PREPROCESSING AND FIRST-ORDER PRIMAL-DUAL ALGORITHMS FOR CONVEX OPTIMIZATION

Yuzixuan Zhu

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Statistics and Operations Research.

Chapel Hill
2020

Approved by:

Shankar Bhamidi

Amarjit Budhiraja

Shu Lu

Gábor Pataki

Quoc Tran-Dinh

ABSTRACT

Yuzixuan Zhu: Preprocessing and First-Order Primal-Dual Algorithms for Convex Optimization
(Under the direction of Gábor Pataki and Quoc Tran-Dinh)

This thesis focuses on two topics in the field of convex optimization: preprocessing algorithms for semidefinite programs (SDPs), and first-order primal-dual algorithms for convex-concave saddle-point problems.

In the first part of this thesis, we introduce Sieve-SDP, a simple facial reduction algorithm to preprocess SDPs. Sieve-SDP inspects the constraints of the problem to detect lack of strict feasibility, deletes redundant rows and columns, and reduces the size of the variable matrix. It often detects infeasibility. It does not rely on any optimization solver: the only subroutine it needs is Cholesky factorization, hence it can be implemented in a few lines of code in machine precision. We present extensive computational results on several problem collections from the literature, with many SDPs coming from polynomial optimization.

In the second part, we develop two first-order primal-dual algorithms to solve a class of convex-concave saddle-point problems involving non-bilinear coupling function, which includes SDP as one of the many special cases. Both algorithms are single-loop and have low per-iteration complexity. Our first algorithm can achieve $\mathcal{O}\left(\frac{1}{k}\right)$ convergence rates on the duality gap in both *ergodic* (averaging) sense and *semi-ergodic* sense, i.e., *non-ergodic* (last-iterate) on the primal, and *ergodic* on the dual. This rate can be further improved on *non-ergodic* primal objective residual using a new parameter update rule. Under strong convexity assumption, our second algorithm can boost these convergence rates to no slower than $\mathcal{O}\left(\frac{1}{k^2}\right)$. Our results can be specified to handle general convex cone constrained problems. We test our algorithms on applications such as two-player games and image processing to compare our algorithms with existing methods.

To Fluffy and Lucky

ACKNOWLEDGEMENTS

First, I would like to thank my advisors, Dr. Quoc Tran-Dinh and Dr. Gábor Pataki. Dr. Pataki gave me this great opportunity to come to UNC and conduct research in optimization, and he introduced me to the beautiful area of semidefinite programming. What I also learn from him is the attention to detail, and that it is always possible to "try harder". I thank Dr. Tran-Dinh for the great deal of guidance, training, and support during my research in numerical optimization. Mathematical analysis was not my strength, but thanks to his patience when mentoring me, and the inspiring discussions, this thesis has been made possible.

I would also like to acknowledge the Department of Statistics and Operations Research at UNC-Chapel Hill, as well as National Science Foundation Awards DMS-1817272 and DMS-1619884, for providing me with the graduate-level education and financial support throughout the five years of my Ph.D. studies. Special thanks to my committee members, Dr. Shu Lu, Dr. Shankar Bhamidi, and Dr. Amarjit Budhiraja, for spending their precious time looking over my thesis and attending my defense.

I am grateful to the countless researchers in my field, as partially listed in the Bibliography. The ideas in this thesis would not exist without your decades of hard work. Your contributions have really made the world a better place, and I am humbled. You, among other pioneers in scientific research, have motivated me to give back by pushing the boundaries of science and technology just a little bit more.

I thank my fellow graduate students in UNC, especially Dr. Tianxiao Sun, Dr. Hongsheng Liu, Dr. Haipeng Gao, Miheer Dewaskar, Alexander Murph, and Aleksandr Touzov for great conversations and good times together throughout my graduate studies. I also thank Deyi Liu for collaborating on a paper together; Yang Luo for taking care of my cats when I was away from home; and Iain Carmichael, Carson Mosso, Taylor Petty, and Kevin O'Connor for being great office mates. Additional thanks to my friends Wenyan Zhu, Xiaotian Fang, Zonghao Zhang, and Yu Zhou. I have known you for almost decades, and your support has always been there.

Thanks to my future colleagues and mentors at ExxonMobil. It was an amazing time working with you as an intern, and thank you for offering me a full-time position as an optimization scientist. It has really given me peace of mind during this time of turmoil in Summer 2020 while I am writing this thesis. I cannot wait to join you.

Last but not least, I would like to thank my family. I thank my husband, Jonathan Hendricks, for accompanying me throughout the past five years. I am so lucky to have you, and could not imagine completing the Ph.D. program without your support. Thanks to our extremely cute cats, Fluffy and Lucky, for filling the home with so much happiness. In the end, I am grateful to my parents, Dr. Xiaokai Zhu and Qun Yu, who have encouraged me to take on this path of a Ph.D. program. Your patience, perspectives, and endless love has helped me through difficult times.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ADMM	alternating direction method of multipliers
APD	Hamedani's accelerated primal-dual method [57]
CP	cone program(ming), or Chambolle & Pock's primal-dual method [20]
DIW	Dressler-Illiman-de Wolff dataset of SDPs [33]
e.g.	for example
i.e.	that is
inf	infimum
infeas	infeasible
LP	linear program(ming)
max	maximum or to maximize
min	minimum or to minimize
ND	negative definite
PD	positive definite
PP	Permenter-Parrilo dataset of SDPs [104]
PSD	positive semidefinite
PSNR	peak signal-to-noise ratio
QCQP	quadratically constrained quadratic program(ming)
resp.	respectively
s.t.	subject to or such that
SDP	semidefinite program(ming)
SP	saddle-point (problem)
sup	supremum
TV	total variation
w/o	without
w.r.t.	with respect to

LIST OF SYMBOLS

$\oplus$	PSD principle block of a matrix
Δ_p	p-dimensional standard simplex, unless specified by context
$\delta_{\mathcal{C}}$	indicator function of a convex set $\mathcal{C}$
$\mathrm{dist}_{\mathcal{C}}$	distance to a convex set $\mathcal{C}$
$\mathrm{proj}_{\mathcal{C}}$	projection onto a convex set $\mathcal{C}$
$\mathrm{ri}\mathcal{C}$	relative interior of a convex set $\mathcal{C}$
$x \in \mathcal{C}$	x is an element of set $\mathcal{C}$
$\mathcal{C}_1 \subseteq \mathcal{C}_2$	set $\mathcal{C}_1$ is a subset of set $\mathcal{C}_2$
$\mathcal{D}$	dual objective value of a convex program
$\mathcal{D}^\star$	dual optimal objective value of a convex program
$\mathrm{argmax}\, f$	maximizer of function f
$\mathrm{argmin}\, f$	minimizer of function f
f^*	Fenchel conjugate of function f
$\mathrm{dom} f$	effective domain of function f
L_f	Lipschitz smoothness modulus of Lipschitz smooth function f
M_f	Lipschitz continuous modulus of Lipschitz continuous function f
μ_f	strong convexity modulus for strongly convex function f
∇f	gradient, subgradient, or Jacobian of function f
prox_f	proximal operator of function f
∂f	subdifferential of function f
$g \circ f$	composition $g(f(x))$ of two functions f and g
I	identity matrix
$\mathcal{K}^*$	the dual cone of convex cone $\mathcal{K}$
$\mathcal{L}$	Lagrangian function of a constrained convex program
$\mathcal{L}_\rho$	augmented Lagrangian function with penalty parameter ρ
$\widetilde{\mathcal{L}}$	the objective function of a primal-dual saddle-point problem
$\mathbb{N}$	the set of nonnegative integers
O	zero matrix

$u_k = \mathcal{O}(v_k)$	$\limsup_{k\to\infty} \frac{u_k}{v_k} < \infty$, where $\{u_k\} \subseteq \mathbb{R}_+$ and $\{v_k\} \subseteq \mathbb{R}_{++}$
$u_k = o(v_k)$	$\lim_{k\to\infty} \frac{u_k}{v_k} = 0$, where $\{u_k\} \subseteq \mathbb{R}_+$ and $\{v_k\} \subseteq \mathbb{R}_{++}$
$u_k = \underline{o}(v_k)$	$\liminf_{k\to\infty} \frac{u_k}{v_k} = 0$, where $\{u_k\} \subseteq \mathbb{R}_+$ and $\{v_k\} \subseteq \mathbb{R}_{++}$
$u_k = \Omega(v_k)$	$\liminf_{k\to\infty} \frac{u_k}{v_k} > 0$, where $\{u_k\} \subseteq \mathbb{R}_+$ and $\{v_k\} \subseteq \mathbb{R}_{++}$
$\mathcal{P}$	primal objective value of a convex program
$\mathcal{P}^\star$	primal optimal objective value of a convex program
$\mathbb{R}^p$	p-dimensional real vector space
$\mathbb{R}^p_+$	$\{x \in \mathbb{R}^p \mid x \geq 0\}$
$\mathbb{R}^p_{++}$	$\{x \in \mathbb{R}^p \mid x > 0\}$
$\mathcal{S}^n$	set of n-by-n symmetric matrices
$\mathcal{S}^n_+$	set of n-by-n symmetric and PSD matrices
$(x^\star, y^\star)$	pair of optimal primal-dual solutions, if exists, to a convex program
$x^\top$, $X^\top$	the transpose of vector x or matrix X
$\langle x, y\rangle$, $x^\top y$	inner product $\sum_i x_i y_i$ of vectors x and y
$\langle X, Y\rangle$, $X \cdot Y$	inner product $\sum_{i,j} X_{ij} Y_{ij}$ of matrices X and Y
$\lambda_{\min}(X)$	the smallest eigenvalue of square matrix X
$\lvert x\rvert$, $\lVert x\rVert_1$	1-norm $\sum_i \lvert x_i\rvert$ of scalar or vector x
$\lVert x\rVert$, $\lVert x\rVert_2$	2-norm $\sqrt{\langle x, x\rangle}$ of vector x
$\lVert X\rVert$, $\lVert X\rVert_2$	2-norm $\sup_{y\neq 0} \frac{\lVert Xy\rVert_2}{\lVert y\rVert_2}$ of matrix X
$\lVert x\rVert_\infty$, $\lVert X\rVert_\infty$	infinity norm $\max_i \lvert x_i\rvert$ of vector x, or $\max_{ij} \lvert X_{ij}\rvert$ of matrix X
$\lVert X\rVert_F$	Frobenius norm $\sqrt{\langle X, X\rangle}$ of matrix X or of a tensor
$R(X)$	range space of matrix X
$X \succeq Y$	square matrix $X - Y$ is PSD
$X \succ Y$	square matrix $X - Y$ is positive definite
$\mathcal{X}$	set that contains primal solutions of a convex program
$\mathcal{Y}$	set that contains dual solutions of a convex program

CHAPTER 1

INTRODUCTION

1.1 Background

The broad field of optimization studies the following three main aspects:

- *Modeling:* To describe a real-world problem as an optimization model;
- *Theory:* To build or discover mathematical structures or properties of optimization models;
- *Algorithms:* To develop efficient methods to solve optimization models.

The use of optimization modeling is prevalent in many areas of modern science and technology, for example, statistics, queueing, scheduling, portfolio, transportation, network flows, signal processing, and machine learning, just to name a few. Once an optimization model is developed, one can apply appropriate optimization algorithms to solve the model to optimum or near-optimum. The resulting solution in turn guides the decision-making of the original real-world problem. In order to design correct and efficient algorithms, one needs theoretical knowledge on the structure of the optimization models, such as duality theory. This relationship between *Modeling*, *Theory*, and *Algorithms* is summarized in Figure 1.1, where the dashed frame denotes the regime of optimization and where it lies in the process of real-world decision-making.

In regards of the above-mentioned three components of optimization, this thesis focuses on the aspect of *Algorithms*.

Specifically, this thesis seek to design efficient algorithms to preprocess or to solve a special type of optimization models: convex program, which is to minimize a convex function over a convex set. We look at the problem of the form:

$$\min_{x\in\mathbb{R}^p} \quad F(x) + H\left(g(x)\right), \tag{P}$$

where $F:\ \mathbb{R}^p \to \mathbb{R}\cup\{+\infty\}$ and $H:\ \mathbb{R}^m \to \mathbb{R}\cup\{+\infty\}$ are closed and convex; the vector function

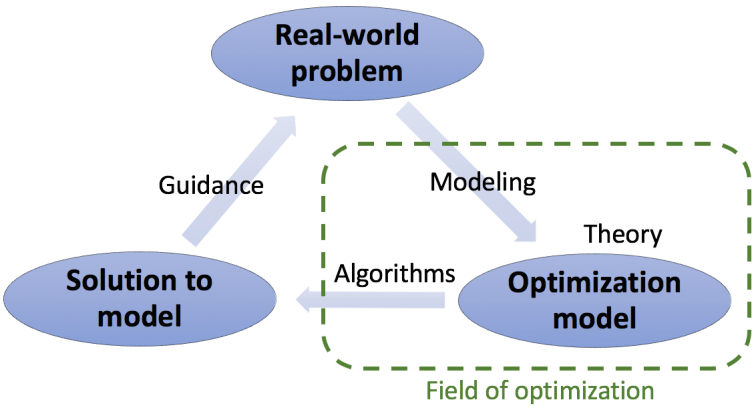

Figure 1.1: Optimization's components and its role in decision-making

$g: \mathbb{R}^p \to \mathbb{R}^m$ is such that $H \circ g$ is convex.

The example below lists some applications of problem (P), roughly sorted from more general to more specific.

Example 1.1. *1.* General finite-dimensional convex program. *When* $H(g(x)) = H(x) = \delta_{\mathcal{C}}(x)$, *where* $\mathcal{C} \subseteq \mathbb{R}^p$, *then* (P) *becomes the general form* $\min_{x \in \mathcal{C}} F(x)$, *which is to minimize a convex function* F *over the finite-dimensional convex set* $\mathcal{C}$.

2. Saddle-point form. *Writing* $H(g(x)) = \max_y \{\langle g(x), y\rangle - H^*(y)\}$, *then we obtain an equivalent saddle-point form of* (P)*:*

$$\min_x \max_y \quad F(x) + \langle g(x), y\rangle - H^*(y). \tag{SP}$$

This problem is convex in x *and convex in* y, *and by minimizing over* x *and maximizing over* y, *the resulting saddle-point can be understood as the strategies that lead to Nash equilibrium in a two-player game.*

For many applications, it is beneficial to cast the problem (P) *into this saddle-point form; see, e.g., machine learning [2, 41, 54, 57, 65, 72, 143], optimal transport [105], robust convex optimization [7, 111], and signal processing [65]. More applications are presented in the book [40]. Furthermore,* (SP) *can serve as the subproblems in algorithms for non-convex optimization problems [10, 127].*

3. Linear g. *When* $g(x) = Ax$ *is linear, we obtain a special problem of minimizing* $F(x) + H(Ax)$.

While being a special case, it already covers various applications in signal and image pro-

cessing [20, 26, 38, 52, 95], machine learning [36], and statistics [52, 58]. More applications are presented in the books [5, 50].

4. Cone-constrained program. *When $H^* \equiv \delta_{\mathcal{K}^*}$, the indicator of the dual cone of a proper cone,*[1] *then* (P) *becomes*

$$\min_{x\in\mathbb{R}^p} F(x) \quad \text{s.t.} \quad g(x) \in -\mathcal{K}. \tag{1.1}$$

It is a convex program, since the assumption that $H \circ g$ being convex ensures that g is $\mathcal{K}$-convex. It generalizes conic programming, as the objective function F is not necessarily linear. It includes many widely used special cases, as seen below.

5. Product-of-cones constraints. *In* (1.1)*, let $\mathcal{K} = \{0\}^n \times \mathbb{R}^m_+$, then we obtain*

$$\min_{x\in\mathbb{R}^p} F(x) \quad \text{s.t.} \quad Ax = b, \quad g(x) \le 0, \tag{1.2}$$

where $A \in \mathbb{R}^{n\times p}$, and $g : \mathbb{R}^p \to \mathbb{R}^m$ is convex. This setting is very common in classical nonlinear programming literature. In particular, if both F and g are quadratic, then we obtain a quadratically constrained quadratic program (QCQP).

6. Semidefinite program (SDP). *In* (1.1)*, let $\mathcal{K} := \mathcal{S}^n_+$, the cone of symmetric positive semidefinite (PSD) matrices of order n, and let $g(x) := \sum_{l=1}^p x_l A_l - C$ for some symmetric matrices $A_1, \ldots, A_p, C \in \mathcal{S}^n$, then clearly g is $\mathcal{S}^n_+$-convex (since g is linear), and we obtain linear matrix constraint in* (1.1)*: $C - \sum_{l=1}^p x_l A_l \succeq 0$. If addition F is linear, then we obtain an SDP in the dual form.*

Alternatively, in (P)*, let the variable be $X \in \mathcal{S}^n$, and let $F := \langle C, X\rangle + \delta_{\mathcal{S}^n_+}(X)$, $H \equiv \delta_b$, and $g(X) := \mathcal{A}(X)$, where $\mathcal{A}(X) := (\langle A_1, X\rangle, \ldots, \langle A_m, X\rangle)^\top$ is a linear operator, then we obtain an SDP in the primal form:*

$$\min_{X\in\mathcal{S}^n_+} \langle C, X\rangle \quad \text{s.t.} \quad \langle A_i, X\rangle = b_i, \quad i = 1, \ldots, m. \tag{1.3}$$

For detail of the primal-dual relationship and the notations used here, see Section 2.1.

SDPs are some of the most versatile, useful, and widespread optimization problems of the last three decades. They find applications in integer programming/combinatorial optimization

[1] Recall that a cone is proper if it is closed, convex, solid, and pointed.

[51, 82], polynomial optimization [73, 97], machine learning [72, 118, 119], and control theory [16, 127], to name just a few areas.

7. Linear program (LP). *In* (1.3)*, if all matrices are diagonal, then we obtain the LP:*

$$\min_{x\in\mathbb{R}^p} \; c^\top x \quad \text{s.t.} \quad Ax = b, \quad x \geq 0.$$

Clearly, it can also be obtained from (1.2)*.*

LP is perhaps the most classical type of optimization problems. Over a century, it has proven to be extremely powerful in areas such as transportation, energy, economics, and engineering, just to name a few. It is also used as subproblems in optimization problems that are nonlinear. □

Example 1.1 shows that the model (P) actually covers a very wide range of optimization problems. Given this fact, this thesis deals with two separate tasks:

- *Task 1:* To develop an algorithm to preprocess the SDP (1.3), before solving it using any given SDP solver;
- *Task 2:* To develop algorithms to solve the composite convex program (P), or its saddle-point form (SP).

We will further introduce the existing work, its limitations, and our achievements on these two tasks in Sections 1.2 and 1.3, respectively.

1.2 Existing Work and Our Contributions on Task 1: Preprocessing SDPs

Existing work. Given the wide applications of SDPs ((1.3) and its dual), several open-source or commercial solvers have been developed to solve them: SeDuMi [120], SDPT3 [134], PENNON [66], SDPA [47, 48], SDPNAL [149, 151], SDPNAL+ [123], and MOSEK [84], just to name a few. However, these solvers are usually slow and inaccurate for large SDPs, and erroneous for "messy" (e.g., non-strictly feasible or weakly infeasible) SDPs.

Therefore, it is useful to reduce the redundancy and to detect lack of strict feasibility of an SDP in the preprocessing stage. The resulting SDP would be smaller and cleaner, thus much easier for the solvers to handle, and to yield a more accurate solution with less computational effort. This

idea of reduction is called facial reduction [11, 12, 99, 100, 138], originally for more general conic linear programs, then specified for SDPs in [133]. It finds the face of PSD cone: $F \subseteq \mathcal{S}_+^n$ so that it contains the original feasible region: $F \cap \{X \mid \mathcal{A}(X) = b\} = \mathcal{S}_+^n \cap \{X \mid \mathcal{A}(X) = b\}$; since F is smaller, the reduced problem is easier to solve. However, finding such a face F is non-trivial, and sometimes as hard as solving the original SDP.

