoDec and ScaledGD by obtaining sparser and lower rank matrices with a lower mean-squared error than via prior attempts, in a comparable amount of computational time. Moreover, our approach scales to successfully solve problem instances with 10000×10000 matrices.

Notation: We let nonbold face characters such as b denote scalars, lowercase bold-faced characters such as $\boldsymbol{x}$ denote vectors, uppercase bold-faced characters such as $\boldsymbol{X}$ denote matrices, and calligraphic uppercase characters such as $\mathcal{Z}$ denote sets. We let $[n]$ denote the set of running indices $\{1, ..., n\}$ and $\langle \cdot, \cdot \rangle$ denote the Euclidean (Frobenius) inner product between two vectors (matrices) of the same dimension. We let $\mathbf{e}$ denote a vector of all 1's, $\mathbf{0}$ denote a vector of all 0's, and $\mathbb{I}$ denote the identity matrix. Finally, we let $\mathcal{S}^n$ ($\mathcal{S}^n_+$) denote the cone of $n \times n$ symmetric (positive semidefinite) matrices.

2 Literature Review and SLR Formulation Properties

In this section, we judiciously characterize Problem (1) and state-of-the-art approaches for addressing it. First, in Section 2.1, we cast a deliberate eye over existing attempts at solving Problem (1) that are currently considered to be state-of-the-art and establish that these approaches are either heuristics that do not provide performance guarantees or branch-and-bound methods that do not scale to even moderate problem sizes. Next, in Section 2.2, we establish several key properties of Problem (1)'s objective function that we invoke throughout the paper. Further, in Section 2.3, we justify the regularization terms in our formulation by interpreting our formulation through the lens of robust optimization. Finally, in Section 2.4, we characterize the conditions under which Problem (1) admits a reduction to matrix completion, a famous and frequently studied cousin of Problem (1) which is notoriously computationally challenging (Candes and Plan, 2010).

2.1 Literature Review

In this section, we selectively review several formulations from the literature that have been employed to solve the sparse plus low-rank decomposition problem and are currently considered to be state-of-the-art. Most of these approaches are heuristic in nature and do not provide valid lower bounds to certify the (sub) optimality of the output solution.

2.1.1 Stable Principal Component Pursuit

Optimizing over low-rank matrices is notoriously computationally challenging in both theory and practice (Recht et al., 2010; Bertsimas et al., 2022). Accordingly, a popular approach is to replace the rank and sparsity terms with their nuclear norm and ℓ_1 norm surrogates, as advocated by Chandrasekaran et al. (2011); Candès et al. (2011) among others. In the presence of noise, this substitution leads to the following formulation, which was originally proposed by Zhou et al. (2010) and is called Stable Principal Component Pursuit (S-PCP):

$$\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad \|\boldsymbol{X}\|_* + \frac{1}{\sqrt{n}}\|\boldsymbol{Y}\|_1 + \frac{1}{2\mu}\|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2. \tag{2}$$

Problem (2) can either be reformulated as a semidefinite problem over a $2n \times 2n$ matrix as advocated by Candès et al. (2011), solved in the original space using a nonsymmetric interior point method as proposed by Skajaa and Ye (2015) or solved in a semidefinite free fashion using an augmented Lagrangian approach as advocated by Yuan and Yang (2013). Unfortunately, all three approaches require repeatedly performing operations such as a singular value decomposition or a Newton step, which has an $O(n^3)$ or higher time/memory cost. Correspondingly, all such semidefinite optimization approaches require too much memory to be successfully implemented in a standard computational environment when $n = 200$, at least with current technology (see Majumdar et al., 2020, for a review of the state-of-the-art in semidefinite optimization). Moreover, these methods are usually only guaranteed to recover a ground truth model under a mutual incoherence condition (or similar) on the ground truth (see Tillmann and Pfetsch, 2013, for a review), which implies that performance guarantees for such semidefinite methods are challenging to obtain indeed.

2.1.2 GoDec

Many existing formulations for SLR employ convex relaxations of the rank function and the ℓ_0 norm function rather than exploiting the inherent discreteness of the problem. An exception to this pattern is the work of Zhou and Tao (2011), who leverage discreteness to obtain higher quality solutions to SLR. Their formulation is given by:

$$\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 \text{ s.t. } \mathrm{Rank}(\boldsymbol{X}) \leq k_0, \|\boldsymbol{Y}\|_0 \leq k_1. \tag{3}$$

Note that (3) differs from (1) by the absence of regularization terms on $\boldsymbol{X}$ and $\boldsymbol{Y}$. Zhou and Tao (2011) obtain a feasible solution to (3) by performing alternating minimization on $\boldsymbol{X}, \boldsymbol{Y}$. Their algorithm, called GoDec, is similar in structure to the algorithm we develop in Section 3 to obtain high-quality solutions to Problem (1). In a related direction, Yan et al. (2015) adopt a similar approach to GoDec in the special case where their design matrix is taken to be the identity. Kyrillidis and Cevher (2012) adopt a similar formulation as GoDec, however, they instead minimize the reconstruction error between an observation vector and a vector-valued linear map of the sum of the low-rank and sparse matrices. In a somewhat different vein, Zhang and Yang (2018) consider an explicit rank constraint but not a sparsity constraint and proceed by leveraging manifold optimization techniques.

2.1.3 Low Rank Matrix Parameterization

An extensively studied family of methods parameterizes the low-rank matrix $\boldsymbol{X}$ as $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{V}^T$ where $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{n\times k_0}$, and performs alternating minimization on $\boldsymbol{U}, \boldsymbol{V}$. Originally proposed in the context of low-rank semidefinite optimization by Burer and Monteiro (2003, 2005) (see also Jain et al., 2013), it has since evolved into an extensively used and practical approach for SLR problems (Netrapalli et al., 2014; Chen and Wainwright, 2015; Gu et al., 2016; Cai et al., 2019). This approach eliminates the rank constraint and can substantially reduce the number of variables when $n \ll k_0$ at the expense of introducing non-convexity in the objective. Remarkably, in many circumstances, the induced non-convexity is benign and the resulting Burer-Monteiro reformulation can be solved efficiently from both a theoretical and a practical perspective. We refer readers to Chi et al. (2019) for a detailed overview.

Two important parametrization-based approaches to SLR are Fast RPCA (Yi et al., 2016) and Scaled Gradient Descent (Tong et al., 2021). In Fast RPCA, after parametrizing the low-rank matrix, Yi et al. (2016) augment the objective with a regularization term on the norm of $(\boldsymbol{U}^T\boldsymbol{U} - \boldsymbol{V}^T\boldsymbol{V}) \in \mathbb{R}^{k_0 \times k_0}$ before performing alternating minimization on $\boldsymbol{U}$ and $\boldsymbol{V}$. In an alternate direction, Tong et al. (2021) performs iterative gradient descent updates on $\boldsymbol{U}$ and $\boldsymbol{V}$ in Scaled Gradient Descent after designing an effective gradient preconditioner that results in desirable convergence behavior even for ill-conditioned problems. However, existing performance guarantees for these approaches rely on assumptions on the structure of the ground truth, such as mutual incoherence, that are difficult to verify without independent access to the ground truth or on being initialized within a “basin of attraction” which similarly is difficult to verify. We point out, however, that one could either use the dual bounds derived in this paper, or side information such as scoring by humans (e.g., in video background separation applications) to provide performance guarantees when the ground truth is not known.

2.1.4 Branch and Bound

To our knowledge, the only existing work that provides guarantees on the quality of solutions to Problem (1) is Lee and Zou (2014), who propose a branch-and-bound algorithm for solving Problem (1) to near-optimality. Specifically, they assume that the spectral norm of $\boldsymbol{X}$ is bounded from above by β, i.e., $\beta \geq \|\boldsymbol{X}\|_\sigma$, and invoke the following inequality to obtain valid lower bounds for each partially specified sparsity pattern (see also Fazel, 2002):

$$\frac{\gamma}{\alpha}\|\boldsymbol{Y}\|_1 + \frac{1}{\beta}\|\boldsymbol{X}\|_* \leq \gamma\|\boldsymbol{Y}\|_0 + \mathrm{Rank}(\boldsymbol{X}), \tag{4}$$

where $\alpha \geq \|\boldsymbol{Y}\|_\infty$ is a bound on the ℓ_∞ norm of $\boldsymbol{Y}$, which can either be taken to be equal to some large fixed constant M (Glover, 1975) or treated as a regularization parameter (Bertsimas et al., 2021). Unfortunately, while Lee and Zou (2014)'s bound is often reasonable, it was not developed by taking the convex envelope of an appropriate substructure of Problem (1), and therefore is not strong enough to solve Problem (1) to optimality at even small problem sizes (see also Bienstock, 2010, for a related discussion on the weakness of big-M bounds). Indeed, the authors reported bound gaps but not optimal solutions for SLR problems when $n = 10$. Nonetheless, this lower bound is potentially interesting in its own right, since it demonstrates that the PCP formulation supplies a valid lower bound on Problem (1) if one is willing to either make a big-M assumption on the spectral norm of the low-rank matrix or compute a valid M (c.f. Bertsimas et al., 2022, Section 3.5).

2.2 Objective Function Properties

We now derive several key properties of Problem (1) that we leverage throughout the paper and present a probabilistic interpretation of (1) which is motivated by Bayesian inference. Specifically, we establish that (1)'s objective is strongly convex, Lipschitz continuous, and the Maximum A Posteriori (MAP) estimator of a suitably defined probabilistic model under a Gaussian prior. Recall that a function $f(\boldsymbol{Z})$ is said to be strongly convex with parameter m (m-strongly convex) if the function $f(\boldsymbol{Z}) - \frac{m}{2}\|\boldsymbol{Z}\|_F^2$ is convex. Similarly, a function $f(\boldsymbol{Z})$ is said to be Lipschitz continuous with constant L (L-Lipschitz) if the function $\frac{L}{2}\|\boldsymbol{Z}\|_F^2 - f(\boldsymbol{Z})$ is convex. Formally, we have the following results (proofs deferred to Appendix A):

Proposition 1 *The function $f(\boldsymbol{X},\boldsymbol{Y}) = \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2$ is jointly m-strongly convex in $(\boldsymbol{X},\boldsymbol{Y})$ over $\mathbb{R}^{n\times n}\times\mathbb{R}^{n\times n}$, i.e., $g(\boldsymbol{X},\boldsymbol{Y}) = f(\boldsymbol{X},\boldsymbol{Y}) - \frac{m}{2}(\|\boldsymbol{X}\|_F^2 + \|\boldsymbol{Y}\|_F^2)$ is jointly convex in $(\boldsymbol{X},\boldsymbol{Y})$, for $m = 2\cdot\min(\lambda,\mu)$.*

Proposition 2 *The function $f(\boldsymbol{X},\boldsymbol{Y}) = \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2$ is L-Lipschitz continuous in $(\boldsymbol{X},\boldsymbol{Y})$ over $\mathbb{R}^{n\times n}\times\mathbb{R}^{n\times n}$ for $L = 2\cdot\max(\lambda,\mu)+6$.*

Note that Propositions 1–2 collectively imply that the condition number κ of $f(\boldsymbol{X},\boldsymbol{Y})$ is

$$\kappa = \frac{L}{m} = \frac{2\cdot\max(\lambda,\mu)+6}{2\cdot\min(\lambda,\mu)}. \tag{5}$$

We now provide a probabilistic interpretation of $f(\boldsymbol{X},\boldsymbol{Y})$. Suppose the data $\boldsymbol{D}\in\mathbb{R}^{n\times n}$ are sampled from

$$\boldsymbol{D} = \boldsymbol{X} + \boldsymbol{Y} + \boldsymbol{\epsilon}, \tag{6}$$

where $\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}$ are unknown parameters to be estimated and $\boldsymbol{\epsilon}\in\mathbb{R}^{n\times n}$, $\epsilon_{ij}\sim N(0,\sigma^2)$ is i.i.d Gaussian noise with variance σ^2. If we adopt independent Gaussian prior beliefs $X_{ij}\sim N(0,\sigma^2/\lambda)$ and $Y_{ij}\sim N(0,\sigma^2/\mu)$ over the parameters $\boldsymbol{X},\boldsymbol{Y}$, then the Maximum A Posteriori (MAP) estimate of $\boldsymbol{X},\boldsymbol{Y}$ after observing $\boldsymbol{D}$ is given by $\arg\min_{\boldsymbol{X},\boldsymbol{Y}} f(\boldsymbol{X},\boldsymbol{Y})$.

To see this, note that the posterior probability after observing $\boldsymbol{D}$ is given by

$$\mathbf{P}(\boldsymbol{X},\boldsymbol{Y}|\boldsymbol{D}) = \frac{\boldsymbol{P}(\boldsymbol{D}|\boldsymbol{X},\boldsymbol{Y})\mathbf{P}(\boldsymbol{X})\mathbf{P}(\boldsymbol{Y})}{\mathbf{P}(\boldsymbol{D})} \propto \boldsymbol{P}(\boldsymbol{D}|\boldsymbol{X},\boldsymbol{Y})\mathbf{P}(\boldsymbol{X})\mathbf{P}(\boldsymbol{Y}). \tag{7}$$

We can now obtain the MAP estimate by maximizing the posterior probability as follows

$$\begin{aligned}\arg\max_{\boldsymbol{X},\boldsymbol{Y}} \boldsymbol{P}(\boldsymbol{D}|\boldsymbol{X},\boldsymbol{Y})\mathbf{P}(\boldsymbol{X})\mathbf{P}(\boldsymbol{Y}) &= \arg\max_{\boldsymbol{X},\boldsymbol{Y}} \prod_{1\le i,j\le n} \frac{e^{\frac{-(D_{ij}-X_{ij}-Y_{ij})^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}\cdot\frac{\sqrt{\lambda}e^{\frac{-\lambda X_{ij}^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}\cdot\frac{\sqrt{\mu}e^{\frac{-\mu Y_{ij}^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}\\ &= \arg\min_{\boldsymbol{X},\boldsymbol{Y}} \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2 = \arg\min_{\boldsymbol{X},\boldsymbol{Y}} f(\boldsymbol{X},\boldsymbol{Y})\end{aligned}$$

where the second equality follows by taking a log transformation and multiplying by $-2\sigma^2$.

2.3 Equivalence Between Regularization and Robustness

Real-world datasets are replete with inaccurate and missing data values, which prevents machine-learning models that do not account for these inconsistencies from generalizing well to unseen data. Accordingly, robustness is a highly desirable attribute for machine learning models, in both theory and practice (Xu et al., 2009; Bertsimas and den Hertog, 2020). In this section, we demonstrate that our regularized problem (1) is equivalent to a robust optimization (RO) problem. This result motivates the inclusion of the Frobenius regularization terms within (1) and verifies that (assuming the hyperparameters in (1) are correctly cross-validated), regularization improves (1)'s out-of-sample performance.

We remark that our results should not be too surprising to readers familiar with the RO literature. Indeed, Bertsimas and Copenhaver (2018) have already derived a similar result for regularized linear regression problems. However, our main result is strictly more general. Indeed, Bertsimas and Copenhaver (2018) prove that augmenting an ℓ_2 loss function with an ℓ_2 regularization penalty is equivalent to solving a RO problem, and conjecture (but do not prove) that their result can be extended to ordinary least squares regression and ridge regularization (with ℓ_2^2 rather than ℓ_2 penalties). On the other hand, we prove a matrix analog of their result and generalize their result to the matrix analog of ℓ_2^2 regularization. Accordingly, this section may be of independent interest to the RO community.

We now connect our work with the work of Bertsimas and Copenhaver (2018) by deriving a conceptually simple analog of their characterization of the equivalence of regularization and robustness for sparse plus low-rank problems. This result sheds insight into the nature of regularization as a robustifying force in Problem (1). Subsequently, we derive an (admittedly more opaque) characterization of Problem (1) itself as a RO problem.

Formally, we have the following results (proofs deferred to Appendix A):

Proposition 3 *Let $\mathcal{U}_\lambda(\boldsymbol{X}) = \{\boldsymbol{\Delta} \in \mathbb{R}^{n\times n} : \|\boldsymbol{\Delta}\|_F \leq \lambda\|\boldsymbol{X}\|_F\}$ for $\boldsymbol{X} \in \mathbb{R}^{n\times n}, \lambda > 0$. Consider the robust optimization problem:*

$$\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \max_{\substack{\boldsymbol{\Delta}_1\in\mathcal{U}_\lambda(\boldsymbol{X}) \\ \boldsymbol{\Delta}_2\in\mathcal{U}_\mu(\boldsymbol{Y})}} \|\boldsymbol{D} + \boldsymbol{\Delta_1} + \boldsymbol{\Delta_2} - \boldsymbol{X} - \boldsymbol{Y}\|_F \text{ s.t. } \boldsymbol{X} \in \mathcal{V}, \boldsymbol{Y} \in \mathcal{W}, \tag{8}$$

where $\mathcal{V}$ and $\mathcal{W}$ are arbitrary subsets of $\mathbb{R}^{n\times n}$. Then, (8) *is equivalent to* (9)*.*

$$\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \|\boldsymbol{D} - \boldsymbol{X} - \boldsymbol{Y}\|_F + \lambda\|\boldsymbol{X}\|_F + \mu\|\boldsymbol{Y}\|_F \text{ s.t. } \boldsymbol{X} \in \mathcal{V}, \boldsymbol{Y} \in \mathcal{W}. \tag{9}$$

Proposition 4 *Problem* (1) *is equivalent to the following robust optimization problem:*

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \max_{\boldsymbol{\Delta}_1,\boldsymbol{\Delta}_2} \quad & \|\boldsymbol{D} - \boldsymbol{X} - \boldsymbol{Y}\|_F^2 + \langle\boldsymbol{X}, \boldsymbol{\Delta_1}\rangle + \langle\boldsymbol{Y}, \boldsymbol{\Delta}_2\rangle - \frac{1}{4\lambda}\|\boldsymbol{\Delta}_1\|_F^2 - \frac{1}{4\mu}\|\boldsymbol{\Delta}_2\|_F^2 \\ \text{s.t.} \quad & \boldsymbol{X} \in \mathcal{V}, \boldsymbol{Y} \in \mathcal{W}.\end{aligned} \tag{10}$$

Taking $\mathcal{V}$ to be the set of matrices with rank at most k_0 and $\mathcal{W}$ to be the set of matrices with ℓ_0 norm at most k_1, Proposition 3 implies that performing SLR decomposition with Frobenius regularization is equivalent to solving a RO problem that allows for adversarial errors in the input data matrix $\boldsymbol{D}$. Moreover, Proposition 4 implies that solving Problem (1) is equivalent to solving a RO problem with a soft robust penalty term in the objective, rather than a hard constraint on the size of the uncertainty set, as such robust equivalent problems usually consist of. This result is perhaps unsurprising in retrospect, since dual problems to quadratically constrained quadratic problems involve quadratic terms in the objective (see also Roos et al., 2020, Section 6.3).

2.4 Connection to Matrix Completion

Low-rank matrix completion is a canonical problem in the Statistics and Machine Learning communities that has been employed in control theory (Boyd et al., 1994), computer vision (Candes and Plan, 2010), and signal processing (Ji et al., 2010) among other applications. Given a partially observed matrix $\boldsymbol{D} \in \mathbb{R}^{n\times n}$ where $\Omega \subset \{(i,j) : 1 \leq i,j \leq n\}$ denotes the set of indices of the revealed entries, the low-rank matrix completion problem is to compute a low-rank matrix $\boldsymbol{X}$ that approximates $\boldsymbol{D}$. Low-rank matrix completion solves

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \quad \sum_{(i,j)\in\Omega} (D_{ij} - X_{ij})^2 \text{ s.t. } \mathrm{Rank}(\boldsymbol{X}) \leq k_0, \tag{11}$$

where k_0 is a predefined target rank.

Although we require $\lambda, \mu > 0$ in our formulation of SLR given by (1), we now show that if we take $\mu = 0$ and also fix a sparsity pattern for the sparse matrix $\boldsymbol{Y}$, then (1) reduces to regularized matrix completion. Let $\boldsymbol{Z} \in \{0,1\}^{n\times n}$ be a matrix such that if $Z_{ij} = 0$, we must have $Y_{ij} = 0$. We refer to $\boldsymbol{Z}$ as a valid sparsity pattern for (1) if $\sum_{ij} Z_{ij} \leq k_1$. Formally, we have (proof deferred to Appendix A):

Proposition 5 *Given a valid sparsity pattern $\boldsymbol{Z}$, if we take $\mu = 0$ then* (1) *reduces to regularized matrix completion with $\Omega = \{(i,j) : Z_{ij} = 0\}$.*

3 An Alternating Minimization Heuristic

In this section, we propose an alternating minimization algorithm that obtains high-quality feasible solutions to (1) in Section 3.2, by iteratively fixing the sparse or low-rank matrix and optimizing the remaining matrix. This is a reasonable strategy, because alternating minimization (AM) strategies are known to obtain high-quality solutions to low-rank problems (Jain et al., 2013) and, as we demonstrate in Section 3.1, when one matrix is fixed the other matrix can be optimized in closed form. Consequently, Problem (1) is amenable to AM techniques. Further, in Section 3.2, we bound the number of iterations required for AM to converge. Finally, in Section 3.3, we establish that for a fixed sparsity pattern and a sufficiently large amount of regularization, AM yields a globally optimal solution to (1). This result provides the basis for the branch-and-bound algorithm we develop in Section 5.

3.1 Two Natural Subproblems

In this subsection, we derive two subproblems of (1) by fixing either the sparse matrix $\boldsymbol{Y}$ (to obtain a low-rank subproblem) or the low-rank matrix $\boldsymbol{Y}$ (to obtain a sparse subproblem). Further, we establish that both subproblems admit closed-form solutions.

Low-Rank Subproblem: First, suppose that we fix a sparse matrix $\boldsymbol{Y}^*$ in Problem (1). Then, (1) becomes:

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \quad \|\bar{\boldsymbol{D}} - \boldsymbol{X}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 \quad \text{s.t.} \quad \mathrm{Rank}(\boldsymbol{X}) \leq k_0, \tag{12}$$

where $\bar{\boldsymbol{D}} = \boldsymbol{D} - \boldsymbol{Y}^*$ and we omit the regularization term on $\boldsymbol{Y}$ since it does not depend on $\boldsymbol{X}$. We refer to Problem (12) as the low-rank subproblem. We now demonstrate that this problem admits a closed-form solution, via the following result:

Proposition 6 *Let $\boldsymbol{X}^*$ be a matrix such that*

$$\boldsymbol{X}^* = \frac{1}{1+\lambda}\bar{\boldsymbol{D}}_{k_0},$$

where $\bar{\boldsymbol{D}}_{k_0}$ is a top-k_0 SVD approximation of $\bar{\boldsymbol{D}}$, i.e., $\bar{\boldsymbol{D}}_{k_0} = \boldsymbol{U}_{k_0}\boldsymbol{\Sigma}_{k_0}\boldsymbol{V}_{k_0}^T$ where $\bar{\boldsymbol{D}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is a singular value decomposition of $\bar{\boldsymbol{D}}$. Then, $\boldsymbol{X}^\star$ is an optimal solution to Problem (12).

Proof It is well known that the solution of the problem

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \quad \|\boldsymbol{A}-\boldsymbol{X}\|_F^2 \quad \text{s.t.} \quad \text{Rank}(\boldsymbol{X}) \leq k_0$$

is given by $\boldsymbol{X}^* = \boldsymbol{A}_{k_0}$, a projection of $\boldsymbol{A}$ onto its first k_0 principal components (Wold et al., 1987). Moreover, since

$$\|\bar{\boldsymbol{D}}-\boldsymbol{X}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 - \frac{\lambda}{1+\lambda}\|\bar{\boldsymbol{D}}\|_F^2 = (1+\lambda)\left\|\frac{1}{1+\lambda}\bar{\boldsymbol{D}} - \boldsymbol{X}\right\|_F^2,$$

it follows that Problem (12) is equivalent to (has the same optimal solution set as) solving

$$\min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \quad \left\|\frac{1}{1+\lambda}\bar{\boldsymbol{D}} - \boldsymbol{X}\right\|_F^2 \quad \text{s.t.} \quad \text{Rank}(\boldsymbol{X}) \leq k_0. \tag{13}$$

■

In Appendix B, we provide an alternate proof of Proposition 6 via strong duality which reveals that (12) exhibits hidden convexity in the sense of Ben-Tal and Den Hertog (2014).

Remark 7 *Observe that $\boldsymbol{X}^\star$ can be computed exactly in $O(n^2k)$ time, since we need not compute a full SVD of $\bar{\boldsymbol{D}}$. Alternatively, it can be computed approximately using randomized SVD in $O(n^2\log k)$ time (Halko et al., 2011).*

Sparse Subproblem: Now, suppose we fix a low-rank matrix $\boldsymbol{X}^*$ in Problem (1). Then, (1) problem becomes:

$$\min_{\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad \|\tilde{\boldsymbol{D}}-\boldsymbol{Y}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2 \quad \text{s.t.} \quad \|\boldsymbol{Y}\|_0 \leq k_1, \tag{14}$$

where $\tilde{\boldsymbol{D}} = \boldsymbol{D} - \boldsymbol{X}^*$ and we have omitted the regularization term on the low-rank matrix because it does not depend on $\boldsymbol{Y}$. We refer to Problem (14) as the sparse matrix subproblem. We now demonstrate that this problem also admits a closed-form solution:

Proposition 8 *Let $\boldsymbol{Y}^*$ be a matrix such that*

$$\boldsymbol{Y}^* = \boldsymbol{S}^* \circ \left(\frac{\tilde{\boldsymbol{D}}}{1+\mu}\right),$$

where $\boldsymbol{S}^*$ is a $n\times n$ binary matrix with k_1 entries $S^\star_{ij} = 1$ such that $S^\star_{i,j} \geq S^\star_{k,l}$ if $|\tilde{D}_{i,j}| \geq |\tilde{D}_{k,l}|$ and $\circ$ denotes the Hadamard product operation $((\boldsymbol{A}\circ\boldsymbol{B})_{ij} = A_{ij}\times B_{ij})$. Then, $\boldsymbol{Y}^\star$ solves Problem (14).

Proof It is straightforward to show that the solution of:

$$\min_{\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad \|\boldsymbol{B}-\boldsymbol{Y}\|_F^2 \quad \text{s.t.} \quad \|\boldsymbol{Y}\|_0 \le k_1$$

is given by $\boldsymbol{Y}^* = \boldsymbol{T}^* \circ \boldsymbol{B}$ where $\boldsymbol{T}^*$ is a $n \times n$ binary matrix with k_1 entries $T^\star_{ij} = 1$ such that $T^\star_{i,j} \ge T^\star_{k,l}$ if $|B_{i,j}| \ge |B_{k,l}|$. Moreover, since

$$\|\tilde{\boldsymbol{D}}-\boldsymbol{Y}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2 - \frac{\mu}{1+\mu}\|\tilde{\boldsymbol{D}}\|_F^2 = (1+\mu)\left\|\frac{1}{1+\mu}\tilde{\boldsymbol{D}}-\boldsymbol{Y}\right\|_F^2,$$

it follows that Problem (14) is equivalent to (i.e., has the same optimal solution set as):

$$\min_{\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad \left\|\frac{1}{1+\mu}\tilde{\boldsymbol{D}}-\boldsymbol{Y}\right\|_F^2 \quad \text{s.t.} \quad \|\boldsymbol{Y}\|_0 \le k_1. \tag{15}$$

■

In Appendix D, we provide an alternative proof of Proposition 8 via strong second-order cone duality which may be of independent interest as it reveals that Problem (15) is equivalent to a convex optimization problem.

Remark 9 *Observe that $\boldsymbol{Y}^\star$ can be computed in $O(n^2)$ time, by forming $\tilde{\boldsymbol{D}}$ and partitioning around its kth largest absolute element via quicksort. Correspondingly, this step is computationally cheaper than computing an optimal low-rank matrix. Moreover, since $\tilde{\boldsymbol{D}} \in \mathbb{R}^{n\times n}$, this operation is linear in the number of entries of $\tilde{\boldsymbol{D}}$.*

3.2 An Alternating Minimization Algorithm

By iteratively solving the sparse matrix subproblem and the low-rank matrix subproblem until we either converge to a stationary point or exceed a prespecified number of iterations, we arrive at a feasible solution to (1). We formalize this iterative procedure in Algorithm 1, and let

$$f(\boldsymbol{X},\boldsymbol{Y}) = \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2,$$

be our overall objective function and $\mathcal{V} = \{\boldsymbol{X} \in \mathbb{R}^{n\times n} : \text{Rank}(\boldsymbol{X}) \le k_0\}$, $\mathcal{W} = \{\boldsymbol{Y} \in \mathbb{R}^{n\times n} : \sum_{ij} \mathbb{1}\{Y_{ij} \ne 0\} \le k_1\}$ denote our respective feasible regions.

We note that the initialization strategy $\boldsymbol{X}_0 \leftarrow \boldsymbol{0}$ and $\boldsymbol{Y}_0 \leftarrow \boldsymbol{0}$ is arbitrary and any initialization strategy could equivalently be employed. For instance, one could employ a greedy rounding of the solution to the semidefinite relaxation we derive in Section 4 as an initialization (see also Bertsimas et al., 2022, Section 4.3). Moreover, Algorithm 1 can be executed multiple times for different initializations of $\boldsymbol{X}_0$ and $\boldsymbol{Y}_0$ to obtain an even higher quality feasible solution to (31). This could be performed in parallel to avoid significantly increasing computational time.

It is well-documented in the optimization and machine learning literature that alternating minimization schemes such as Algorithm 1 produce a sequence of non-increasing iterates that converge to a local minimum; for Algorithm 1, this can be shown as a straightforward corollary of (Zhou and Tao, 2011, Theorem 1). Building upon this, we now demonstrate that, for a given relative improvement tolerance ϵ, Algorithm 1 terminates in a

```
Algorithm 1: Alternating Minimization Heuristic
  Data: $\boldsymbol{D} \in \mathbb{R}^{n\times n}, \lambda, \mu > 0, k_0, k_1 \in \mathbb{Z}^+$, tolerance parameter $\epsilon > 0$.
  Result: $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ feasible and stationary for Problem (31)
  $\boldsymbol{X}_0 \leftarrow \boldsymbol{0}$; $\boldsymbol{Y}_0 \leftarrow \boldsymbol{0}$;
  $f_0 \leftarrow f(\boldsymbol{X}_0, \boldsymbol{Y}_0)$;
  $t \leftarrow 0$;
  do
      $t \leftarrow t + 1$;
      $\boldsymbol{Y}_t \leftarrow \arg\min_{\boldsymbol{Y}\in\mathcal{W}} f(\boldsymbol{X}_{t-1}, \boldsymbol{Y})$;
      $\boldsymbol{X}_t \leftarrow \arg\min_{\boldsymbol{X}\in\mathcal{V}} f(\boldsymbol{X}, \boldsymbol{Y}_t)$;
      $f_t \leftarrow f(\boldsymbol{X}_t, \boldsymbol{Y}_t)$;
  while $f_t > 0$ and $\frac{f_{t-1}-f_t}{f_t} \geq \epsilon$;
  return $\bar{\boldsymbol{X}} = \boldsymbol{X}_t$, $\bar{\boldsymbol{Y}} = \boldsymbol{Y}_t$
```

finite number of iterations. Indeed, Algorithm 1 terminates at iteration t if either $f_t = 0$ or $f_t > \left(\frac{1}{1+\epsilon}\right) f_{t-1}$. For any iteration t, the update rules for $\boldsymbol{X}_{t+1}$ and $\boldsymbol{Y}_{t+1}$ imply that $f_{t+1} = f(\boldsymbol{X}_{t+1}, \boldsymbol{Y}_{t+1}) \leq f(\boldsymbol{X}_t, \boldsymbol{Y}_{t+1}) \leq f(\boldsymbol{X}_t, \boldsymbol{Y}_t) = f_t$. This implies that the sequence $\{f_t\}$ is strictly non-increasing.

Proposition 10 *Algorithm 1 terminates after at most* $\frac{\log \frac{\mu+\lambda+\mu\lambda}{\mu\lambda}}{\log 1+\epsilon}$ *iterations.*

Proof Assume that $\boldsymbol{D} \neq 0$. The case when $\boldsymbol{D} = 0$ is trivial as in this setting, Algorithm 1 terminates immediately because $f_0 = 0$. Suppose Algorithm 1 has yet to terminate after iteration t. This implies that

$$0 < f_t \leq \left(\frac{1}{1+\epsilon}\right) f_{t-1} \leq \left(\frac{1}{1+\epsilon}\right)^t f_0.$$

Recall that $f_0 = f(\boldsymbol{0}, \boldsymbol{0}) = \|\boldsymbol{D}\|_F^2$. Moreover, for all t we must have

$$f_t \geq \min_{\boldsymbol{X}\in\mathcal{V}, \boldsymbol{Y}\in\mathcal{W}} f(\boldsymbol{X}, \boldsymbol{Y}) \geq \min_{\boldsymbol{X}, \boldsymbol{Y}\in\mathbb{R}^{n\times n}} f(\boldsymbol{X}, \boldsymbol{Y}).$$

Simple unconstrained minimization gives $\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} f(\boldsymbol{X}, \boldsymbol{Y}) = \frac{\mu\lambda}{\mu+\lambda+\mu\lambda}\|\boldsymbol{D}\|_F^2$. Combining the above inequalities, we obtain

$$\frac{\mu\lambda}{\mu+\lambda+\mu\lambda}\|\boldsymbol{D}\|_F^2 \leq f_t \leq \left(\frac{1}{1+\epsilon}\right)^t \|\boldsymbol{D}\|_F^2.$$

The result follows by noting that the above inequality is violated if $t > \frac{\log \frac{\mu+\lambda+\mu\lambda}{\mu\lambda}}{\log 1+\epsilon}$. ■

In Section 4, we complement this result by introducing a lower bound that can be used to certify the quality of the solution returned by Algorithm 1. Moreover, in Section 6, we demonstrate numerically that Algorithm 1 produces high-quality solutions to (31).

3.3 Optimality of Algorithm 1 for a Fixed Sparsity Pattern

In this section, we establish the optimality of Algorithm 1 for a fixed sparsity pattern under certain easy-to-verify conditions that often hold in practice. Accordingly, here and throughout this section, we assume we are given a collection of indices $\mathcal{I}_0 \subset \{(i,j) : 1 \leq i,j \leq n\}$, $\|\mathcal{I}_0\| = n^2 - k_1$ that correspond to entries of the sparse matrix $\boldsymbol{Y}$ that must take value 0, and that $\boldsymbol{S}^\star$ is a binary matrix that encodes this sparsity pattern. The collection $\mathcal{I}_0$ specifies a complete feasible sparsity pattern for the matrix $\boldsymbol{Y}$.

Given the sparsity pattern specified by $\mathcal{I}_0$, Problem (1) reduces to

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\cdot\|\boldsymbol{X}\|_F^2 + \mu\cdot\|\boldsymbol{Y}\|_F^2 \\ \text{s.t.} \quad & \text{Rank}(\boldsymbol{X}) \leq k_0,\ Y_{ij} = 0\ \forall (i,j)\in\mathcal{I}_0.\end{aligned} \tag{16}$$

Algorithm 1 can be easily adapted to produce a feasible solution to Problem (16). Indeed, by Proposition 8, an optimal binary matrix $\boldsymbol{Y}^\star$ in (16) is given by

$$\boldsymbol{Y}^* = \boldsymbol{S}^* \circ \left(\frac{\boldsymbol{D}-\boldsymbol{X}}{1+\mu}\right).$$

Moreover, applying Algorithm 1 with a fixed sparsity pattern and fixed low-rank matrix recovers this sparse matrix automatically. Thus, applying Algorithm 1 to Problem (16) is equivalent to solving the following non-convex optimization problem:

$$\begin{aligned}\min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \quad & \left\|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{S}^*\circ\left(\frac{\boldsymbol{D}-\boldsymbol{X}}{1+\mu}\right)\right\|_F^2 + \lambda\cdot\|\boldsymbol{X}\|_F^2 + \mu\cdot\left\|\boldsymbol{S}^*\circ\left(\frac{\boldsymbol{D}-\boldsymbol{X}}{1+\mu}\right)\right\|_F^2 \\ \text{s.t.} \quad & \text{Rank}(\boldsymbol{X}) \leq k_0.\end{aligned} \tag{17}$$

Let us now define some additional notation: let $g(\boldsymbol{X})$ denote the objective value function of (17), $\Omega = \{\boldsymbol{X}\in\mathbb{R}^{n\times n} : \text{Rank}(\boldsymbol{X}) \leq k_0\}$ denote the set of n-by-n matrices with rank at most k_0, $\mathcal{P}_{\mathcal{X}}(\cdot)$ denote the projection operator onto a set $\mathcal{X} \subseteq \mathbb{R}^{n\times n}$, i.e., $\mathcal{P}_{\mathcal{X}}(\boldsymbol{Y}) = \arg\min_{\boldsymbol{X}\in\mathcal{X}} \|\boldsymbol{Y}-\boldsymbol{X}\|_F^2$, and let $\gamma_k(\boldsymbol{X}) = {\sigma_{k+1}(\boldsymbol{X})}/{\sigma_k(\boldsymbol{X})} \leq 1$ denote the ratio between the $(k+1)$th and the kth singular values of $\boldsymbol{X}$.

We have the following result (proof deferred to Appendix A):

Proposition 11 *Given a full sparsity pattern $\mathcal{I}_0 \subset \{(i,j) : 1 \leq i,j \leq n\}$, $\|\mathcal{I}_0\| = n^2 - k_1$, if we constrain the binary matrix $\boldsymbol{S}^*$ in the solution of the sparse matrix subproblem* (14) *to satisfy $S^*_{ij} = 0 \iff (i,j)\in\mathcal{I}_0$, then Algorithm 1 is equivalent to performing Projected Gradient Descent on* (17) *given by $\boldsymbol{X}_{t+1} = \mathcal{P}_\Omega(\boldsymbol{X}_t - \eta\nabla g(\boldsymbol{X}_t))$ with step size $\eta = \frac{1}{2(1+\lambda)}$. By equivalent, we mean that the two algorithms produce the same sequence of feasible low-rank iterates $\boldsymbol{X}_t$ and that we have $f(\boldsymbol{X}_t, \boldsymbol{Y}_t) = g(\boldsymbol{X}_t)$ for all iterations t where $\boldsymbol{Y}_t$ denotes the sparse matrix iterates produced by Algorithm 1.*

We are now ready to establish the main result. We have:

Theorem 12 *Given a full sparsity pattern $\mathcal{I}_0 \subset \{(i,j) : 1 \leq i,j \leq n\}$, $\|\mathcal{I}_0\| = n^2 - k_1$, let $\boldsymbol{S}^\star$ be the binary matrix satisfying $S^\star_{ij} = 0 \iff (i,j)\in\mathcal{I}_0$. Let $\boldsymbol{X}^\star$ denote the optimal low-rank matrix for* (17) *and define $\tilde{\boldsymbol{D}} = \left(\frac{1}{1+\lambda}\left[\boldsymbol{D} - \boldsymbol{S}^*\circ\left(\frac{\boldsymbol{D}-\boldsymbol{X}^\star}{1+\mu}\right)\right]\right)$.*

Assume $\text{Rank}(\boldsymbol{X}^\star) = k_0$ *and suppose that the following two conditions hold:*

1. $\lambda + \frac{2\mu}{1+\mu} - 1 > 0$*;*

2. $\gamma_{k_0}(\tilde{\boldsymbol{D}}) < \frac{1}{1+\lambda}\Big(\lambda + \frac{2\mu}{1+\mu} - 1.\Big)$.

Alternatively, assume $\text{Rank}(\boldsymbol{X}^\star) < k_0$ *and suppose only the first condition listed above holds. In both of these two settings, Algorithm 1 converges linearly to the unique optimal solution of Problem* (16) *(where we constrain the binary matrix* $\boldsymbol{S}^*$ *in the solution of the sparse subproblem* (14) *to satisfy* $S^*_{ij} = 0 \iff (i,j) \in \mathcal{I}_0$*). Specifically, letting* $\{(\boldsymbol{X}_t, \boldsymbol{Y}_t)\}_{t=1}^{\infty}$ *denote the sequence of iterates generated by Algorithm 1 and* $(\boldsymbol{X}^*, \boldsymbol{Y}^*)$ *denote the optimal solution of* (16)*, we have*

$$\frac{f(\boldsymbol{X}_{t+1}, \boldsymbol{Y}_{t+1}) - f(\boldsymbol{X}^*, \boldsymbol{Y}^*)}{f(\boldsymbol{X}_t, \boldsymbol{Y}_t) - f(\boldsymbol{X}^*, \boldsymbol{Y}^*)} \leq \frac{1}{(2\lambda+1)(1+\mu)+\mu} \quad \forall\ t.$$

Note that the first condition on the regularization parameters λ and μ in Theorem 12 is equivalent to requiring that the objective function of (17) has a small condition number. The second condition is a more technical one that requires that the gradient of the objective function at the optimal solution of (17) is never too large.

Remark 13 *Theorem 12 implies that there is a phase transition in Problem* (1)*'s difficulty as the amount of regularization increases. Indeed, when* $\mu = 0$ *and the sparsity pattern is fixed, Problem* (1) *is equivalent to matrix completion (Proposition 5), which is a problem that may admit multiple local minima (Bertsimas et al., 2022), and this may cause Algorithm 1 to converge to a non-global local optimum. On the other hand, our main result implies that, with a sufficiently large regularization term, Problem* (1) *can be solved to certifiable optimality by enumerating the sparsity patterns and running alternating minimization on each fixed sparsity pattern. Thus, regularization partially controls the complexity of* (1)*.*

Proof We establish the result by invoking Theorem 3.3 from Ha et al. (2020). We prove the result for the more involved case where $\text{Rank}(\boldsymbol{X}^\star) = k_0$. The proof for the case where $\text{Rank}(\boldsymbol{X}^\star) < k_0$ follows similar reasoning by combining Proposition 11 with (Ha et al., 2020, Theorem 3.3). We observe that the objective function $g(\boldsymbol{X})$ of (17) is m-strongly convex and L-Lipschitz continuous with $m = 2\lambda + \frac{2\mu}{1+\mu}$ and $L = 2\lambda + 2$. To see this, note that we have

$$g(\boldsymbol{X}) - \frac{m}{2}\|\boldsymbol{X}\|_F^2 = \Big(\lambda - \frac{m}{2}\Big)\|\boldsymbol{X}\|_F^2 + \sum_{(i,j)\in\mathcal{I}_0} (D_{ij} - X_{ij})^2 + \sum_{(i,j)\notin\mathcal{I}_0} (D_{ij} - X_{ij})^2 \cdot \frac{\mu}{1+\mu},$$

which is convex when $m = 2\lambda + \frac{2\mu}{1+\mu}$. Similarly, we have

$$\frac{L}{2}\|\boldsymbol{X}\|_F^2 - g(\boldsymbol{X}) = \Big(\frac{L}{2} - \lambda\Big)\|\boldsymbol{X}\|_F^2 - \sum_{(i,j)\in\mathcal{I}_0} (D_{ij} - X_{ij})^2 - \sum_{(i,j)\notin\mathcal{I}_0} (D_{ij} - X_{ij})^2 \cdot \frac{\mu}{1+\mu},$$

which is convex when $L = 2\lambda + 2$. Suppose that $\boldsymbol{X}^\star$ is a global minimizer of (17). We claim that gradient of $g(\boldsymbol{X})$ at $\boldsymbol{X}^\star$ satisfies:

$$\|\nabla g(\boldsymbol{X}^\star)\|_\sigma = 2(1+\lambda)\gamma_{k_0}(\tilde{\boldsymbol{D}})\sigma_{k_0}(\boldsymbol{X}^\star),$$

where $\|\boldsymbol{X}\|_\sigma = \sigma_1(\boldsymbol{X})$ denotes the spectral norm of $\boldsymbol{X}$. To see this, note that since $\boldsymbol{X}^\star$ is an optimal solution, it must be a fixed point of (37). Thus, we have

$$\begin{aligned}
\|\nabla g(\boldsymbol{X}^\star)\|_\sigma &= 2(1+\lambda)\left\|\boldsymbol{X}^\star - \frac{1}{1+\lambda}\left(\boldsymbol{D} - \boldsymbol{S}^* \circ \left(\frac{\boldsymbol{D}-\boldsymbol{X}^\star}{1+\mu}\right)\right)\right\|_\sigma \\
&= 2(1+\lambda)\|\boldsymbol{X}^\star - \tilde{\boldsymbol{D}}\|_\sigma \\
&= 2(1+\lambda)\sigma_{k_0+1}(\tilde{\boldsymbol{D}}) \\
&= 2(1+\lambda)\gamma_{k_0}(\tilde{\boldsymbol{D}})\sigma_{k_0}(\tilde{\boldsymbol{D}}) \\
&= 2(1+\lambda)\gamma_{k_0}(\tilde{\boldsymbol{D}})\sigma_{k_0}(\boldsymbol{X}^\star),
\end{aligned}$$

where the third and fifth equalities follow from $\boldsymbol{X}^\star$ being a fixed point of (37) and the fourth equality follows from the definition of $\gamma_{k_0}(\tilde{\boldsymbol{D}})$. It is easy to verify that when the first condition of Theorem 12 holds, the condition number $\kappa = \frac{L}{m}$ of $g(\boldsymbol{X})$ satisfies $\kappa < 2$. Moreover, when the second condition of Theorem 12 holds, it can similarly be verified that the gradient of $g(\boldsymbol{X})$ at $\boldsymbol{X}^\star$ satisfies $\|\nabla g(\boldsymbol{X}^\star)\|_\sigma < (2m - L)\sigma_{k_0}(\boldsymbol{X}^\star)$. Invoking the result of Theorem 3.3 from Ha et al. (2020), $\boldsymbol{X}^\star$ is the unique fixed point of Projected Gradient Descent with step size $\eta = \frac{1}{2(1+\lambda)}$. Invoking Proposition 11, this immediately implies that Algorithm 1 converges to $\boldsymbol{X}^\star$.

Finally, it is known that Projected Gradient Descent converges linearly with rate $\frac{\kappa-1}{\kappa+1}$ for strongly convex functions (Recht, 2012). Combining this with Proposition 11, we have

$$\frac{g(\boldsymbol{X}_{t+1}) - g(\boldsymbol{X}^*)}{g(\boldsymbol{X}_t) - g(\boldsymbol{X}^*)} = \frac{f(\boldsymbol{X}_{t+1}, \boldsymbol{Y}_{t+1}) - f(\boldsymbol{X}^*, \boldsymbol{Y}^*)}{f(\boldsymbol{X}_t, \boldsymbol{Y}_t) - f(\boldsymbol{X}^*, \boldsymbol{Y}^*)} \leq \frac{\kappa - 1}{\kappa + 1} = \frac{1}{(2\lambda+1)(1+\mu)+\mu},$$

which holds for all t. This completes the proof. ■

4 A Convex Relaxation

In this section, we reformulate (1) as a mixed-integer, mixed-projection optimization problem. We then employ the (matrix) perspective relaxation (Günlük and Linderoth, 2012; Bertsimas et al., 2022, 2023) to construct a convex relaxation of (1). We illustrate the power of our convex relaxation in Section 4.1, by demonstrating that it reflects the hidden convexity of the low-rank subproblem we derived in the previous section and allows this subproblem to be solved via convex optimization. Further, we compare our convex relaxation to the previously derived relaxation of Lee and Zou (2014) in Section 4.2 and demonstrate that when both relaxations make the same assumptions, our relaxation is at least as powerful, and sometimes strictly more powerful. Finally, in Section 4.3, we interpret (a slightly modified version of, where the sparsity and rank are penalized rather than constrained) our convex relaxation as a convex penalty.

To model the sparsity pattern of the sparse matrix $\boldsymbol{Y}$, we introduce binary variables $\boldsymbol{Z} \in \{0,1\}^{n \times n}$ and require that $Y_{ij} = 0$ if $Z_{i,j} = 0$ by imposing the nonlinear constraint $Y_{i,j} = Y_{i,j} Z_{i,j}$, and also require that $\sum_{ij} Z_{ij} \leq k_1$. To model the column space of $\boldsymbol{X}$, we introduce an orthogonal projection matrix $\boldsymbol{P} \in \mathcal{P}$ and require that $\mathrm{tr}(\boldsymbol{P}) \leq k_0$ and

$\boldsymbol{X} = \boldsymbol{P}\boldsymbol{X}$. Let $\mathcal{Z}_{k_1} = \{\boldsymbol{Z} \in \{0,1\}^{n\times n} : \sum_{ij} Z_{ij} \leq k_1\}$ and $\mathcal{P}_{k_0} = \{\boldsymbol{P} \in \mathcal{S}^n : \boldsymbol{P}^2 = \boldsymbol{P}, \operatorname{tr}(\boldsymbol{P}) \leq k_0\}$. This gives the following reformulation of (1):

$$\begin{aligned} \min_{\boldsymbol{Z}\in\mathcal{Z}_{k_1},\boldsymbol{P}\in\mathcal{P}_{k_0}} \min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\cdot\|\boldsymbol{X}\|_F^2 + \mu\cdot\|\boldsymbol{Y}\|_F^2 \\ \text{s.t.} \quad & \boldsymbol{X} = \boldsymbol{P}\boldsymbol{X}, \boldsymbol{Y} = \boldsymbol{Z}\circ\boldsymbol{Y}. \end{aligned} \tag{18}$$

We now have the following result (proof deferred to Appendix A):

Proposition 14 *Problem* (18) *is a valid reformulation of Problem* (1)*.*

The constraints $\boldsymbol{X} = \boldsymbol{P}\boldsymbol{X}$ and $\boldsymbol{Y} = \boldsymbol{Z}\circ\boldsymbol{Y}$ in (18) are complicating because they are non-convex in the decision variables $(\boldsymbol{Z},\boldsymbol{P},\boldsymbol{X},\boldsymbol{Y})$. Accordingly, to model these constraints in a convex manner, we invoke the (matrix) perspective reformulation (Günlük and Linderoth, 2012; Bertsimas et al., 2022, 2023). Specifically, to model the sparse matrix $\boldsymbol{Y}$, we introduce variables $\boldsymbol{\alpha} \in \mathbb{R}^{n\times n}$ where α_{ij} models Y_{ij}^2, and the constraint $\alpha_{ij}Z_{ij} \geq Y_{ij}^2$, which is second-order cone representable. To model the low-rank matrix $\boldsymbol{X}$, we introduce a variable $\boldsymbol{\Theta} \in \mathbb{R}^{n\times n}$ that models $\boldsymbol{X}^T\boldsymbol{X}$, and the constraint $\left(\begin{smallmatrix} \boldsymbol{\Theta} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{P}\end{smallmatrix}\right) \succeq 0$.