Thus, some simplified and implementable versions of facial reduction algorithms have been proposed, and theoretical studies have been conducted; see, e.g., [34, 35, 46, 67, 103, 104, 124].

Our contributions. In this thesis, we propose a version of facial reduction algorithm, called Sieve-SDP, which differs in several aspects from the above-mentioned algorithms:

- It is extremely simple, and it works in machine precision.
- Unlike the algorithms in [104], it does not rely on any optimization solver.
- We have developed Sieve-SDP as a software package in MATLAB, and thus it is ready to use and to be integrated into solvers that can be called from MATLAB.
- Finally, we present extensive computational results on general SDPs, which, to the best of our knowledge, are not yet available for such a simple algorithm.

1.3 Existing Work and Our Contributions on Task 2: Solving (P) or (SP)

Existing work. As shown in Example 1.1, the prototype (P) covers a very wide range of optimization problems, as broad as general finite-dimensional convex programming (Item 1), and as special as LP (Item 7). Therefore, we do not exhaust all the existing algorithms for each special type of problems, but refer to books [1, 5, 8, 9, 17, 44, 45, 93, 94, 109, 114, 116], some of which are classical textbooks, for a comprehensive treatment.

Here, we limit our brief review to the work that are closely related to ours, i.e., we focus on the the regime of *first-order primal-dual* algorithms for solving the composite problem (P) or its saddle-point form (SP). This deserves some remarks:

- *First-order algorithms.* Optimization algorithms tend to be iterative: analytical solution, in most cases, is either unavailable due to the complexity of objective function or the feasible region; or, the problem size is very large, and thus computing the analytical solution is

too expensive. Therefore, an optimization algorithm would start from an initial point x^0, then generate a sequence of iterates $\{x^k\}$, where k is the iteration counter, such that the corresponding objective values approach the true objective value.

In iteration k, the algorithm queries some information, called oracle or feedback, to proceed and generate the next iterate x^{k+1}. The so-called *first-order algorithms* are the algorithms that requires only the first-order information: the values, gradients, and proximal points of objective or constraint functions at current or previous iterative points. Unlike second-order methods, which requires computing (approximate) Hessians, first-order methods have much lower per-iteration complexity, which is favorable in the big data era, where many optimization problems are too large to allow the expensive computation of Hessians.

- *Primal-dual algorithms.* It is usually beneficial to design primal-dual algorithms for the saddle-point form, (SP), using duality theory. These algorithms provide primal iterates $\{x^k\}$ and dual iterates $\{y^k\}$, in order to (in a sense) decrease the primal objective value and increase the dual objective value. The problem (SP) is considered solved if the duality gap is sufficiently small.

With above remarks, we are ready to review the following limitations of existing related work:

- *Strong model assumptions.* Although some existing methods [57, 64, 88] apply to seemingly more general problems than (SP), i.e., when the coupling term $\Phi(x, y)$ has a more general form instead of just $\langle g(x), y\rangle$. However, strong assumptions are imposed - they require y in (SP) to be bounded, which excludes many important cases such as (1.1). Furthermore, many methods have been developed to solve only some special cases listed in Example 1.1, and thus they are not able to tackle the general template (P):
  - For Item 3 of Example 1.1 where g is linear, it is common to use splitting methods or smoothing techniques, see [20, 21, 25, 27, 28, 38, 39, 49, 52, 59, 60, 69–71, 85–87, 90, 91, 106, 107, 125, 126, 128, 132, 135, 150], or books such as [50] for more comprehensive review. The most well-known algorithm in this category is perhaps the so-called alternating direction method of multipliers, or ADMM [15, 32, 37, 53, 80], although its performance varies depending on different applications.
  - There are few algorithms dedicated to the cone program (1.1) of Item 4. More often,

the existing literature discusses the special case when g is linear (see above), or when $\mathcal{K}$ is special (see below).

  - For Item 5, see, e.g., [8, 76–78, 115, 144–147].
  - SDP in Item 6 is often solved using second-order methods such as interior-point methods; see Section 1.2. There exist first-order methods [68, 141], which are often applications of the general solvers for Item 3.
  - LP in Item 7, due to its simple form, can be solved using zero-order methods. However, in the big data era, first-order methods as mentioned above are more and more popular when solving large-scale LPs.

- *High per-iteration complexity.* Several methods, including [79, 148], require double loops, where the inner loop approximately solves an inner problem, e.g., the maximization problem in y, and the outer loop handles the minimization problem; hence, the complexity of each outer iteration is often high. Another drawback of using double-loop is that the involved parameters are hard to tune.
- *Inconsistency between theoretical convergence guarantees and empirical solutions.* Existing work [57, 64, 88, 92, 144] for general problem (P) can only show the convergence rates on the *ergodic* (or, the *averaging*) sequences $\{\bar{x}^k\}$ defined as

$$\bar{x}^k := \frac{1}{k}\sum_{j=1}^{k} x^j,$$

  or its weighted variants, via, e.g., primal objective residual: $\mathcal{P}(\bar{x}^k) - \mathcal{P}^\star = \mathcal{O}\left(\frac{1}{k}\right)$, where $\mathcal{P}(x)$ is the objective value of problem (P) at point $\bar{x}^k$, and $\mathcal{P}^\star$ is the optimal objective value. The primal-dual gap function is also often used. However, in practical implementation, researchers and users often adopt the *non-ergodic* (or, the *last-iterate*) sequence, which is the $\{x^k\}$ itself, output by the algorithm after each iteration;[2] it is because:

  - The non-ergodic iterates often perform better (i.e., converge faster) empirically;
  - By taking the average over the past iterates, the ergodic iterates destroys the special structures sometimes desired of the solution such as sparsity in feature selection, sharp-

[2]In the literature, "ergodic" and "averaging" are used interchangeably; same goes for "non-ergodic" and "last-iterate". In this thesis, we will mainly use "ergodic", "non-ergodic", and sometimes "semi-ergodic".

edgedness in images, and low-rankness in matrix approximation.

Unfortunately, as far as we know, there have not been an algorithm in the literature for the general problem (P) with theoretical convergence guarantees based on the last-iterate sequence $\{x^k\}$.

Our contributions. Faced with the above limitations in the existing literature, we develop two first-order primal-dual algorithms with the following features:

- *Mild model assumptions.* Our algorithms are designed for the general problem (P) or its saddle-point form (SP), and with mild assumptions. Therefore, all special cases listed in Example 1.1 are covered.
- *Low per-iteration complexity.* Our algorithms has only single-loop, thus the per-iteration complexity is low: They only require gradient computations, proximal operations, and function evaluations, at most twice each in each iteration.
- *Consistency between theoretical convergence guarantees and empirical solutions.* Our algorithms have non-ergodic primal and ergodic dual convergence rates. It means that for problem (P), we have convergence guarantees on $\mathcal{P}(x^k) - \mathcal{P}^\star$, which is consistent with the solution for practical use. If g is linear, than by seeing the dual as primal, we would also have non-ergodic rate on the dual.
- *Optimal and faster convergence rates.* When problem size p satisfies $k = \mathcal{O}(p)$, our *non-ergodic* convergence rate is the *optimal* $\mathcal{O}\left(\frac{1}{k}\right)$, which can be achieved by existing algorithms only via *ergodic* sequence. When $k > \mathcal{O}(p)$, our method achieves an even faster $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ convergence rate. When F in (P) is strongly convex, we can boost this rate to $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$.

1.4 Outline

This thesis is organized as follows.

In Chapter 2, we address Task 1 introduced in Section 1.2, and describe the algorithm Sieve-SDP to preprocess SDPs, and present the numerical experiments. This chapter is based on the following paper:

[153] Yuzixuan Zhu, Gábor Pataki, and Quoc Tran-Dinh. Sieve-SDP: A simple facial reduction algorithm to preprocess semidefinite programs. *Mathematical Programming Computation,* 11(3):503–586, 2019.

In Chapter 3, we address Task 2 introduced in Section 1.3, and propose the two first-order primal-dual algorithms to solve composite convex programs. This chapter is based on the following two papers:

[130] Quoc Tran-Dinh and Yuzixuan Zhu. Non-stationary first-order primal-dual algorithms with fast non-ergodic convergence rates. *arXiv preprint arXiv:1903.05282,* 2020. Accepted by *SIAM Journal on Optimization*.

[152] Yuzixuan Zhu, Deyi Liu, and Quoc Tran-Dinh. Accelerated primal-dual algorithms for a class of convex-concave saddle-point problems with non-bilinear coupling term. *arXiv preprint arXiv:2006.09263,* 2020. Submitted to *Mathematical Programming*.

Here, [130] discusses the case where g is linear in (P), and developes some important techniques. Then, [152] generalizes the theory and algorithms in [130] for nonlinear g. Since [152] is more general, it is the major component of Chapter 3.

Finally, we put supplemental content for Chapters 2 and 3, such as extended experiment results and technical proofs, into Appendices A and B, respectively.

CHAPTER 2

SIEVE-SDP: A SIMPLE FACIAL REDUCTION ALGORITHM TO PREPROCESS SEMIDEFINITE PROGRAMS

2.1 Introduction

We introduce Sieve-SDP, a simple facial reduction algorithm to preprocess semidefinite programs (SDPs). Sieve-SDP inspects the constraints of the problem to detect lack of strict feasibility, deletes redundant rows and columns, and reduces the size of the variable matrix. It often detects infeasibility. It does not rely on any optimization solver: the only subroutine it needs is Cholesky factorization, hence it can be implemented in a few lines of code in machine precision. We present extensive computational results on several problem collections from the literature, with many SDPs coming from polynomial optimization.

This chapter is based on [153], a joint work with Dr. Gábor Pataki and Dr. Quoc Tran-Dinh.

2.1.1 Problem statement and the preprocessing algorithm

Consider an SDP in the form

$$\begin{aligned} \inf_X \quad & C \cdot X \\ \text{s.t. } & A_i \cdot X = b_i, \quad i = 1, \ldots, m, \\ & X \succeq 0, \end{aligned} \tag{P}$$

where the A_i and C are $n \times n$ symmetric matrices, the b_i are scalars, $X \succeq 0$ means that X is in $\mathcal{S}^n_+$, the set of symmetric, positive semidefinite (PSD) matrices, and the $\cdot$ inner product of symmetric matrices is the trace of their regular product, which is also the sum of element-wise products. Sometimes, the inner product $C \cdot X$ is also written as $\langle C, X \rangle$.

SDPs are some of the most versatile, useful, and widespread optimization problems of the last three decades. They find applications in control theory, integer programming, and combinatorial

optimization, to name just a few areas. Several good solvers are available to solve SDPs: see for example SeDuMi [120], SDPT3 [134], PENNON [66], SDPA [47, 48], SDPNAL [149, 151], SDPNAL+ [123], and MOSEK [84].

SDPs – as all optimization problems – often have redundant variables and/or constraints. The redundancy we address is lack of *strict feasibility*, i.e., when there is no feasible positive definite X in (P). Figure 2.1 shows the feasible region of a 2-by-2 SDP where the linear space does not pass through the interior of the PSD cone. When (P) is not strictly feasible, the optimal value of (P) and of its dual may differ, and the latter may not be attained.[1] Hence, when attempting to solve such an SDP, solvers often struggle or fail.

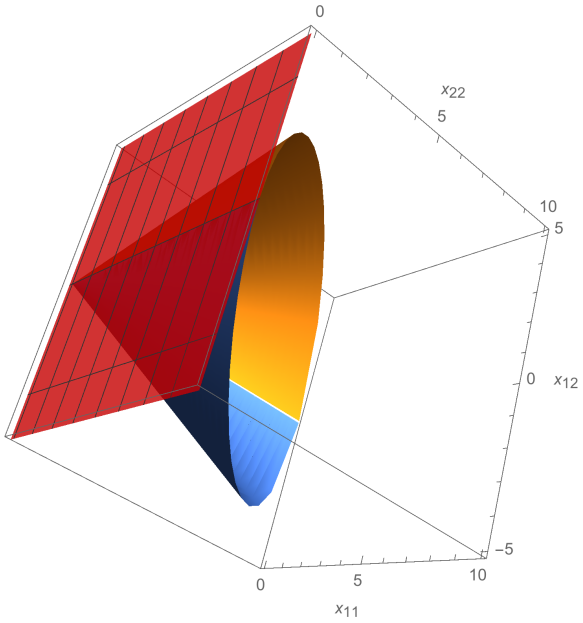

Figure 2.1: The feasible region of a non-strictly feasible SDP

It is, therefore, useful to detect lack of strict feasibility in a preprocessing stage. This chapter describes a very simple preprocessing algorithm for SDPs, called Sieve-SDP, which belongs to the class of facial reduction algorithms [12, 34, 35, 67, 99, 100, 104, 133, 138]. Sieve-SDP can detect lack of strict feasibility, reduce the size of the problem, and can be implemented in a few lines of code in machine precision.

To motivate our algorithm, let us consider an example.

[1] More precisely, when (P) *is* strictly feasible, *strong duality* holds between (P) and its dual, i.e., their values agree and the latter is attained.

Example 2.1. *The SDP instance (with an arbitrary objective function)*

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \cdot X = 0, \qquad \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \cdot X = -1, \qquad X \succeq 0, \tag{2.1}$$

is infeasible. Indeed, suppose $X = (x_{ij})_{i,j=1}^{3}$ *is feasible in* (2.1)*. Then* $x_{11} = 0$*, hence the first row and column of* X *are zero by PSD-ness, so the second constraint implies that* $x_{22} = -1$*, which is a contradiction.* □

Note that if we replace -1 in the second constraint of (2.1) by a positive number, then (2.1) can be restated over the set of PSD matrices with first row and column equal to zero. Thus, even if we do not detect infeasibility, such preprocessing is still useful.

Our algorithm Sieve-SDP repeats the Basic Step shown in Figure 2.2. Hereafter $D \succ 0$ means that a symmetric matrix D is positive definite.

BASIC STEP

1. Find $i \in \{1, \ldots, m\}$ (if any) such that the i-th constraint of (P), after permuting rows and columns, and possibly multiplying both sides by -1, is of the form

$$\begin{pmatrix} D_i & 0 \\ 0 & 0 \end{pmatrix} \cdot X = b_i, \tag{2.2}$$

   where $D_i \succ 0$, and $b_i \leq 0$. If there is no such i, STOP; problem (P) cannot be preprocessed further.

2. If $b_i < 0$, then STOP; (P) is infeasible.

3. If $b_i = 0$, then delete this constraint. Also delete all rows and columns in the other constraints that correspond to rows and columns of D_i.

Figure 2.2: The Basic Step of Sieve-SDP

Example 2.2 (Example 2.1 continued)**.** *When we first execute the Basic Step on* (2.1)*, we find the first constraint, delete it, and also delete the first row and column from the second constraint matrix. Next, we find the constraint*

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \cdot X = -1,$$

and declare that (2.1) *is infeasible.* □

We call our algorithm in Figure 2.2 Sieve-SDP, since by shading the deleted rows and columns in the variable matrix X (and the coefficient matrices A_i's), we obtain a sieve-like structure, as shown in Figure 2.3.

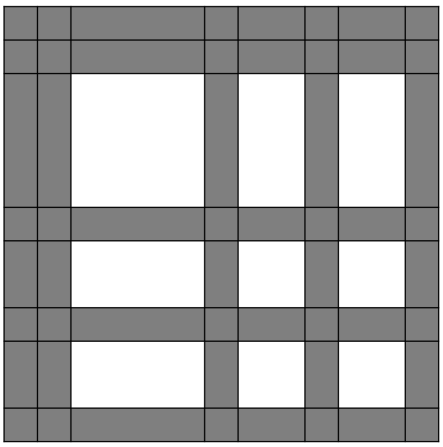

Figure 2.3: The sieve structure

Sieve-SDP is easy to implement: it only needs an incomplete Cholesky factorization subroutine to check positive definiteness. We can delete rows and columns using fast matrix operations. Even the worst case complexity of Sieve-SDP is reasonable: an easy calculation shows that it can fully preprocess (P) using $\mathcal{O}(\min\{m,n\}n^3m)$ arithmetic operations.

Sieve-SDP is heuristic: it does not always detect infeasibility, or lack of strict feasibility. For example, it would not work on problem (2.1), if we apply a similarity transformation $T^\top(\cdot)T$ to all A_i's, where T is a random invertible matrix.

Given the simplicity of Sieve-SDP, it is natural to ask whether it can work in practice. Precisely, the main question we address is: Can Sieve-SDP help us compute more accurate solutions and reduce the computing time on a broad range of SDPs?

We will answer this question in the affirmative.

2.1.2 Related work

Our algorithm belongs to the family of *facial reduction algorithms*, which we now describe. If (P) is not strictly feasible, one can replace the constraint $X \in \mathcal{S}^n_+$ by

$$X \in F,$$

where F is a proper face of $\mathcal{S}_+^n$.[2] Since any such face can be written as (see e.g. [98])

$$F = V \mathcal{S}_+^r V^\top, \tag{2.3}$$

where $r < n$, and V is an $n \times r$ matrix, the reduced problem can be restated over a smaller PSD cone. Facial reduction algorithms – for more general conic programs – originated in the papers [11, 12]. Later simplified, more easily implementable variants were given in [99, 100, 138], and in [133] for the SDP case. A recent, very concise version with a short proof of convergence is in [81].

Facial reduction algorithms, when applied to (P), find face F by solving a sequence of SDP subproblems, which may be as hard to solve as (P) itself. Thus one is led to seek simpler alternatives.

Simplified and implementable versions of facial reduction is described in [104], where the algorithms reduce the feasible set of (P) (or of an SDP in a different shape) by solving linear programs (LPs) instead of SDPs. Thus they do not find *all* reductions, but still simplify the SDPs in many cases. They are available as public domain codes, and we will compare them with Sieve-SDP in Section 2.2. A facial reduction algorithm embedded in an interior point method was implemented in [103].

The idea of reducing SDPs by simply inspecting constraints appears in several papers. For example, [46] notes that if $A \cdot X = 0$ is a constraint in (P) with $A \succeq 0$, then we can restrict X to belong to a face of the form (2.3), where V spans the nullspace of A. A similar idea was used in [67] to reduce Euclidean distance matrix completion problems. For a rigorous derivation of the algorithm in [67], see [35], which used an intermediate step of analyzing the semidefinite completion problem. For follow-up work, see [34] on the noisy version of the same problem; and [124] for a more theoretical study.

We finally mention two very accurate SDP solvers, which do not rely on facial reduction. The first is SDPA-GMP [47], which uses the GMP library and computes solutions of (P) and of its dual using several hundred digits of accuracy. We will use SDPA-GMP in Subsection 2.3.6 to check accuracy of the solutions computed by Sieve-SDP and MOSEK. Another solver is SPECTRA [63],

[2] That is, $F \neq \mathcal{S}_+^n$, F is convex, and $X, Y \in \mathcal{S}_+^n$, $\frac{1}{2}(X+Y) \in F$ implies that X and Y are in F.

which computes a feasible solution of (P) in exact arithmetic. Although these methods cannot handle large SDPs, they can solve small ones accurately.

2.1.3 Our contributions

Sieve-SDP differs in several aspects from the above-mentioned algorithms:

- It needs only Cholesky factorization as a subroutine, and, unlike the algorithms in [104], it does not rely on any optimization solver.
- It detects very simple redundancies, which are easy to explain even to a user not trained in optimization, and can help him/her better formulate other problems.
- As soon as Sieve-SDP finds a reducing constraint, it deletes this constraint, and it also deletes the corresponding rows and columns from the constraint matrices. Hence errors do not accumulate. Thus Sieve-SDP is as accurate as Cholesky factorization, which works in machine precision [131, Theorem 23.2].
- Sieve-SDP can also detect infeasibility.
- It is easy to run in a *safe mode* (explained in Section 2.2) to even better safeguard against numerical errors.
- Finally, we present extensive computational results on general SDPs, which, to the best of our knowledge, are not yet available for such a simple algorithm.