This yields the following reformulation of (18):

$$\begin{aligned} \min_{\boldsymbol{Z}\in\mathcal{Z},\boldsymbol{P}\in\mathcal{P}} \min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\cdot\operatorname{tr}(\boldsymbol{\Theta}) + \mu\cdot\langle\boldsymbol{E},\boldsymbol{\alpha}\rangle \\ \text{s.t.} \quad & \boldsymbol{Y}\circ\boldsymbol{Y} \leq \boldsymbol{\alpha}\circ\boldsymbol{Z},\ \begin{pmatrix} \boldsymbol{\Theta} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{P}\end{pmatrix} \succeq 0, \end{aligned} \tag{19}$$

where $\boldsymbol{E}$ denotes a matrix of all ones of appropriate dimension.

Problem (19) is a reformulation of Problem (1) where the problem's non-convexity is entirely captured by the non-convex sets $\mathcal{Z}_{k_1}$ and $\mathcal{P}_{k_0}$. We now obtain a convex relaxation of (1) by solving (19) with $\boldsymbol{Z} \in \operatorname{conv}(\mathcal{Z}_{k_1})$ and $\boldsymbol{P} \in \operatorname{conv}(\mathcal{P}_{k_0})$ where $\operatorname{conv}(\mathcal{X})$ denotes the convex hull of the set $\mathcal{X}$. It is straightforward to see that $\operatorname{conv}(\mathcal{Z}_{k_1}) = \{\boldsymbol{Z} \in [0,1]^{n\times n} : \sum_{ij} Z_{ij} \leq k_1\}$. Moreover, we have $\operatorname{conv}(\mathcal{P}_{k_0}) = \{\boldsymbol{P} \in \mathcal{S}_+^n : \mathbb{I} - \boldsymbol{P} \succeq 0, \operatorname{tr}(\boldsymbol{P}) \leq k_0\}$(Overton and Womersley, 1992). This gives the following convex optimization problem:

$$\begin{aligned} \min_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{P},\boldsymbol{\Theta},\boldsymbol{\alpha}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\cdot\operatorname{tr}(\boldsymbol{\Theta}) + \mu\cdot\langle\boldsymbol{E},\boldsymbol{\alpha}\rangle \\ \text{s.t.} \quad & \boldsymbol{Y}\circ\boldsymbol{Y} \leq \boldsymbol{\alpha}\circ\boldsymbol{Z},\ \langle\boldsymbol{E},\boldsymbol{Z}\rangle \leq k_1,\ \boldsymbol{0} \leq \boldsymbol{Z} \leq \boldsymbol{E}, \\ & \boldsymbol{P} \succeq 0,\ \mathbb{I} - \boldsymbol{P} \succeq 0,\ \operatorname{tr}(\boldsymbol{P}) \leq k_0,\ \begin{pmatrix} \boldsymbol{\Theta} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{P}\end{pmatrix} \succeq 0. \end{aligned} \tag{20}$$

We now have the following result (proof deferred to Appendix A):

Theorem 15 *Problem* (20) *is a valid convex relaxation of* (1)*.*

Note that Problem (20) only produces a nontrivial lower bound to (1) when the regularization parameters satisfy $\lambda, \mu > 0$. If either $\lambda = 0$ or $\mu = 0$, it can easily be shown that the optimal value of (20) is 0. In Section 6, we employ this convex relaxation to produce bounds for feasible solutions returned by Algorithm 1. Moreover, we show that (20) can be embedded within a branch-and-bound framework.

4.1 Hidden Convexity in the Low Rank Subproblem

In this section, we demonstrate that the low-rank subproblem derived in the previous section exhibits hidden convexity in the sense of Ben-Tal and Den Hertog (2014). This result allows us to establish the strength of our overall convex relaxation in the next section. Formally, we have the following result (proof deferred to Appendix C):

Theorem 16 *Consider the semidefinite optimization problem:*

$$\begin{aligned}\min_{\boldsymbol{P},\boldsymbol{\Theta}\in\mathcal{S}^n_+,\boldsymbol{X}\in\mathcal{S}^n} \quad & \|\bar{\boldsymbol{D}}\|_F^2+(1+\lambda)\cdot\mathrm{tr}(\boldsymbol{\Theta})-2\cdot\langle\boldsymbol{X},\bar{\boldsymbol{D}}\rangle\\ \text{s.t.}\quad & \mathrm{tr}(\boldsymbol{P})\le k_0,\ \mathbb{I}-\boldsymbol{P}\succeq 0,\ \begin{pmatrix}\boldsymbol{\Theta} & \boldsymbol{X}\\ \boldsymbol{X}^T & \boldsymbol{P}\end{pmatrix}\succeq 0.\end{aligned} \tag{21}$$

Solving Problem (12) *is equivalent to solving Problem* (21) *in that both problems have the same optimal objective value and given an optimal solution to either problem, an optimal solution to the other problem can be constructed efficiently.*

4.2 Comparison With the Relaxation of Lee and Zou

To illustrate the power of our convex relaxation, we now present a formal comparison between (20) and the relaxation proposed by Lee and Zou (2014) and demonstrate that our relaxation is at least as powerful and sometimes strictly more powerful. Accordingly, here and throughout this subsection, we assume that the spectral norm of the low-rank matrix $\boldsymbol{X}$ and the infinity norm of the sparse matrix $\boldsymbol{Y}$ are bounded as otherwise the relaxation proposed by Lee and Zou (2014) yields a lower bound of zero. Explicitly, we assume that $\|\boldsymbol{X}\|_\sigma=\max_i\sigma_i(\boldsymbol{X})\le\beta$ and $\|\boldsymbol{Y}\|_\infty=\max_{ij}|Y_{ij}|\le\gamma$ where $\sigma_i(\boldsymbol{X})$ denotes the i^{th} singular value of $\boldsymbol{X}$ for $\beta,\gamma\in\mathbb{R}_+$.

Lee and Zou (2014) obtain their relaxation by noting that under the spectral and infinity norm boundedness assumptions, convex lower bounds of the non-convex rank and ℓ_0 norm functions can be obtained as $\mathrm{Rank}(\boldsymbol{X})\ge\frac{1}{\beta}\|\boldsymbol{X}\|_\star$ and $\|\boldsymbol{Y}\|_0\ge\frac{1}{\gamma}\|\boldsymbol{Y}\|_1$ respectively. Noting that the ℓ_1 norm can be trivially linearized and that the nuclear norm of a matrix $\boldsymbol{X}$ admits a well-known semidefinite characterization given by

$$\min_{\boldsymbol{W}_1,\boldsymbol{W}_2\in\mathcal{S}^n}\quad\frac{1}{2}\mathrm{tr}(\boldsymbol{W}_1+\boldsymbol{W}_2)\ \text{s.t.}\ \begin{pmatrix}\boldsymbol{W}_1 & \boldsymbol{X}\\ \boldsymbol{X}^T & \boldsymbol{W}_2\end{pmatrix}\succeq 0,$$

we can express Lee and Zou (2014)'s relaxation of (1) as follows:

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{V},\boldsymbol{W}_1,\boldsymbol{W}_2\in\mathbb{R}^{n\times n}}\quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\lambda\cdot\|\boldsymbol{X}\|_F^2+\mu\cdot\|\boldsymbol{Y}\|_F^2\\ \text{s.t.}\quad & -\boldsymbol{V}\le\boldsymbol{Y}\le\boldsymbol{V},\ \frac{1}{\gamma}\langle\boldsymbol{E},\boldsymbol{V}\rangle\le k_1,\\ & \frac{1}{2\beta}\mathrm{tr}(\boldsymbol{W}_1)+\frac{1}{2\beta}\mathrm{tr}(\boldsymbol{W}_2)\le k_0,\ \begin{pmatrix}\boldsymbol{W}_1 & \boldsymbol{X}\\ \boldsymbol{X}^T & \boldsymbol{W}_2\end{pmatrix}\succeq 0.\end{aligned} \tag{22}$$

To allow for a fair comparison between our relaxation and that given by (22), we note that under the assumptions $\|\boldsymbol{X}\|_\sigma\le\beta$ and $\|\boldsymbol{Y}\|_\infty\le\gamma$, we can strengthen (20) as follows:

$$\begin{aligned}
\min_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{P}_c,\boldsymbol{P}_r,\boldsymbol{\Theta},\boldsymbol{\alpha}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\lambda\cdot\operatorname{tr}(\boldsymbol{\Theta})+\mu\cdot\langle\boldsymbol{E},\boldsymbol{\alpha}\rangle \\
\text{s.t.} \quad & \boldsymbol{Y}\circ\boldsymbol{Y}\leq\boldsymbol{\alpha}\circ\boldsymbol{Z},\ \langle\boldsymbol{E},\boldsymbol{Z}\rangle\leq k_1,\ \boldsymbol{0}\leq\boldsymbol{Z}\leq\boldsymbol{E},\ -\gamma\boldsymbol{Z}\leq\boldsymbol{Y}\leq\gamma\boldsymbol{Z}, \\
& \boldsymbol{P}_c\succeq 0,\ \mathbb{I}-\boldsymbol{P}_c\succeq 0,\ \operatorname{tr}(\boldsymbol{P}_c)\leq k_0, \\
& \boldsymbol{P}_r\succeq 0,\ \mathbb{I}-\boldsymbol{P}_r\succeq 0,\ \operatorname{tr}(\boldsymbol{P}_r)\leq k_0, \\
& \begin{pmatrix}\boldsymbol{\Theta} & \boldsymbol{X}\\ \boldsymbol{X}^T & \boldsymbol{P}_c\end{pmatrix}\succeq 0,\ \begin{pmatrix}\beta\boldsymbol{P}_r & \boldsymbol{X}\\ \boldsymbol{X}^T & \beta\boldsymbol{P}_c\end{pmatrix}\succeq 0.
\end{aligned} \tag{23}$$

The constraint $-\gamma Z_{ij}\leq Y_{ij}\leq\gamma Z_{ij}$ in (23) emerges immediately from the bound on the infinity norm of the sparse matrix. The last four constraints in (23) follow from the bound on the spectral norm of the low-rank matrix. The variable $\boldsymbol{P}_c$ plays the role of $\boldsymbol{P}$ in (20) and models the k_0 dimensional column space of $\boldsymbol{X}$ as before while the variable $\boldsymbol{P}_r$ models the k_0 dimensional row space of $\boldsymbol{X}$. To see that these four constraints are valid, consider any matrix $\bar{\boldsymbol{X}}$ satisfying $\|\bar{\boldsymbol{X}}\|_\star\leq\beta$ and $\operatorname{Rank}(\bar{\boldsymbol{X}})\leq k_0$, and let $\bar{\boldsymbol{X}}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ be its singular value decomposition. Define $\bar{\boldsymbol{P}}_c=\boldsymbol{U}\boldsymbol{U}^T$ and $\bar{\boldsymbol{P}}_r=\boldsymbol{V}\boldsymbol{V}^T$. We have $\beta^2\bar{\boldsymbol{P}}_r\succeq\bar{\boldsymbol{P}}_r\bar{\boldsymbol{X}}^T\bar{\boldsymbol{X}}=\bar{\boldsymbol{X}}^T\bar{\boldsymbol{P}}_c\bar{\boldsymbol{X}}=\bar{\boldsymbol{X}}^T\bar{\boldsymbol{P}}_c^\dagger\bar{\boldsymbol{X}}$ so we have $\begin{pmatrix}\beta\bar{\boldsymbol{P}}_r & \bar{\boldsymbol{X}}\\ \bar{\boldsymbol{X}}^T & \beta\bar{\boldsymbol{P}}_c\end{pmatrix}\succeq 0$. Feasibility of $\bar{\boldsymbol{P}}_c$ and $\bar{\boldsymbol{P}}_r$ for the remaining constraints follows the same reasoning employed in Theorem 15. Note that if we restrict $\boldsymbol{X}$ to be symmetric, we can take $\boldsymbol{P}_r=\boldsymbol{P}_c$ in (23) as the row space and the column space of $\boldsymbol{X}$ will be the same.

Proposition 17 *For any input data $\boldsymbol{D},k_0,k_1$ and hyperparameters λ,μ, the optimal value of* (23) *is no less than the optimal value of* (22)*.*

Proof To establish the proposition, we show that for any feasible solution to (23) we can construct a feasible solution to (22) that achieves the same or lower objective value.

Fix any input data $\boldsymbol{D}\in\mathbb{R}^{n\times n},k_0,k_1\in\mathbb{N}_+$ and any hyperparameters $\lambda,\mu>0$. Consider an arbitrary feasible solution $\mathcal{S}_1=(\bar{\boldsymbol{X}},\bar{\boldsymbol{Y}},\bar{\boldsymbol{Z}},\bar{\boldsymbol{P}}_c,\bar{\boldsymbol{P}}_r,\bar{\boldsymbol{\Theta}},\bar{\boldsymbol{\alpha}})$ to (23). Let $\bar{\boldsymbol{V}}=\gamma\bar{\boldsymbol{Z}},\bar{\boldsymbol{W}}_1=\beta\bar{\boldsymbol{P}}_c$ and $\bar{\boldsymbol{W}}_2=\beta\bar{\boldsymbol{P}}_r$. We will show that the solution $\mathcal{S}_2=(\bar{\boldsymbol{X}},\bar{\boldsymbol{Y}},\bar{\boldsymbol{V}},\bar{\boldsymbol{W}}_1,\bar{\boldsymbol{W}}_2)$ is feasible to (22) and achieves an objective value that is no larger than the objective value achieves by $\mathcal{S}_2$ in (23). From feasibility of $\mathcal{S}_1$ in (23), we have $-\gamma\bar{Z}_{ij}\leq\bar{Y}_{ij}\leq\gamma\bar{Z}_{ij}\implies-\bar{V}_{ij}\leq\bar{Y}_{ij}\leq\bar{V}_{ij}$ and $\langle\boldsymbol{E},\bar{\boldsymbol{Z}}\rangle\leq k_1\implies\frac{1}{\gamma}\langle\boldsymbol{E},\bar{V}_{ij}\rangle\leq k_1$. Moreover, we have

$$\frac{1}{2\beta}\operatorname{tr}(\bar{\boldsymbol{W}}_1+\bar{\boldsymbol{W}}_2)=\frac{1}{2\beta}\operatorname{tr}(\beta\bar{\boldsymbol{P}}_c+\beta\bar{\boldsymbol{P}}_r)=\frac{1}{2}\operatorname{tr}(\bar{\boldsymbol{P}}_c)+\frac{1}{2}\operatorname{tr}(\bar{\boldsymbol{P}}_c)\leq\frac{k_0}{2}+\frac{k_0}{2}=k_0$$

We conclude that $\mathcal{S}_2$ is feasible to (22) by noting that the last constraint in (22) reduces to the fourth from last constraint in (23) after substituting the definitions of $\bar{\boldsymbol{W}}_1$ and $\bar{\boldsymbol{W}}_2$. We observe that $\mathcal{S}_2$ achieves an objective value in (22) no greater than that achieved by $\mathcal{S}_1$ in (23) by noting that feasibility of $\mathcal{S}_1$ implies that $\operatorname{tr}(\bar{\boldsymbol{\Theta}})\geq\|\bar{\boldsymbol{X}}\|_F^2$ and $\langle\boldsymbol{E},\bar{\boldsymbol{\alpha}}\rangle\geq\|\bar{\boldsymbol{Y}}\|_F^2$. Since this construction holds for every feasible solution to (23), it must hold for any optimal solution, which implies that the optimal value of (22) is no greater than the optimal value of (23). This completes the proof. ■

Proposition 17 establishes that our relaxation is at least as strong as (22), but does not in and of itself demonstrate its utility since it does not preclude the possibility of the optimal value of (23) always coinciding with the optimal value of (22). To address this, Proposition 18 which establishes the existence of problem instances for which the optimal value of (23) is strictly greater than the optimal value of (22). Taken together, Propositions 17 and 18 show that (23) is a (strictly) stronger convex relaxation to (1) than (22).

Proposition 18 *There exists input data $\boldsymbol{D}, k_0, k_1$ and hyperparameters λ, μ such that the optimal value of* (23) *is strictly greater than the optimal value of* (22).

Proof We establish the result constructively. Let $n = 2, \boldsymbol{D} = \mathbb{I}_2, k_0 = 1, k_1 = 0, \lambda = 1$ and $\mu = 1$. With these values, (1) reduces to

$$\min_{\boldsymbol{X} \in \mathbb{R}^{2\times 2}} \quad \|\mathbb{I}_2 - \boldsymbol{X}\|_F^2 + \|\boldsymbol{X}\|_F^2 \text{ s.t. } \mathrm{Rank}(\boldsymbol{X}) \leq 1. \tag{24}$$

It follows immediately from Proposition 6 that the optimal solution to (24) is $\boldsymbol{X}^\star = \begin{pmatrix} 0.5 & 0 \\ 0 & 0 \end{pmatrix}$ and the optimal objective value is $\frac{3}{2}$. Let $\beta = 2$ and $\gamma = 1$. Note that γ can be chosen arbitrarily since the optimal sparse matrix is $\boldsymbol{Y}^\star = \boldsymbol{0}$. Consider solving (23) and (22) for this problem data. From Theorem 16, it follows that the optimal value of (23) coincides with the optimal value of (24). Next, note that if we ignore the rank constraint, it can easily be verified that the unconstrained minimum of (24) is given by $\tilde{\boldsymbol{X}} = \frac{1}{2}\mathbb{I}$ and achieves an objective value of 1. Finally, observe that taking $\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{V}} = \boldsymbol{0}, \tilde{\boldsymbol{W}}_1 = \tilde{\boldsymbol{W}}_2 = \mathbb{I}$, the solution $(\tilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}, \tilde{\boldsymbol{V}}, \tilde{\boldsymbol{W}}_1, \tilde{\boldsymbol{W}}_2)$ is feasible to (22) and achieves an objective value of 1. This completes the proof. ■

4.3 Penalty Interpretation of Relaxation

We now consider instances where the sparsity and rank of the matrices are penalized in the objective rather than constrained and interpret the resulting relaxation as a penalty function in the tradition of Fazel (2002); Recht et al. (2010); Pilanci et al. (2015); Bertsimas et al. (2022) among others. Formally, we have the following result[1], which can be deduced by combining (Pilanci et al., 2015, Corollary 3) with (Bertsimas et al., 2022, Lemma 6):

Proposition 19 *The following two optimization problems are equivalent:*

$$\begin{aligned} \min_{\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{P}, \boldsymbol{\Theta}, \boldsymbol{\alpha} \in \mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D} - \boldsymbol{X} - \boldsymbol{Y}\|_F^2 + \lambda \cdot \mathrm{tr}(\boldsymbol{\Theta}) + \mu \cdot \langle \boldsymbol{E}, \boldsymbol{\alpha} \rangle + \rho_1 \cdot \mathrm{tr}(\boldsymbol{P}) + \rho_2 \cdot \langle \boldsymbol{E}, \boldsymbol{Z} \rangle \\ \text{s.t.} \quad & \boldsymbol{Y} \circ \boldsymbol{Y} \leq \boldsymbol{\alpha} \circ \boldsymbol{Z}, \ \boldsymbol{0} \leq \boldsymbol{Z} \leq \boldsymbol{E}, \ \boldsymbol{P} \succeq 0, \ \mathbb{I} - \boldsymbol{P} \succeq 0, \ \begin{pmatrix} \boldsymbol{\Theta} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{P} \end{pmatrix} \succeq 0. \end{aligned} \tag{25}$$

[1] Note that the statement of our result is slightly different to the statement in Pilanci et al. (2015), because, as noted by Dong et al. (2015), the original result contains some minor typos.

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\sum_{i\in[n]}\min\left(\sqrt{\rho_1\lambda}\sigma_i(\boldsymbol{X}),\rho_1+\lambda\sigma_i(\boldsymbol{X})^2\right)\\ & \qquad +\sum_{i,j\in[n]}\min\left(\sqrt{\mu\rho_2}Y_{i,j},\rho_2+\mu Y_{i,j}^2\right).\end{aligned} \tag{26}$$

The above result demonstrates that our regularized relaxation generalizes the reverse Huber penalty (c.f. Pilanci et al., 2015) to sparse plus low-rank optimization problems. This is quite different from unregularized low-rank problems. Indeed, it follows directly from (Bertsimas et al., 2022, Lemma 7) that under a standard big-M assumption on the ℓ_∞ norm of the sparse matrix and the spectral norm of the low-rank matrix, an unregularized relaxation of the form

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{P}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\rho_1\mathrm{tr}(\boldsymbol{P})+\rho_2\langle\boldsymbol{E},\boldsymbol{Z}\rangle\\ \text{s.t.} \quad & |Y_{ij}|\leq mZ_{ij}\ \forall i,j\in[n],\ \boldsymbol{0}\leq\boldsymbol{Z}\leq\boldsymbol{E},\\ & \boldsymbol{P}\succeq 0,\ \mathbb{I}-\boldsymbol{P}\succeq 0,\ \begin{pmatrix}MP & \boldsymbol{X}\\ \boldsymbol{X}^T & M\boldsymbol{P}\end{pmatrix}\succeq 0\end{aligned} \tag{27}$$

is equivalent to the Lasso and nuclear norm regularized problem

$$\min_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{P}\in\mathbb{R}^{n\times n}} \quad \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\frac{\rho_1}{M}\|\boldsymbol{X}\|_*+\frac{\rho_2}{m}\|\boldsymbol{Y}\|_1. \tag{28}$$

Moreover, as demonstrated by Pilanci et al. (2015); Bertsimas et al. (2020) among others, reverse Huber penalties outperform Lasso penalties for sparse regression problems both theoretically—by requiring fewer data to recover the ground truth under a restricted isometry model Pilanci et al. (2015), and empirically—by providing a significantly lower false discovery rate and comparable accuracy rate after observing the same amount of data Bertsimas et al. (2020). This is because Lasso-type penalties are robust estimators but not sparse estimators (Bertsimas and Copenhaver, 2018), while reverse Huber penalties are sparse estimators that recover the ground truth after observing slightly more data than via an exact approach (c.f. Askari et al., 2022). Since SLR decomposition is a generalization of sparse regression, this partially explains the superior numerical performance of our alternating minimization method compared to GoDec, as reflected in Section 6.

5 Branch and Bound

In this section, we propose a branch-and-bound algorithm in the sense of (Land and Doig, 2010; Little, 1966) that computes certifiably (near) optimal solutions to Problem (1) in a practical amount of time. Specifically, we state explicitly our subproblem strategy in Section 5.1, before stating our overall algorithmic approach in Section 5.2. We also provide a sufficient condition for branch-and-bound to obtain a globally optimal solution in Section 5.2. We remark that branch-and-bound strategies have previously been leveraged for matrix optimization problems (Bertsimas et al., 2017; Lee and Zou, 2014).