2.1.4 Chapter organization

The rest of this chapter is organized as follows. In Section 2.2 we describe how we implemented Sieve-SDP, the computational setup, and the criteria for comparison with competing codes. In this section, we also give a small SDP with a positive duality gap in Example 2.3, and show how to construct a pair of primal-dual solutions with arbitrarily small constraint violation and arbitrarily small duality gap.

In Section 2.3 we comment in detail on the results on some of the problems, and on the strengths and weaknesses of the preprocessors. For example, we examine whether they help to find the correct solution of numerically difficult SDPs, and how fast they are on large-scale problems.

In Section 2.4 we summarize the preprocessing results, and conclude the paper.

We have four appendices. In Appendix A.1 we present very detailed computational results on all problems. In Appendix A.2 we give the core MATLAB code of Sieve-SDP, containing only about 40-80 lines. In Appendix A.3 we provide the definition of the DIMACS errors for completeness. In Appendix A.4 we discuss the issue of recovering an optimal solution of the dual of (P) from the optimal solution of the dual of the reduced problem.

2.2 The Setup for Numerical Experiments

Implementation and computing. We implemented Sieve-SDP in MATLAB R2015a, using the standard Cholesky factorization (subroutine `chol`) to check positive definiteness. We ran both Sieve-SDP and the competing preprocessors on a MacBook Pro with processor Intel Core i5 running at 2.7GHz, and with 8GB of RAM.

Safe mode. To safeguard against numerical errors, we use a *safe mode*. We set

$$\varepsilon := 2^{-52} \approx 2.2204 \cdot 10^{-16},$$

which is the machine precision in MATLAB. In the Basic Step in Figure 2.2, if we find a constraint of type (2.2), then, instead of checking $b_i < 0$ we check whether

$$b_i < -\sqrt{\varepsilon}\max\{1,\ \|b\|_\infty\}.$$

If this test fails, then instead of checking $b_i = 0$ we check whether

$$b_i > -\varepsilon\max\{1,\ \|b\|_\infty\}.$$

This step is correct, because in the Basic Step we already ensured $b_i \leq 0$.

Preprocessors used for comparison. We compare Sieve-SDP with the algorithms proposed by Permenter and Parrilo in [104]. Their algorithms solve LP subproblems to reduce the size of an SDP. They can work either on the problem (P), which we call the *primal*; or on its *dual*:

$$\begin{aligned} \sup_y \quad & \sum_{i=1}^m y_i b_i \\ \text{s.t.} \quad & \sum_{i=1}^m y_i A_i \preceq C. \end{aligned} \tag{D}$$

They can use either diagonal, or diagonally dominant reductions (for details, see [104]).

Thus, there are four algorithms from [104] that we tested: pd1, pd2, dd1 and dd2. Here, pd1 stands for primal diagonal; pd2 for primal diagonally dominant; dd1 for dual diagonal; and dd2 for dual diagonally dominant.

Remark 2.1. *In the theoretical description of the preprocessing in [104] the SDP called the primal is actually our dual* (D)*. However, in their implementation and their code posted on the github website, their primal is the same as our primal* (P)*.*

The datasets. We tested Sieve-SDP and competing methods on five datasets, which contain 771 problems overall:

- The Permenter-Parrilo (PP) dataset from [104], which contains 68 problems originally from [4, 14, 18, 24, 31, 42, 102, 108, 110, 136, 137, 139]. Although a few problems in this dataset are randomly generated, most come from applications. This dataset contains problems that are notoriously difficult for SDP solvers, and some are known to be not strictly feasible. We have excluded two problems from [104]: "copos_5" and "cprank_3", since they were too large to be solved by MOSEK on our computer.
- The Mittelmann dataset obtained from Hans Mittelmann's website, which we call the Milttelmann dataset. It contains 31 problems.
- The Dressler-Illiman-de Wolff (DIW) dataset, a collection of SDP relaxations of polynomial optimization based on the paper [33]. It contains 155 problems.
- The Henrion-Toh dataset kindly provided to us by Didier Henrion and Kim-Chuan Toh. It contains 98 problems.
- The Toh-Sun-Yang dataset kindly provided to us by Kim-Chuan Toh. It contains 419 problems, whose description is in [122] and [149].

Our datasets contain many different types of SDPs and, not surprisingly, the performance of the preprocessors on them varies widely. Many SDPs that come from applications may be strictly feasible, and on these SDPs even more sophisticated preprocessors would not find reductions. For example, on the Toh-Sun-Yang dataset the preprocessors did not find any reductions. However, Sieve-SDP and pd1 only took a negligible amount of time to deliver the "no reduction found" result, so it did not hurt to preprocess.

Yet, even in the datasets other than the PP dataset, many SDPs *were* reduced by some preprocessor. In the Henrion-Toh dataset, pd1, pd2, and Sieve-SDP all reduced 18 problems, whereas dd1 and dd2 reduced none. In the Mittelmann dataset, pd1, pd2, and Sieve-SDP reduced 8 problems; dd1 and dd2 reduced none.

Strikingly, in the DIW dataset Sieve-SDP proved infeasibility of 59 problems out of 155, and reduced total solving time by a factor of more than a hundred! The method pd1 did slightly worse.

For illustration, we refer to Figure 2.4, which shows the size and sparsity structure of the problem "ex4.2_order20"[3] before (on the left) and after (on the right) applying Sieve-SDP. Each row in corresponds to an A_i matrix stretched out as a vector. Red dots correspond to positive entries, blue dots correspond to negative entries, and white areas correspond to zero entries.

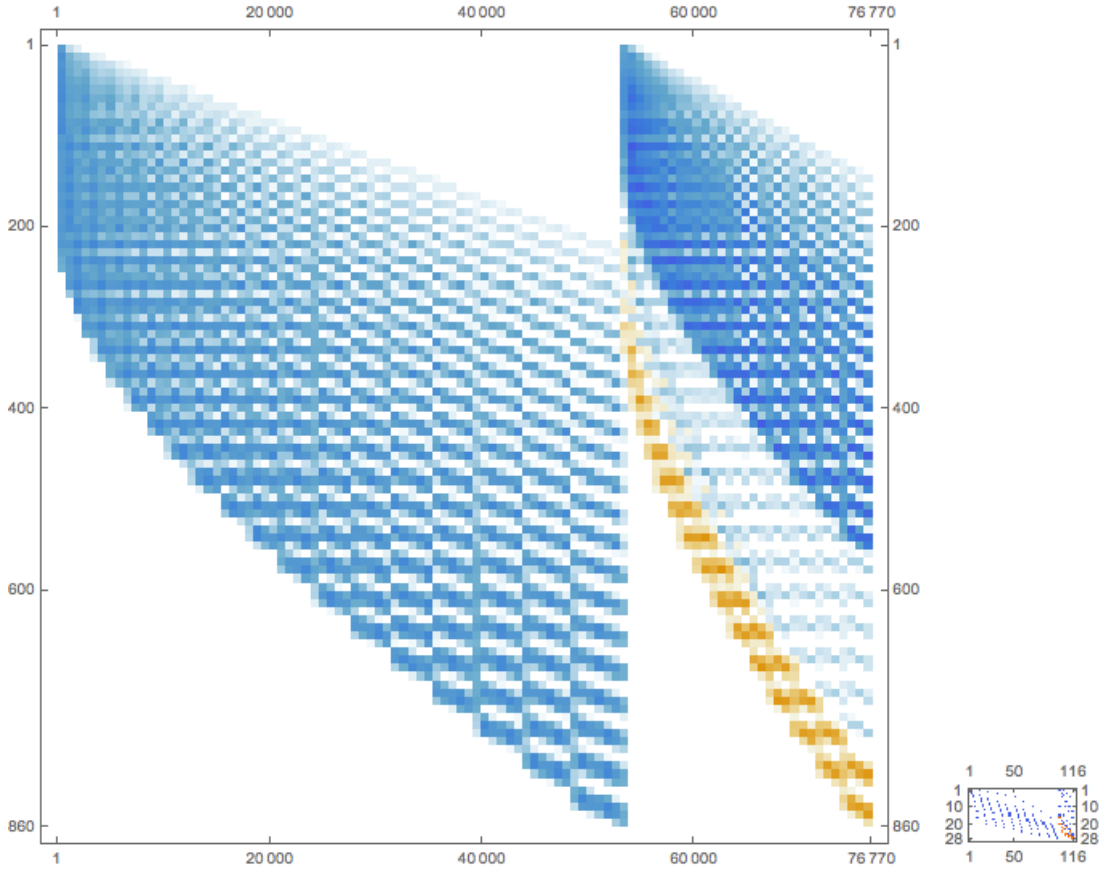

Figure 2.4: Problem "ex4.2_order20": size and sparsity before and after preprocessing

Internal format and input/output format. Internally we store the A_i matrices as an $n \times (nm)$

[3]This SDP is from the DIW dataset.

sparse matrix of the form

$$\begin{pmatrix} A_1 & A_2 & \cdots & A_m \end{pmatrix}$$

(i.e., the A_i's are stored side-by-side), and C as an $n \times n$ sparse matrix. The input and the output format of the preprocessors is the widely used Mosekopt format.

The choice of the SDP and LP solver. For all preprocessors we use MOSEK 8.1.0.27 (from now on, simply "MOSEK") as SDP solver: we solve the SDPs with MOSEK before and after preprocessing. We also solve the LP subproblems in the algorithms of [104] by MOSEK. We consider MOSEK as the best choice, since it is a reliable commercial SDP and LP solver, and it is being actively developed and improved.

Our settings are different from the ones used in [104], where SeDuMi [120] format is used as input format, MOSEK as LP solver, and SeDuMi as SDP solver. With our settings, the algorithms of [104] work faster, because MOSEK is much faster than SeDuMi. Although we must convert the data from Mosekopt format to SeDuMi format (to do the preprocessing), and then back (to solve the preprocessed problem with MOSEK), the total conversion time is negligible: for each of pd1, pd2, dd1 and dd2 it is less than 100 seconds on *all* 771 SDPs. To be fair, in the detailed comparison tables of Appendix A.1 we list conversion time and preprocessing time separately.

Criteria for comparison. Let us recall the main question addressed in the paper: Can Sieve-SDP help us compute more accurate solutions and reduce the computing effort on a broad range of SDPs?

To answer this question, we propose the following criteria, in order of priority:

1. *Does preprocessing help detect infeasibility? If not, does it help find a correct optimal solution?* Precisely, suppose MOSEK reports an incorrect optimal value of an SDP before preprocessing. Does MOSEK find a correct optimal value after preprocessing (assuming that the optimal value of the SDP is known analytically)?
2. *Does preprocessing reduce computing time?* This criterion is secondary, since preprocessing is often essential to computing an accurate solution: see Subsections 2.3.1 through 2.3.3. Thus, we believe that we should always do preprocessing, as long as it is with very high precision, even if preprocessing *increases* the solving time.

3. *Does preprocessing improve numerical accuracy measured by the six DIMACS errors [102]?* Precisely, let

$$\mathrm{DIMACS}_{\mathrm{before}} \quad \text{and} \quad \mathrm{DIMACS}_{\mathrm{after}}$$

be the largest absolute value of the DIMACS errors before and after preprocessing, respectively. We say that a method *improves* the DIMACS error, if it does not detect infeasibility, and

$$\mathrm{DIMACS}_{\mathrm{before}} > 10^{-6} \quad \text{and} \quad \frac{\mathrm{DIMACS}_{\mathrm{after}}}{\mathrm{DIMACS}_{\mathrm{before}}} < \frac{1}{10}.$$

This last criterion must be taken with a grain of salt. While the DIMACS errors are very natural (they measure constraint violation and duality gap), Example 2.3 below shows that they do not always measure accurately how good a solution is. In fact, a *larger* DIMACS error may correspond to a *better* solution!

Example 2.3. *Consider the SDP*

$$\begin{aligned} \min_X \quad & \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \cdot X \\ \text{s.t.} \quad & \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \cdot X = 0, \quad \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \cdot X = 1, \quad X = (x_{ij}) \succeq 0, \end{aligned} \tag{2.4}$$

and its dual

$$\begin{aligned} \max_y \quad & y_2 \\ \text{s.t.} \quad & y_1 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} + y_2 \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \preceq \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \end{aligned} \tag{2.5}$$

We claim that the duality gap between them is 1. *Indeed, let* X *be a feasible solution of* (2.4). *Since* $x_{11} = 0$, *the first row and column of* X *must be zero, hence*

$$X = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

is an optimal solution with objective value 1. *In turn, in* (2.5) *we have* $y_2 = 0$ *for all feasible* y, *so its optimal value is* 0.

Next, let $\varepsilon > 0$ *and define* $M_\varepsilon > 0$ *so that*

$$X_\varepsilon := \begin{pmatrix} \varepsilon & 0 & \frac{1-\varepsilon}{2} \\ 0 & \varepsilon & 0 \\ \frac{1-\varepsilon}{2} & 0 & M_\varepsilon \end{pmatrix}$$

is positive semidefinite. Then X_ε *is an approximate solution of* (2.4), *which violates only the first constraint (by* ε*) and has objective value* 2ε.

If we feed the pair (2.4)-(2.5) *to MOSEK, it returns a solution with DIMACS errors*

$$0.5000,\ 0,\ 0.7071,\ 0,\ -5.5673 \times 10^{-9},\ 5.9077 \times 10^{-17}.$$

The first and third errors are large, so we cannot conclude the problem has been "solved".

However, let us apply a similarity transformation $T^\top(\cdot)T$ *to all matrices in* (2.4) *with*

$$T = \begin{pmatrix} 3 & 5 & -2 \\ 4 & 1 & 1 \\ -4 & -4 & 5 \end{pmatrix}.$$

Then the resulting primal-dual pair still has a duality gap of 1. *However, MOSEK now returns a solution with DIMACS errors*

$$1.6093 \times 10^{-6},\ 0,\ 5.2111 \times 10^{-9},\ 3.287 \times 10^{-12},\ -8.1484 \times 10^{-5},\ 3.0511 \times 10^{-5},$$

which may seem "essentially all zero" to a user. □

Such "fake" solutions can arise in any SDP pair with positive duality gap. Indeed, suppose

$$\mathcal{D}^\star < \mathcal{P}^\star,$$

where $\mathcal{P}^\star$ is the optimal objective value of (P), and $\mathcal{D}^\star$ is that of (D). Then by the theory of asymptotic duality [112, Chapter 3], there is a sequence $\{X_\varepsilon \succeq 0 \mid \varepsilon > 0\}$ such that X_ε violates each primal constraint by at most ε, and

$$\langle C, X_\varepsilon \rangle \to \mathcal{D}^\star \quad \text{as} \quad \varepsilon \to 0.$$

As Example 2.3 shows, such "fake" or approximate solutions are sometimes indeed found by some SDP solvers.

We note that [24] also presented computational results on SDPs with positive duality gaps, and noted that SeDuMi often gave an incorrect solution on such problems. However, [24] did not report the DIMACS errors.

2.3 Detailed Comments on Some Preprocessing Results

We now report in detail how the preprocessors perform on some of the problems. We examine them from several angles: for example, can they help to find known optimal solutions of difficult SDPs? How do they perform on large-scale SDPs? How fast are they when they do not reduce an SDP by much, or at all?

We first look at how the preprocessors perform on the "Compact", "unbound", and "Example" problems, for which the exact optimal values are known, but are hard to compute. (These problems are from the PP dataset.) We examine whether preprocessing helps to find these optimal values.

We note that Sieve-SDP does not change the optimal value of (P), since it deletes rows and columns from the variable matrix X that are always zero anyway. However, when it deletes rows and columns in the constraint matrices, in the dual (D) we require only a principal minor of $C - \sum_{i=1}^m y_i A_i$ to be PSD. Thus applying Sieve-SDP may increase the optimal value of (D).

To make this argument more precise, let us write ($\mathrm{P_r}$) and ($\mathrm{D_r}$) for the primal and dual problems after preprocessing by Sieve-SDP, respectively. Then,

$$\mathcal{D}^\star \leq \mathcal{D}_\mathrm{r}^\star \leq \mathcal{P}_\mathrm{r}^\star = \mathcal{P}^\star, \tag{2.6}$$

where $\mathcal{P}_{\mathrm{r}}^\star$ and $\mathcal{D}_{\mathrm{r}}^\star$ are optimal objective values of ($\mathrm{P_r}$) and ($\mathrm{D_r}$), respectively. For example, in Example 2.3 Sieve-SDP deletes the first row and first column in all constraint matrices, and it is easy to check that the corresponding optimal values are $0 < 1 = 1 = 1$, respectively. In detail, for this example ($\mathrm{D_r}$) is

$$\begin{array}{ll} \sup_{y_2} & y_2 \\ \text{s.t.} & y_2 \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \preceq \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \end{array} \tag{2.7}$$

whose optimal value is 1. On the other hand, suppose $\mathcal{P}^\star = \mathcal{D}^\star$. Then (2.6) implies that Sieve-SDP changes neither the primal, nor the dual optimal values.

We should also expect the primal optimal value (but not that of the dual) to remain the same, if we preprocess (P) by pd1 and pd2, since these algorithms also reduce the primal. On the other hand, algorithms dd1 and dd2 reduce the dual problem (D), so they keep the optimal value of the dual (D) the same. However, they may change the optimal value of the primal (P).

In all tables we use the following convention: among the reported objective values the first is the primal and the second is the dual.

2.3.1 "Compact" problems – 10 problems from [137]

These instances are *weakly infeasible*, i.e., the affine subspace

$$H = \{X \mid A_i \cdot X = b_i,\ i = 1, \ldots, m\}$$

does not intersect $\mathcal{S}_+^n$, but the distance of H to $\mathcal{S}_+^n$ is zero. Weakly infeasible SDPs are particularly challenging to SDP solvers. However, we refer to a recent algorithm in [63], which can detect (in)feasibility of small SDPs in exact arithmetic; and to [81] for an algorithm that is tailored to detect weak infeasibility.

On these problems pd1 and pd2 produced the same results, while dd1 and dd2 reduced none of them; pd1 and pd2 combined with MOSEK correctly detected primal infeasibility of all problems, while Sieve-SDP correctly proved their primal infeasibility without using MOSEK. (Since it found the primal infeasibility, we did not compute a dual solution.)

The results are in Table 2.1.

Table 2.1: Results on the "Compact" problems

Problem	Correct obj (P, D)	Before prep.	After pd1/pd2	After dd1/dd2	After Sieve-SDP
CompactDim2R1	Infeas, $+\infty$	3.79e+06, 4.20e+06	Infeas, 1	3.79e+06, 4.20e+06	Infeas, -
CompactDim2R2	Infeas, $+\infty$	6.41e-10, 6.81e-10	Infeas, 2	6.41e-10, 6.81e-10	Infeas, -
CompactDim2R3	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
CompactDim2R4	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
CompactDim2R5	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
CompactDim2R6	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
CompactDim2R7	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
CompactDim2R8	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
CompactDim2R9	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
CompactDim2R10	Infeas, $+\infty$	1.5, 1.5	Infeas, 2	1.5, 1.5	Infeas, -
correctness %	100%, 100%	0%, 0%	100%, 0%	0%, 0%	100%, -

We mention here another set of (weakly) infeasible SDPs. They were presented in [81], and are available from `http://gaborpataki.web.unc.edu/infeasible-and-weakly-infeasible-sdps/`. Some of these SDPs are classified as "clean" and some of them as "messy". In the "clean" instances the structure that proves infeasibility is apparent, while in the "messy" instances that structure was obscured by two kinds of operations: random elementary row operations on the constraints and a random similarity transformation.

Indeed, in our testing all clean instances were found infeasible by Sieve-SDP, pd1 and pd2. In contrast, no messy instances were reduced by any of the preprocessing methods.

2.3.2 "Unbound" problems – 10 problems from [139]

The mathematically correct optimal values of both the primal and the dual are 0 in this problem collection. However, before preprocessing MOSEK returned wrong optimal values for 6 out of 10 problems. Although MOSEK found solutions with almost correct optimal value in problems 2, 3 and 4, these solutions are inaccurate, as the DIMACS errors are of the order 10^{-1} (this is marked by "*" symbols in Table 2.2).