Let $h(\boldsymbol{Z},\boldsymbol{P})$ denote the optimal value of the inner minimization problem in (18), i.e.:

$$
\begin{aligned}
h(\boldsymbol{Z},\boldsymbol{P}) := \min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\cdot\|\boldsymbol{X}\|_F^2 + \mu\cdot\|\boldsymbol{Y}\|_F^2 \\
\text{s.t.} \quad & \boldsymbol{X}=\boldsymbol{P}\boldsymbol{X}, \boldsymbol{Y}=\boldsymbol{Z}\circ\boldsymbol{Y}.
\end{aligned}
$$

Proposition 14 established that solving (1) is equivalent to solving $\min_{\boldsymbol{Z}\in\mathcal{Z}_{k_1},\boldsymbol{P}\in\mathcal{P}_{k_0}} h(\boldsymbol{Z},\boldsymbol{P})$. In Section 4, we illustrated how to obtain a lower bound for the optimal value of (1) by solving $\min_{\boldsymbol{Z}\in\mathrm{conv}(\mathcal{Z}_{k_1}),\boldsymbol{P}\in\mathrm{conv}(\mathcal{P}_{k_0})} h(\boldsymbol{Z},\boldsymbol{P})$ which we formulated as a semidefinite program in (20). Suppose we wanted to compute a stronger lower bound for (1). Two natural Lagrangean relaxations to consider are:

$$
\min_{\boldsymbol{Z}\in\mathrm{conv}(\mathcal{Z}_{k_1}),\boldsymbol{P}\in\mathcal{P}_{k_0}} h(\boldsymbol{Z},\boldsymbol{P}), \tag{29}
$$

$$
\min_{\boldsymbol{Z}\in\mathcal{Z}_{k_1},\boldsymbol{P}\in\mathrm{conv}(\mathcal{P}_{k_0})} h(\boldsymbol{Z},\boldsymbol{P}). \tag{30}
$$

It is not immediately clear which of these two problems produces a stronger lower bound for (1). However, as there does not yet exist an efficient method to branch over the set of $n\times n$ orthogonal projection matrices with trace at most k_0 (Bertsimas et al., 2022), we focus on developing a branch-and-bound algorithm that can solve the second problem, (30). Moreover, Theorem 12 provides sufficient conditions under which we can exactly compute $\min_{\boldsymbol{P}\in\mathcal{P}_{k_0}} h(\boldsymbol{Z}_0,\boldsymbol{P})$ for any fixed $\boldsymbol{Z}_0\in\mathcal{Z}_{k_1}$. Thus, provided these conditions hold, we can solve $\min_{\boldsymbol{Z}\in\mathcal{Z}_{k_1},\boldsymbol{P}\in\mathcal{P}_{k_0}} h(\boldsymbol{Z},\boldsymbol{P})$ to optimality by branching over the set $\mathcal{Z}_{k_1}$.

5.1 Subproblems

We construct an enumeration tree that branches on the entries of the binary matrix $\boldsymbol{Z}$, which models the sparsity pattern of the sparse matrix $\boldsymbol{Y}$. Each node in the tree is defined by a (partial or complete) sparsity pattern, described by collections $\mathcal{I}_0,\mathcal{I}_1\subset\{(i,j):1\le i,j\le n\}$ where we have $|\mathcal{I}_0|\le n^2-k_1$, $|\mathcal{I}_1|\le k_1$ and $\mathcal{I}_0\cap\mathcal{I}_1=\emptyset$, and has an accompanying subproblem. We note that Berk and Bertsimas (2019) use a similar notion of partially-determined support when developing a custom branch-and-bound algorithm for the Sparse Principal Component Analysis problem. For indices $(i,j)\in\mathcal{I}_0$, we constrain $Z_{ij}=0$ and for indices $(i,j)\in\mathcal{I}_1$, we constrain $Z_{ij}=1$. We say that $\mathcal{I}_0$ and $\mathcal{I}_1$ define a complete sparsity pattern if either $|\mathcal{I}_0|=n^2-k_1$ or $|\mathcal{I}_1|=k_1$, otherwise we say that $\mathcal{I}_0$ and $\mathcal{I}_1$ define a partial sparsity pattern. A terminal node is a node in the tree that can be described by a complete sparsity pattern.

At any given node in the enumeration defined by collections $\mathcal{I}_0$ and $\mathcal{I}_1$, we consider the subproblem given by:

$$
\begin{aligned}
\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\cdot\|\boldsymbol{X}\|_F^2 + \mu\cdot\|\boldsymbol{Y}\|_F^2 \\
\text{s.t.} \quad & \mathrm{Rank}(\boldsymbol{X})\le k_0, \sum_{(i,j)\notin\mathcal{I}_0\cup\mathcal{I}_1} \mathbb{1}\{Y_{ij}\neq 0\}\le k_1-|\mathcal{I}_1|,\ Y_{ij}=0\ \forall (i,j)\in\mathcal{I}_0.
\end{aligned} \tag{31}
$$

This subproblem can equivalently be expressed as

$$
\min_{\boldsymbol{Z}\in\mathcal{Z}_{k_1},\boldsymbol{P}\in\mathcal{P}_{k_0}} h(\boldsymbol{Z},\boldsymbol{P}) \text{ s.t. } Z_{ij}=0\ \forall (i,j)\in\mathcal{I}_0,\ Z_{ij}=1\ \forall (i,j)\in\mathcal{I}_1. \tag{32}
$$

Note that if $\mathcal{I}_0=\mathcal{I}_1=\emptyset$, (31) and (32) are equivalent to (1).

5.1.1 Subproblem Upper Bound

We adapt Algorithm 1 to compute feasible solutions to (31). Suppose that we fix a sparse matrix $\boldsymbol{Y}^*$ in Problem (31). Then, the problem exactly reduces to (12), which we know how to solve by Proposition 6. Suppose we fix a low-rank matrix $\boldsymbol{X}^*$ in Problem (31). Then, the problem becomes:

$$\begin{aligned} \min_{\boldsymbol{Y} \in \mathbb{R}^{n \times n}} \quad & \|\tilde{\boldsymbol{D}} - \boldsymbol{Y}\|_F^2 + \mu \cdot \|\boldsymbol{Y}\|_F^2 \\ \text{s.t.} \quad & \sum_{(i,j) \notin \mathcal{I}_0 \cup \mathcal{I}_1} \mathbb{1}\{Y_{ij} \neq 0\} \leq k_1 - |\mathcal{I}_1|, \; Y_{ij} = 0 \; \forall (i,j) \in \mathcal{I}_0. \end{aligned} \tag{33}$$

where $\tilde{\boldsymbol{D}} = \boldsymbol{D} - \boldsymbol{X}^*$ and we have omitted the regularization term on the low-rank matrix because it does not depend on $\boldsymbol{Y}$. Similarly to (14), (33) admits a closed-form solution:

Proposition 20 *Let $\boldsymbol{Y}^*$ be a matrix such that*

$$\boldsymbol{Y}^* = \boldsymbol{S}^* \circ \left(\frac{\tilde{\boldsymbol{D}}}{1 + \mu} \right),$$

where $\boldsymbol{S}^*$ is a $n \times n$ binary matrix with k_1 entries $S^\star_{ij} = 1$ such that $S^\star_{ij} = 0 \; \forall \, (i,j) \in \mathcal{I}_0, S^\star_{ij} = 1 \; \forall \, (i,j) \in \mathcal{I}_1$ and $S^\star_{i,j} \geq S^\star_{k,l}$ if $|\tilde{D}_{i,j}| \geq |\tilde{D}_{k,l}| \; \forall \, (i,j), (k,l) \notin \mathcal{I}_0 \cup \mathcal{I}_1$. Then, $\boldsymbol{Y}^\star$ solves Problem (33)*.*

Thus, by replacing the update $\boldsymbol{Y}_t \leftarrow \arg\min_{\boldsymbol{Y} \in \mathcal{W}} f(\boldsymbol{X}_{t-1}, \boldsymbol{Y})$ in Algorithm 1 by the update $\boldsymbol{Y}_t \leftarrow \arg\min_{\boldsymbol{Y} \in \bar{\mathcal{W}}} f(\boldsymbol{X}_{t-1}, \boldsymbol{Y})$ where $\bar{\mathcal{W}} = \{\boldsymbol{Y} \in \mathbb{R}^{n \times n} : \sum_{ij} \mathbb{1}\{Y_{ij} \neq 0\} \leq k_1 - |\mathcal{I}_1|, Y_{ij} = 0 \; \forall \, (i,j) \in \mathcal{I}_0\}$ using the result of Proposition 20, Algorithm 1 can be readily adapted to obtain high quality feasible solutions to (31).

5.1.2 Subproblem Lower Bound

To obtain a lower bound for the objective value of a subproblem given by (32), we solve the relaxation given by

$$\min_{\boldsymbol{Z} \in \text{Conv}(\mathcal{Z}_{k_1}), \; \boldsymbol{P} \in \text{Conv}(\mathcal{P}_{k_0})} \quad h(\boldsymbol{Z}, \boldsymbol{P}) \text{ s.t. } Z_{ij} = 0 \; \forall (i,j) \in \mathcal{I}_0, \; Z_{ij} = 1 \; \forall (i,j) \in \mathcal{I}_1. \tag{34}$$

From Section 4, it follows that (34) can be expressed as the following semidefinite problem:

$$\begin{aligned} \min_{\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{P}, \boldsymbol{\Theta}, \boldsymbol{\alpha} \in \mathbb{R}^{n \times n}} \quad & \|\boldsymbol{D} - \boldsymbol{X} - \boldsymbol{Y}\|_F^2 + \lambda \cdot \text{tr}(\boldsymbol{\Theta}) + \mu \cdot \text{tr}\langle \boldsymbol{E}, \boldsymbol{\alpha} \rangle \\ \text{s.t.} \quad & \boldsymbol{Y} \circ \boldsymbol{Y} \leq \boldsymbol{\alpha} \circ \boldsymbol{Z}, \; \langle \boldsymbol{E}, \boldsymbol{Z} \rangle \leq k_1, \; \boldsymbol{0} \leq \boldsymbol{Z} \leq \boldsymbol{E}, \\ & \boldsymbol{P} \succeq 0, \; \mathbb{I} - \boldsymbol{P} \succeq 0, \; \text{tr}(\boldsymbol{P}) \leq k_0, \; \begin{pmatrix} \boldsymbol{\Theta} & \boldsymbol{X} \\ \boldsymbol{X}^T & \boldsymbol{P} \end{pmatrix} \succeq 0, \\ & Z_{ij} = 0 \; \forall (i,j) \in \mathcal{I}_0, \; Z_{ij} = 1 \; \forall (i,j) \in \mathcal{I}_1. \end{aligned} \tag{35}$$

5.2 Branch and Bound Algorithm

Having specified the subproblem we consider at each node in the tree and how we compute upper bounds (feasible solutions) and lower bounds by leveraging Algorithm 1 and the convex relaxation given by (35), it remains to specify the branching rule and the node selection rule. Algorithm 2 describes our approach. Branching and node selection rules for branch-and-bound form a rich literature (Morrison et al., 2016). In our current implementation of Algorithm 2, we employ the most fractional branching rule. Specifically, for an arbitrary non-terminal node p, let $\boldsymbol{Z}^*$ be the optimal matrix $\boldsymbol{Z}$ of the node's convex relaxation given by (35). We branch on entry $(i^*, j^*) = \arg\min_{(i,j)\notin \mathcal{I}_0\cup\mathcal{I}_1} |Z_{ij} - 0.5|$. When selecting which node to investigate in the tree, we choose a node having a lower bound equal to the current global lower bound. Let $\{(\bar{\boldsymbol{X}}_i, \bar{\boldsymbol{Y}}_i)\}_i$ denote the collection of feasible solutions produced by Algorithm 1 across all nodes that are visited during the execution of Algorithm 2 and let $g(\mathcal{I}_0, \mathcal{I}_1)$ denote the optimal value of Problem (35). The final upper bound returned by Algorithm 2 is given by $\min_i f(\bar{\boldsymbol{X}}_i, \bar{\boldsymbol{Y}}_i)$, the smallest objective value achieved by the feasible solution returned by Algorithm 1 for any subproblem explored during the execution of Algorithm 2. The final lower bound returned by Algorithm 2 is given by $\min_{(\mathcal{I}_0,\mathcal{I}_1)\in\mathcal{N}} g(\mathcal{I}_0, \mathcal{I}_1)$ where $\mathcal{N}$ denotes the set of nodes that have not been discarded upon the termination of Algorithm 2.

Theorem 21 *Algorithm 2 terminates in a finite number of iterations and either returns an ϵ globally optimal solution to* (1) *or returns the solution of* (30)*.*

Proof To see that Algorithm 2 terminates in a finite number of iterations, it suffices to note that Algorithm 2 can never visit a node more than once and that there is a finite number of partial and complete sparsity patterns (each corresponding to a possible tree node) because the set $\mathcal{Z}_{k_1}$ is discrete.

Upon termination, we must have either $\frac{ub-lb}{ub} \leq \epsilon$ or $|\mathcal{N}| = 0$ (or both). Suppose that $\frac{ub-lb}{ub} \leq \epsilon$. Then, by definition, the output solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ is ϵ globally optimal to problem (1) since lb consists of a global lower bound and $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ is feasible to (1). Suppose instead that $|\mathcal{N}| = 0$. Algorithm 2 partitions the space of feasible solutions to (30) and only discards elements of the partition that are guaranteed not to contain the globally optimal solution. If $|\mathcal{N}| = 0$ upon termination, then Algorithm 2 has explored (or pruned) the entire space of feasible solutions so the output value lb is the optimal objective of (30). ■

Theorem 22 *Suppose $\lambda + \frac{2\mu}{1+\mu} - 1 > 0$ and for every full sparsity pattern $\mathcal{I}_0 \subset \{(i,j) : 1 \leq i,j \leq n\}$, $\|\mathcal{I}_0\| = n^2 - k_1$, we have*

$$\gamma_{k_0}(\tilde{\boldsymbol{D}}) < \frac{1}{1+\lambda}\Big(\lambda + \frac{2\mu}{1+\mu} - 1\Big),$$

where $\tilde{\boldsymbol{D}}$ is defined in Theorem 12. Then Algorithm 2 returns an ϵ-optimal solution to (1)*.*

Proof Upon termination of Algorithm 2, we must have either $\frac{ub-lb}{ub} \leq \epsilon$ or $|\mathcal{N}| = 0$ (or both). Suppose that $\frac{ub-lb}{ub} \leq \epsilon$. Then, by definition, the output solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ is ϵ globally optimal to problem (1). Suppose instead that $|\mathcal{N}| = 0$. Then it must be the case that

$ub = lb$. To see this, note that Algorithm 2 partitions the space of feasible solutions to (1) and only discards elements of the partition that are guaranteed not to contain the optimal solution. Moreover, at nodes that correspond to complete sparsity patterns, Theorem 12 guarantees that Algorithm 2 computes the exact solution of (16). Thus, if $|\mathcal{N}| = 0$ upon termination, Algorithm 2 has explored (or pruned) the entire space of feasible solutions so the output value lb is equal to ub and is the optimal objective of (1). ■

```
Algorithm 2: Near-Optimal SLR Decomposition
  Data: $\boldsymbol{D} \in \mathbb{R}^{n\times n}, \lambda, \mu \in \mathbb{R}^+, k_0, k_1 \in \mathbb{Z}^+$ . Tolerance parameter $\epsilon \geq 0$.
  Result: $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ that solves (1) within the optimality tolerance $\epsilon$.
  $p_0 \leftarrow (\mathcal{I}_0, \mathcal{I}_1) = (\emptyset, \emptyset)$;
  $\mathcal{N} \leftarrow \{p_0\}$;
  $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}}) \leftarrow$ solution returned by Algorithm 1;
  $ub \leftarrow f(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ ;
  $lb \leftarrow$ optimal value of (20);
  while $\frac{ub-lb}{ub} > \epsilon$ and $|\mathcal{N}| > 0$ do
      select $(\mathcal{I}_0, \mathcal{I}_1) \in \mathcal{N}$;
      select some element $(i,j) \notin \mathcal{I}_0 \cup \mathcal{I}_1$;
      for $k = 0, 1$ do
          $l \leftarrow (k+1)$ mod 2;
          newnode $\leftarrow \Big(\big(\mathcal{I}_k \cup (i,j)\big), \mathcal{I}_l\Big)$;
          $upper \leftarrow$ upperBound(newnode) with feasible point $(\boldsymbol{X}^*, \boldsymbol{Y}^*)$;
          $lower \leftarrow$ lowerBound(newnode);
          if upper $< ub$ then
              $ub \leftarrow upper$;
              $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}}) \leftarrow (\boldsymbol{X}^*, \boldsymbol{Y}^*)$ ;
              remove any node in $\mathcal{N}$ with $lower \geq ub$;
          end
          if lower $< ub$ then
              add newnode to $\mathcal{N}$
          end
      end
      remove $(\mathcal{I}_0, \mathcal{I}_1)$ from $\mathcal{N}$;
      update $lb$ to be the lowest value of $lower$ over $\mathcal{N}$;
  end
  return $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$, $lb$
```

6 Computational Results

In this section, we evaluate the performance of our alternating minimization heuristic (Algorithm 1) and our branch-and-bound method (Algorithm 2) implemented in Julia 1.5.2 using the `JuMP.jl` package version 0.21.7 and solved using `Mosek` version 9.2 for the semidefinite subproblems (20). We compare our methods against GoDec given by (3), Stable Principal Component Pursuit (S-PCP) given by (2), Fast RPCA (fRPCA) (Yi et al., 2016) , Accelerated Alternating Projections (AccAltProj) (Cai et al., 2019) and Scaled Gradient Descent (ScaledGD) (Tong et al., 2021). All experiments were performed using synthetic data, and run on MIT's Supercloud Cluster (Reuther et al., 2018), which hosts Intel Xeon Platinum 8260 processors. The maximum RAM used across all trials was 192GB. To bridge the gap between theory and practice, we have made our code freely available on `GitHub` at `github.com/NicholasJohnson2020/SparseLowRankSoftware`. For experiments involving AccAltProj, we employ the MATLAB implementation of the method written by Cai et al. (2019) which is available publicly at `https://github.com/caesarcai/AccAltProj_for_RPCA/tree/master`.

We aim to answer the following questions:

1. How does the performance of Algorithm 1 compare to state-of-the-art convex and non-convex methods such as GoDec, S-PCP, AccAltProj, fRPCA and ScaledGD?
2. How does the performance of the accelerated implementation of Algorithm 1 (described in Section 6.4) compare to its exact implementation?
3. How is the performance of Algorithm 1 affected by the dimension of the data matrix $\boldsymbol{D}$, the signal-to-noise level, the rank of the underlying low-rank matrix, and the sparsity of the underlying sparse matrix?
4. How does the performance of Algorithm 2 compare to Algorithm 1?

6.1 Synthetic Data Generation

All experiments were performed using synthetic data. To generate a synthetic data matrix $\boldsymbol{D}$, we first fix a problem dimension n, a desired rank for the low-rank matrix k_0, a desired sparsity for the sparse matrix k_1 and a value $\sigma > 0$ that controls the signal to noise ratio. Next, we generate a random rank k_0 matrix and k_1 sparse matrix. To generate the low-rank matrix $\boldsymbol{L} \in \mathbb{R}^{n\times n}$, we set $\boldsymbol{L} = \boldsymbol{V}\boldsymbol{V}^T$ where $\boldsymbol{V} \in \mathbb{R}^{n\times k_0}$ and $V_{ij} \sim N(0, \frac{\sigma^2}{n})$. To generate the sparse matrix $\boldsymbol{S} \in \mathbb{R}^{n\times n}$, we randomly select a symmetric set of indices $\mathcal{S} \subset \{(i,j) : 1 \leq i, j \leq n\}$ with cardinality $|\mathcal{S}| = k_1$ and let $S_{ij} \sim U(-5,5)$ if $(i,j) \in \mathcal{S}$ and $S_{ij} = 0$ otherwise. Finally, we set $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{S} + \boldsymbol{N}$ where $N_{ij} = N_{ji} \sim N(0,1)$. Note that this data generation process is similar to that employed by Candès et al. (2011).

6.2 Hyperparameter Tuning

We tune the hyperparameters of Algorithm 1, fRPCA, and ScaledGD using 30-fold cross-validation, as proposed by Owen and Perry (2009). For each fold, we randomly sample l columns and rows from the input data matrix $\boldsymbol{D}$ and permute the columns and rows of $\boldsymbol{D}$ to obtain $\tilde{\boldsymbol{D}} = \begin{pmatrix} \boldsymbol{D}_{val} & \boldsymbol{D}_{UR} \\ \boldsymbol{D}_{LL} & \boldsymbol{D}_{train} \end{pmatrix}$ where $\boldsymbol{D}_{val} \in \mathbb{R}^{l\times l}$ is the submatrix corresponding to the randomly sampled rows and columns of $\boldsymbol{D}$, $\boldsymbol{D}_{train} \in \mathbb{R}^{(n-l)\times(n-l)}$, and $\boldsymbol{D}_{UR}, \boldsymbol{D}_{LL}^T \in \mathbb{R}^{l\times(n-l)}$. We set $l = \lfloor n \cdot (1 - \sqrt{0.7}) \rfloor$ so that the training set $\boldsymbol{D}_{train}$ contains at least 70% of the input data. For a given choice of hyperparameters, we perform a SLR decomposition on $\boldsymbol{D}_{train}$. Letting $\hat{\boldsymbol{X}}$ denote the estimated low-rank matrix, we compute the validation score for a single fold as $\frac{\|\boldsymbol{D}_{val}-\boldsymbol{D}_{UR}\hat{\boldsymbol{X}}^{\dagger}\boldsymbol{D}_{LL}\|_F{}^2}{\|\boldsymbol{D}_{val}\|_F{}^2}$. The final validation score for a given set of hyperparameters is the average over 30 folds.

For experiments reported in Section 6.3 and Section 6.4, we tune the hyperparameters (λ, μ) for Algorithm 1 from the collection $\left(\frac{10^{-2}}{\sqrt{n}}, \frac{10^{-1}}{\sqrt{n}}, \frac{10^{0}}{\sqrt{n}}, \frac{10^{1}}{\sqrt{n}}\right) \times \left(\frac{10^{-2}}{\sqrt{n}}, \frac{10^{-1}}{\sqrt{n}}, \frac{10^{0}}{\sqrt{n}}, \frac{10^{1}}{\sqrt{n}}\right)$ and we set the hyperparameter $\gamma = \alpha\frac{k_1}{n^2}$ for fRPCA and ScaledGD where α is tuned (independently for each method) from the collection $(0.01, 0.05, 0.1, 0.5, 1, 2, 4, 6, 8, 10)$. For subsequent experiments in Section 6.1 and beyond, the hyperparameters of Algorithm 1, fRPCA, and ScaledGD are fixed respectively to the best-performing hyperparameters selected via cross-validation in Section 6.3 and Section 6.4. For experiments employing Algorithm 2, we set $\lambda = \mu = \frac{1}{\sqrt{n}}$. We terminate Algorithm 1, GoDec, fRPCA, and ScaledGD when $\frac{f_{t-1}-f_t}{f_t} < 0.001$ where f_t denotes the objective value achieved by the estimate of the low-rank matrix $\boldsymbol{X}$ and the sparse matrix $\boldsymbol{Y}$ at iteration t.

6.3 A Comparison Between the Performance of Algorithm 1, GoDec, S-PCP, AccAltProj, fRPCA and ScaledGD

We present a comparison of Algorithm 1, GoDec, S-PCP, AccAltProj, fRPCA, and ScaledGD as we vary the dimension n of the input data matrix $\boldsymbol{D}$, the rank k_0 of the underlying low-rank matrix $\boldsymbol{L}$ and the sparsity level k_1 of the underlying sparse matrix $\boldsymbol{S}$. We report results for the exact implementations of Algorithm 1 ("Alg 1 Exact") and GoDec where the singular value decomposition is computed exactly at each step. We fix $\sigma = 10$ across all trials. For each value of (n, k_0, k_1), we perform 10 trials.

In Table 2, we report the low-rank matrix reconstruction error (L Error) of each method and the rank and sparsity of the solution returned by S-PCP. Let $\hat{\boldsymbol{L}}$ denote the low-rank matrix returned by one of the five methods. We define the low-rank matrix reconstruction error to be $\frac{\|\hat{\boldsymbol{L}}-\boldsymbol{L}\|_F^2}{\|\boldsymbol{L}\|_F^2}$. Let $\hat{\boldsymbol{L}}$ and $\hat{\boldsymbol{S}}$ denote the low-rank and sparse matrices returned by S-PCP. We define the rank of a solution returned by S-PCP to be $\sum_{i=1}^{n} \mathbb{1}\{\sigma_i(\hat{\boldsymbol{L}}) > 10^{-2}\}$, the number of singular values of $\hat{\boldsymbol{L}}$ that are greater than 10^{-2}. Similarly, we define the sparsity of a solution returned by S-PCP to be $\sum_{ij} \mathbb{1}\{\hat{S}_{ij} > 10^{-2}\}$, the number of entries of $\hat{\boldsymbol{S}}$ that are greater than 10^{-2}.

For every parameter configuration explored, Algorithm 1 outperforms all benchmark methods by producing a solution that has a comparable although slightly lower low-rank matrix reconstruction error and a lower sparse matrix reconstruction error. Moreover, the solutions returned by S-PCP always have an average rank that is far greater than the target rank k_0 and a sparsity level that is far greater than the target sparsity level k_1. Further, the numerical threshold used to compute the rank and sparsity of S-PCP solutions, 10^{-2}, is quite generous. Indeed, using a more common, more restrictive threshold for numerical tolerance would further amplify this discrepancy.

In Table 3, we report the low-rank matrix reconstruction error of each method, the bound gap between the solution returned by Algorithm 1 and the solution of (20), and the time required to solve (20). Letting $\hat{f}$ denote the objective value achieved by the solution returned by Algorithm 1 and letting f^* denote the optimal value of (20), we define the bound gap as $\frac{\hat{f}-f^*}{\hat{f}}$. Thus, not only does Algorithm 1 outperform S-PCP, GoDec, fRPCA and ScaledGD, but, by using the relaxation given by (20), we obtain a certificate of Algorithm 1's instance-wise quality.