In summary, 9 out of 10 problems in this dataset need preprocessing to obtain a reasonable solution. Sieve-SDP, pd1 and pd2 corrected all objective values, as Table 2.2 shows.

The authors in [139] computed the correct optimal solution of these instances using SDPA-GMP [47], a high-precision SDP solver that carries several hundred significant digits. However, doing so is more time consuming than running MOSEK combined with Sieve-SDP.

Table 2.2: Results on the "unbound" problems

Problem	Correct obj (P, D)	Before prep.	After pd1/pd2	After dd1/dd2	After Sieve-SDP
unboundDim1R1	0, 0	1.33e-09, -7.05e-10	1.33e-09, -7.05e-10	1.33e-09, -7.05e-10	0, 0
unboundDim1R2	0, 0	-8.19e-15*, -8.01e-15*	0, 0	-8.19e-15*, -8.01e-15*	0, 0
unboundDim1R3	0, 0	-2.04e-11*, -2.02e-11*	0, 0	-2.04e-11*, -2.02e-11*	0, 0
unboundDim1R4	0, 0	-2.34e-10*, -2.32e-10*	0, 0	-2.34e-10*, -2.32e-10*	0, 0
unboundDim1R5	0, 0	-1, -1	0, 0	-1, -1	0, 0
unboundDim1R6	0, 0	-1, -1	0, 0	-1, -1	0, 0
unboundDim1R7	0, 0	-1, -1	0, 0	-1, -1	0, 0
unboundDim1R8	0, 0	-1, -1	0, 0	-1, -1	0, 0
unboundDim1R9	0, 0	-1, -1	0, 0	-1, -1	0, 0
unboundDim1R10	0, 0	-1, -1	0, 0	-1, -1	0, 0
correct%	100%, 100%	10%, 10%	100%, 100%	10%, 10%	100%, 100%

2.3.3 "Example" problems – 8 problems from [24]

The mathematically correct objective values are reported in [24, Table 12.1]. (In [24], our primal is considered as the dual, and vice versa, so that table must be read accordingly.)

Table 2.3 shows the objective values before and after preprocessing. We consider an objective value correct if it is less than 10^{-6} away from the true optimal value.

Table 2.3: Results on the "Example" problems

Problem	Correct (P, D)	Before prep.	After pd1/pd2	After dd1/dd2	After Sieve-SDP
Example1	0, 0	0, 0	0, 0	0, 0	0, 0
Example2	1, 0	3.33e-01, 3.33e-01	1, 1	4.73e-15, 1.82e-14	1, 1
Example3	0, 0	3.33e-01, 3.33e-01	1.17e-07, 1.69e-07	4.73e-15, 1.82e-14	1.17e-07, 1.69e-07
Example4	Infeas, 0	Infeas, 3.74e-07	Infeas, 1	0, 0	Infeas, -
Example6	1, 1	1, 1	1, 1	1, 1	1, 1
Example7	0, 0	0, 0	0, 0	0, 0	0, 0
Example9size20	Infeas, 0	Infeas, 3.39e-01	Infeas, 1	0, 0	Infeas, -
Example9size100	Infeas, 0	Infeas, 3.43e-01	Infeas, 1	0, 0	Infeas, -
correctness %	100%, 100%	75%, 50%	100%, 50%	50%, 100%	100%, 50%

We excluded "Example5" of [24] from this table, since in [24, Table 12.1] the optimal value is not reported. For all other problems, except for "Example9size20" and "Example9size100", we manually verified the correctness of the optimal values in exact arithmetic.

Note that the comparison in Table 2.3 is somewhat unfair to Sieve-SDP: if it found a problem infeasible, it did not compute a dual solution.

2.3.4 "Finance" problems – 4 problems from [14]

The PP dataset contains four "finance" problems: "leverage_limit", "long_only", "sector_neutral" and "unconstrained". We report on these problems in detail, since these are the largest in the PP dataset. For example, "long_only" has 100 semidefinite variable blocks of order 91 and another 100 of order 30.

Table 2.4 shows how much the preprocessors reduced these SDPs: here n_{psd} is the total size of the semidefinite blocks; n_{nonneg} is the total number of nonnegative variables; n_{free} is the total number of free variables; m is the total number of constraints; and Nnz is the total number of nonzero entries in the constraints.

Table 2.4: Results on the "finance" problems

Method	n_{psd}	n_{nonneg}	n_{free}	m	Nnz
None	60,400	51,100	0	251,777	2,895,756
pd1	60,400	51,100	0	251,777	2,895,756
pd2	60,280	51,100	0	249,797	2,880,876
dd1	27,429	51,100	2,286,000	251,777	2,844,756
dd2	36,400	51,100	2,521,005	251,777	2,605,807
Sieve-SDP	56,766	50,873	0	215,210	2,466,573

While dd1 and dd2 significantly reduced the size of the PSD blocks, they added many free variables. Sieve-SDP reduced the size of the PSD blocks without adding free variables, and it eliminated the most constraints. We mention here that after preprocessing with dd2, MOSEK detected that problem "leverage_limit" is "dual infeasible". This may be due to numerical instability.

We remark that preprocessing actually *increased* the solving time on these problems, though not by much. For example, the total time spent on preprocessing with Sieve-SDP plus solving with MOSEK is about 21% higher than the solving time with MOSEK without preprocessing. Still, since the primary goal of preprocessing is to improve solution accuracy, we believe that we should do it whenever we can.

Furthermore, on these instances Sieve-SDP performed a large number of iterations, and deleted only a small submatrix in each one. Thus, we could easily reduce the time spent by Sieve-SDP by limiting the maximum number of iterations it is allowed to perform. We do not report results with such a setting, since we do not want to "overtune" our code.

2.3.5 Dressler-Illiman-de Wolff dataset (155 problems)

Let us consider the polynomial optimization problem

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \geq 0, \quad i = 1, \ldots, m, \end{aligned} \tag{2.8}$$

where f and the g_i are multivariate polynomials.

As shown in the seminal work of Lasserre [73], the optimal value of (2.8) can be lower bounded by solving SDPs. Under suitable conditions the lower bounds obtained from these SDPs converge to the optimal value of (2.8), as the so-called Lasserre relaxation order increases. See [97] for a related scheme to construct SDP relaxations of (2.8). However, no useful lower bound is obtained when the SDPs are infeasible.

Since solving the Lasserre SDPs can be challenging, Dressler, Illiman and de Wolff [33] proposed an alternative relaxation based on so-called nonnegative circuit polynomials, and they compared their approach with the SDP-based one.

We constructed the SDPs in the DIW dataset by taking the polynomial optimization problems from [33] and using GloptiPoly 3 [62] to generate their SDP relaxations. We describe our SDPs in Table 2.5 with their Lasserre relaxation order, which ranges from the lowest possible (half the degree of the highest degree monomial in the polynomial) to 20. For example, the SDP named "ex3.3_order4" is obtained by applying the Lasserre relaxation of order 4 to [33, Example 3.3].

Table 2.5: Relaxation orders for examples in [33]

Examples	3.3	4.1	4.2	4.3	4.4	5.4	5.5	5.6	5.7
Relaxation orders	$6, \ldots, 20$	$3, \ldots, 20$	$6, \ldots, 20$	$2, \ldots, 20$	$3, \ldots, 20$	$5, \ldots, 20$	$4, \ldots, 20$	$4, \ldots, 20$	$5, \ldots, 20$

Table 2.6 shows the results. Here, "n" is the sum of the orders of all PSD and nonnegative blocks, and column "m" is the sum of the number of constraints in all problems.

The results are quite striking. Sieve-SDP, pd1, and pd2 ran fast, reduced all problems in this collection, detected infeasibility of more than a third, and reduced overall computing time by a factor of more than a hundred! Sieve-SDP was the best in all aspects, with pd1 a close second. On the other hand, without preprocessing, MOSEK failed to detect infeasibility of any of these SDPs.

These results are somewhat surprising, since [33] solved some of these SDPs to approxi-

Table 2.6: Results on the DIW dataset

Method	# Reduced	n	m	Preprocessing (s)	Solving (s)	# Infeas
None	-	53,523	186,225	-	139,493.56	-
pd1	155	1,450	3,278	1632.43	128.46	56
pd2	155	1,450	3,278	10,831.32	124.44	56
dd1	0	53,523	186,225	65.18	139,493.56	0
dd2	0	53,523	186,225	22,152.57	139,493.56	0
Sieve-SDP	155	1,385	3,204	1,232.27	87.53	59

mate optimality, and managed to extract approximate optimal solutions of the original polynomial optimization problems. Similar results for similar SDPs were obtained earlier in [61]. In fact, [61] took the view that numerical inaccuracy of the SDP solvers actually *helps* find near-optimal solutions of the polynomial optimization problems. See [74] for a more recent and thorough study of the same issue. We remark that these SDPs are likely to be weakly infeasible.

We were thus motivated to double check that Sieve-SDP indeed reduced these SDPs correctly. Precisely, we verified that in the Basic Step in Figure 2.2 it only eliminated constraints that were in one of the following forms: either of the form

$$\begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \cdot X = 0,$$

where D is positive definite diagonal of order 1 or 2, and the smallest diagonal element is 1 or 0.5 or $1/3 = 0.3333\ldots$; or of the form

$$O \cdot X = 0,$$

where O is the zero matrix. Furthermore, Sieve-SDP always detected infeasibility by finding a constraint of the form

$$\begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \cdot X = \beta,$$

where D is as above, and $\beta = -3$ or -8.

The zeroes in all these constraints are zero in absolute machine precision, i.e., in the sparse SDPs returned by GloptiPoly 3, these entries do not appear at all. Thus Sieve-SDP performed all reductions correctly.

2.3.6 Henrion-Toh dataset (98 problems)

This dataset was kindly provided to us by Didier Henrion and Kim-Chuan Toh. The problems come mostly from polynomial optimization.

Among these problems, 18 were reduced by pd1, pd2, or Sieve-SDP, and none by dd1 or dd2. Table 2.7 shows the time details in seconds. The last column "Pre. vs. Solve" shows the time spent on preprocessing as a percentage of time spent on solving. It is

$$\frac{\text{preprocessing time}}{\text{solving time without preprocessing}} \times 100\%. \tag{2.9}$$

Table 2.7: Time results on the Henrion-Toh dataset

Method	Preprocessing (s)	Solving (s)	Pre. vs. solve
None	-	1420.02	-
pd1	10.27	1373.70	0.72%
pd2	49.84	1374.31	3.51%
dd1	3.93	1420.02	0.28%
dd2	29.24	1420.02	2.06%
Sieve-SDP	4.58	1376.27	0.32%

On this dataset the preprocessors are less successful: pd1, pd2, and Sieve-SDP detected infeasibility of only one problem (of "sedumi-l4") and they reduced solving time only a little. However, the preprocessing times are small, or even negligible: for example, Sieve-SDP spent only about 0.3% of the time that it took for MOSEK to solve the problems.

In Figure 2.5 we illustrate how Sieve-SDP works on the problem instance "sedumi-fp32": we show the sparsity structure of the constraints of the original problem (on the left), and after we applied Sieve-SDP (on the right). Just like in Figure 2.4, each row corresponds to an A_i matrix stretched out as a vector. Red dots correspond to positive entries, blue dots correspond to negative entries, and white areas correspond to zero entries.

Here, we also discuss problem "sedumi-fp33" on which preprocessing by Sieve-SDP makes the DIMACS error *worse*. Since this is the only such instance, we looked at it in more detail. The worst DIMACS error (of a solution computed by MOSEK) before Sieve-SDP is 3.36×10^{-7}, which is acceptable. After Sieve-SDP the worst error is about 0.0928, which is unacceptable.

We also solved this instance using the high accuracy SDP solver SDPA-GMP [47]. The DIMACS errors were

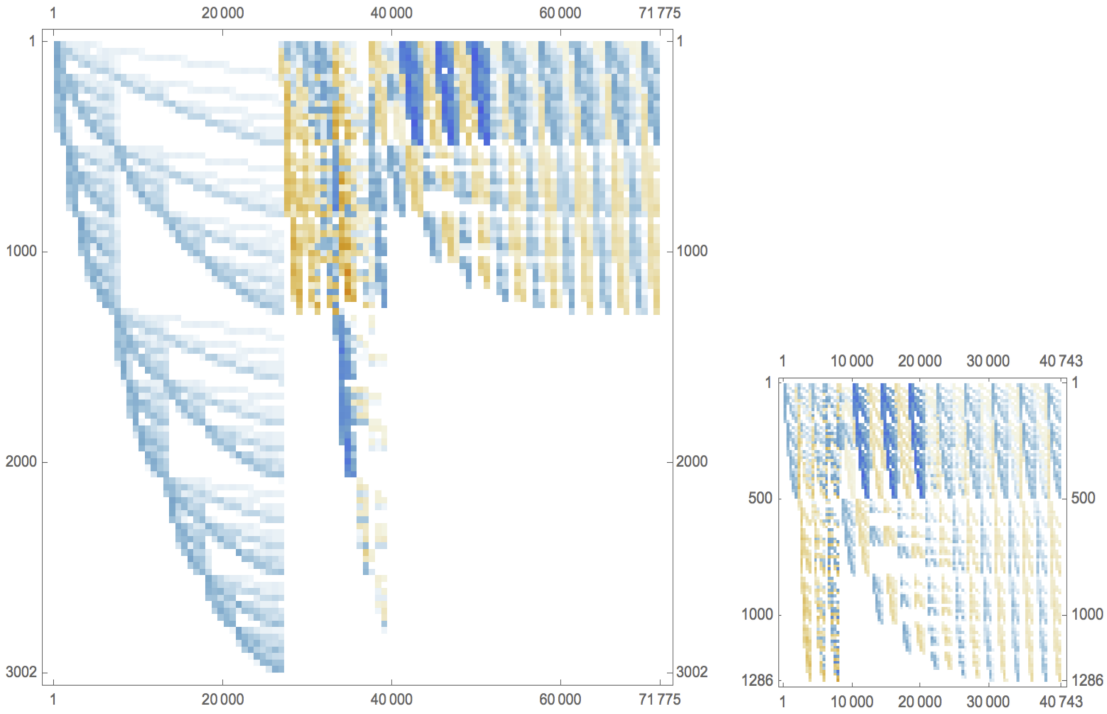

Figure 2.5: Problem "sedumi-fp32": size and sparsity before and after preprocessing.

$$2.3497 \times 10^{2},\ 0.0000,\ 1.8552 \times 10^{1},\ 0.0000,\ -9.9999 \times 10^{-1},\ 8.5173 \times 10^{-2}$$

before Sieve-SDP, and

$$3.4075 \times 10^{2},\ 0.0000,\ 1.9636 \times 10^{1},\ 0.0000,\ -9.9999 \times 10^{-1},\ 6.1901 \times 10^{-1}$$

after Sieve-SDP. In both cases, the largest error is unacceptably large. Given the high accuracy of SDPA-GMP, this problem seems not to be accurately solved by current fast SDP solvers, and the *worse* DIMACS error returned by MOSEK after Sieve-SDP alerts the user to this fact: this problem may actually have a positive duality gap (cf. Example 2.3).

2.3.7 Toh-Sun-Yang dataset (419 problems) from [122, 149]

Although none of the five methods reduced the SDPs in this collection, we still comment on them in detail, since pd1, dd1 and Sieve-SDP spent only a negligible amount time on preprocessing. Thus, using these three methods it does not hurt to preprocess: see Table 2.8. The last column "Pre. vs. Solve" defined in (2.9) shows the time spent on preprocessing as a percentage of time

spent on solving. Pd2 and dd2, on the other hand, spent considerably more time on preprocessing.

Table 2.8: Time results on the Toh-Sun-Yang dataset

Method	Preprocessing (s)	Solving (s)	Pre. vs. solve
pd1	220.18	27,635.46	0.80%
pd2	4,029.61	27,635.46	14.58%
dd1	134.64	27,635.46	0.49%
dd2	2,428.82	27,635.46	8.79%
Sieve-SDP	152.14	27,635.46	0.55%

2.4 Conclusions

We now give an overall comparison of all methods in Tables 2.9, 2.10 and 2.11.

Table 2.9: Infeasibility detection and error reduction on all 771 problems

Method	# Reduced	# Infeas detected	# DIMACS error improved	Crashed
pd1	209	67	74	0
pd2	230	67	78	6
dd1	14	0	2	0
dd2	21	0	4	4
Sieve-SDP	216	73	74	0

Table 2.10: Time results on all 771 problems

Method	Preprocessing (s)	Solving (s)	Prep vs. solve	Time reduction
None	-	272,427.23	-	-
pd1	2,486.51	132,356.63	0.91%	50.50%
pd2	23,323.07	131,636.47	8.56%	43.12%
dd1	587.93	272,244.62	0.22%	-0.15%
dd2	35,984.45	272,031.04	13.21%	-13.16%
Sieve-SDP	2,170.13	131,837.25	0.80%	51.81%

Table 2.11: Size reduction on all 771 problems

Method	# Reduced	Red. on n	Red. on m	Extra free vars	Nnz
none	-	-	-	-	300,989,332
pd1	209	15.47%	17.79%	0	211,299,702
pd2	230	15.59%	18.23%	0	211,257,726
dd1	14	6.74%	0.00%	2,293,495	300,936,120
dd2	21	9.28%	0.00%	2,315,849	299,272,012
Sieve-SDP	216	16.55%	20.66%	0	206,061,059

In Table 2.9 the second column shows how many problems were reduced. The third column shows how many problems were detected to be infeasible. The fourth column shows on how many

instances the preprocessing improved the DIMACS errors, as we discussed in Section 2.2. The last column "Crashed" shows how many times a method crashed or ran out of memory: this happened with pd2 six times and with dd2 four times. To ensure fair reporting, we reran these methods on the same instances on a machine with 24 GB RAM, and the results were the same.

Table 2.10 shows the preprocessing and solving times in seconds. The second column shows the preprocessing time and the third shows the solving time by MOSEK after preprocessing. Column "Prep vs. solve" shows the relative speed of the preprocessors; see (2.9). The last column, "Time reduction", displays by how much preprocessing reduced the solving time. It is

$$\frac{\text{solving time w/o preprocessing} - (\text{preprocessing time} + \text{solving time after preprocessing})}{\text{solving time w/o preprocessing}} \times 100\%.$$

Of course, the higher this percentage, the more a preprocessor reduces solving time. A negative percentage means that preprocessing actually *increased* the total time.

Finally, Table 2.11 shows by "how much" the problems were reduced. As in Table 2.9, the second column shows the number of problems reduced by each method. To explain the other columns, let us fix an SDP in the primal form (P) with potentially several PSD block variables (some of which may be of order 1, i.e., nonnegative variables). Let n_{before} and n_{after} be the total size of the PSD blocks before and after reduction. We define the reduction rate on n as

$$\frac{\sum n_{\text{before}} - \sum n_{\text{after}}}{\sum n_{\text{before}}},$$

where the sum is over all 771 problems. Similarly, let m_{before} and m_{after} be the number of constraints in a problem before and after reduction. We define the reduction rate on m as

$$\frac{\sum m_{\text{before}} - \sum m_{\text{after}}}{\sum m_{\text{before}}},$$

where the sum is again taken over all 771 problems. Methods dd1 and dd2 added free variables, and the fifth column in Table 2.11 shows how many. The sixth column "Nnz" shows the total number of nonzeros in the constraint matrices.

Given these tables, we now summarize the findings. In all aspects Sieve-SDP is competitive with the other preprocessing methods. In detail:

- It is competitive considering the number of problems reduced.
- It is competitive in computing known optimal solutions; see Tables 2.1, 2.2 and 2.3.
- The time spent on preprocessing with Sieve-SDP vs. solving is negligible. It is also negligible for pd1 and dd1, but less so for pd2 and dd2. See Table 2.10.

In several aspects Sieve-SDP is the best.