6.4 An Accelerated Implementation of Algorithm 1 and its Performance

As noted in Section 3, the main bottleneck in our implementation of Algorithm 1 is the singular value decomposition step that must be performed at each iteration. One commonly proposed technique in the literature to circumvent this difficulty is to employ a randomized SVD (c.f. Halko et al., 2011), which computes a low-rank matrix less accurately but in significantly less time than via an exact SVD. Accordingly, in this section, we investigate the use of a randomized SVD in Algorithm 1 ("Alg 1 Acc") against an exact SVD step ("Alg 1 Exact"). In the accelerated implementation of Algorithm 1, we compute a randomized SVD at every iteration except the final one, where we employ an exact SVD.

We now present a comparison of the exact and accelerated implementations of Algorithm 1 as we vary the dimension n of the input data matrix $\boldsymbol{D}$, the rank k_0 of the underlying low-rank matrix $\boldsymbol{L}$ and the sparsity level k_1 of the underlying sparse matrix $\boldsymbol{S}$. We fix $\sigma = 10$ across all trials. For each value of (n, k_0, k_1), we performed 10 trials.

In Table 4, we report the low-rank matrix reconstruction error and the execution time of the exact and accelerated implementations of Algorithm 1. The execution time reported is the average total runtime of each method which includes the time required to perform cross-validation for the hyperparameters λ and μ. The exact implementation of Algorithm 1 produces a lower reconstruction error than the accelerated implementation across all trials. This behavior is expected given that at each iteration, the exact implementation of Algorithm 1 solves the low-rank subproblem (12) to optimality, whereas the accelerated implementation only computes a high-quality solution to this subproblem (except at the last step). Further, across all trials, the accelerated implementation of Algorithm 1 has a faster average execution time than the exact implementation, which is consistent with the $O(n^2 \log k)$ complexity of the low-rank update in the accelerated implementation compared to the $O(n^2 k)$ complexity in the exact implementation.

6.5 Scalability of Algorithm 1

We present a comparison of Algorithm 1 with GoDec, AccAltProj and ScaledGD as we vary the dimension of the input data matrix $\boldsymbol{D} \in \mathbb{R}^{n \times n}$. We report results for the exact implementations of Algorithm 1 and GoDec. For the first experiment, we fixed $k_0 = 5$, $k_1 = 500$, $\sigma = 10$ across all trials, considered values of $n \in \{200, 250, 300, ..., 1000\}$, and performed 50 trials for each n. For the second experiment, we fixed $k_0 = 2$, $k_1 = 500$, $\sigma = 10$, considered values of $n \in \{2000, 4000, ..., 10000\}$, and performed 5 trials for each n. We fixed the hyperparameters $(\lambda, \mu) = \left(\frac{0.1}{\sqrt{n}}, \frac{10}{\sqrt{n}}\right)$ (resp. $\gamma = \frac{k_1}{2n^2}$) for Algorithm 1 (resp. ScaledGD) for these and all subsequent experiments.

We report the low-rank matrix reconstruction error, the sparse matrix reconstruction error, the sparse support discovery rate, and the execution time for each method in Figures 1–2. We additionally report the low-rank matrix reconstruction error, the sparse matrix reconstruction error and the execution time for Algorithm 1, GoDec and ScaledGD in Table 5 of Appendix D. Let $\hat{\boldsymbol{S}}$ denote the sparse matrix returned by either Algorithm 1 or GoDec. We define the sparse matrix reconstruction error analogously to the low-rank matrix reconstruction error as $\frac{\|\hat{\boldsymbol{S}}-\boldsymbol{S}\|_F^2}{\|\boldsymbol{S}\|_F^2}$. Let $\mathcal{I}(\boldsymbol{S}) = \{(i,j) : S_{ij} \neq 0\}$ denote the support of the sparse matrix $\boldsymbol{S}$, i.e., the set of indices for which the matrix $\boldsymbol{S}$ takes non zero values. Then, we define the sparse support discovery rate to be $\frac{1}{k_1}\sum_{(i,j)\in\mathcal{I}(\boldsymbol{S})}\mathbb{1}(\hat{S}_{ij}\neq 0)$. The execution time reported is the average runtime for a single trial of a given method. We note that if AccAltProj were implemented in Julia, it would very likely exhibit more favorable runtimes than its publicly available MATLAB implementation (Bezanson et al., 2017). The performance metric of greatest interest is the low-rank matrix reconstruction error followed by the sparse matrix reconstruction error.

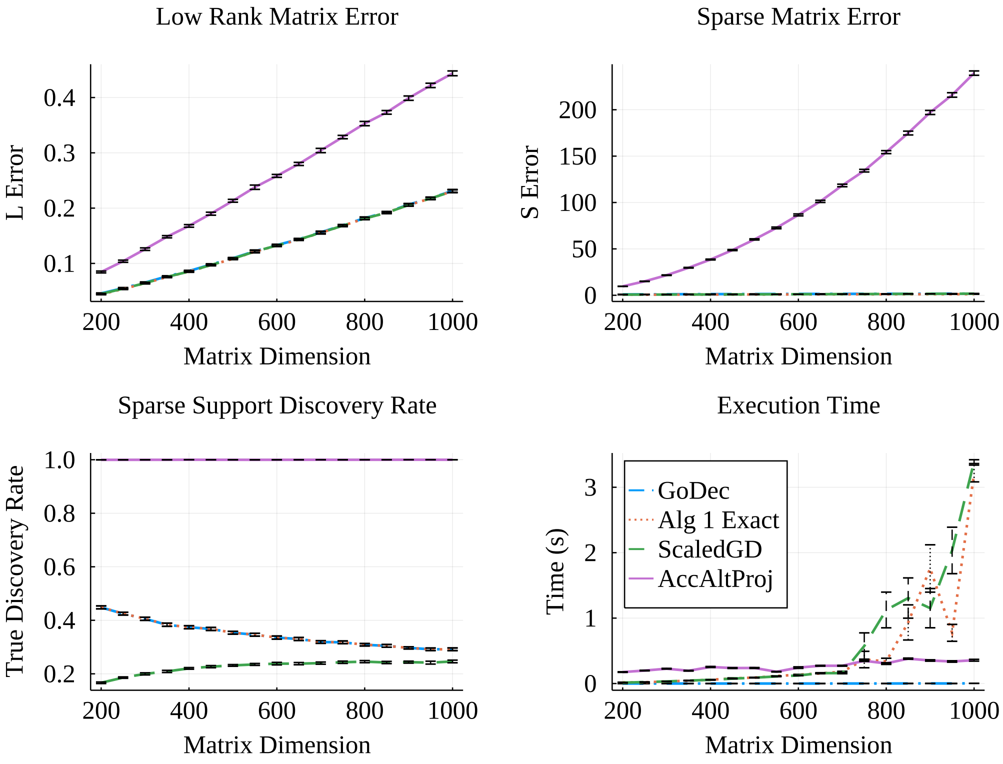

Figure 1: Low-rank matrix reconstruction error (top left), sparse matrix reconstruction error (top right), sparse support discovery rate (bottom left) and execution time (bottom right) versus n with $k_0 = 5$, $k_1 = 500$ and $\sigma = 10$. Averaged over 50 trials for each parameter configuration.

Our main findings from this set of experiments are:

1. Algorithm 1 outperforms GoDec, AccAltProj and ScaledGD across most trials by obtaining lower sparse and low-rank reconstruction errors, while having a comparable execution time.

2. The low-rank matrix reconstruction error scales linearly with matrix dimension for Algorithm 1, AccAltProj, ScaledGD, and GoDec. It can be shown that for our data generation process, $\lim_{n\to\infty}\mathbb{E}[\|\boldsymbol{L}\|_F^2] = C(k_0,\sigma)$ where $C(k_0,\sigma)$ is a constant that

depends only on the rank of $\boldsymbol{L}$ and the signal-to-noise level. This implies that for all methods, $\mathbb{E}[\|\hat{\boldsymbol{L}} - \boldsymbol{L}\|_F^2]$ is $\Theta(n)$.

3. The sparse matrix reconstruction error appears to scale linearly with matrix dimension for Algorithm 1, ScaledGD, and GoDec, while scaling superlinearly with the matrix dimension for AccAltProj. Note that AccAltProj does not allow the cardinality of the sparse matrix to be explicitly constrained. Accordingly, AccAltProj tends to return a sparse matrix that is considerably denser than the desired level. This produces a high sparse support discovery rate (true positive rate) at the expense of a high false discovery rate.. The sparse support discovery rate declines as the matrix dimension increases for GoDec and Algorithm 1in the regime investigated in Figure 1. ScaledGD underperforms GoDec and Algorithm 1 with respect to sparse support discovery rate in low-dimensional settings (Figure 1) but outperforms in high-dimensional settings (Figure 2). This is to be expected as with increasing matrix dimension while k_1 is held fixed, it becomes increasingly difficult to identify the underlying sparsity pattern.

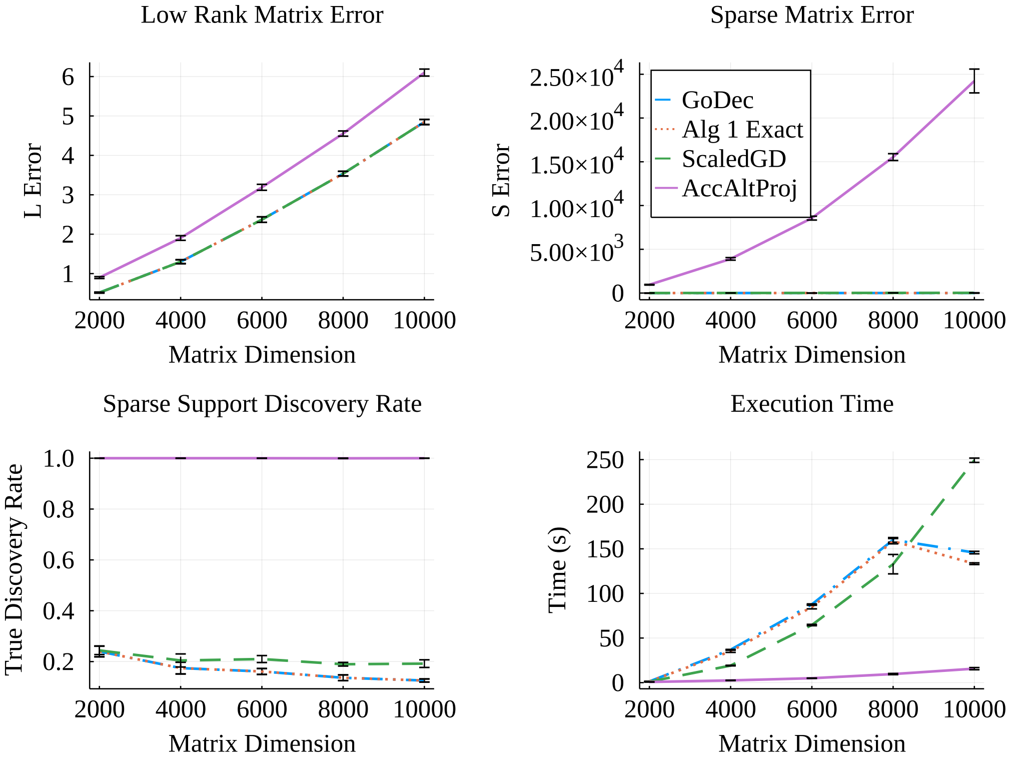

Figure 2: Low-rank matrix reconstruction error (top left), sparse matrix reconstruction error (top right), sparse support discovery rate (bottom left) and execution time (bottom right) versus n with $k_0 = 2$, $k_1 = 500$ and $\sigma = 10$. Averaged over 5 trials for each parameter configuration.

6.6 Sensitivity to Noise

We present a comparison of Algorithm 1 with GoDec, AccAltProj and ScaledGD as we vary the signal to noise level σ of the input data matrix $\boldsymbol{D}$. Large values of σ correspond to a greater signal in the low-rank matrix $\boldsymbol{L}$ compared to the perturbation matrix $\boldsymbol{N}$. We report results for the exact implementations of Algorithm 1 and GoDec that exactly compute the singular value decomposition step. We fixed $n = 100$, $k_0 = 5$, $k_1 = 500$ across all trials and considered values of $\sigma \in \{1, 2, 3, ..., 30\}$. For each value of σ, we performed 50 trials.

We report the low-rank matrix reconstruction error, the sparse matrix reconstruction error, the sparse support discovery rate, and the execution time for each method in Figure 3. Figure 3 includes only results for values of $\sigma \in [10, 30]$ to aid visualization due to significant differences in scale between these results and those for $\sigma \in [1, 10]$. We report the results for the full range $\sigma \in [1, 30]$ in Figure 8 of Appendix D.

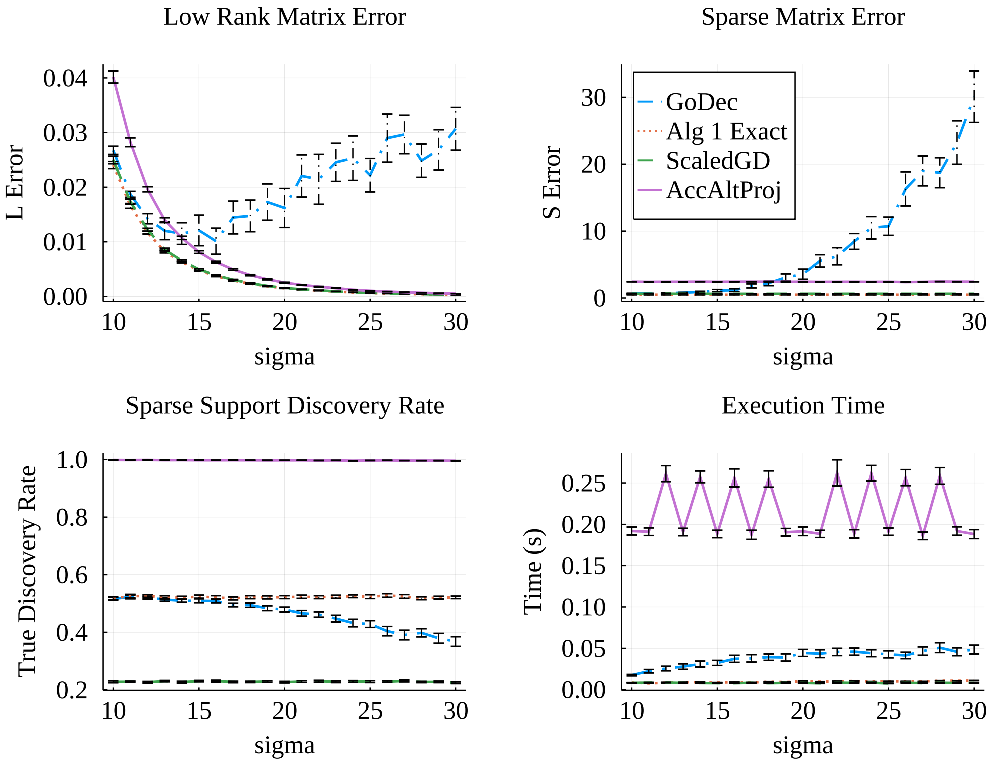

Figure 3: Low-rank matrix reconstruction error (top left), sparse matrix reconstruction error (top right), sparse support discovery rate (bottom left) and execution time (bottom right) versus σ with $n = 100$, $k_0 = 5$ and $k_1 = 500$. Averaged over 50 trials for each parameter configuration.

Our main findings from this set of experiments are:

1. Consistent with previous experiments, Algorithm 1 outperforms GoDec, AccAltProj and ScaledGD across most trials by obtaining a lower sparse and low-rank matrix

reconstruction error while maintaining a comparable execution time and exhibiting superior sparse support discovery rates (compared to GoDec and ScaledGD). The superior performance of Algorithm 1 relative to GoDec becomes more extreme as the signal-to-noise ratio increases.

2. The low-rank reconstruction error of Algorithm 1 decreases as σ increases. This is consistent with the intuition that larger values of σ correspond to easier problem instances, so it should be easier to recover the low-rank matrix. Further, the plotted trend suggests that should σ be further increased, Algorithm 1 would exactly recover $\boldsymbol{L}$. Somewhat surprisingly, the performance of GoDec appears to break down at higher levels of σ. The sparse matrix reconstruction error of Algorithm 1 also declines as σ increases, whereas that of GoDec again breaks down. ScaledGD exhibits a poor sparse recovery rate in these experiments.
3. The sparse support discovery rate of Algorithm 1 slightly declines as σ increases, whereas that of GoDec drops sharply. Though one might expect the sparse support discovery rate to increase with the signal-to-noise level, recall that σ controls the signal-to-noise level of the low-rank matrix compared to the noise matrix and not that of the sparse matrix. Consequently, as σ increases, it should become easier to recover the low-rank matrix but more difficult to recover the sparse matrix.

6.7 Sensitivity to Rank

We present a comparison of Algorithm 1 with GoDec, AccAltProj and ScaledGD as we vary the rank k_0 of the underlying low-rank matrix $\boldsymbol{L}$. We report results for the exact implementations of Algorithm 1 and GoDec that exactly compute the singular value decomposition step. We fixed $n = 100$, $k_1 = 500$, $\sigma = 10$ across all trials and considered values of $k_0 \in \{2, 4, 6, ..., 50\}$. For each value of k_0, we performed 50 trials.

We report the low-rank matrix reconstruction error, the sparse matrix reconstruction error, the sparse support discovery rate, and the runtime for each method in Figure 4.

Our main findings from this set of experiments are:

1. Consistent with previous experiments, Algorithm 1 outperforms GoDec, AccAltProj and ScaledGD across all trials by obtaining a lower low-rank matrix reconstruction error and sparse matrix reconstruction error while having a lesser (in the case of GoDec and AccAltProj) or comparable (in the case of ScaledGD) execution time and exhibiting superior sparse support discovery rates than GoDec and ScaledGD. The superior performance of Algorithm 1 becomes more extreme as the rank increases.
2. The low-rank reconstruction error of Algorithm 1 and that of AccAltProj decrease as k_0 increases whereas the low-rank reconstruction error of GoDec increases with increasing k_0 and that of ScaledGD remains roughly constant.
3. Algorithm 1's and ScaledGD's sparse matrix reconstruction error increases slightly, while GoDec's error increases significantly and AccAltProj's decreases slightly.

6.8 Sensitivity to Sparsity

We present a comparison of Algorithm 1 with GoDec, AccAltProj and ScaledGD as we vary the sparsity level k_1 of the underlying sparse matrix $\boldsymbol{S}$. We report results for the

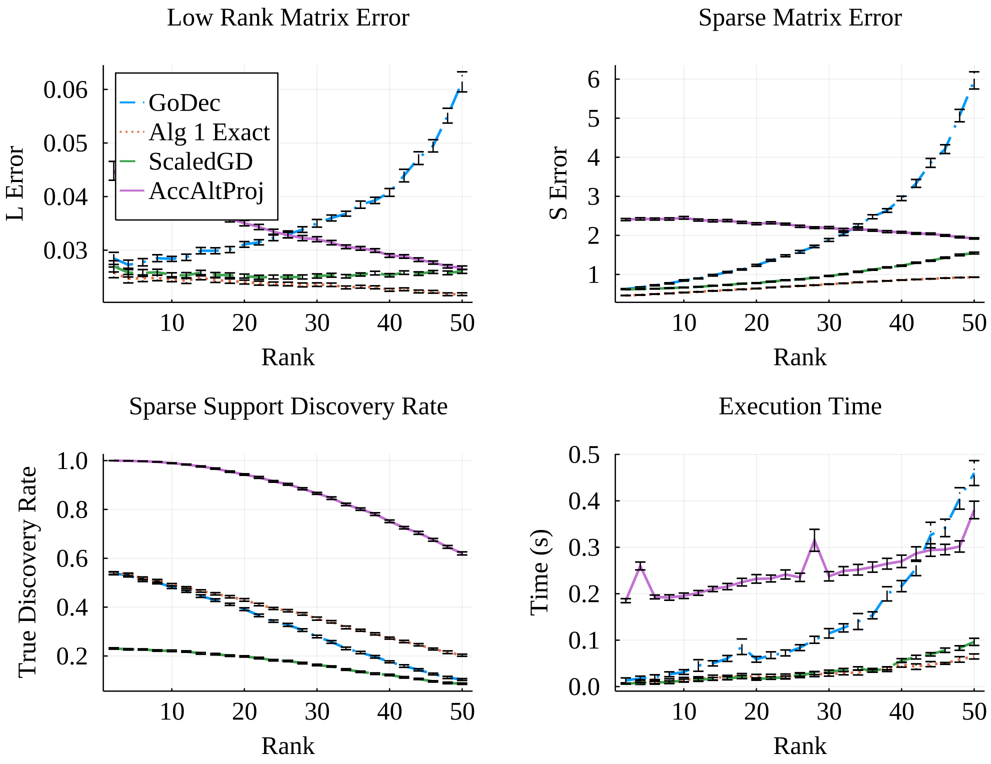

Figure 4: Low-rank matrix reconstruction error (top left), sparse matrix reconstruction error (top right), sparse support discovery rate (bottom left) and execution time (bottom right) versus k_0 with $n = 100$, $k_1 = 500$ and $\sigma = 10$. Averaged over 50 trials for each parameter configuration.

exact implementations of Algorithm 1 and GoDec that exactly compute the singular value decomposition step. We fixed $n = 100$, $k_0 = 5$, $\sigma = 10$ across all trials and considered values of $k_1 \in \{50, 100, 150, ..., 1000\}$. For each value of k_1, we performed 50 trials.

We report the low-rank matrix reconstruction error, the sparse matrix reconstruction error, the sparse support discovery rate, and the runtime for each method in Figure 5.

Our main findings from this set of experiments are:

1. Consistent with previous experiments, Algorithm 1 outperforms GoDec, AccAltProj and ScaledGD across all trials by obtaining a lower low-rank matrix reconstruction error and sparse matrix reconstruction error while having a lesser execution time. Algorithm 1 also exhibits a superior accuracy rate than GoDec and ScaledGD.

2. The low-rank reconstruction error of Algorithm 1, GoDec, AccAltProj and ScaledGD increase as k_1 increases. This is consistent with the intuition that as the sparsity of the underlying spare matrix increases, it becomes more difficult to identify the true low-rank matrix.

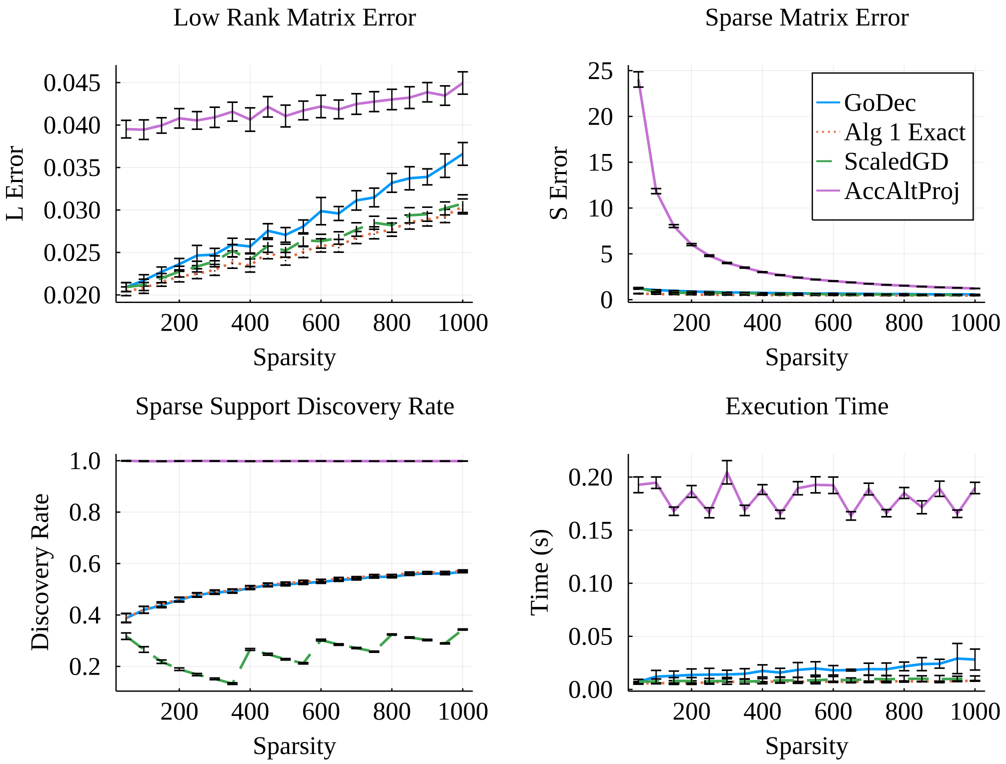

Figure 5: Low-rank matrix reconstruction error (top left), sparse matrix reconstruction error (top right), sparse support discovery rate (bottom left) and execution time (bottom right) versus k_1 with $n = 100$, $k_0 = 5$ and $\sigma = 10$. Averaged over 50 trials for each parameter configuration.