- It is best in detecting infeasibility; see Table 2.9. It is important that Sieve-SDP detects infeasibility without using any optimization solver, whereas the other methods rely on MOSEK.
- It reduced solving time the most, with pd1 a close second; see Table 2.10.
- It reduced the size of the instances the most: see Table 2.11.
- It needs very little additional memory, precisely $O(nm)$; see Appendix A.2.
- It is as accurate as Cholesky factorization, which works in machine precision. Sieve-SDP is also easily implemented in a *safe mode*; see Section 2.2.
- It is the simplest: the core code consists of only 40-80 lines; see Appendix A.2.

The code is available from

`https://github.com/unc-optimization/SieveSDP`

CHAPTER 3

ACCELERATED PRIMAL-DUAL ALGORITHMS FOR A CLASS OF CONVEX-CONCAVE SADDLE-POINT PROBLEMS

3.1 Introduction

We develop two new first-order primal-dual algorithms to solve a class of convex-concave saddle-point problems involving non-bilinear coupling function, which covers many existing and brand-new applications as special cases. Our approach relies on a novel combination of non-convex augmented Lagrangian and Nesterov's accelerated schemes, and homotopy strategy. Both algorithms are single-loop and only require one or at most two proximal operators of the objective function, one gradient of the coupling function, and possibly one gradient of the smooth objective term per iteration. They do not require to solve any complex subproblem as in standard augmented Lagrangian or penalty methods.

When the objective function is merely convex, our first algorithm can achieve $\mathcal{O}\left(\frac{1}{k}\right)$ convergence rates through three different criteria (primal objective residual, dual objective residual, and primal-dual gap), on either the *ergodic* sequence or the *non-ergodic* sequence. This rate can potentially be even faster than $\mathcal{O}\left(\frac{1}{k}\right)$ on *non-ergodic* primal objective residual using a new parameter update rule. If the objective function is strongly convex, our second algorithm can boost these convergence rates to no slower than $\mathcal{O}\left(\frac{1}{k^2}\right)$. To the best of our knowledge, these are the first algorithms that can achieve such fast convergence rates on *non-ergodic* sequences for non-bilinear convex-concave saddle-point problems.

As a by-product, we specify our results to handle general cone-constrained convex problems. We test our algorithms on quadratically constrained quadratic programs, convex-concave game problems, and image processing problems, to verify the algorithms' performance as well as to compare them with existing methods.

This chapter is based on papers [129, 130, 152], the joint works with Dr. Quoc Tran-Dinh and

Deyi Liu. Here, [129] develops some basic proof techniques, and provides the numerical experiments on image processing. Then [130] discusses the case where the coupling term is bilinear, and improves the convergence rates. Finally, [152] generalizes the theory and algorithms in [130] for non-bilinear coupling terms. The last paper considers the most general problem, and thus it is the major component of this chapter.

Problem statement. Our goal is to develop novel first-order primal-dual algorithms to solve the following convex-concave saddle-point problem involving non-bilinear coupling function:

$$\min_{x\in\mathbb{R}^p}\max_{y\in\mathbb{R}^m}\left\{\widetilde{\mathcal{L}}(x,y) := F(x) + \langle g(x), y\rangle - H^*(y)\right\}, \tag{SP}$$

where functions $F:\ \mathbb{R}^p \to \mathbb{R}\cup\{+\infty\}$ and $H:\ \mathbb{R}^m \to \mathbb{R}\cup\{+\infty\}$ are proper, closed, and convex, but not necessarily smooth, H^* is the Fenchel conjugate of H, and $g:\ \mathbb{R}^p \to \mathbb{R}^m$ is a smooth vector function such that $\langle g(x), y\rangle$ is convex for all $y \in \mathrm{dom}H^*$. Under this assumption on g, we have that $H\circ g$ is convex in x, and we can formulate (SP) into the following primal composite convex minimization problem:

$$\mathcal{P}^\star := \min_{x\in\mathbb{R}^p}\left\{\mathcal{P}(x) := F(x) + \max_{y\in\mathbb{R}^m}\{\langle g(x), y\rangle - H^*(y)\} \equiv F(x) + H\left(g(x)\right)\right\}. \tag{P}$$

The corresponding dual problem is also convex and can be written as

$$\mathcal{D}^\star := \max_{y\in\mathbb{R}^m}\left\{\mathcal{D}(y) := \min_{x\in\mathbb{R}^p}\{F(x) + \langle g(x), y\rangle\} - H^*(y)\right\}. \tag{D}$$

The saddle-point problem (SP) and its primal form (P) has numerous applications, as described in Example 1.1 in Section 1.1. Among them, an important special case (1.1) is when $H^* \equiv \delta_{\mathcal{K}^*}$, the indicator of the dual cone of a proper cone, and (P) becomes

$$\min_{x\in\mathbb{R}^p} F(x) \quad \text{s.t.} \quad g(x) \in -\mathcal{K}. \tag{3.1}$$

It is a convex program, and it generalizes conic programming, as the objective function F is not necessarily linear.

For a review of the existing methods for solving (SP) or (P), see Section 1.3, where we have

pointed out that existing methods have three types of limitations:

- Strong model assumptions;
- High per-iteration complexity; and
- Inconsistency between theoretical convergence guarantees and empirical solutions.

Our contributions. Faced with the above limitations in the existing literature, we develop two first-order primal-dual algorithms with the following features:

1. *Mild model assumption.* Our algorithms are designed for the general problem (P) or its saddle-point form (SP), and with mild assumptions. Unlike [57, 64, 79], the domain of y in (SP) can be assumed to be unbounded.
2. *Low per-iteration complexity.* Our algorithms have only single loop, thus the per-iteration complexity is low - they only require gradient computations, proximal operations, and function evaluations, at most twice each for each iteration.
3. *Optimal* $\mathcal{O}\left(\frac{1}{k}\right)$ *ergodic and semi-ergodic convergence rates when* $k = \mathcal{O}(p)$.[1]
   - We specify a parameter update rule for Algorithm 1 and establish its $\mathcal{O}\left(\frac{1}{k}\right)$ *ergodic* convergence rate on the duality gap.
   - We establish $\mathcal{O}\left(\frac{1}{k}\right)$ convergence rate on the primal *non-ergodic* and dual *ergodic* sequences of Algorithm 1. We call this the *semi-ergodic* rate. Unlike existing works, we characterize three different criteria: gap function, primal objective residual, and dual objective residual.
4. *Faster* $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ *primal non-ergodic convergence rate when* $k > \mathcal{O}(p)$. We modify the parameter update rule of Algorithm 1 to achieve both $\mathcal{O}\left(\frac{1}{k}\right)$ and $\underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)$ *non-ergodic* convergence rates for (P). To the best of our knowledge, this is the first time that a first-order method for (P) attains such fast convergence rates.
5. *Boosted convergence rates when* F *is strongly convex.* When function F in (SP) is strongly convex, Algorithm 2 can boost the rates in Items 3 and 4 up to $\mathcal{O}\left(\frac{1}{k^2}\right)$ and $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$, resp.

[1]The $\mathcal{O}$ and the soon-to-appear $\underline{o}$ notations will be defined in (3.2)-(3.3).

The above contributions in terms of convergence rates, specified by Items 3-5, are summarized in Table 3.1. Here, $\mathcal{P}(x) - \mathcal{D}(y)$ and $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x, y)$ are both gap functions to be defined in Section 3.2.3; $\{x^k\}$ and $\{\bar{x}^k\}$ denote the primal non-ergodic (last-iterate) and ergodic (averaging) sequences, respectively, and $\{\bar{y}^k\}$ is the dual ergodic sequence.

Table 3.1: Summary of our main contributions

Algorithm	Parameter update rule	Convergence criteria	Convergence rate	Theorem
Algorithm 1	option 1: (3.43) (3.44)	$\mathcal{P}(\bar{x}^k) - \mathcal{D}(\bar{y}^k),\ \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(\bar{x}^k, \bar{y}^k)$	$\mathcal{O}\left(\frac{1}{k}\right)$	Theorem 3.1
	option 2: (3.50) (3.51)	$\mathcal{P}(x^k) - \mathcal{D}(\bar{y}^k),\ \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x^k, \bar{y}^k)$	$\mathcal{O}\left(\frac{1}{k}\right)$	Theorem 3.2
	option 3: (3.54) (3.55)	$\mathcal{P}(x^k) - \mathcal{P}^\star$	$\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$	Theorem 3.3
Algorithm 2	option 1: (3.72) (3.73)	$\mathcal{P}(\bar{x}^k) - \mathcal{D}(\bar{y}^k),\ \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(\bar{x}^k, \bar{y}^k)$	$\mathcal{O}\left(\frac{1}{k^2}\right)$	Theorem 3.4
	option 2: (3.79) (3.80)	$\mathcal{P}(x^k) - \mathcal{D}(\bar{y}^k),\ \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x^k, \bar{y}^k)$	$\mathcal{O}\left(\frac{1}{k^2}\right)$	Theorem 3.5
	option 3: (3.86) (3.87)	$\mathcal{P}(x^k) - \mathcal{P}^\star$	$\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$	Theorem 3.6

Our approach. Our approach relies on a novel combination of the following techniques:

- We utilize an augmented Lagrangian function to penalize the constraints of (3.10). Different from standard augmented Lagrangian methods [8, 115, 144–147], our augmented Lagrangian function is *globally non-convex*, but *locally convex* in x. This function plays a role as a merit function to measure the optimality.
- We apply Nesterov's accelerated methods [89] to minimize the augmented Lagrangian function w.r.t. the primal variable x, in order to achieve the optimal $\mathcal{O}\left(\frac{1}{k}\right)$ and $\mathcal{O}\left(\frac{1}{k^2}\right)$ convergence rates, resp. for merely convex and strongly convex F.
- We exploit homotopy strategy in [126, 130] to simultaneously update penalty parameter and stepsizes, making the algorithms converge with optimal rates in the primal *non-ergodic* sense.
- We use the techniques in [3] to develop the even faster $\underline{o}$-rates.

Chapter organization. The rest of this chapter is organized as follows.

In the main text, Section 3.2 recalls some basic concepts, and defines our augmented Lagrangian function and characterizes its property. Section 3.3 develops our first algorithm, Algorithm 1, for solving (SP), and discusses how its three variants lead to different types of convergence guarantees. In Section 3.4, we develop the second algorithm, Algorithm 2, to handle the strongly convex case, and we prove the algorithm's convergence rates. Section 3.5 specifies our methods to solve cone-constrained convex problem (3.1). Section 3.6 provides several numerical examples to verify our

theoretical results. Finally, we draw the conclusions in Section 3.7.

For the clarity of presentation, we put some supplemental proofs in the appendices. Appendix B.1 presents a useful basic lemma. Appendices B.2 and B.3 provide supplenmental proofs for the key lemmas and theorems in Sections 3.3 and 3.4, respectively.

3.2 Fundamental Assumptions and Mathematical Tools

Let us first recall some basic notations and concepts, and describe our assumptions imposed on (SP). Then, we state the optimality condition and introduce the gap functions. Finally, we reformulate (SP) into a non-convex constrained problem and introduce the associated non-convex augmented Lagrangian function. We also prove a key property of this function, which will be used in the sequel.

3.2.1 Basic notations and concepts

We work with Euclidean spaces $\mathbb{R}^p$ and $\mathbb{R}^m$ equipped with standard inner product $\langle x, w\rangle := \sum_i x_i w_i$ and norm $\|x\| := \langle x, x\rangle^{1/2}$. For any nonempty, closed, and convex set $\mathcal{X}$ in $\mathbb{R}^p$, we use $\mathrm{ri}\mathcal{X}$ to denote the relative interior of $\mathcal{X}$, and $\delta_{\mathcal{X}}$ to denote the indicator of $\mathcal{X}$. If $\mathcal{K}$ is a proper cone, then $\mathcal{K}^* := \{w \in \mathbb{R}^p \mid \langle w, x\rangle \geq 0,\ \forall x \in \mathcal{K}\}$ denotes its dual cone.

For any proper, closed, and convex function $f : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$, let $\mathrm{dom} f := \{x \in \mathbb{R}^p \mid f(x) < +\infty\}$ be its (effective) domain, let $f^*(w) := \sup_x\{\langle w, x\rangle - f(x)\}$ be its Fenchel conjugate, let $\partial f(x) := \{w \in \mathbb{R}^p \mid f(x') - f(x) \geq \langle w,\ x' - x\rangle,\ \forall x' \in \mathrm{dom} f\}$ be the subdifferential of f at x, and let ∇f be the gradient or subgradient of f. We also denote $\mathrm{prox}_f(x) := \mathrm{argmin}_{x'}\{f(x') + \frac{1}{2}\|x' - x\|^2\}$ as the proximal operator of f at x. If f is the indicator of a convex set $\mathcal{X}$, then prox_f reduces to the projection $\mathrm{proj}_{\mathcal{X}}$ onto $\mathcal{X}$. For a vector function $g : \mathbb{R}^p \to \mathbb{R}^m$, we use $\nabla g \in \mathbb{R}^{m\times p}$ to denote its Jacobian.

A function $f : \mathbb{R}^p \to \mathbb{R}^m$ is called M_f-Lipschitz continuous on $\mathrm{dom} f$ with a Lipschitz constant $M_f \in [0, +\infty)$ if $\|f(x) - f(x')\| \leq M_f\|x - x'\|$ for all $x, x' \in \mathrm{dom} f$. If f is differentiable on $\mathrm{dom} f$ and ∇f is Lipschitz continuous with a Lipschitz constant $L_f \in [0, +\infty)$, i.e., $\|\nabla f(x) - \nabla f(x')\| \leq L_f\|x - x'\|$ for $x, x' \in \mathrm{dom} f$, then we say that f is L_f-smooth. If $f(\cdot) - \frac{\mu_f}{2}\|\cdot\|^2$ is still convex for some $\mu_f > 0$, then we say that f is μ_f-strongly convex with a strong convexity parameter μ_f.

Clearly, if $\mu_f = 0$, then f is merely convex.

Finally, $\mathbb{R}_+$ and $\mathbb{R}_{++}$ are the sets of nonnegative and positive real numbers, respectively, and $\mathbb{N}$ is the set of nonnegative integers. We use $\mathcal{O}(\cdot)$, $o(\cdot)$ and $\Omega(\cdot)$ to denote the order of complexity as usual: for two sequences of scalars $\{u_k\} \subseteq \mathbb{R}_+$ and $\{v_k\} \subseteq \mathbb{R}_{++}$, we say

$$\begin{cases} u_k = \mathcal{O}(v_k), & \text{if } \limsup_{k\to\infty} \frac{u_k}{v_k} < \infty, \\ u_k = o(v_k), & \text{if } \lim_{k\to\infty} \frac{u_k}{v_k} = 0, \\ u_k = \Omega(v_k), & \text{if } \liminf_{k\to\infty} \frac{u_k}{v_k} > 0. \end{cases} \tag{3.2}$$

We further define a new $\underline{o}(\cdot)$ notation as follows:

$$u_k = \underline{o}(v_k) \quad \text{if} \quad \liminf_{k\to\infty} \frac{u_k}{v_k} = 0, \tag{3.3}$$

that is, there is a subsequence $\{k_j\} \subseteq \mathbb{N}$ such that $u_{k_j} = o(v_{k_j})$.

3.2.2 Fundamental assumptions

Throughout this chapter, we will rely on the following two assumptions imposed on (SP) to develop our algorithms and analyze their convergence guarantees.

Assumption 3.1. *The set of saddle-points $\mathcal{X}^\star \times \mathcal{Y}^\star$ of* (SP) *is nonempty, i.e., there exists $(x^\star, y^\star) \in \mathcal{X}^\star \times \mathcal{Y}^\star$ such that:*

$$\widetilde{\mathcal{L}}(x^\star, y) \le \widetilde{\mathcal{L}}(x^\star, y^\star) \le \widetilde{\mathcal{L}}(x, y^\star), \quad \forall (x, y) \in \mathbb{R}^p \times \mathbb{R}^m. \tag{3.4}$$

Assumption 3.1 is standard in saddle-point problems. With this, we can show the following connection between the objective values of the primal (P) and of its dual (D):

$$\mathcal{D}(y) \le \mathcal{D}(y^\star) = \mathcal{D}^\star = \mathcal{P}^\star = \mathcal{P}(x^\star) \le \mathcal{P}(x), \quad \forall (x, y) \in \mathbb{R}^p \times \mathbb{R}^m, \tag{3.5}$$

where $\mathcal{P}$ and $\mathcal{D}$ are the primal and dual objectives defined in (P) and (D), respectively.[2]

[2]Indeed, $\mathcal{P}^\star := \mathcal{P}(x^\star) \overset{(\mathrm{P})}{=} \max_y \widetilde{\mathcal{L}}(x^\star, y) = \widetilde{\mathcal{L}}(x^\star, y^\star) \overset{(3.4)}{\le} \widetilde{\mathcal{L}}(x, y^\star) \le \max_y \widetilde{\mathcal{L}}(x, y) \overset{(\mathrm{P})}{=} \mathcal{P}(x)$. The dual direction can be proved analogously.

We also impose the following assumption on (SP).

Assumption 3.2. *The functions F, H, and g in* (SP) *satisfy the following conditions:*

1. *The function $F(x) = f(x) + h(x)$ is defined on $\mathbb{R}^p$, where both f and h are proper, closed and convex, and f is L_f-smooth for some Lipschitz constant $L_f \in [0, \infty)$.*
2. *The function $H : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ is proper, closed, and convex.*
3. *The function g is such that $\langle g(x), y\rangle$ is convex in x for any $y \in \mathrm{dom} H^*$, and:*
   (a) *Function g is $\mathbf{M}_g$-uniformly Lipschitz continuous for some $\mathbf{M}_g \in [0, \infty)^m$, i.e.,*

$$\|g(x) - g(x')\| \le \|\mathbf{M}_g\|\|x - x'\| = M_g\|x - x'\|, \quad \forall x, x' \in \mathrm{dom}\mathcal{P}, \tag{3.6}$$

   where $M_g := \|\mathbf{M}_g\|$.
   (b) *For any $y \in \mathrm{dom} H^*$, function $[\nabla g(\cdot)]^\top y$ is $\mathbf{L}_g(y)$-Lipschitz continuous for some $\mathbf{L}_g(y) \in [0, +\infty)$ depending on y, i.e.,*

$$\left\|[\nabla g(x)]^\top y - [\nabla g(x')]^\top y\right\| \le [\mathbf{L}_g(y)]\|x - x'\|, \quad \forall x, x' \in \mathrm{dom}\mathcal{P}.$$

   In addition, $\mathbf{L}_g(y)$ satisfies $0 \le \mathbf{L}_g(y) \le L_g\|y\|$ for some $L_g \in [0, +\infty)$.

The condition (3.6) is equivalent to the M_{g_i}-Lipschitz continuity of g_i, where M_{g_i} is the i-th component of $\mathbf{M}_g$, and g_i is the i-th component of mapping g, where $i = 1, \ldots, m$. Clearly, if $g(x) = Ax$ is bilinear, then it automatically satisfy Assumption 3.2(3). Assumption 3.2 is standard in primal-dual methods for solving (SP) as used in [57, 64, 79]. However, unlike these works, $\mathbf{L}_g(y)$ in Assumption 3.2 can depend on y, which allows us to cover cone constrained problem (3.1) without requiring the boundedness of $\mathrm{dom} F$ or $\mathrm{dom} H^*$. Note that in item 3, the convexity and $\mathbf{L}_g(y)$-smoothness of $\langle g(x), y\rangle$ imply that

$$0 \le \langle y,\ g(x') - g(x) - [\nabla g(x)](x' - x)\rangle \le \frac{\mathbf{L}_g(y)}{2}\|x' - x\|^2, \quad \forall x, x' \in \mathrm{dom}\mathcal{P}. \tag{3.7}$$

In particular, if $\mathrm{dom}\mathcal{P}$ is nonempty, convex, and compact, and g is continuously differentiable on $\mathrm{dom}\mathcal{P}$, then g is $\mathbf{M}_g$-Lipschitz continuous and $\langle g(x), y\rangle$ is $\mathbf{L}_g(y)$-smooth on $\mathrm{dom}\mathcal{P}$. Some existing works [6, 71, 147] impose these conditions, but we do not require $\mathrm{dom}\mathcal{P}$ to be bounded.