3. The sparse matrix reconstruction error of Algorithm 1, ScaledGD, AccAltProj and GoDec decline as k_1 increases.

6.9 Performance of Algorithm 2

We report the performance of Algorithm 2 on several problem instances. In these experiments, calls that Algorithm 2 make to Algorithm 1 employ the exact implementation of Algorithm 1. We fix $\sigma = 10$ and set $\epsilon = 0.05$, meaning that Algorithm 1 terminates when it has computed a solution to (1) that is certifiably within 5% of the globally optimal solution. We report the optimality gap between the root node upper bound and the root node lower bound, the total number of nodes explored, and the execution time of Algorithm 2 for 14 problem instances in Table 1.

As expected, when the root node optimality gap is less than ϵ, no additional nodes are explored. The total number of possible terminal nodes in any branch-and-bound instance is equal to the number of distinct sparsity patterns, given by $\binom{n^2}{k_1}$. This implies that the

Table 1: Performance of Algorithm 2 for $\epsilon = 0.05$. Reported root node gap is a percentage.

N	k_0	k_1	Root Node Gap	Nodes Explored	Time (s)
10	1	10	5.66	3	41
10	1	15	2.94	1	43
10	2	20	2.37	1	43
15	1	22	7.34	33	58
15	2	33	5.08	3	47
15	3	45	3.26	1	40
20	1	20	5.48	5	44
20	2	40	6.44	123	126
20	3	60	4.33	1	40
20	4	80	4.15	1	41
25	1	31	7.43	205	479
25	2	62	8.30	14709	28977
25	3	93	6.60	1053	2485
25	5	125	7.50	653	1631

total number of possible nodes in any branch-and-bound instance is given by $2 \cdot \binom{n^2}{k_1} - 1$. In the case of the last instance given in Table 1, this quantity is roughly equal to 5.3×10^{134}. Thus, the results of Table 1 indicate that Algorithm 2 is able to prune the vast majority of possible nodes in the branch-and-bound tree. We note that the execution time explodes as the number of nodes explored increases. One of the main limitations of the current implementation of Algorithm 2 is that it requires solving (35), a semidefinite optimization problem, at every node that is explored. This becomes a computational bottleneck as the most efficient interior point solvers for SDPs exhibit poor scaling.

Figure 6 illustrates that Algorithm 2 only occasionally updates the global upper bound and that the vast majority of computational time is spent certifying optimality. This behavior is consistent across all problem instances in which the root node upper bound is not already ϵ optimal. Moreover, Figure 7 illustrates that Algorithm 2 successfully solves instances where $n = 15$ for all values of σ, and is fastest when there is the least amount of noise.

6.10 Summary of Findings From Numerical Experiments

We are now in a position to answer the four questions introduced at the start of this section. Our findings are as follows:

1. Algorithm 1 outperforms GoDec across all trials by obtaining a lower low-rank matrix reconstruction error and sparse matrix reconstruction error while having a lesser execution time and exhibiting superior sparse support discovery rates. The superior performance of Algorithm 1 is most extreme in regimes where the signal-to-noise level σ is high and separately when the rank k_0 of the underlying low-rank matrix is high. Further, Algorithm 1 outperforms S-PCP, AccAltProj and fRPCA across all trials by obtaining lower low-rank and sparse matrix reconstruction errors. With cross-

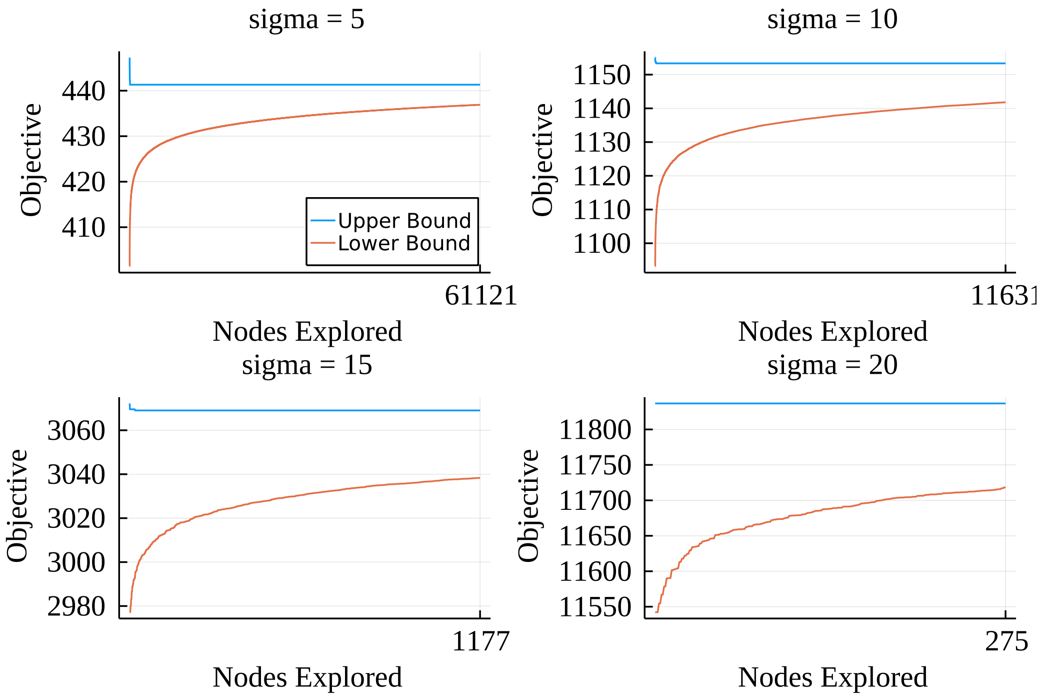

Figure 6: Algorithm 2 upper and lower bound evolution (for a single instance) for $\sigma = 5$ (top left), $\sigma = 10$ (top right), $\sigma = 15$ (bottom left) and $\sigma = 20$ (bottom right) with $n = 15$, $k_0 = 1$, $k_1 = 22$ and $\epsilon = 0.01$.

validation, Algorithm 1 obtains low-rank matrices with a lower rank and a comparable reconstruction error than ScaledGD, and with a rank constraint on both methods it obtains a lower low-rank error that ScaledGD on all but 3 trials. Moreover, it always achieves a lesser sparse matrix reconstruction error than ScaledGD.

2. The exact implementation of Algorithm 1 outperforms the accelerated implementation by achieving a lower reconstruction error across all trials. However, across all trials, the accelerated implementation of Algorithm 1 has a faster average execution time than the exact implementation.

3. (a) Increasing the matrix dimension n results in linear increases in the low-rank matrix reconstruction error and the sparse matrix reconstruction error for Algorithm 1, GoDec and ScaledGD. Increasing the matrix dimension n results in a linear increase in the low-rank matrix reconstruction error and a superlinear increase in the sparse matrix reconstruction error for AccAltProj. The sparse support discovery rate decreases with n for Algorithm 1 and GoDec while the execution time of each method scales superlinearly with n.

   (b) The low-rank matrix and sparse matrix reconstruction errors of Algorithm 1, AccAltProj and ScaledGD decrease with increasing values of σ and that of Algorithm 1 appears to converge towards 0. The sparse support discovery rate of Algorithm 1 decreases slightly with σ while its execution time remains roughly

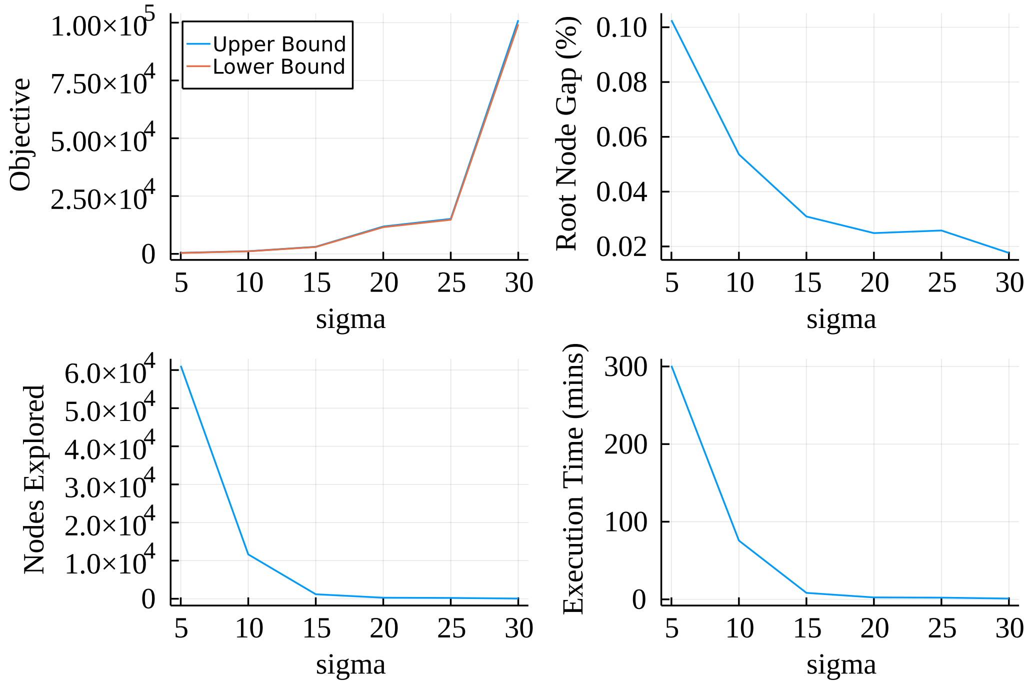

Figure 7: Algorithm 2 root node upper and lower bound (top left), root node optimality gap (top right), number of nodes explored (bottom left) and execution time (bottom right) versus σ with $n = 15$, $k_0 = 1$, $k_1 = 50$ and $\epsilon = 0.01$.

constant. Conversely, the low-rank matrix and sparse matrix reconstruction errors of GoDec explode for large values of σ. GoDec's sparse support discovery rate declines sharply in the high signal-to-noise level regime. ScaledGD generally has poor sparse support discovery.AccAltProj tends to exhibit high sparse support discovery rate because the sparse matrix selected by AccAltProj is in general substantially more dense than the ground truth sparse matrix.

(c) Increasing the rank of the low-rank matrix results in a slight decrease in the low-rank matrix reconstruction error and a slight increase in the sparse matrix reconstruction error for Algorithm 1 and ScaledGD. In contrast, the low-rank matrix and sparse matrix reconstruction errors grow superlinearly for GoDec with increasing rank. The sparse support discovery rate , of Algorithm 1, GoDec and ScaledGD, and the execution time of all methods grow with increasing rank.

(d) Algorithm 1, ScaledGD and GoDec exhibit similar behaviour as a function of sparsity k_1. As the sparsity level of the underlying sparse matrix increases, the low-rank matrix reconstruction error, sparse support discovery rate, and execution time of each of these methods increase while the sparse matrix reconstruction error decreases.

4. Algorithm 2 solves (1) to certifiable optimality for small problem instances (up to $n = 25$) in reasonable wall clock time. The majority of Algorithm 2's execution time is

spent certifying optimality. This implies that the final solution returned by Algorithm 2 is, in general, only marginally better than the solution returned by Algorithm 1.

7 Conclusion

In this paper, we introduced a novel formulation (1) for SLR that exploits discreteness and leverages regularization. We presented Algorithm 1, an alternating minimization heuristic that can compute high-quality feasible solutions to (1) and can scale to $n = 10000$ in minutes. We developed a strong semidefinite relaxation (20) that can certify the quality of the solutions returned by Algorithm 1. Finally, we presented Algorithm 2, a branch-and-bound method that solves (1) to certifiable near-optimality and scales to $n = 25$ in minutes. Moreover, we established sufficient conditions under which Algorithm 2 is optimal. Further work could focus on increasing the scalability of our branch-and-bound method. When executing Algorithm 2, a semidefinite optimization problem must be solved at every node in the branch-and-bound tree to compute a lower bound. This computation is quite costly. A possible extension would be to compute a second-order cone lower bound at each node which would be more scalable at the expense of being less tight. Algorithm 2 can also potentially be further improved by adopting an alternate branching rule.

Acknowledgements

We are very grateful to three anonymous referees for their insightful and helpful comments that improved the paper significantly.

References

Gérard Ben Arous, Alexander S Wein, and Ilias Zadik. Free energy wells and overlap gap property in sparse PCA. In *Conference on Learning Theory*, pages 479–482. PMLR, 2020.

Armin Askari, Alexandre d'Aspremont, and Laurent El Ghaoui. Approximation bounds for sparse programs. *SIAM Journal on Mathematics of Data Science*, 4(2):514–530, 2022.

Sumanta Basu, Xianqi Li, and George Michailidis. Low rank and structured modeling of high-dimensional vector autoregressions. *IEEE Transactions on Signal Processing*, 67(5):1207–1222, 2019.

Aharon Ben-Tal and Dick Den Hertog. Hidden conic quadratic representation of some nonconvex quadratic optimization problems. *Mathematical Programming*, 143(1):1–29, 2014.

Lauren Berk and Dimitris Bertsimas. Certifiably optimal sparse principal component analysis. *Mathematical Programming Computation*, 11(3):381–420, 2019.

Dimitri P Bertsekas. *Nonlinear programming.* Athena Scientific Belmont MA, 3rd edition, 2016.

Dimitris Bertsimas and Martin S Copenhaver. Characterization of the equivalence of robustification and regularization in linear and matrix regression. *European Journal of Operational Research*, 270(3):931–942, 2018.

Dimitris Bertsimas and Dick den Hertog. *Robust and adaptive optimization.* Dynamic Ideas LLC, 2020.

Dimitris Bertsimas, Martin S Copenhaver, and Rahul Mazumder. Certifiably optimal low rank factor analysis. *Journal of Machine Learning Research*, 18(1):907–959, 2017.

Dimitris Bertsimas, Jean Pauphilet, and Bart Van Parys. Sparse regression: Scalable algorithms and empirical performance. *Statistical Science*, 35(4):555–578, 2020.

Dimitris Bertsimas, Ryan Cory-Wright, and Jean Pauphilet. A unified approach to mixed-integer optimization problems with logical constraints. *SIAM Journal on Optimization*, 31(3):2340–2367, 2021.

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

Dimitris Bertsimas, Ryan Cory-Wright, and Jean Pauphilet. A new perspective on low-rank optimization. *Mathematical Programming, articles in advance*, pages 1–46, 2023.

Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

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

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, 1994.

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

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

HanQin Cai, Jian-Feng Cai, and Ke Wei. Accelerated alternating projections for robust principal component analysis. *Journal of Machine Learning Research*, 20(1):685–717, 2019.

Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):1–37, 2011.

Venkat Chandrasekaran, Sujay Sanghavi, Pablo A Parrilo, and Alan S Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.

Junbo Chen, Shouyin Liu, and Min Huang. Low-rank and sparse decomposition model for accelerating dynamic MRI reconstruction. *Journal of Healthcare Engineering*, 2017, 2017.

Yudong Chen and Martin J Wainwright. Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*, 2015.

Yuejie Chi, Yue M Lu, and Yuxin Chen. Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Transactions on Signal Processing*, 67(20):5239–5269, 2019.

Hongbo Dong, Kun Chen, and Jeff Linderoth. Regularization vs. relaxation: A conic optimization perspective of statistical variable selection. *arXiv preprint arXiv:1510.06083*, 2015.

Maryam Fazel. *Matrix rank minimization with applications*. PhD thesis, Stanford University, 2002.

David Gamarnik. The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences*, 118(41):e2108492118, 2021.

Nicolas Gillis and François Glineur. Low-rank matrix approximation with weights or missing data is NP-hard. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1149–1165, 2011.

Fred Glover. Improved linear integer programming formulations of nonlinear integer problems. *Management Science*, 22(4):455–460, 1975.

Quanquan Gu, Zhaoran Wang Wang, and Han Liu. Low-rank and sparse structure pursuit via alternating minimization. In *Artificial Intelligence and Statistics*, pages 600–609. PMLR, 2016.

Oktay Günlük and Jeff Linderoth. Perspective reformulation and applications. In *Mixed Integer Nonlinear Programming*, pages 61–89. Springer, 2012.

Wooseok Ha, Haoyang Liu, and Rina Foygel Barber. An equivalence between critical points for rank constraints versus low-rank factorizations. *SIAM Journal on Optimization*, 30(4):2927–2955, 2020.

Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth Annual ACM Symposium on Theory of Computing*, pages 665–674, 2013.

Hui Ji, Chaoqiang Liu, Zuowei Shen, and Yuhong Xu. Robust video denoising using low rank matrix completion. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1791–1798. IEEE, 2010.

Anastasios Kyrillidis and Volkan Cevher. Matrix ALPS: Accelerated low rank and sparse matrix reconstruction. In *2012 IEEE Statistical Signal Processing Workshop (SSP)*, pages 185–188. IEEE, 2012.

Ailsa H. Land and Alison G. Doig. *An Automatic Method for Solving Discrete Programming Problems*, pages 105–132. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

Jon Lee and Bai Zou. Optimal rank-sparsity decomposition. *Journal of Global Optimization*, 60(2):307–315, 2014.

John DC Little. *Branch and bound methods for combinatorial problems.* PhD thesis, MIT, 1966.

Anirudha Majumdar, Georgina Hall, and Amir Ali Ahmadi. Recent scalability improvements for semidefinite programming with applications in machine learning, control, and robotics. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:331–360, 2020.

David R. Morrison, Sheldon H. Jacobson, Jason J. Sauppe, and Edward C. Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19: 79–102, 2016.

Sahand Negahban and Martin J Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *The Annals of Statistics*, pages 1069–1097, 2011.

Praneeth Netrapalli, UN Niranjan, Sujay Sanghavi, Animashree Anandkumar, and Prateek Jain. Non-convex robust PCA. *arXiv preprint arXiv:1410.7660*, 2014.

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

Art B. Owen and Patrick O. Perry. Bi-cross-validation of the SVD and the nonnegative matrix factorization. *The Annals of Applied Statistics*, 3(2):564 – 594, 2009.

Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

Mert Pilanci, Martin J Wainwright, and Laurent El Ghaoui. Sparse learning via Boolean relaxations. *Mathematical Programming*, 151(1):63–87, 2015.

Benjamin Recht. Projected gradient methods. *Course Notes*, 2012.

Benjamin Recht, Maryam Fazel, and Pablo A Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010.

Albert Reuther, Jeremy Kepner, Chansup Byun, Siddharth Samsi, William Arcand, David Bestor, Bill Bergeron, Vijay Gadepally, Michael Houle, Matthew Hubbell, Michael Jones, Anna Klein, Lauren Milechin, Julia Mullen, Andrew Prout, Antonio Rosa, Charles Yee, and Peter Michaleas. Interactive supercomputing on 40,000 cores for machine learning and data analysis. In *2018 IEEE High Performance extreme Computing Conference (HPEC)*, pages 1–6. IEEE, 2018.

Kees Roos, Marleen Balvert, Bram L Gorissen, and Dick den Hertog. A universal and structured way to derive dual optimization problem formulations. *INFORMS Journal on Optimization*, 2(4):229–255, 2020.

Anders Skajaa and Yinyu Ye. A homogeneous interior-point algorithm for nonsymmetric convex conic optimization. *Mathematical Programming*, 150(2):391–422, 2015.

Andreas M Tillmann and Marc E Pfetsch. The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing. *IEEE Transactions on Information Theory*, 60(2):1248–1259, 2013.

Tian Tong, Cong Ma, and Yuejie Chi. Accelerating ill-conditioned low-rank matrix estimation via scaled gradient descent. *Journal of Machine Learning Research*, 22(1):6639–6701, 2021.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1):37–52, 1987. Proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists.

Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10(7), 2009.

Qi Yan, Jieping Ye, and Xiaotong Shen. Simultaneous pursuit of sparseness and rank structures for matrix decomposition. *Journal of Machine Learning Research*, 16(1):47–75, 2015.

Xinyang Yi, Dohyung Park, Yudong Chen, and Constantine Caramanis. Fast algorithms for robust PCA via gradient descent. *Advances in Neural Information Processing Systems*, 29, 2016.

Xiaoming Yuan and Junfeng Yang. Sparse and low-rank matrix decomposition via alternating direction methods. *Pacific Journal of Optimization*, 9(1):167–180, 2013.

Teng Zhang and Yi Yang. Robust PCA by manifold optimization. *The Journal of Machine Learning Research*, 19(1):3101–3139, 2018.

Tianyi Zhou and Dacheng Tao. Godec: Randomized low-rank & sparse matrix decomposition in noisy case. *Proceedings of the 28th International Conference on Machine Learning*, 35:33–40, 2011.

Zihan Zhou, Xiaodong Li, John Wright, Emmanuel Candès, and Yi Ma. Stable principal component pursuit. In *2010 IEEE International Symposium on Information Theory*, pages 1518–1522, 2010.

Appendix A. SLR Formulation Properties Omitted Proofs

Recall that Proposition 1 states that $f(\boldsymbol{X},\boldsymbol{Y}) = \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2$ is jointly m-strongly convex in $(\boldsymbol{X},\boldsymbol{Y})$. We prove this fact below:

Proof Consider any two points $(\boldsymbol{X}_1,\boldsymbol{Y}_1),(\boldsymbol{X}_2,\boldsymbol{Y}_2) \in \mathbb{R}^{n\times n}\times\mathbb{R}^{n\times n}$ and any $t\in[0,1]$. We have

$$\begin{aligned}
g(t\boldsymbol{X}_1 + (1-t)\boldsymbol{X}_2,\, t\boldsymbol{Y}_1 + (1-t)\boldsymbol{Y}_2) = \|\boldsymbol{D} - t\boldsymbol{X}_1 + (1-t)\boldsymbol{X}_2 - t\boldsymbol{Y}_1 + (1-t)\boldsymbol{Y}_2\|_F^2 + \\
(\lambda - \min(\lambda,\mu))\|t\boldsymbol{X}_1 + (1-t)\boldsymbol{X}_2\|_F^2 + (\mu - \min(\lambda,\mu))\|t\boldsymbol{Y}_1 + (1-t)\boldsymbol{Y}_2\|_F^2 \\
(\lambda - \min(\lambda,\mu))\|t\boldsymbol{X}_1 + (1-t)\boldsymbol{X}_2\|_F^2 + (\mu - \min(\lambda,\mu))\|t\boldsymbol{Y}_1 + (1-t)\boldsymbol{Y}_2\|_F^2 \\
\le t\cdot\Big[\|\boldsymbol{D} - \boldsymbol{X}_1 - \boldsymbol{Y}_1\|_F^2 + (\lambda - \min(\lambda,\mu))\|\boldsymbol{X}_1\|_F^2 + (\mu - \min(\lambda,\mu))\|\boldsymbol{Y}_1\|_F^2\Big] + \\
(1-t)\Big[\|\boldsymbol{D} - \boldsymbol{X}_2 - \boldsymbol{Y}_2\|_F^2 + (\lambda - \min(\lambda,\mu))\|\boldsymbol{X}_2\|_F^2 + (\mu - \min(\lambda,\mu))\|\boldsymbol{Y}_2\|_F^2\Big] \\
= t\cdot g(\boldsymbol{X}_1,\, \boldsymbol{Y}_1) + (1-t)\cdot g(\boldsymbol{X}_2,\, \boldsymbol{Y}_2).
\end{aligned}$$ ■

Recall that Proposition 2 states that $f(\boldsymbol{X},\boldsymbol{Y}) = \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \lambda\|\boldsymbol{X}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2$ is L-Lipschitz continuous in $(\boldsymbol{X},\boldsymbol{Y})$. We prove this fact below:

Proof To establish Proposition 2, it suffices to show that $h(\boldsymbol{X},\boldsymbol{Y}) = \frac{L}{2}(\|\boldsymbol{X}\|_F^2 + \|\boldsymbol{Y}\|_F^2) - f(\boldsymbol{X},\boldsymbol{Y})$ is convex for $L = 2\cdot\max(\lambda,\mu)+6$. We have

$$\begin{aligned}
h(\boldsymbol{X},\,\boldsymbol{Y}) &= \frac{L}{2}(\|\boldsymbol{X}\|_F^2 + \|\boldsymbol{Y}\|_F^2) - \lambda\|\boldsymbol{X}\|_F^2 - \mu\|\boldsymbol{Y}\|_F^2 - \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2 \\
&= \left(\frac{L}{2} - \lambda - 1\right)\|\boldsymbol{X}\|_F^2 + \left(\frac{L}{2} - \mu - 1\right)\|\boldsymbol{Y}\|_F^2 + 2\Big(\langle\boldsymbol{D},\boldsymbol{X}\rangle + \langle\boldsymbol{D},\boldsymbol{Y}\rangle - \langle\boldsymbol{X},\boldsymbol{Y}\rangle\Big) - \|\boldsymbol{D}\|_F^2 \\
&= \left(\frac{L}{2} - \lambda - 2\right)\|\boldsymbol{X}\|_F^2 + \left(\frac{L}{2} - \mu - 2\right)\|\boldsymbol{Y}\|_F^2 + \|\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \\
&\qquad 2\Big(\langle\boldsymbol{D},\boldsymbol{X}\rangle + \langle\boldsymbol{D},\boldsymbol{Y}\rangle + \|\boldsymbol{D}\|_F^2\Big) - 3\|\boldsymbol{D}\|_F^2 \\
&= \left(\frac{L}{2} - \lambda - 3\right)\|\boldsymbol{X}\|_F^2 + \left(\frac{L}{2} - \mu - 3\right)\|\boldsymbol{Y}\|_F^2 + \|\boldsymbol{X}-\boldsymbol{Y}\|_F^2 + \\
&\qquad \|\boldsymbol{X}-\boldsymbol{D}\|_F^2 + \|\boldsymbol{Y}-\boldsymbol{D}\|_F^2 - 3\|\boldsymbol{D}\|_F^2.
\end{aligned}$$

Taking $L = 2\cdot\max(\lambda,\mu)+6$, we have $\frac{L}{2} - \lambda - 3 = \max(\lambda,\mu) - \lambda \ge 0$ and $\frac{L}{2} - \mu - 3 = \max(\lambda,\mu) - \mu \ge 0$. Thus, we have written $h(\boldsymbol{X},\boldsymbol{Y})$ as the sum of convex quadratic functions of $(\boldsymbol{X},\boldsymbol{Y})$ which immediately implies $h(\boldsymbol{X},\boldsymbol{Y})$'s joint convexity. ■

Recall that Proposition 3 states if we let $\mathcal{U}_\lambda(\boldsymbol{X}) = \{\boldsymbol{\Delta}\in\mathbb{R}^{n\times n} : \|\boldsymbol{\Delta}\|_F \le \lambda\|\boldsymbol{X}\|_F\}$ for $\boldsymbol{X}\in\mathbb{R}^{n\times n}, \lambda > 0$, then (8) is equivalent to (9). We prove this result below:

Proof Consider the inner maximization problem in (8) and first note that by applying the triangle inequality for the Frobenius norm, we have

$$\max_{\substack{\boldsymbol{\Delta}_1\in\mathcal{U}_\lambda(\boldsymbol{X}) \\ \boldsymbol{\Delta}_2\in\mathcal{U}_\mu(\boldsymbol{Y})}} \|\boldsymbol{D} + \boldsymbol{\Delta_1} + \boldsymbol{\Delta_2} - \boldsymbol{X} - \boldsymbol{Y}\|_F \le \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F + \lambda\|\boldsymbol{X}\|_F + \mu\|\boldsymbol{Y}\|_F.$$

Next, note that by taking

$$\boldsymbol{\Delta_1^*} = \frac{\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}}{\|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F}\cdot\lambda\|\boldsymbol{X}\|_F, \text{ and}$$
$$\boldsymbol{\Delta_2^*} = \frac{\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}}{\|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F}\cdot\mu\|\boldsymbol{Y}\|_F,$$

the upper bound on the maximization problem is attained:

$$\begin{aligned}\|\boldsymbol{D}+\boldsymbol{\Delta_1^*}+\boldsymbol{\Delta_2^*}-\boldsymbol{X}-\boldsymbol{Y}\|_F &= \left\|(\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y})\cdot\left(1+\frac{\lambda\|\boldsymbol{X}\|_F+\mu\|\boldsymbol{Y}\|_F}{\|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F}\right)\right\|_F\\ &= \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F+\lambda\|\boldsymbol{X}\|_F+\mu\|\boldsymbol{Y}\|_F.\end{aligned}$$

The proof is concluded by noting that we have $\boldsymbol{\Delta_1^*}\in\mathcal{U}_\lambda(\boldsymbol{X})$ and $\boldsymbol{\Delta_2^*}\in\mathcal{U}_\mu(\boldsymbol{Y})$. ■

We now provide a formal proof of Proposition 4:

Proof Let us rewrite Problem (1) as

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y},\boldsymbol{U},\boldsymbol{V}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\lambda\|\boldsymbol{U}\|_F^2+\mu\|\boldsymbol{V}\|_F^2\\ \text{s.t.} \quad & \text{Rank}(\boldsymbol{X})\le k_0,\ \|\boldsymbol{Y}\|_0\le k_1, \boldsymbol{X}=\boldsymbol{U}, \boldsymbol{Y}=\boldsymbol{V},\end{aligned}$$

and associate matrices of dual multipliers $\boldsymbol{\alpha},\boldsymbol{\beta}$ with the linear constraints $\boldsymbol{X}=\boldsymbol{U}$ and $\boldsymbol{Y}=\boldsymbol{V}$ respectively. Then, this problem can be rewritten as

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y}}\min_{\boldsymbol{U},\boldsymbol{V}}\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & \|\boldsymbol{D}-\boldsymbol{X}-\boldsymbol{Y}\|_F^2+\lambda\|\boldsymbol{U}\|_F^2+\mu\|\boldsymbol{V}\|_F^2+\langle\boldsymbol{\alpha},\boldsymbol{X}-\boldsymbol{U}\rangle+\langle\boldsymbol{\beta},\boldsymbol{Y}-\boldsymbol{V}\rangle\\ \text{s.t.} \quad & \text{Rank}(\boldsymbol{X})\le k_0,\ \|\boldsymbol{Y}\|_0\le k_1.\end{aligned}$$

Therefore, let us fix $\boldsymbol{X},\boldsymbol{Y}$ and use a standard minimax theorem (see, e.g., Bertsekas, 2016, Chap. 6) to exchange the order of minimizing $\boldsymbol{U},\boldsymbol{V}$ and maximizing $\boldsymbol{\alpha},\boldsymbol{\beta}$. This gives the following subproblem in $\boldsymbol{U},\boldsymbol{V}$ for a fixed $\boldsymbol{\alpha},\boldsymbol{\beta}$:

$$\min_{\boldsymbol{U},\boldsymbol{V}} \quad \lambda\|\boldsymbol{U}\|_F^2+\mu\|\boldsymbol{V}\|_F^2+\langle\boldsymbol{\alpha},-\boldsymbol{U}\rangle+\langle\boldsymbol{\beta},-\boldsymbol{V}\rangle.$$

By differentiating and setting the gradient to zero, it is not too hard to see that this subproblem takes the value $\frac{-1}{4\lambda}\|\boldsymbol{\alpha}\|_F^2-\frac{1}{4\mu}\|\boldsymbol{\beta}\|_F^2$. This implies the result. ■

Recall that Proposition 5 establishes that (1) reduces to regularized matrix completion with $\Omega=\{(i,j):Z_{ij}=0\}$ where $\boldsymbol{Z}$ denotes a valid sparsity pattern and we take $\mu=0$. We prove this result below:

Proof Given a valid sparsity pattern $\boldsymbol{Z}$ and letting $\Omega=\{(i,j):Z_{ij}=0\}$, Problem (1) can be expressed as

$$\begin{aligned}\min_{\boldsymbol{X},\boldsymbol{Y}\in\mathbb{R}^{n\times n}} \quad & \lambda\|\boldsymbol{X}\|_F^2+\sum_{(i,j)\in\Omega}(D_{ij}-X_{ij}-Y_{ij})^2+\mu Y_{ij}^2+\sum_{(i,j)\notin\Omega}(D_{ij}-X_{ij}-Y_{ij})^2+\mu Y_{ij}^2\\ \text{s.t.} \quad & \text{Rank}(\boldsymbol{X})\le k_0,\ Y_{ij}=0\ \forall\ (i,j)\in\Omega.\end{aligned}$$

Simple unconstrained minimization gives $Y_{ij} = \frac{D_{ij}-X_{ij}}{1+\mu}$ for $(i,j) \notin \Omega$. Using this relationship, Problem (1) can be further simplified to

$$\begin{aligned} \min_{\boldsymbol{X}\in\mathbb{R}^{n\times n}} \quad & \lambda \cdot \|\boldsymbol{X}\|_F^2 + \sum_{(i,j)\in\Omega} (D_{ij} - X_{ij})^2 + \frac{\mu}{1+\mu} \cdot \sum_{(i,j)\notin\Omega} (D_{ij} - X_{ij})^2 \cdot \\ \text{s.t.} \quad & \text{Rank}(\boldsymbol{X}) \le k_0. \end{aligned} \tag{36}$$

The result then follows by observing that the last term in the objective function of (36) disappears when $\mu = 0$. Moreover, if we take $\lambda = 0$, then (36) exactly becomes (11). ■

We now provide a formal proof of Proposition 11:
Proof First, note that given the full sparsity pattern, the iterates $(\boldsymbol{X}_t^{AM}, \boldsymbol{Y}_t^{AM})$ produced by Algorithm 1 satisfy $\boldsymbol{Y}_{t+1}^{AM} = \boldsymbol{S}^* \circ \left(\frac{\boldsymbol{D}-\boldsymbol{X}_t^{AM}}{1+\mu}\right)$ and $\boldsymbol{X}_{t+1}^{AM} = \frac{1}{1+\lambda}\mathcal{P}_\Omega(\boldsymbol{D} - \boldsymbol{Y}_{t+1}^{AM})$. This implies that

$$\boldsymbol{X}_{t+1}^{AM} = \mathcal{P}_\Omega\left(\frac{1}{1+\lambda}\left[\boldsymbol{D} - \boldsymbol{S}^* \circ \left(\frac{\boldsymbol{D} - \boldsymbol{X}_t^{AM}}{1+\mu}\right)\right]\right). \tag{37}$$

Next, note that the gradient of $g(\boldsymbol{X}_t)$ is given by

$$\nabla g(\boldsymbol{X}_t) = 2\left((1+\lambda)\boldsymbol{X}_t - \boldsymbol{D} + \boldsymbol{S}^* \circ \left(\frac{\boldsymbol{D} - \boldsymbol{X}_t}{1+\mu}\right)\right).$$

The result follows by noting that the Projected Gradient Descent update $\boldsymbol{X}_{t+1} = \mathcal{P}_\Omega(\boldsymbol{X}_t - \eta\nabla g(\boldsymbol{X}_t))$ is the same as the update given by (37) when $\eta = \frac{1}{2(1+\lambda)}$. ■

We now provide a formal proof of Proposition 14:
Proof We show that given a feasible solution to (18), we can construct a feasible solution to (1) that achieves the same objective value and vice versa.

Consider an arbitrary feasible solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}}, \bar{\boldsymbol{Z}}, \bar{\boldsymbol{P}})$ to (18). Since $\bar{\boldsymbol{Z}} \in \mathcal{Z}_{k_1}$ and $\bar{\boldsymbol{Y}} = \bar{\boldsymbol{Z}} \circ \bar{\boldsymbol{Y}}$, we have $\|\bar{\boldsymbol{Y}}\|_0 \le k_1$. Moreover, since $\bar{\boldsymbol{P}} \in \mathcal{P}_{k_0}$ and $\bar{\boldsymbol{X}} = \bar{\boldsymbol{P}}\bar{\boldsymbol{X}}$, we have $\text{Rank}(\bar{\boldsymbol{X}}) \le k_0$. Thus, $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ is feasible to (1). Since both (18) and (1) have the same objective function, $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ achieves the same objective in (1) as $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}}, \bar{\boldsymbol{Z}}, \bar{\boldsymbol{P}})$ does in (18).

Consider an arbitrary feasible solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ to (1). Let $\bar{\boldsymbol{Z}} \in \{0,1\}^{n\times n}$ be the binary matrix such that $\bar{Z}_{ij} = 1$ if $\bar{Y}_{ij} \ne 0$ and $\bar{Z}_{ij} = 0$ otherwise. Further, let $\bar{\boldsymbol{P}} = \boldsymbol{U}\boldsymbol{U}^T$ where $\bar{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is a singular value decomposition of $\bar{\boldsymbol{X}}$. By construction, we have $\bar{\boldsymbol{Z}} \in \mathcal{Z}_{k_1}$ and $\bar{\boldsymbol{P}} \in \mathcal{P}_{k_0}$ since $\|\bar{\boldsymbol{Y}}\|_0 \le k_1$ and $\text{Rank}(\bar{\boldsymbol{X}}) \le k_0$. Thus, $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}}, \bar{\boldsymbol{Z}}, \bar{\boldsymbol{P}})$ is feasible to (18) and achieves the same objective as $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ does in (1). This completes the proof. ■

We now provide a formal proof of Theorem 15:
Proof Clearly Problem (20) is a convex optimization problem. We will show that given any feasible solution to Problem (1), we can construct a feasible solution to (20) that achieves the same objective value.

Consider an arbitrary feasible solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ to (1). Let $\bar{\boldsymbol{Z}} \in \{0,1\}^{n\times n}$ be the binary matrix such that $\bar{Z}_{ij} = 1$ if $\bar{Y}_{ij} \ne 0$ and $\bar{Z}_{ij} = 0$ otherwise and let $\bar{\boldsymbol{\alpha}} \in \mathbb{R}^{n\times n}$ be the matrix such that $\bar{\alpha}_{ij} = \bar{Y}_{ij}^2$. Further, let $\bar{\boldsymbol{P}} = \boldsymbol{U}\boldsymbol{U}^T$ where $\bar{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is a singular value decomposition of $\bar{\boldsymbol{X}}$ and let $\bar{\boldsymbol{\Theta}} = \bar{\boldsymbol{X}}^T\bar{\boldsymbol{X}}$. By construction, we have $\bar{\boldsymbol{Z}} \in \mathcal{Z}_{k_1}$ and $\bar{\boldsymbol{P}} \in \mathcal{P}_{k_0}$ since $\|\bar{\boldsymbol{Y}}\|_0 \leq k_1$ and $\text{Rank}(\bar{\boldsymbol{X}}) \leq k_0$ which implies that $\text{tr}(\boldsymbol{E}\bar{\boldsymbol{Z}}) \leq k_1, 0 \leq \bar{\boldsymbol{Z}} \leq 1, \bar{\boldsymbol{P}} \succeq 0, \mathbb{I} - \bar{\boldsymbol{P}} \succeq 0$ and $\text{tr}(\bar{\boldsymbol{P}}) \leq k_0$. It is straightforward to see that we have $\bar{Y}_{ij}^2 \leq \bar{\alpha}_{ij}\bar{Z}_{ij} \;\forall\, (i,j)$. Finally, we have $\bar{\boldsymbol{\Theta}} = \bar{\boldsymbol{X}}^T\bar{\boldsymbol{X}} = \bar{\boldsymbol{X}}^T\bar{\boldsymbol{P}}\bar{\boldsymbol{X}} = \bar{\boldsymbol{X}}^T\bar{\boldsymbol{P}}^\dagger\bar{\boldsymbol{X}}$ so we have $\begin{pmatrix} \bar{\boldsymbol{\Theta}} & \bar{\boldsymbol{X}} \\ \bar{\boldsymbol{X}}^T & \bar{\boldsymbol{P}} \end{pmatrix} \succeq 0$. Thus, we have shown that $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}}, \bar{\boldsymbol{Z}}, \bar{\boldsymbol{P}}, \bar{\boldsymbol{\Theta}}, \bar{\boldsymbol{\alpha}})$ is feasible to (20). This achieves an objective of

$$\begin{aligned}\|\boldsymbol{D} - \bar{\boldsymbol{X}} - \bar{\boldsymbol{Y}}\|_F^2 + \lambda\text{tr}(\bar{\boldsymbol{\Theta}}) + \mu\text{tr}(\boldsymbol{E}\bar{\boldsymbol{\alpha}}) &= \|\boldsymbol{D} - \bar{\boldsymbol{X}} - \bar{\boldsymbol{Y}}\|_F^2 + \lambda\text{tr}(\bar{\boldsymbol{X}}^T\bar{\boldsymbol{X}}) + \mu\sum_{ij}\bar{Y}_{ij}^2 \\ &= \|\boldsymbol{D} - \bar{\boldsymbol{X}} - \bar{\boldsymbol{Y}}\|_F^2 + \lambda\|\bar{\boldsymbol{X}}\|_F^2 + \mu\|\bar{\boldsymbol{Y}}\|_F^2.\end{aligned}$$

which is the same objective achieved by $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{Y}})$ in (1). This completes the proof. ■

Appendix B. Alternative Proof of Proposition 6

Proof Clearly, $\boldsymbol{X}^*$ is feasible for (12). Let $\boldsymbol{P}^* = \boldsymbol{U}_{k_0}\boldsymbol{U}_{k_0}^T$ and $\boldsymbol{\Theta}^* = \boldsymbol{X}^{*T}\boldsymbol{X}^*$. As established in the proof of Theorem 16, $(\boldsymbol{X}^*, \boldsymbol{P}^*, \boldsymbol{\Theta}^*)$ is feasible to (21) and achieves the same objective as $\boldsymbol{X}^*$ does in (12). We prove Proposition 6 by deriving the dual of (21) and constructing a dual feasible solution that achieves the same objective value as $(\boldsymbol{X}^*, \boldsymbol{P}^*, \boldsymbol{\Theta}^*)$ achieves in (21). By duality, this then implies that $(\boldsymbol{X}^*, \boldsymbol{P}^*, \boldsymbol{\Theta}^*)$ is optimal for (21) which in turn implies that $\boldsymbol{X}^*$ is optimal for (12).

The dual of (21) is given by

$$\begin{aligned}\max_{\boldsymbol{A},\boldsymbol{B}\in\mathcal{S}_+^n,\sigma\geq 0} \quad & \|\bar{\boldsymbol{D}}\|_F^2 + \sigma(n-k_0) - \text{tr}(\boldsymbol{B}) \\ \text{s.t.} \quad & (1+\lambda)\mathbb{I} \succeq \boldsymbol{A},\; \boldsymbol{B} \succeq \sigma\mathbb{I},\; \begin{pmatrix} \boldsymbol{A} & \bar{\boldsymbol{D}} \\ \bar{\boldsymbol{D}}^T & \boldsymbol{B} \end{pmatrix} \succeq 0.\end{aligned} \tag{38}$$

Let $\{\phi_i\}_{i=1}^n$ denote the collection of singular values of $\bar{\boldsymbol{D}}$ in non-increasing order (so that $\phi_i \geq \phi_{i+1} \;\forall\, i$). Let $\sigma^* = \frac{1}{1+\lambda}\phi_{k_0}^2$. Let $\nu_i^* = \frac{1}{1+\lambda}\phi_i^2 \;\forall\, i < k_0$ and let $\nu_i^* = \sigma^* \;\forall\, k_0 \leq i \leq n$. Let $\boldsymbol{A}^* = (1+\lambda)\mathbb{I}$ and $\boldsymbol{B}^* = \boldsymbol{U}Diag(\boldsymbol{\nu})\boldsymbol{U}^T$ where $\bar{\boldsymbol{D}} = \boldsymbol{U}\boldsymbol{\Phi}\boldsymbol{U}^T$ is a spectral decomposition of $\bar{\boldsymbol{D}}$ and $Diag(\boldsymbol{\nu})$ denotes the $n \times n$ diagonal matrix with diagonal entries given by the entries of $\boldsymbol{\nu}$. Note that the solution $(\boldsymbol{A}^*, \boldsymbol{B}^*, \sigma^*)$ is feasible to (38). To see this, observe that by construction, we have $\boldsymbol{A}^*, \boldsymbol{B}^* \in \mathcal{S}_+^n, \sigma^* \geq 0$, and $(1+\lambda)\mathbb{I} \succeq \boldsymbol{A}^*$. Moreover, since $\{\phi_i\}_{i=1}^n$ are in non-increasing order, we have $\min_i \nu_i \geq \sigma^*$ which implies $\boldsymbol{B}^* \succeq \sigma\mathbb{I}$. Finally, we have $\nu_i \geq \frac{1}{1+\lambda}\phi_i^2 \;\forall\, i$ which implies that $\boldsymbol{B}^* \succeq \bar{\boldsymbol{D}}^T\boldsymbol{A}^{*-1}\bar{\boldsymbol{D}}$ and $\begin{pmatrix} \boldsymbol{A}^* & \bar{\boldsymbol{D}} \\ \bar{\boldsymbol{D}}^T & \boldsymbol{B}^* \end{pmatrix} \succeq 0$. The feasible solution $(\boldsymbol{A}^*, \boldsymbol{B}^*, \sigma^*)$ achieves an objective of:

$$\begin{aligned}\|\bar{\boldsymbol{D}}\|_F^2 + \sigma^*(n-k_0) - \text{tr}(\boldsymbol{B}^*) &= \sum_{i=1}^n \phi_i^2 + \frac{n-k_0}{1+\lambda}\phi_{k_0}^2 - \frac{1}{1+\lambda}\sum_{i=1}^{k_0-1}\phi_i^2 - \frac{1}{1+\lambda}\sum_{i=k_0}^n \phi_{k_0}^2 \\ &= \frac{\lambda}{1+\lambda}\sum_{i=1}^{k_0}\phi_i^2 + \sum_{i=k_0+1}^n \phi_i^2\end{aligned}$$

in (38). Moreover, the solution $(\boldsymbol{X}^*, \boldsymbol{P}^*, \boldsymbol{\Theta}^*)$ achieves the same objective in (21):

$$\begin{aligned}\|\bar{\boldsymbol{D}}\|_F^2 + (1+\lambda)\mathrm{tr}(\bar{\boldsymbol{\Theta}}^*) - 2\cdot \mathrm{tr}(\bar{\boldsymbol{X}}^*\bar{\boldsymbol{D}}) &= \sum_{i=1}^{n} \phi_i^2 + \frac{1}{1+\lambda}\sum_{i+1}^{k_0}\phi_i^2 - \frac{2}{1+\lambda}\sum_{i=1}^{k_0}\phi_i^2 \\ &= \frac{\lambda}{1+\lambda}\sum_{i=1}^{k_0}\phi_i^2 + \sum_{i=k_0+1}^{n}\phi_i^2.\end{aligned}$$

By duality, the objective value of any feasible solution to (38) provides a lower bound on the objective of (21). Since $(\boldsymbol{X}^*, \boldsymbol{P}^*, \boldsymbol{\Theta}^*)$ is primal feasible and achieves the same objective as a feasible dual solution, it must be optimal for (21). This in turn implies that $\boldsymbol{X}^*$ is optimal to (12) by Theorem 16. This completes the proof. ■

Appendix C. Proof of Convexity in the Low-Rank Subproblem

Proof We prove the equivalence in two steps. First, we show that given a feasible solution to (12), we can construct a feasible solution to (21) that achieves the same objective value. Second, we show that given a feasible solution to (21), we can construct a feasible solution to (12) that achieves the same or lower objective. Given an arbitrary feasible solution to (21), we construct a linear optimization problem in which feasible solutions correspond to feasible solutions to (21) and extreme points of the feasible set of the linear optimization problem correspond to feasible solutions to (12). The initial feasible solution to (21) is feasible to this linear optimization problem, so there is an extreme point corresponding to a feasible solution to (12) that achieves an equal or lower objective value.