3.2.3 Optimality condition and gap functions

Optimality condition. In view of Assumption 3.1 and the Fermat's rule, there exists a pair of optimal solutions $(x^\star, y^\star) \in \mathbb{R}^p \times \mathbb{R}^m$ to the primal problem (P) and its dual formulation (D), which satisfies the following optimality condition:

$$0 \in \partial F(x^\star) + [\nabla g(x^\star)]^\top y^\star \quad \text{and} \quad 0 \in g(x^\star) - \partial H^*(y^\star). \tag{3.8}$$

Gap function. We consider two types of duality gap functions at a pair of solutions (x, y). The first one is the standard primal-dual gap $\mathcal{P}(x) - \mathcal{D}(y)$, which is nonnegative due to the weak duality as shown in (3.5), and it vanishes, i.e., $\mathcal{P}(x) - \mathcal{D}(y) = 0$, if and only if (x, y) is a saddle point of (SP) due to strong duality, by Assumption 3.1.

Another gap function is defined as

$$\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x, y) := \sup_{x' \in \mathcal{X}, y' \in \mathcal{Y}} \left\{ \widetilde{\mathcal{L}}(x, y') - \widetilde{\mathcal{L}}(x', y) \right\} = \sup_{y' \in \mathcal{Y}} \widetilde{\mathcal{L}}(x, y') - \inf_{x' \in \mathcal{X}} \widetilde{\mathcal{L}}(x', y). \tag{3.9}$$

When $\mathcal{X} \times \mathcal{Y}$ contains a saddle-point, it is clear that

$$\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x, y) \geq \widetilde{\mathcal{L}}(x, y^\star) - \widetilde{\mathcal{L}}(x^\star, y) \overset{(3.4)}{\geq} \widetilde{\mathcal{L}}(x^\star, y^\star) - \widetilde{\mathcal{L}}(x^\star, y^\star) = 0.$$

Moreover, when (x, y) is a saddle-point, $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x, y) = 0$.[3] This gap function is widely used in the literature on primal-dual convergence theory [13, 20, 28, 88].

It is clear that $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x, y) \leq \mathcal{G}_{\mathbb{R}^p\times\mathbb{R}^m}(x, y) = \mathcal{P}(x) - \mathcal{D}(y)$. In our analysis, we will have convergence guarantees on both types of duality gaps. For the gap $\mathcal{P}(x) - \mathcal{D}(y)$, we would require additional conditions such as the Lipschitz continuity on H and/or F^*; in contrast, convergence guarantees on $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}$ do not require such conditions.

3.2.4 The augmented Lagrangian function and its properties

Non-convex constrained reformulation. To solve (SP), we can write (P) as

[3] To be more specific, $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}$ could also vanish at non-saddle-points. However, if (x, y) is in the interior of $\mathcal{X} \times \mathcal{Y}$, then $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x, y) = 0$ if and only if (x, y) is a saddle-point of (SP) [20].

$$\min_{(x,s)\in\mathbb{R}^p\times\mathbb{R}^m}\{F(x)+H(-s)\quad\text{s.t.}\quad g(x)+s=0\},\tag{3.10}$$

where s is the slack variable. If g is non-affine, then (3.10) is non-convex. Moreover, the Lagrange function associated with (3.10) can be written as

$$\mathcal{L}(x,s,y):=F(x)+H(-s)+\langle y,\ g(x)+s\rangle,\tag{3.11}$$

where $y\in\mathbb{R}^m$ is a Lagrange multiplier. If $(x^\star,y^\star)$ is optimal to (SP), i.e., satisfies (3.8), then $(x^\star,y^\star,s^\star)$ is optimal to (3.10), where $s^\star=-g(x^\star)$. Thus (3.8) can be written as

$$0\in\partial F(x^\star)+[\nabla g(x^\star)]^\top y^\star\quad\text{and}\quad -g(x^\star)=s^\star\in-\partial H^*(y^\star).\tag{3.12}$$

By the Fenchel theorem, we have $H(-s)+H^*(y)\geq-\langle s,y\rangle$, where the equality holds if and only if $s\in-\partial H^*(y)$, or equivalently, $y\in\partial H(-s)$. Therefore, it holds that

$$\widetilde{\mathcal{L}}(x,y)\leq\mathcal{L}(x,s,y)\quad\text{and}\quad\widetilde{\mathcal{L}}(x,y)=\mathcal{L}(x,s,y)\text{ iff }s\in-\partial H^*(y).\tag{3.13}$$

Consequently, for any $(x,s,y)\in\mathbb{R}^p\times\mathbb{R}^m\times\mathbb{R}^m$, (3.4) implies that

$$\widetilde{\mathcal{L}}(x^\star,y)\leq\mathcal{L}(x^\star,s^\star,y)=\widetilde{\mathcal{L}}(x^\star,y^\star)=\mathcal{L}(x^\star,s^\star,y^\star)\leq\widetilde{\mathcal{L}}(x,y^\star)\leq\mathcal{L}(x,s,y^\star).\tag{3.14}$$

Augmented Lagrangian function. The augmented Lagrangian of (3.10) is defined as

$$\begin{aligned}\mathcal{L}_\rho(x,s,y)&:=\mathcal{L}(x,s,y)+\tfrac{\rho}{2}\|g(x)+s\|^2\\&\overset{(3.11)}{=}F(x)+H(-s)+\langle y,\ g(x)+s\rangle+\tfrac{\rho}{2}\|g(x)+s\|^2,\end{aligned}\tag{3.15}$$

where the scalar $\rho>0$ is a penalty parameter. Note that if g is not affine, then $\mathcal{L}_\rho$ is *not convex* in x. Some existing works [144–147] minimize $\mathcal{L}_\rho$ over s to obtain a standard convex augmented Lagrangian function, first proposed in [113]; however, such formulation does not allow linear updates in y, preventing a clear analysis when applying Nesterov's acceleration technique. Therefore, we preserve s and keep the non-convex form of $\mathcal{L}_\rho$, so that it is linear in y. As will be shown, in our analysis, we do not need the global convexity, but rather the *local convexity* of $\mathcal{L}_\rho$ in x. We can

view this function as a smoothed approximation of the constrained reformulation (3.10) of (P), where the smoothness parameter is in fact the penalty parameter [91].

Augmented Lagrangian term. Let us introduce

$$\phi_\rho(x,s,y) := \langle y,\ g(x)+s\rangle + \frac{\rho}{2}\|g(x)+s\|^2. \tag{3.16}$$

Then, by (3.15), we have $\mathcal{L}_\rho(x,s,y) = F(x) + H(-s) + \phi_\rho(x,s,y)$. It is easy to see that at an optimal solution, i.e., a $(x^\star,s^\star,y^\star)$-tuple that satisfies (3.12), we have $\phi_\rho(x^\star,s^\star,y^\star)=0$. Moreover, we can directly compute the first-order derivatives of ϕ_ρ as

$$\begin{cases} \nabla_x\phi_\rho(x,s,y) \ = \ [\nabla g(x)]^\top\left(y+\rho[g(x)+s]\right), \\ \nabla_s\phi_\rho(x,s,y) \ = \ y+\rho[g(x)+s], \\ \nabla_y\phi_\rho(x,s,y) \ = \ g(x)+s, \end{cases} \tag{3.17}$$

where $\nabla g(x)\in\mathbb{R}^{m\times p}$ is the Jacobian of g at x. For $d\in\mathbb{R}^p$, the Hessian of ϕ_ρ in x to the direction of d is given by

$$\nabla_x^2\phi_\rho(x,s,y)[d,d] = \rho\|\nabla g(x)[d]\|^2 + \sum_{i=1}^m \left(y_i+\rho[g_i(x)+s_i]\right)\nabla^2 g_i(x)[d,d].$$

By Assumption 3.2(3), when $y' := y+\rho[g(x)+s]\in \mathrm{dom}H^*$, we have that $\langle g(x),y'\rangle$ is convex in x, i.e., the last term in the last equality is nonnegative, and thus $\phi_\rho(x,s,y')$ is locally convex in x. If we view ϕ_ρ as a function of g, then it is convex and ρ-smooth in g.

These important properties of ϕ_ρ leads to Lemma 3.1, which will be used to prove descent lemmas in Sections 3.3 and 3.4.

Lemma 3.1. *Let ϕ_ρ be as in (3.16). For any $x,x'\in\mathbb{R}^p$, $s,s'\in\mathbb{R}^m$ and $y\in\mathbb{R}^m$ such that $y+\rho[g(x)+s]\in\mathrm{dom}H^*$, we define the residual Δ_ρ of a linearization of ϕ_ρ at (x,s,y) as*

$$\begin{aligned}\Delta_\rho(x',s';x,s,y) \ := \ & \phi_\rho(x',s',y)-\phi_\rho(x,s,y) \\ & -\langle\nabla_x\phi_\rho(x,s,y),\ x'-x\rangle - \langle\nabla_s\phi_\rho(x,s,y),\ s'-s\rangle.\end{aligned} \tag{3.18}$$

Then, we have the following estimate:

$$0 \leq \Delta_\rho(x', s'; x, s, y) - \frac{\rho}{2}\|[g(x') + s'] - [g(x) + s]\|^2 \leq \frac{\mathbf{L}_g\left(y + \rho[g(x) + s]\right)}{2}\|x' - x\|^2, \tag{3.19}$$

where $\mathbf{L}_g\left(y + \rho[g(x) + s]\right)$ *is the Lipschitz modulus defined by Assumption 3.2(3c).*

Proof. By definition of Δ_ρ in (3.18), we can use the definition of $\phi_\rho(x, s, y)$ in (3.16) and its partial gradients w.r.t. x and s in (3.17) to explicitly write Δ_ρ as

$$\begin{aligned}\Delta_\rho(x', s'; x, s, y) = &\langle y,\ [g(x') + s'] - [g(x) + s]\rangle + \tfrac{\rho}{2}\left(\|g(x') + s'\|^2 - \|g(x) + s\|^2\right)\\ &-\langle y + \rho[g(x) + s],\ [\nabla g(x)](x' - x) + (s' - s)\rangle \\ = &\langle y + \rho[g(x) + s],\ g(x') - g(x) - [\nabla g(x)](x' - x)\rangle + \tfrac{\rho}{2}\|[g(x') + s'] - [g(x) + s]\|^2.\end{aligned} \tag{3.20}$$

By the $\mathbf{L}_g(\cdot)$-smoothness of $\langle \cdot, \nabla g(x)\rangle$ w.r.t. x, and that $y + \rho[g(x) + s] \in \mathrm{dom} H^*$, we can apply (3.7) with $y \leftarrow y + \rho[g(x) + s]$ to get

$$0 \leq \left\langle y + \rho[g(x) + s],\ g(x') - g(x) - [\nabla g(x)](x' - x)\right\rangle \leq \frac{\mathbf{L}_g\left(y + \rho[g(x) + s]\right)}{2}\|x' - x\|^2. \tag{3.21}$$

Combining (3.20) and (3.21), we immediately get

$$\begin{cases} \Delta_\rho(x', s'; x, s, y) \ \geq\ \frac{\rho}{2}\|[g(x') + s'] - [g(x) + s]\|^2 \\ \Delta_\rho(x', s'; x, s, y) \ \leq\ \frac{\rho}{2}\|[g(x') + s'] - [g(x) + s]\|^2 + \frac{\mathbf{L}_g(y + \rho[g(x) + s])}{2}\|x' - x\|^2, \end{cases}$$

which is exactly (3.19). □

3.3 Our First Primal-Dual Algorithm: General Convex-Concave Case

In this section, we develop a novel algorithm to solve (SP) under the general convexity-concavity assumption, i.e., F and H^* are convex, but not necessarily strongly convex.

3.3.1 The derivation and the complete algorithm

Our main idea is to exploit the augmented Lagrangian $\mathcal{L}_\rho$ defined in (3.15) as a merit function to measure the progress of the iterate sequence $\left\{(x^k, y^k)\right\}$. Since this function not only involves x but also the dual variable y and the slack variable s, we also need to update them accordingly. To

accelerate, we inject Nesterov's accelerated steps [89] in x. Recall that ϕ_ρ is not convex in x, but thanks to Lemma 3.1, we can still utilize its local convexity.

Step by step, we derive our scheme to solve (SP) as follows.

1. We first update the slack variable s^{k+1} by minimizing $\mathcal{L}_{\rho_k}(\hat{x}^k, s, \tilde{y}^k)$ w.r.t. s:

$$\begin{aligned} s^{k+1} &:= \arg\min_{s\in\mathbb{R}^m} \left\{ H(-s) + \langle \tilde{y}^k,\ g(\hat{x}^k) + s\rangle + \frac{\rho_k}{2}\|g(\hat{x}^k) + s\|^2 \right\} \\ &= -\mathrm{prox}_{H/\rho_k}\left(\frac{\tilde{y}^k}{\rho_k} + g(\hat{x}^k)\right). \end{aligned} \tag{3.22}$$

2. To update x^{k+1}, we would attempt to minimize $\mathcal{L}_{\rho_k}(x, s^{k+1}, \tilde{y}^k)$ w.r.t. x. However, since minimizing this function directly is difficult, we instead linearize f and $\phi_{\rho_k}(\cdot, s^{k+1}, \tilde{y}^k)$ at point $\hat{x}^k$, respectively:

$$\begin{cases} f(x) & \approx\ f(\hat{x}^k) + \langle \nabla f(\hat{x}^k),\ x - \hat{x}^k\rangle + \frac{L_k^f}{2}\|x - \hat{x}^k\|^2, \\ \phi_{\rho_k}(x, s^{k+1}, \tilde{y}^k) & \approx\ \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x - \hat{x}^k\rangle + \frac{L_k^\phi}{2}\|x - \hat{x}^k\|^2, \end{cases}$$

   for some $L_k^f > 0$ and $L_k^\phi > 0$, respectively. Writing $\beta_k := \frac{1}{L_k^f + L_k^\phi}$, we can combine the above two approximations and update x^{k+1} as

$$\begin{aligned} x^{k+1} &:= \arg\min_{x\in\mathbb{R}^p} \left\{ h(x) + \langle \nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x - \hat{x}^k\rangle + \tfrac{1}{2\beta_k}\|x - \hat{x}^k\|^2 \right\} \\ &= \mathrm{prox}_{\beta_k h}\left(\hat{x}^k - \beta_k[\nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k)]\right). \end{aligned} \tag{3.23}$$

   Here, the actual value of $\beta_k > 0$ will be appropriately updated in our analysis.

3. To accelerate the descent progress on the primal variable, we update $\hat{x}^k$ by applying Nesterov's acceleration technique [89]:

$$\hat{x}^{k+1} := x^{k+1} + \frac{\tau_{k+1}(1 - \tau_k)}{\tau_k}(x^{k+1} - x^k),$$

   where the step-size $\tau_k \in (0, 1]$ will be updated appropriately.

4. We update the dual variable $\tilde{y}^k$ as follows:

$$\tilde{y}^{k+1} := \mathrm{proj}_{\mathcal{B}_k}\left(\tilde{y}^k + \eta_k\left([g(x^{k+1}) + s^{k+1}] - (1 - \tau_k)[g(x^k) + s^k]\right)\right), \tag{3.24}$$

where $\mathcal{B}_k \subseteq \mathbb{R}^m$ is a norm ball, which will be specified later.

5. Finally, we define the dual variable

$$y^{k+1} := \mathrm{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right). \tag{3.25}$$

However, by Moreau's identity,

$$\rho_k s^{k+1} \overset{(3.22)}{=} \mathrm{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right) - [\tilde{y}^k + \rho_k g(\hat{x}^k)] = y^{k+1} - [\tilde{y}^k + \rho_k g(\hat{x}^k)]. \tag{3.26}$$

Thus, we can in fact eliminate variable s^{k+1} from the expression of x^{k+1} in (3.23) by noting that $\nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) = [\nabla g(\hat{x}^k)]^\top y^{k+1}$. Similarly, the presence of s^k and s^{k+1} in the update of $\tilde{y}^{k+1}$ in (3.24) can also be eliminated. In this way, we can make our algorithm into the primal-dual form [20, 38].

Combining the above steps, we arrive at our complete algorithm as in Algorithm 1.

Algorithm 1 Our first primal-dual algorithm: General convex-concave case

1: **Initialization:** Choose an initial primal-dual point $(x^0, y^0) \in \mathbb{R}^p \times \mathbb{R}^m$.
2: Set $\hat{x}^0 := x^0$, $\tilde{y}^0 := y^0$, and $\Theta_0 := 0$.
3: Choose appropriate initial parameters, according to (3.43), (3.50), or (3.54).
4: **For** $k = 0$ **to** $k_{\max}$
5: Update the parameters according to (3.44), (3.51), or (3.55), consistent with Step 3.
6: Update $(x^k, \hat{x}^k, y^k, \tilde{y}^k)$ as follows:

$$\begin{cases} y^{k+1} & := \mathrm{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right), \\ x^{k+1} & := \mathrm{prox}_{\beta_k h}\left(\hat{x}^k - \beta_k\left(\nabla f(\hat{x}^k) + [\nabla g(\hat{x}^k)]^\top y^{k+1}\right)\right), \\ \hat{x}^{k+1} & := x^{k+1} + \frac{\tau_{k+1}(1-\tau_k)}{\tau_k}(x^{k+1} - x^k), \\ \Theta_{k+1} & := g(x^{k+1}) - g(\hat{x}^k) + \frac{1}{\rho_k}(y^{k+1} - \tilde{y}^k), \\ \tilde{y}^{k+1} & := \mathrm{proj}_{\mathcal{B}_k}\left(\tilde{y}^k + \eta_k[\Theta_{k+1} - (1-\tau_k)\Theta_k]\right). \end{cases} \tag{3.27}$$

7: **EndFor**

Per-iteration complexity. We analyze the per-iteration complexity of Step 6 in Algorithm 1. The dominate computation includes:

1. The first line requires one function evaluation of g and a proximal operation of H^*.
2. The second line needs to compute one Jacobian $\nabla g(\hat{x}^k)$, one gradient ∇f, and one proximal

operation of h.

3. The fourth line essentially uses one function evaluation of g at x^{k+1}.
4. The fifth line requires one projection on $\mathcal{B}_k$ if necessary, i.e., when $\mathcal{B}_k \neq \mathbb{R}^m$.

This break-down of complexity shows that Algorithm 1 essentially has the same complexity as other state-of-the-art algorithms of the same type [57, 64, 88]. However, [64] assumes strong convexity of f, and [57, 88] does not separate functions f and h, thus their subproblems corresponding to the second line of (3.27) could be non-trivial to solve.

3.3.2 Convergence rate analysis

The following lemma provides a recursive inequality based on scheme (3.27), and will serve as a key estimate to analyze the global convergence rates of Algorithm 1.

Lemma 3.2. *Define $\mathcal{L}$ as in (3.11), $\mathcal{L}_\rho$ as in (3.15), and L_f, M_g, and $\mathbf{L}_g$ as in Assumption 3.2. Let $\{(x^k, \hat{x}^k, y^k, \tilde{y}^k)\}$ be generated by (3.27) with $\tau_k \in (0, 1]$ and $\rho_k > \eta_k$. Let $\{s^k\}$ be defined in (3.22). Further introduce*

$$L_k := \mathbf{L}_g(y^{k+1}), \quad \tilde{x}^k := \frac{1}{\tau_k}[\hat{x}^k - (1-\tau_k)x^k], \quad \text{and} \quad \breve{y}^{k+1} := (1-\tau_k)\breve{y}^k + \tau_k y^{k+1}. \tag{3.28}$$

Then, for all $k \in \mathbb{N}$ and for any $(x, s, y) \in \mathbb{R}^p \times \mathbb{R}^m \times \mathcal{B}_k$, it holds that

$$\begin{aligned} \mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, y) - \mathcal{L}(x, s, \breve{y}^{k+1}) &\leq (1-\tau_k)[\mathcal{L}_{\rho_{k-1}}(x^k, s^k, y) - \mathcal{L}(x, s, \breve{y}^k)] \\ &+ \tfrac{\tau_k^2}{2\beta_k}(\|\tilde{x}^k - x\|^2 - \|\tilde{x}^{k+1} - x\|^2) + \tfrac{1}{2\eta_k}(\|\tilde{y}^k - y\|^2 - \|\tilde{y}^{k+1} - y\|^2) \\ &- \tfrac{(1-\tau_k)[\rho_{k-1} - (1-\tau_k)\rho_k]}{2}\|g(x^k) + s^k\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\beta_k} - L_k - L_f - \tfrac{\rho_k^2 M_g^2}{\rho_k - \eta_k}\right)\|x^{k+1} - \hat{x}^k\|^2. \end{aligned} \tag{3.29}$$

Proof. For readability, we first claim that for any $(x, s) \in \mathbb{R}^p \times \mathbb{R}^m$,

$$\begin{aligned} \mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \leq\ & \mathcal{L}_{\rho_k}(x, s, \tilde{y}^k) + \tfrac{1}{\beta_k}\langle x^{k+1} - \hat{x}^k,\ x - x^{k+1}\rangle \\ & - \tfrac{\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2 + \tfrac{L_k + L_f + \rho_k M_g^2}{2}\|x^{k+1} - \hat{x}^k\|^2. \end{aligned} \tag{3.30}$$

The proof of this claim is deferred as Lemma B.2 in Appendix B.2.