Consider an arbitrary feasible solution $\bar{\boldsymbol{X}}$ to Problem (12). Since $\boldsymbol{D}$ is symmetric, we can restrict ourselves to considering symmetric feasible solutions. Since we have $\mathrm{Rank}(\bar{\boldsymbol{X}}) \leq k$ and $\bar{\boldsymbol{X}}$ is symmetric, we can factor $\bar{\boldsymbol{X}}$ as $\bar{\boldsymbol{X}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T$ where $\boldsymbol{U} \in \mathbb{R}^{n\times k_0}$, $\boldsymbol{U}^T\boldsymbol{U} = \mathbb{I}_{k_0}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{k_0\times k_0}$ and $\boldsymbol{\Sigma}$ is diagonal. Let $\bar{\boldsymbol{P}} = \boldsymbol{U}\boldsymbol{U}^T$. $\bar{\boldsymbol{P}}$ is the orthogonal projection matrix onto the k_0 dimensional column space of $\boldsymbol{U}$. This implies that $\bar{\boldsymbol{P}} \succeq 0$, $\mathbb{I} - \bar{\boldsymbol{P}} \succeq 0$ and $\mathrm{tr}(\bar{\boldsymbol{P}}) \leq k_0$. Let $\bar{\boldsymbol{\Theta}} = \bar{\boldsymbol{X}}^T\bar{\boldsymbol{X}} \succeq 0$. Note that $\bar{\boldsymbol{P}}\bar{\boldsymbol{X}} = \bar{\boldsymbol{X}}$ and $\bar{\boldsymbol{P}} = \bar{\boldsymbol{P}}^\dagger$, where $\bar{\boldsymbol{P}}^\dagger$ denotes the pseudo-inverse of $\bar{\boldsymbol{P}}$, since $\bar{\boldsymbol{P}}$ is an orthogonal projection matrix. Thus, we have $\bar{\boldsymbol{\Theta}} - \bar{\boldsymbol{X}}^T\bar{\boldsymbol{P}}^\dagger\bar{\boldsymbol{X}} = 0 \implies \begin{pmatrix}\bar{\boldsymbol{\Theta}} & \bar{\boldsymbol{X}} \\ \bar{\boldsymbol{X}}^T & \bar{\boldsymbol{P}}\end{pmatrix} \succeq 0$. We have shown that $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{P}}, \bar{\boldsymbol{\Theta}})$ is feasible to (21). To see that this solution achieves the same objective as $\bar{\boldsymbol{X}}$ achieves in (12), note that

$$\begin{aligned}\|\bar{\boldsymbol{D}} - \bar{\boldsymbol{X}}\|_F^2 + \lambda\|\bar{X}\|_F^2 &= \|\bar{\boldsymbol{D}}\|_F^2 + (1+\lambda)\|\bar{\boldsymbol{X}}\|_F^2 - 2\cdot\mathrm{tr}(\bar{\boldsymbol{X}}\bar{\boldsymbol{D}}) \\ &= \|\bar{\boldsymbol{D}}\|_F^2 + (1+\lambda)\mathrm{tr}(\bar{\boldsymbol{\Theta}}) - 2\cdot\mathrm{tr}(\bar{\boldsymbol{X}}\bar{\boldsymbol{D}}).\end{aligned}$$

Now, consider an arbitrary feasible solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{P}}, \bar{\boldsymbol{\Theta}})$ to (21). Since the objective function of (21) includes the term $\mathrm{tr}(\boldsymbol{\Theta})$ and feasibility requires $\boldsymbol{\Theta} \succeq \boldsymbol{X}^T\boldsymbol{P}^\dagger\boldsymbol{X}$, we can take $\boldsymbol{\Theta}' = \bar{\boldsymbol{X}}^T\bar{\boldsymbol{P}}^\dagger\bar{\boldsymbol{X}}$ and the solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{P}}, \boldsymbol{\Theta}')$ will be feasible to (21) with an objective value no greater than that of the original feasible solution. Since $\bar{\boldsymbol{P}}$ is PSD, it can be written as $\bar{\boldsymbol{P}} = \sum_{i=1}^n \phi_i u_i u_i^T$ where $u_i^T u_i = 1$ for all i, $u_i^T u_j = 0$ for all $i \neq j$ and the feasibility of $\bar{\boldsymbol{P}}$ implies $0 \leq \phi_i \leq 1$ for all i. Moreover, we have $\bar{\boldsymbol{P}}^\dagger = \sum_{i:\phi_i\neq 0}\frac{1}{\phi_i}u_i u_i^T$. Further, since the feasibility condition $\begin{pmatrix} \boldsymbol{\Theta}' & \bar{\boldsymbol{X}} \\ \bar{\boldsymbol{X}}^T & \bar{\boldsymbol{P}} \end{pmatrix} \succeq 0$ implies that $\bar{\boldsymbol{X}} = \bar{\boldsymbol{P}}^\dagger \bar{\boldsymbol{P}} \bar{\boldsymbol{X}}$ by the generalized Schur complement lemma (see Boyd et al. 1994, Equation 2.41) and $\bar{\boldsymbol{X}}$ is symmetric, without loss of generality it can be written as $\bar{\boldsymbol{X}} = \sum_{i=1}^n \sigma_i u_i u_i^T$. The solution $(\bar{\boldsymbol{X}}, \bar{\boldsymbol{P}}, \boldsymbol{\Theta}')$ achieves an objective of

$$\begin{aligned} h(\bar{\boldsymbol{X}}, \bar{\boldsymbol{P}}, \boldsymbol{\Theta}') &= \|\bar{\boldsymbol{D}}\|_F^2 + (1+\lambda)\mathrm{tr}(\boldsymbol{\Theta}') - 2 \cdot \mathrm{tr}(\bar{\boldsymbol{X}}\bar{\boldsymbol{D}}) \\ &= \|\bar{\boldsymbol{D}}\|_F^2 + \sum_{i:\phi_i \neq 0} \left[\frac{1+\lambda}{\phi_i} \sigma_i^2 - 2 \cdot \sigma_i \mathrm{tr}(u_i u_i^T \bar{\boldsymbol{D}}) \right]. \end{aligned}$$

Note that if we view the above as a function of σ_i and ϕ_i (denoted $f(\phi, \sigma)$), then this expression corresponds to the objective value achieved by some feasible solution to (21) provided we constrain $0 \leq \phi_i \leq 1$ and $\sum_i \phi_i \leq k_0$. $h(\phi, \sigma)$ is a convex quadratic in σ_i. It is minimized when $\nabla_{\sigma_i} h(\phi, \sigma) = \frac{2(1+\lambda)}{\phi_i}\sigma_i - 2\mathrm{tr}(u_i u_i^T \bar{\boldsymbol{D}}) = 0 \implies \sigma_i = \frac{\phi_i}{1+\lambda}\mathrm{tr}(u_i u_i^T \bar{\boldsymbol{D}})$. Substituting the optimal value of σ_i into $h(\phi, \sigma)$, we obtain

$$h(\phi) = \min_\sigma f(\phi, \sigma) = \|\bar{\boldsymbol{D}}\|_F^2 - \sum_{i:\phi_i \neq 0} \frac{\phi_i}{1+\lambda} [\mathrm{tr}(u_i u_i^T \bar{\boldsymbol{D}})]^2 = \|\bar{\boldsymbol{D}}\|_F^2 - \sum_{i=1}^n \frac{\phi_i}{1+\lambda} [\mathrm{tr}(u_i u_i^T \boldsymbol{D}^*)]^2.$$

$h(\phi)$ is a linear function of ϕ. Therefore, the minimum of $h(\phi)$ over the set $0 \leq \phi_i \leq 1$ for all i, $\sum_i \phi_i \leq k_0$ is achieved at some $\phi^* \in \{0,1\}^{n \times n}$. Let $\boldsymbol{P}^* = \sum_{i=1}^n \phi_i^* u_i u_i^T$, $\boldsymbol{X}^* = \sum_{i=1}^n \phi_i^* \mathrm{tr}(u_i u_i^T \bar{\boldsymbol{D}}) u_i u_i^T$ and $\boldsymbol{\Theta}^* = \boldsymbol{X}^{*T} \boldsymbol{P}^* \boldsymbol{X}$. Then $(\boldsymbol{X}^*, \boldsymbol{P}^*, \boldsymbol{\Theta}^*)$ is feasible to (21) and achieves objective $h(\phi^*)$. By construction, we have

$$h(\phi^*) \leq h(\bar{\boldsymbol{X}}, \bar{\boldsymbol{P}}, \boldsymbol{\Theta}') \leq h(\bar{\boldsymbol{X}}, \bar{\boldsymbol{P}}, \bar{\boldsymbol{\Theta}}).$$

Further, since $\phi^* \in \{0,1\}^{n \times n}$ and $\sum_i \phi_i^* \leq k_0$, we have $\mathrm{Rank}(\boldsymbol{X}^*) \leq k_0$ which means that $\boldsymbol{X}^*$ is feasible to (12) and achieves objective $h(\phi^*)$. This completes the proof. ■

Appendix D. Alternative Proof of Proposition 8

Proof Let $f(\boldsymbol{Y}) = \|\tilde{\boldsymbol{D}} - \boldsymbol{Y}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2$, the objective function of Problem (14). We can rewrite $f(\boldsymbol{Y})$ as:

$$\begin{aligned} f(\boldsymbol{Y}) = \|\tilde{\boldsymbol{D}} - \boldsymbol{Y}\|_F^2 + \mu\|\boldsymbol{Y}\|_F^2 &= \sum_{ij} (\tilde{d}_{ij} - y_{ij})^2 + \mu \sum_{ij} y_{ij}^2 \\ &= \sum_{ij} \left[(\tilde{d}_{ij} - y_{ij})^2 + y_{ij}^2 \right] = \sum_{ij} f_{ij}(y), \end{aligned}$$

where we define $f_{ij}(y) = (\tilde{d}_{ij} - y)^2 + y^2$. We have shown that the objective function is separable, so Problem (14) can be solved by minimizing each function $f_{ij}(y)$. $f_{ij}(y)$ is a convex quadratic function, and simple univariate calculus allows us to conclude that it achieves its minimum when $y^* = \frac{\tilde{d}_{ij}}{1+\mu}$. The minimum value of f_{ij} is therefore $f_{ij}(y^*) = \frac{\mu}{1+\mu}\tilde{d}_{ij}^2$. However, due to the sparsity constraint on $\boldsymbol{Y}$, at most k_1 entries of $\boldsymbol{Y}$ can be non-zero. By introducing binary variables s_{ij} and noting that $f_{ij}(0) = \tilde{d}_{ij}^2$, we can rewrite the objective of problem 2 as a function of the binary matrix $\boldsymbol{S}$:

$$f(\boldsymbol{S}) = \sum_{ij} \left[s_{ij} \cdot \frac{\mu}{1+\mu} \tilde{d}_{ij}^2 + (1 - s_{ij}) \cdot \tilde{d}_{ij}^2 \right].$$

Due to the sparsity constraint, at most k_1 of the variables s_{ij} can be 1 while all others must be 0. If $s_{ij} = 0$, the objective increases by $\tilde{d}_{ij}^2$ whereas if $s_{ij} = 1$, the objective only increases by $\frac{\mu}{1+\mu} \tilde{d}_{ij}^2$. It follows immediately that the objective will be minimized when $s_{ij} = 1$ if and only if $\tilde{d}_{ij}$ is one of the k_1 largest entries in absolute value of the matrix $\tilde{\boldsymbol{D}}$. Note that in the case that the k_1^{th} largest entry in absolute value and the $(k_1 + 1)^{th}$ largest entry in absolute value are not distinct, the tie can be broken arbitrarily. Letting $\boldsymbol{S}^*$ represent the binary matrix formed by an optimal choice of the binary variables s_{ij}, the solution to Problem (14) is given by $\boldsymbol{Y}^* = \boldsymbol{S}^* \circ \left(\frac{\tilde{\boldsymbol{D}}}{1+\mu} \right)$. ■

Appendix E. Supplemental Computational Results

Table 2: Comparison of average low-rank matrix reconstruction error generated by S-PCP, GoDec, ScaledGD, AccAltProj, fR-PCA, and Algorithm 1. Results are reported for the exact SVD implementation of GoDec. Averaged over 10 trials for each parameter configuration.

					L Error					
N	k_0	k_1	S-PCP Rank	S-PCP Sparsity	S-PCP	GoDec	ScaledGD	AccAltProj	fRPCA	Alg 1 Exact
20	1	20	5.7	95.6	0.0176	0.0101	0.0082	0.0111	0.0088	**0.0072**
20	2	40	12.0	197.4	0.0178	0.0430	0.0062	0.0074	0.0068	**0.0057**
20	3	60	15.3	275.3	0.1123	0.1136	0.0084	0.0083	0.0077	**0.0075**
20	4	80	17.5	341.1	0.1510	0.3247	0.0087	0.0092	0.0088	**0.0079**
40	2	80	5.4	286.6	0.0233	0.0121	0.0147	0.0168	0.0174	**0.0110**
40	4	160	16.4	417.2	0.0272	0.0189	0.0122	0.0143	0.0136	**0.0113**
40	6	240	27.3	731.3	0.0334	0.0996	0.0159	0.0171	0.0165	**0.0145**
40	8	320	36.7	1365.1	0.0453	0.3225	0.0170	0.0178	0.0157	**0.0149**
60	3	180	7.8	631.6	0.0311	0.0158	0.0182	0.0231	0.0197	**0.0149**
60	6	360	13.0	777.6	0.0328	0.0247	0.0171	0.0222	0.0177	**0.0150**
60	9	540	36.3	1181.1	0.0439	0.0520	0.0236	0.0251	0.0226	**0.0202**
60	12	720	55.9	2930.5	0.0577	0.2696	0.0236	0.0316	0.0242	**0.0209**
80	4	320	10.9	1128.5	0.0345	0.0176	0.0230	0.0272	0.0238	**0.0166**
80	8	640	15.4	1380.1	0.0448	0.0293	0.0240	0.0314	0.0248	**0.0223**
80	12	960	34.0	1634.6	0.0569	0.0537	0.0271	0.0307	0.0269	**0.0246**
80	16	1280	62.7	3316.8	0.0737	0.2989	0.0339	0.0378	0.0339	**0.0300**
100	5	500	13.8	1771.6	0.0443	0.0255	0.0288	0.0383	0.0267	**0.0239**
100	10	1000	19.2	2139.9	0.0531	0.0357	0.0318	0.0385	0.0345	**0.0271**
100	15	1500	36.4	2525.9	0.0640	0.0679	0.0356	0.0392	0.0330	**0.0304**
100	20	2000	63.4	3145.1	0.0840	0.3675	0.0399	0.0471	0.0395	**0.0381**
120	12	1440	21.3	3067.7	0.0644	0.0423	0.0368	0.0474	0.0400	**0.0333**
120	18	2160	38.8	3628.4	0.0789	0.0858	0.0440	0.0497	0.0424	**0.0388**
120	24	2880	72.0	4288.3	0.0968	0.3838	0.0512	0.0570	0.0498	**0.0464**
140	7	980	19.3	3436.0	0.0613	0.0365	0.0386	0.0553	0.0375	**0.0331**
140	21	2940	37.9	4911.8	0.0868	0.0910	0.0506	0.0573	0.0479	**0.0442**
140	28	3920	76.7	5790.9	0.1085	0.4156	0.0607	0.0695	0.0598	**0.0566**

Table 3: Bound gap of Algorithm 1 derived using (20). Averaged over 10 trials for each parameter configuration.

			L Error							
N	k_0	k_1	S-PCP	GoDec	ScaledGD	AccAltProj	fRPCA	Alg 1 Exact	Alg 1 Bound Gap	Bound Time (s)
20	1	20	.0176	.0101	0.0082	0.0111	0.0088	**0.0072**	0.7052	3.7200
60	6	360	.0328	.0247	0.0171	0.0222	0.0177	**0.0150**	0.8543	189.1900
60	9	540	.0439	.052	0.0236	0.0251	0.0226	**0.0202**	0.8601	184.9500
60	12	720	.0577	.2696	0.0236	0.0316	0.0242	**0.0209**	0.7709	155.2800
80	4	320	.0345	.0176	0.0230	0.0272	0.0238	**0.0166**	0.9180	577.8400
80	8	640	.0448	.0293	0.0240	0.0314	0.0248	**0.0223**	0.9267	765.9100
80	12	960	.0569	.0537	0.0271	0.0307	0.0269	**0.0246**	0.7944	691.5500
80	16	1280	.0737	.2989	0.0339	0.0378	0.0339	**0.0300**	0.7803	611.4700
100	5	500	.0443	.0255	0.0288	0.0383	0.0267	**0.0239**	0.9592	1936.2600
100	10	1000	.0531	.0357	0.0318	0.0385	0.0345	**0.0271**	0.9382	2987.0800
100	15	1500	.064	.0679	0.0356	0.0392	0.0330	**0.0304**	0.9062	2224.6100
20	2	40	.0178	.043	0.0062	0.0074	0.0068	**0.0057**	0.5935	3.8200
100	20	2000	.084	.3675	0.0399	0.0471	0.0395	**0.0381**	0.8145	2188.6600
120	12	1440	.0644	.0423	0.0368	0.0474	0.0400	**0.0333**	0.8951	6759.9200
120	18	2160	.0789	.0858	0.0440	0.0497	0.0424	**0.0388**	0.8968	6878.3600
120	24	2880	.0968	.3838	0.0512	0.0570	0.0498	**0.0464**	0.7877	5310.5800
140	7	980	.0613	.0365	0.0386	0.0553	0.0375	**0.0331**	0.9014	14731.2500
140	21	2940	.0868	.091	0.0506	0.0573	0.0479	**0.0442**	0.8854	11260.5200
140	28	3920	.1085	.4156	0.0607	0.0695	0.0598	**0.0566**	0.8116	11840.3000
20	3	60	.1123	.1136	0.0084	0.0083	0.0077	**0.0075**	0.5443	3.9600
20	4	80	.151	.3247	0.0087	0.0092	0.0088	**0.0079**	0.7146	4.0500
40	2	80	.0233	.0121	0.0147	0.0168	0.0174	**0.0110**	0.8214	30.6200
40	4	160	.0272	.0189	0.0122	0.0143	0.0136	**0.0113**	0.8804	27.9200
40	6	240	.0334	.0996	0.0159	0.0171	0.0165	**0.0145**	0.7937	28.4700
40	8	320	.0453	.3225	0.0170	0.0178	0.0157	**0.0149**	0.7051	23.8700
60	3	180	.0311	.0158	0.0182	0.0231	0.0197	**0.0149**	0.8075	154.9000

Table 4: Running time of the exact implementation of Algorithm 1 and the accelerated implementation of Algorithm 1. In the exact implementation, the SVD step is computed exactly, whereas in the accelerated implementation, a randomized SVD is employed in all but the final SVD step. Averaged over 10 trials for each parameter configuration.

			L Error		Time (s)		
N	k_0	k_1	Alg 1 Exact	Alg 1 Acc	Alg 1 Exact	Alg 1 Acc	Time Decrease (%)
20	1	20	**0.0072**	0.0094	0.1351	**0.0986**	27.06
20	2	40	**0.0057**	0.0084	0.2342	**0.1071**	54.27
20	3	60	**0.0075**	0.0084	0.5713	**0.1394**	75.59
20	4	80	**0.0079**	0.0085	0.8126	**0.1519**	81.31
40	2	80	**0.0110**	0.0123	0.4157	**0.1982**	52.31
40	4	160	**0.0113**	0.0139	0.9250	**0.2536**	72.59
40	6	240	**0.0145**	0.0183	2.0046	**0.3574**	82.17
40	8	320	**0.0149**	0.0192	2.8281	**0.4309**	84.76
60	3	180	**0.0149**	0.0178	0.7407	**0.3964**	46.47
60	6	360	**0.0150**	0.0198	2.2547	**0.5103**	77.37
60	9	540	**0.0202**	0.0286	4.4260	**0.6930**	84.34
60	12	720	**0.0209**	0.0300	7.2143	**0.8724**	87.91
80	4	320	**0.0166**	0.0199	1.2156	**0.6214**	48.88
80	8	640	**0.0223**	0.0331	4.1513	**0.8543**	79.42
80	12	960	**0.0246**	0.0399	8.0393	**1.1153**	86.13
80	16	1280	**0.0300**	0.0488	13.5348	**1.2970**	90.42
100	5	500	**0.0239**	0.0289	1.5669	**0.9722**	37.95
100	10	1000	**0.0271**	0.0439	6.4084	**1.2111**	81.10
100	15	1500	**0.0304**	0.0540	12.8520	**1.5614**	87.85
100	20	2000	**0.0381**	0.0671	13.5619	**1.4767**	89.11
120	12	1440	**0.0333**	0.0564	9.2897	**1.6930**	81.78
120	18	2160	**0.0388**	0.0752	18.0824	**2.1187**	88.28
120	24	2880	**0.0464**	0.0932	19.8079	**1.9967**	89.92
140	7	980	**0.0331**	0.0428	2.6152	**1.6039**	38.67
140	21	2940	**0.0442**	0.0922	18.1729	**2.1653**	88.08
140	28	3920	**0.0566**	0.1296	29.6370	**2.6352**	91.11

Table 5: Low-rank matrix reconstruction error, sparse matrix reconstruction error and execution time of Algorithm 1, GoDec and ScaledGD.

			L Error			S Error			Time (s)		
N	k_0	k_1	Alg 1 Exact	GoDec	ScaledGD	Alg 1 Exact	GoDec	ScaledGD	Alg 1 Exact	GoDec	ScaledGD
200	5	500	**0.0442**	0.0458	0.0449	**0.5677**	0.9246	0.7379	0.0185	0.0187	**0.0134**
250	5	500	**0.0538**	0.0553	0.0544	**0.6176**	1.0208	0.7417	**0.0191**	0.0250	0.0225
300	5	500	**0.0641**	0.0654	0.0644	**0.6725**	1.1036	0.7741	0.0314	**0.0290**	0.0321
350	5	500	**0.0755**	0.0766	0.0757	**0.7307**	1.1955	0.8259	0.0436	**0.0411**	0.0454
400	5	500	**0.0852**	0.0863	0.0854	**0.7716**	1.2483	0.8578	0.0574	**0.0517**	0.0562
450	5	500	**0.0970**	0.0980	0.0972	**0.8038**	1.2918	0.9134	0.0792	**0.0712**	0.0751
500	5	500	**0.1083**	0.1093	0.1085	**0.8530**	1.3585	0.9746	0.0906	0.0918	**0.0895**
550	5	500	**0.1213**	0.1222	0.1215	**0.8918**	1.4021	1.0518	0.1138	**0.1049**	0.1083
600	5	500	**0.1322**	0.1331	0.1324	**0.9377**	1.4593	1.1210	0.1357	0.1384	**0.1228**
650	5	500	**0.1430**	0.1438	0.1433	**0.9624**	1.4842	1.1881	**0.1538**	0.1693	0.1590
700	5	500	**0.1554**	0.1562	0.1556	**1.0126**	1.5524	1.2712	0.1810	0.2022	**0.1587**
750	5	500	**0.1681**	0.1689	0.1682	**1.0244**	1.5587	1.3332	**0.3668**	0.5669	0.5734
800	5	500	**0.1812**	0.1820	**0.1812**	**1.0676**	1.6062	1.4105	**0.3395**	0.5000	1.1244
850	5	500	0.1918	0.1925	**0.1917**	**1.0967**	1.6372	1.4958	**0.9337**	1.0395	1.3067
900	5	500	0.2057	0.2064	**0.2056**	**1.1348**	1.6852	1.5847	1.7587	1.5853	**1.1520**
950	5	500	**0.2174**	0.2181	0.2175	**1.1543**	1.6942	1.6608	0.7749	**0.7494**	2.0345
1000	5	500	0.2306	0.2313	**0.2305**	**1.1783**	1.7207	1.7417	3.2104	**3.1600**	3.3916
2000	2	500	**0.5171**	0.5177	0.5173	**1.5707**	2.1098	3.6181	1.3195	1.3155	**1.0648**
4000	2	500	**1.3013**	1.3019	1.3018	**2.1207**	2.6438	7.9775	35.1148	36.9397	**19.1202**
6000	2	500	**2.3694**	2.3700	2.3704	**2.3058**	2.7742	11.9133	84.7058	87.7330	**64.5782**
8000	2	500	**3.5365**	3.5373	**3.5365**	**2.5880**	3.0463	16.8837	158.5785	160.0202	**132.8005**
10000	2	500	**4.8465**	4.8472	4.8486	**2.7586**	3.1967	21.5332	**133.3238**	145.8102	249.2882

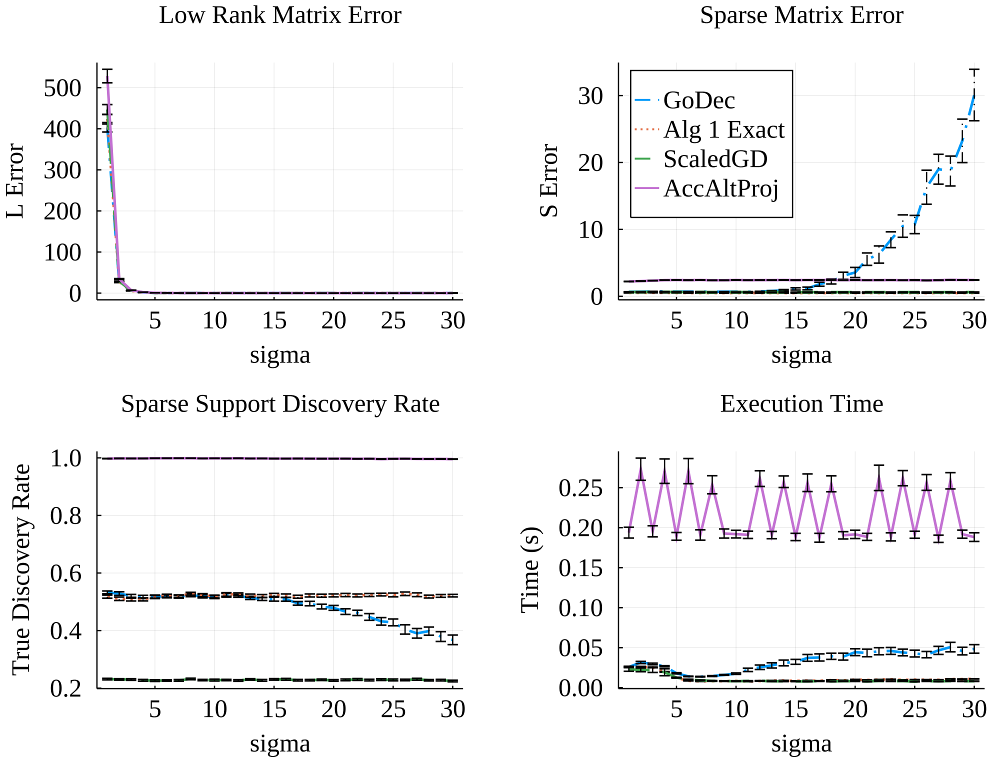

Figure 8: Low-rank matrix reconstruction error (top left), sparse matrix reconstruction error (top right), sparse support discovery rate (bottom left) and execution time (bottom right) versus σ with $n = 100$, $k_0 = 5$ and $k_1 = 500$. Averaged over 50 trials for each parameter configuration.