Plugging $(x, s) := (x^k, s^k)$ in (3.30), we obtain

$$\begin{aligned}\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},\tilde{y}^k) \le\ & \mathcal{L}_{\rho_k}(x^k,s^k,\tilde{y}^k)+\tfrac{1}{\beta_k}\langle x^{k+1}-\hat{x}^k,\ x^k-x^{k+1}\rangle\\ & -\tfrac{\rho_k}{2}\|[g(x^k)+s^k]-[g(\hat{x}^k)+s^{k+1}]\|^2+\tfrac{L_k+L_f+\rho_k M_g^2}{2}\|x^{k+1}-\hat{x}^k\|^2.\end{aligned}$$

Now, multiplying the above estimate above by $1-\tau_k\in[0,1)$, and (3.30) by $\tau_k\in(0,1]$, and then summing up the results, we get

$$\begin{aligned}\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},\tilde{y}^k) \le\ & (1-\tau_k)\mathcal{L}_{\rho_k}(x^k,s^k,\tilde{y}^k)+\tau_k\mathcal{L}_{\rho_k}(x,s,\tilde{y}^k)+\tfrac{\tau_k^2}{\beta_k}\langle\tilde{x}^{k+1}-\tilde{x}^k,\ x-\tilde{x}^{k+1}\rangle\\ & -\tfrac{(1-\tau_k)\rho_k}{2}\|[g(x^k)+s^k]-[g(\hat{x}^k)+s^{k+1}]\|^2\\ & -\tfrac{\tau_k\rho_k}{2}\|[g(x)+s]-[g(\hat{x}^k)+s^{k+1}]\|^2+\tfrac{L_k+L_f+\rho_k M_g^2}{2}\|x^{k+1}-\hat{x}^k\|^2,\end{aligned}\tag{3.31}$$

where we have used $(1-\tau_k)x^k+\tau_k x-x^{k+1}=\tau_k(x-\tilde{x}^{k+1})$ and $x^{k+1}-\hat{x}^k=\tau_k(\tilde{x}^{k+1}-\tilde{x}^k)$ derived from the definition of $\tilde{x}^k$ in (3.28).

Next, by the definition of $\mathcal{L}_{\rho_k}$, for any $y\in\mathcal{B}_k$, we have

$$\begin{aligned}&\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},y)-(1-\tau_k)\mathcal{L}_{\rho_k}(x^k,s^k,y)\\ &\quad=\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},\tilde{y}^k)-(1-\tau_k)\mathcal{L}_{\rho_k}(x^k,s^k,\tilde{y}^k)\\ &\qquad+\underbrace{\langle y-\tilde{y}^k,\ [g(x^{k+1})+s^{k+1}]-(1-\tau_k)[g(x^k)+s^k]\rangle}_{\mathcal{T}_1}.\end{aligned}\tag{3.32}$$

To analyze the last term $\mathcal{T}_1$ in (3.32), we denote

$$u^{k+1}:=\eta_k\left([g(x^{k+1})+s^{k+1}]-(1-\tau_k)[g(x^k)+s^k]\right)\overset{(3.26)}{=}\eta_k[\Theta_{k+1}-(1-\tau_k)\Theta_k].\tag{3.33}$$

Then, by the update of $\tilde{y}^{k+1}$ in (3.27) and the fact that $y\in\mathcal{B}_k$, we can use the non-expansive property of the projection $\mathrm{proj}_{\mathcal{B}_k}$ to get

$$\|\tilde{y}^{k+1}-y\|=\|\mathrm{proj}_{\mathcal{B}_k}(\tilde{y}^k+u^{k+1})-\mathrm{proj}_{\mathcal{B}_k}(y)\|\le\|\tilde{y}^k+u^{k+1}-y\|.\tag{3.34}$$

Therefore, $\mathcal{T}_1$ becomes

$$\begin{aligned}\mathcal{T}_1 &\overset{(3.33)}{=}\ \tfrac{1}{\eta_k}\langle y-\tilde{y}^k,\ u^{k+1}\rangle=\tfrac{1}{\eta_k}\langle\tilde{y}^k-y,\ (y-u^{k+1})-y\rangle\\ &=\ \tfrac{1}{2\eta_k}\left(\|\tilde{y}^k-y\|^2+\|y-u^{k+1}-y\|^2-\|\tilde{y}^k+u^{k+1}-y\|^2\right)\\ &\overset{(3.34)}{\le}\ \tfrac{1}{2\eta_k}(\|\tilde{y}^k-y\|^2-\|\tilde{y}^{k+1}-y\|^2+\|u^{k+1}\|^2).\end{aligned}\tag{3.35}$$

Substituting (3.35) into (3.32), and then combining with (3.31), we can further derive

$$\begin{aligned}\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},y) \le\ & \overbrace{(1-\tau_k)\mathcal{L}_{\rho_k}(x^k,s^k,y)}^{\mathcal{T}_2}\\ &+\underbrace{\tau_k\mathcal{L}_{\rho_k}(x,s,\tilde{y}^k)-\tfrac{\tau_k\rho_k}{2}\|[g(x)+s]-[g(\hat{x}^k)+s^{k+1}]\|^2}_{\mathcal{T}_3}\\ &+\underbrace{\tfrac{\tau_k^2}{\beta_k}\langle\tilde{x}^{k+1}-\tilde{x}^k,\ x-\tilde{x}^{k+1}\rangle+\tfrac{L_k+L_f+\rho_kM_g^2}{2}\|x^{k+1}-\hat{x}^k\|^2}_{\mathcal{T}_4}\\ &+\tfrac{1}{2\eta_k}(\|\tilde{y}^k-y\|^2-\|\tilde{y}^{k+1}-y\|^2)+\tfrac{1}{2\eta_k}\|u^{k+1}\|^2\\ &-\tfrac{(1-\tau_k)\rho_k}{2}\|[g(x^k)+s^k]-[g(\hat{x}^k)+s^{k+1}]\|^2.\end{aligned}\tag{3.36}$$

We now estimate the terms $\mathcal{T}_2$, $\mathcal{T}_3$, and $\mathcal{T}_4$ above. It is easy to see that

$$\mathcal{T}_2=(1-\tau_k)\left[\mathcal{L}_{\rho_{k-1}}(x^k,s^k,y)+\frac{\rho_k-\rho_{k-1}}{2}\|g(x^k)+s^k\|^2\right].\tag{3.37}$$

By (3.26) as well as definition of $\breve{y}^{k+1}$ in (3.28), we have

$$\mathcal{T}_3=\mathcal{L}(x,s,\breve{y}^{k+1})-(1-\tau_k)\mathcal{L}(x,s,\breve{y}^k)-\frac{\tau_k\rho_k}{2}\|g(\hat{x}^k)+s^{k+1}\|^2.\tag{3.38}$$

Using the relation $\tilde{x}^{k+1}-\tilde{x}^k=\frac{1}{\tau_k}(x^{k+1}-\hat{x}^k)$, we further have

$$\begin{aligned}\mathcal{T}_4 &= \tfrac{\tau_k^2}{2\beta_k}(\|\tilde{x}^k-x\|^2-\|\tilde{x}^{k+1}-x\|^2-\|\tilde{x}^{k+1}-\tilde{x}^k\|^2)+\tfrac{L_k+L_f+\rho_kM_g^2}{2}\|x^{k+1}-\hat{x}^k\|^2\\ &= \tfrac{\tau_k^2}{2\beta_k}(\|\tilde{x}^k-x\|^2-\|\tilde{x}^{k+1}-x\|^2)-\tfrac{1}{2}\left(\tfrac{1}{\beta_k}-L_k-L_f-\rho_kM_g^2\right)\|x^{k+1}-\hat{x}^k\|^2.\end{aligned}\tag{3.39}$$

Substituting (3.37)-(3.39) into (3.36), we get

$$\begin{aligned}\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},y)-\mathcal{L}(x,s,\breve{y}^{k+1}) \le\ & (1-\tau_k)[\mathcal{L}_{\rho_{k-1}}(x^k,s^k,y)-\mathcal{L}(x,s,\breve{y}^k)]\\ &+\tfrac{\tau_k^2}{2\beta_k}(\|\tilde{x}^k-x\|^2-\|\tilde{x}^{k+1}-x\|^2)\\ &+\tfrac{1}{2\eta_k}(\|\tilde{y}^k-y\|^2-\|\tilde{y}^{k+1}-y\|^2)\\ &-\tfrac{1}{2}\left(\tfrac{1}{\beta_k}-L_k-L_f-\rho_kM_g^2\right)\|x^{k+1}-\hat{x}^k\|^2+\mathcal{T}_5,\end{aligned}\tag{3.40}$$

where

$$
\begin{aligned}
\mathcal{T}_5 \;:=\; & \tfrac{1}{2\eta_k}\|u^{k+1}\|^2 + \tfrac{(1-\tau_k)(\rho_k-\rho_{k-1})}{2}\|g(x^k)+s^k\|^2 - \tfrac{\tau_k\rho_k}{2}\|g(\hat{x}^k)+s^{k+1}\|^2 \\
& -\tfrac{(1-\tau_k)\rho_k}{2}\|[g(x^k)+s^k]-[g(\hat{x}^k)+s^{k+1}]\|^2 \\
=\; & \tfrac{1}{2\eta_k}\|u^{k+1}\|^2 - \tfrac{\rho_k}{2}\|[g(\hat{x}^k)+s^{k+1}]-(1-\tau_k)[g(x^k)+s^k]\|^2 \\
& -\tfrac{(1-\tau_k)[\rho_{k-1}-(1-\tau_k)\rho_k]}{2}\|g(x^k)+s^k\|^2 \\
\leq\; & \tfrac{\rho_k\eta_k}{2(\rho_k-\eta_k)}\|g(x^{k+1})-g(\hat{x}^k)\|^2 - \tfrac{(1-\tau_k)[\rho_{k-1}-(1-\tau_k)\rho_k]}{2}\|g(x^k)+s^k\|^2 \\
\overset{(3.6)}{\leq}\; & \tfrac{\rho_k\eta_k M_g^2}{2(\rho_k-\eta_k)}\|x^{k+1}-\hat{x}^k\|^2 - \tfrac{(1-\tau_k)[\rho_{k-1}-(1-\tau_k)\rho_k]}{2}\|g(x^k)+s^k\|^2,
\end{aligned}
\tag{3.41}
$$

where in the first inequality (second to last line) above, we have used Lemma B.1(1) and $\rho_k > \eta_k$. Substituting (3.41) into (3.40), we eventually get

$$
\begin{aligned}
\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},y) - \mathcal{L}(x,s,\breve{y}^{k+1}) \leq\; & (1-\tau_k)[\mathcal{L}_{\rho_{k-1}}(x^k,s^k,y) - \mathcal{L}(x,s,\breve{y}^k)] \\
& + \tfrac{\tau_k^2}{2\beta_k}(\|\tilde{x}^k-x\|^2 - \|\tilde{x}^{k+1}-x\|^2) + \tfrac{1}{2\eta_k}(\|\tilde{y}^k-y\|^2 - \|\tilde{y}^{k+1}-y\|^2) \\
& - \tfrac{1}{2}\left(\tfrac{1}{\beta_k} - L_k - L_f - \rho_k M_g^2 - \tfrac{\rho_k\eta_k M_g^2}{\rho_k-\eta_k}\right)\|x^{k+1}-\hat{x}^k\|^2 \\
& - \tfrac{(1-\tau_k)[\rho_{k-1}-(1-\tau_k)\rho_k]}{2}\|g(x^k)+s^k\|^2,
\end{aligned}
$$

which is exactly (3.29). □

Now, we analyze the convergence rates of Algorithm 1 for three parameter initialization (Step 3) and update (Step 6) options. To abbreviate the notation, given $x^0 \in \mathbb{R}^p$, $y^0 \in \mathbb{R}^m$, and $\beta_0,\ \eta_0 > 0$, we frequently use the following quantity:

$$
\mathcal{R}_0^2(x,y) := \frac{1}{\beta_0}\|x^0-x\|^2 + \frac{1}{\eta_0}\|y^0-y\|^2 \tag{3.42}
$$

to characterize the weighted square-distance from the initial point (x^0,y^0) to (x,y).

3.3.2.1 The $\mathcal{O}\left(\frac{1}{k}\right)$ ergodic convergence rate

The following theorem shows the $\mathcal{O}\left(\frac{1}{k}\right)$ ergodic convergence rate of Algorithm 1.

Theorem 3.1. *Suppose Assumptions 3.1 and 3.2 hold for* (SP). *Let* $\left\{(x^k,y^k)\right\}_{k\geq 0}$ *be generated by Algorithm 1 with the following parameter configurations:*

- Initialization: *Choose* $\rho,\ \beta,\ \eta,\ C > 0$, *and* $\gamma \in (0,1)$ *such that*

$$\begin{cases} \beta := \frac{\gamma}{\gamma L_f + \rho(\gamma C + M_g^2)}, \quad \eta := (1-\gamma)\rho, \\ L_g[\|y^\star\| + (\sqrt{\eta} + \rho\sqrt{\beta} M_g)\mathcal{R}_0(x^\star, y^\star)] \le \rho C. \end{cases} \tag{3.43}$$

- Update: *For all $k \in \mathbb{N}$, fix the parameters at*

$$\tau_k \equiv 1, \quad \rho_k \equiv \rho, \quad \beta_k \equiv \beta, \quad \eta_k \equiv \eta, \quad \textit{and} \quad \mathcal{B}_k \equiv \mathbb{R}^m. \tag{3.44}$$

Let $\left\{(\bar{x}^k, \bar{y}^k)\right\}_{k\ge 1}$ be the ergodic sequence defined as

$$(\bar{x}^k,\ \bar{y}^k) := \frac{1}{k}\sum_{j=1}^{k}(x^j,\ y^j). \tag{3.45}$$

Then, for all $k \ge 1$, the following bounds hold:

$$\begin{cases} \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(\bar{x}^k, \bar{y}^k) & \le \frac{1}{2k} \sup\limits_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \mathcal{R}_0^2(x,y), \\ \mathcal{P}(\bar{x}^k) - \mathcal{P}^\star & \le \frac{1}{2k}\left[\frac{\|x^0 - x^\star\|^2}{\beta} + \frac{(\|y^0\| + M_H)^2}{\eta}\right], \\ \mathcal{D}^\star - \mathcal{D}(\bar{y}^k) & \le \frac{1}{2k}\left[\frac{(\|x^0\| + M_{F^*})^2}{\beta} + \frac{\|y^0 - y^\star\|^2}{\eta}\right], \\ \mathcal{P}(\bar{x}^k) - \mathcal{D}(\bar{y}^k) & \le \frac{1}{2k}\left[\mathcal{R}_0^2(x^\star, y^\star) + \frac{(\|x^0\| + M_{F^*})^2}{\beta} + \frac{(\|y^0\| + M_H)^2}{\eta}\right], \end{cases} \tag{3.46}$$

where $\mathcal{R}_0$ is defined by (3.42), and the $M_H,\ M_{F^*} \in [0,\infty]$ are the Lipschitz constants of H and F^*, respectively.

As a result, Algorithm 1 has $\mathcal{O}\left(\frac{1}{k}\right)$ ergodic convergence rate on the primal objective residual, the dual objective residual, and the primal-dual gaps.

Proof. For readability, we first claim that for all $k \in \mathbb{N}$,

$$L_g(\|y^\star\| + \|\tilde{y}^k - y^\star\| + \rho M_g \|x^k - x^\star\|) \le \rho C. \tag{3.47}$$

The proof of this claim is deferred as Lemma B.3 in Appendix B.2.

By (3.47), we can follow the same lines as (B.8) and (B.9) to show that $\frac{1}{\beta} - L_k - L_f - \frac{\rho^2 M_g^2}{\rho - \eta} \ge 0$. Therefore, similar to (B.10), for any $y \in \mathbb{R}^m$ and any $j \in \mathbb{N}$, we have

$$\mathcal{L}(x^{j+1}, s^{j+1}, y) - \mathcal{L}(x, s, y^{j+1}) \le \frac{1}{2\beta}(\|x^j - x\|^2 - \|x^{j+1} - x\|^2) + \frac{1}{2\eta}(\|\tilde{y}^j - y\|^2 - \|\tilde{y}^{j+1} - y\|^2).$$

Summing up this inequality from $j := 0$ to $j := k-1$, we get

$$\sum_{j=0}^{k-1}[\mathcal{L}(x^{j+1}, s^{j+1}, y) - \mathcal{L}(x, s, y^{j+1})] \le \frac{1}{2}\left(\frac{1}{\beta}\|x^0 - x\|^2 + \frac{1}{\eta}\|y^0 - y\|^2\right) = \frac{\mathcal{R}_0^2(x,y)}{2}.$$

Dividing the above inequality by $k \ge 1$, and using the convexity of $\mathcal{L}$ in x and s, and its concavity in y, with $\bar{x}^k$ and $\bar{y}^k$ defined in (3.45) and $\bar{s}^k := \frac{1}{k}\sum_{j=1}^k s^j$, we get

$$\mathcal{L}(\bar{x}^k, \bar{s}^k, y) - \mathcal{L}(x, s, \bar{y}^k) \overset{(3.45)}{\le} \frac{1}{k}\sum_{j=1}^{k}[\mathcal{L}(x^j, s^j, y) - \mathcal{L}(x, s, y^j)] \le \frac{\mathcal{R}_0^2(x,y)}{2k}. \tag{3.48}$$

Now, by (3.13), we have $\widetilde{\mathcal{L}}(\bar{x}^k, y) \le \mathcal{L}(\bar{x}^k, \bar{s}^k, y)$ and $\widetilde{\mathcal{L}}(x, \bar{y}^k) = \mathcal{L}(x, \breve{s}^k, \bar{y}^k)$ for $\breve{s}^k \in -\partial H^*(\bar{y}^k)$. Hence, $\widetilde{\mathcal{L}}(\bar{x}^k, y) - \widetilde{\mathcal{L}}(x, \bar{y}^k) \le \mathcal{L}(\bar{x}^k, \bar{s}^k, y) - \mathcal{L}(x, \breve{s}^k, \bar{y}^k)$. Substituting $s := \breve{s}^k$ and this inequality into (3.48), we obtain $\widetilde{\mathcal{L}}(\bar{x}^k, y) - \widetilde{\mathcal{L}}(x, \bar{y}^k) \le \frac{\mathcal{R}_0^2(x,y)}{2k}$. Taking the supremum on both sides over $\mathcal{X} \times \mathcal{Y}$ and recalling the definition of $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}$ in (3.9), we prove the first assertion of (3.46).

Next, if H is M_H-Lipschitz continuous, then we let $\breve{y}^k := \frac{M_H}{\|g(\bar{x}^k) + \bar{s}^k\|}[g(\bar{x}^k) + \bar{s}^k]$, and substitute $(x, s, y) := (x^\star, s^\star, \breve{y}^k)$ in (3.48) to get

$$\begin{aligned}
\mathcal{P}(\bar{x}^k) - \mathcal{P}^\star &\overset{(\mathrm{P})}{=} F(\bar{x}^k) + H(g(\bar{x}^k)) - \mathcal{P}^\star \le F(\bar{x}^k) + H(-\bar{s}^k) + |H(g(\bar{x}^k)) - H(-\bar{s}^k)| - \mathcal{P}^\star \\
&\le F(\bar{x}^k) + H(-\bar{s}^k) + M_H|g(\bar{x}^k) + \bar{s}^k| - \mathcal{P}^\star \\
&= F(\bar{x}^k) + H(-\bar{s}^k) + \langle \breve{y}^k,\ g(\bar{x}^k) + \bar{s}^k\rangle - \mathcal{P}^\star \le \mathcal{L}(\bar{x}^k, \bar{s}^k, \breve{y}^k) - \mathcal{P}^\star \overset{(3.48)}{\le} \tfrac{\mathcal{R}_0^2(x^\star, \breve{y}^k)}{2k}.
\end{aligned}$$

Using $\|y^0 - \breve{y}^k\|^2 \le (\|y^0\| + \|\breve{y}^k\|)^2 = (\|y^0\| + M_H)^2$ to upper bound $\mathcal{R}_0^2(x^\star, \breve{y}^k)$ in the last estimate, we obtain the second assertion of (3.46).

On the other hand, let $\breve{x}^k$ satisfy $0 \in [\nabla g(\breve{x}^k)]^\top \bar{y}^k + \partial F(\breve{x}^k)$, then by the form of (D), we have $\mathcal{D}(\bar{y}^k) = \widetilde{\mathcal{L}}(\breve{x}^k, \bar{y}^k) = \mathcal{L}(\breve{x}^k, \breve{s}^k, \bar{y}^k)$ for $\breve{s}^k \in -\partial H^*(\bar{y}^k)$. Moreover, notice that $\mathcal{D}^\star = \mathcal{L}(x^\star, s^\star, y^\star) \le \mathcal{L}(\bar{x}^k, \bar{s}^k, y^\star)$ in (3.14). Therefore, substituting $(x, s, y) := (\breve{x}^k, \breve{s}^k, y^\star)$ into (3.48), we can derive

$$\mathcal{D}^\star - \mathcal{D}(\bar{y}^k) \le \mathcal{L}(\bar{x}^k, \bar{s}^k, y^\star) - \mathcal{L}(\breve{x}^k, \breve{s}^k, \bar{y}^k) \overset{(3.48)}{\le} \frac{\mathcal{R}_0^2(\breve{x}^k, y^\star)}{2k}.$$

Since $0 \in [\nabla g(\breve{x}^k)]^\top \bar{y}^k + \partial F(\breve{x}^k)$, we have $\breve{x}^k \in \partial F^*\left(-[\nabla g(\breve{x}^k)]^\top \bar{y}^k\right)$. If F^* is M_{F^*}-Lipschitz continuous, then $\|\breve{x}^k\| = \left\|\nabla F^*\left(-[\nabla g(\breve{x}^k)]^\top \bar{y}^k\right)\right\| \le M_{F^*}$, thus $\|x^0 - \breve{x}^k\|^2 \le (\|x^0\| + M_{F^*})^2$. Substituting this into $\mathcal{R}_0^2(\breve{x}^k, y^\star)$ of the last inequality leads to the third assertion of (3.46).

Finally, combining the second and third assertions of (3.46), we have immediately proved the last assertion on the primal-dual gap $\mathcal{P}(\bar{x}^k) - \mathcal{D}(\bar{y}^k)$. □

The first convergence guarantee on $\mathcal{G}_{\mathcal{X}\times\mathcal{Y}}$ in (3.46) is independent of M_H and M_{F^*}, while the last one depends on both M_H and M_{F^*}. Note that under the update rule (3.44), Step 6 of Algorithm 1 can be simply written as

$$\begin{cases} y^{k+1} := \operatorname{prox}_{\rho H^*}\left(\tilde{y}^k + \rho g(x^k)\right), \\ x^{k+1} := \operatorname{prox}_{\beta h}\left(x^k - \beta\left(\nabla f(x^k) + [\nabla g(x^k)]^\top y^{k+1}\right)\right), \\ \tilde{y}^{k+1} := \tilde{y}^k + \eta\left[g(x^{k+1}) - g(x^k) + \frac{1}{\rho}(y^{k+1} - \tilde{y}^k)\right]. \end{cases}$$

This scheme requires one proximal operation of H^* and h each, one evaluation of g, one evaluation of gradient ∇f and one evaluation of Jocobian ∇g. If $H = \delta_{\mathbb{R}^m_+\times\{0\}^n}$, the indicator of $\mathbb{R}^m_+ \times \{0\}^n$, then this scheme is similar to [144, Algorithm 1] for solving (1.2). However, our dual step $\tilde{y}^k$ is still different from the algorithm in [144].

Remark 3.1 (Initialization in (3.43))**.** *In fact, for any choice of $\rho > 0$ and $\gamma \in (0, 1)$, we can find $C > 0$ that satisfies* (3.43)*. For example, we can simply set*

$$\rho := 1, \quad \gamma := \frac{1}{2}, \quad \text{and} \quad C := \max\{L_f + 2M_g^2 + 2,\ L_g D(L_g D + 4M_g + 2)\}, \tag{3.49}$$

where $D \geq \max\{\|x^0 - x^\star\|,\ \|y^0 - y^\star\|,\ \|y^\star\|\}$ is an upper estimate. As shown in Appendix B.2, the choice given in (3.49) *is feasible to* (3.43)*. Notice that C presented in* (3.49) *is not tight, since we have loosened this estimate to get simple expressions. One may choose different ρ's and smaller C's, which also solve* (3.43)*, for better practical performance.* □

Remark 3.2 (Optimal rate)**.** *It was shown in [75, Section 5] and [142, Theorem 1] that, under Assumption 3.1, the rate $\mathcal{O}\left(\frac{1}{k}\right)$ is optimal, in the sense that for any algorithm $\mathcal{A}$ for solving* (P)*, in order to achieve the bound $\mathcal{P}(x^k) - \mathcal{P}^\star < \varepsilon$, there exists an instance of functions F, H and g, with their arguments' dimensions p and m dependent on ε, such that $\mathcal{A}$ makes $\Omega\left(\frac{1}{\varepsilon}\right)$ queries to the first-order oracle of F and $H \circ g$. In other words, the convergence rate of $\mathcal{A}$ can not exceed $\mathcal{O}\left(\frac{1}{k}\right)$ rate under Assumption 3.1 when the problem dimensions p and m are much larger than the number of iterations k. Consequently, Algorithm 1 indeed achieves optimal convergence rate.* □

3.3.2.2 The $\mathcal{O}\left(\frac{1}{k}\right)$ semi-ergodic convergence rate

The following theorem shows $\mathcal{O}\left(\frac{1}{k}\right)$ semi-ergodic convergence rate of Algorithm 1 for solving (SP) using the last-iterate sequence $\{x^k\}$ and on the averaging sequence $\{\bar{y}^k\}$.

Theorem 3.2. *Suppose Assumptions 3.1 and 3.2 hold for* (SP)*, and assume*

1. $\|g(x)\| \leq B_g$ *for all* $x \in \mathrm{dom} g \cap \mathrm{dom} F$ *for some* $B_g \in [0, \infty]$ *such that* $L_g B_g < +\infty$*. In particular, if* g *is affine, then* $L_g = 0$*, and we allow* $B_g = \infty$ *(i.e., in this case, no boundedness on* g *is required).*
2. *there exist* $y_* \in \partial H(0)$ *and* $s_* \in -\partial H^*(0)$*.*

Let $\{(x^k, y^k)\}_{k\geq 0}$ *be generated by Algorithm 1 with the following parameter configurations:*

- Initialization: *Choose*

$$\rho_0 > 0 \quad \textit{and} \quad \gamma \in (0, 1). \tag{3.50}$$

- Update: *For all* $k \in \mathbb{N}$*, fix* $\mathcal{B}_k \equiv \mathbb{R}^m$*, and update*

$$\begin{cases} \tau_k := \frac{1}{k+1}, \quad \rho_k := \frac{\rho_0}{\tau_k}, \quad \eta_k := (1-\gamma)\rho_k, \quad \textit{and} \\ \beta_k := \frac{\gamma}{\gamma(L_f + 2L_g\|y_*\|) + \rho_k\left(L_g\left[\frac{\|y^0\|}{\rho_0} + (2-\gamma)B_g + 2(1-\gamma)\|s_*\|\right] + M_g^2\right)}. \end{cases} \tag{3.51}$$

Let $\{\bar{y}^k\}_{k\geq 1}$ *be the ergodic sequence defined in* (3.45)*. Then, for* $k \geq 1$*, the following holds:*

$$\begin{cases} \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x^k, \bar{y}^k) & \leq \frac{1}{2k} \sup\limits_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \mathcal{R}_0^2(x, y), \\ \mathcal{P}(x^k) - \mathcal{P}^\star & \leq \frac{1}{2k}\left[\frac{\|x^0 - x^\star\|^2}{\beta_0} + \frac{(\|y^0\| + M_H)^2}{\eta_0}\right], \\ \mathcal{D}^\star - \mathcal{D}(\bar{y}^k) & \leq \frac{1}{2k}\left[\frac{(\|x^0\| + M_{F^*})^2}{\beta_0} + \frac{\|y^0 - y^\star\|^2}{\eta_0}\right], \\ \mathcal{P}(x^k) - \mathcal{D}(\bar{y}^k) & \leq \frac{1}{2k}\left[\mathcal{R}_0^2(x^\star, y^\star) + \frac{(\|x^0\| + M_{F^*})^2}{\beta_0} + \frac{(\|y^0\| + M_H)^2}{\eta_0}\right], \end{cases}$$

where $\mathcal{R}_0$ *is defined in* (3.42)*, and the* $M_H,\ M_{F^*} \in [0, \infty]$ *are the Lipschitz constants of* H *and* F^**, respectively.*

As a result, Algorithm 1 has $\mathcal{O}\left(\frac{1}{k}\right)$ *non-ergodic convergence rate on primal objective residual, ergodic rate on dual objective residual, and semi-ergodic rate on primal-dual gaps.*

Proof. For readability, we first claim that for $k \in \mathbb{N}$,

$$\frac{\|\tilde{y}^k\|}{\rho_k} \leq \frac{1}{\gamma}\left[\frac{\|y^0\|}{\rho_0} + 2(1-\gamma)(B_g + \|s_*\|)\right]. \tag{3.52}$$

The proof of this claim is deferred as Lemma B.4 in Appendix B.2.

By definition of y_*, we have $y_* = \mathrm{prox}_{\rho_k H^*}(y_*)$ for any $\rho_k > 0$. Using this relation, the non-expansiveness of $\mathrm{prox}_{\rho_k H^*}$, and (3.52), we can prove that

$$\begin{aligned} L_k &\overset{(3.28)}{=} \mathbf{L}_g(y^{k+1}) \overset{(3.25)}{\leq} L_g\|\mathrm{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right)\| \\ &= L_g\|\mathrm{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right) - \mathrm{prox}_{\rho_k H^*}(y_*) + y_*\| \\ &\leq L_g\left(2\|y_*\| + \|\tilde{y}^k + \rho_k g(\hat{x}^k)\|\right) \\ &\overset{(3.52)}{\leq} L_g\left(2\|y_*\| + \rho_k B_g + \tfrac{\rho_k}{\gamma}\left[\tfrac{\|y^0\|}{\rho_0} + 2(1-\gamma)(B_g + \|s_*\|)\right]\right). \end{aligned} \tag{3.53}$$

Therefore, by the update rule of β_k and η_k in (3.51), and (3.53), we can easily show that

$$\begin{aligned} &\tfrac{1}{\beta_k} - L_k - L_f - \tfrac{\rho_k^2 M_g^2}{\rho_k - \eta_k} \geq \tfrac{\rho_k}{\gamma}\left[L_g\left(\tfrac{\|y^0\|}{\rho_0} + (2-\gamma)B_g + 2(1-\gamma)\|s_*\|\right) + M_g^2\right] \\ &\quad L_f + 2L_g\|y_*\| - L_g\left[2\|y_*\| + \tfrac{\rho_k}{\gamma}\left(\tfrac{\|y^0\|}{\rho_0} + (2-\gamma)B_g + 2(1-\gamma)\|s_*\|\right)\right] - L_f - \tfrac{\rho_k M_g^2}{\gamma} \\ &= \tfrac{\rho_k}{\gamma}\left[L_g\left(\tfrac{\|y^0\|}{\rho_0} + (2-\gamma)B_g\right) + M_g^2 - L_g\left(\tfrac{\|y^0\|}{\rho_0} + (2-\gamma)B_g\right) - M_g^2\right] = 0. \end{aligned}$$

Furthermore, other conditions in (3.51) ensure that

$$\rho_k > \eta_k, \quad \frac{1}{2\eta_k} = \frac{1-\tau_k}{2\eta_{k-1}}, \quad \rho_{k-1} - (1-\tau_k)\rho_k = 0, \quad \text{and} \quad \frac{\tau_k^2}{2\beta_k} \leq \frac{(1-\tau_k)\tau_{k-1}^2}{2\beta_{k-1}}.$$

Utilizing these relations, we can simplify estimate (3.29) of Lemma 3.2 to get

$$\begin{aligned} &\mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, y) - \mathcal{L}(x, s, \bar{y}^{k+1}) + \tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^{k+1} - x\|^2 + \tfrac{1}{2\eta_k}\|\tilde{y}^{k+1} - y\|^2 \\ &\qquad \leq (1-\tau_k)\left[\mathcal{L}_{\rho_{k-1}}(x^k, s^k, y) - \mathcal{L}(x, s, \bar{y}^k) + \tfrac{\tau_{k-1}^2}{2\beta_{k-1}}\|\tilde{x}^k - x\|^2 + \tfrac{1}{2\eta_{k-1}}\|\tilde{y}^k - y\|^2\right], \end{aligned}$$

for any $(x, s, y) \in \mathbb{R}^p \times \mathbb{R}^m \times \mathbb{R}^m$. Here, we have used the fact that $\bar{y}^k$ as defined in (3.45) is equal to $\breve{y}^k$ as defined in (3.28), since $\tau_k = \frac{1}{k+1}$. By induction, this inequality implies that

$$\begin{aligned} &\mathcal{L}(x^k, s^k, y) - \mathcal{L}(x, s, \bar{y}^k) \leq \mathcal{L}_{\rho_{k-1}}(x^k, s^k, y) - \mathcal{L}(x, s, \bar{y}^k) \\ &\quad \leq \left[\textstyle\prod_{j=1}^{k-1}(1-\tau_j)\right]\left[\mathcal{L}_{\rho_0}(x^1, s^1, y^\star) - \mathcal{L}(x, s, \bar{y}^1) + \tfrac{\tau_0^2}{2\beta_0}\|\tilde{x}^1 - x\|^2 + \tfrac{1}{2\eta_0}\|\tilde{y}^1 - y\|^2\right] \\ &\quad \overset{(3.29)(3.51)}{\leq} \tfrac{1}{k}\left((1-\tau_0)[\mathcal{L}_{\rho_0}(x^0, s^0, y^\star) - \mathcal{L}(x, s, y^0)] + \tfrac{\tau_0^2}{2\beta_0}\|\tilde{x}^0 - x\|^2 + \tfrac{1}{2\eta_0}\|y^0 - y\|^2\right) \\ &\quad \overset{\tau_0=1}{=} \tfrac{1}{2k}\left(\tfrac{1}{\beta_0}\|x^0 - x\|^2 + \tfrac{1}{\eta_0}\|y^0 - y\|^2\right) = \tfrac{\mathcal{R}_0^2(x,y)}{2k}. \end{aligned}$$

Take $\check{s}^k \in -\partial H^*(y^k)$. Using the argument immediately following (3.48), we can show that

$$\widetilde{\mathcal{L}}(x^k, y) - \widetilde{\mathcal{L}}(x, \bar{y}^k) \leq \mathcal{L}(x^k, s^k, y) - \mathcal{L}(x, \check{s}^k, \bar{y}^k) \leq \frac{\mathcal{R}_0^2(x, y)}{2k}.$$

The rest of the proof of Theorem 3.2 is similar to the lines after (3.48) in the proof of Theorem 3.1, except that we replace $\bar{x}^k$ there by x^k. Thus we omit the verbatim here. □

Remark 3.3. *Condition 1 in Theorem 3.2 is not a strong assumption. When H^* is separable in y, e.g., when $H^*(y) = \delta_{\mathbb{R}^m_+}(y)$, the indicator of non-negative orthant, then condition $L_g B_g < +\infty$ can be relaxed to $\sum_{i=1}^m L_{g_i} B_{g_i} < +\infty$, where L_{g_i} is the Lipschitz smoothness modulus of g_i, and B_{g_i} is the bound for g_i. Therefore, our condition allows both linear (where $L_{g_i} = 0$) and bounded nonlinear constraint functions.* □

Remark 3.4 (Symmetry)**.** *If g is linear, then the primal-dual problems* (P) *and* (D) *are symmetric. Therefore, to obtain a non-ergodic convergence rate on the dual problem* (D)*, we could apply Algorithm 1 to the dual-primal pair instead of the primal-dual pair.* □

3.3.2.3 The $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ non-ergodic convergence rate

We show in Theorem 3.3 that if we modify the update rule of τ_k, then we can boost the convergence rate of Algorithm 1 up to $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ in the non-ergodic sense on the primal objective residual, where $\underline{o}(\cdot)$ is defined in (3.3). Here, since $\underline{o}$-rate is not necessarily strictly faster than $\mathcal{O}$-rate, we use the "min" to imply that our rate is no slower than $\mathcal{O}\left(\frac{1}{k}\right)$.[4]

Theorem 3.3. *Suppose Assumptions 3.1 and 3.2 hold for* (SP)*, and assume*

1. *$\|g(x)\| \leq B_g$ for all $x \in \mathrm{dom} g \cap \mathrm{dom} F$ for some $B_g \in [0,\ \infty]$ such that $L_g B_g < +\infty$. In particular, if g is affine, then $L_g = 0$, and we allow $B_g = \infty$.*
2. *there exists $y_* \in \partial H(0)$.*

Let $\{(x^k, y^k)\}_{k\geq 0}$ be generated from Algorithm 1 with the following parameter configurations:

- Initialization: *Choose*

[4] In fact, the numerical experiments in Section 3.6 shows that the parameter update provided in Theorem 3.3 greatly boosts the performance of Algorithm 1.

$$\rho_0 > 0, \quad \gamma \in (0,1), \quad c > 1, \quad \textit{and} \quad R_y \geq \frac{\|y^\star\|}{\rho_0}. \tag{3.54}$$

- Update: *For $k \in \mathbb{N}$, set*

$$\begin{cases} \tau_k := \frac{c}{k+c}, \quad \rho_k := \frac{\rho_0}{\tau_k}, \quad \eta_k := (1-\gamma)\rho_k, \quad \mathcal{B}_k := \{y \mid \|y\| \leq \rho_k R_y\}, \\ \textit{and} \quad \beta_k := \frac{\gamma}{\gamma(L_f + 2L_g\|y_*\|) + \rho_k\left[\gamma L_g(R_y + B_g) + M_g^2\right]}. \end{cases} \tag{3.55}$$

Then, the following bounds hold:

$$\mathcal{P}(x^k) - \mathcal{P}^\star \leq \frac{R_P^2}{k+c-1} \quad \textit{for} \quad \forall k \geq 1 \quad \textit{and} \quad \liminf_{k\to\infty} k\sqrt{\log k}[\mathcal{P}(x^k) - \mathcal{P}^\star] = 0, \tag{3.56}$$

where $R_P^2 := \Delta_0^2 + \sqrt{2c/\rho_0}(\|y^\star\| + M_H)\Delta_0$, where $\Delta_0^2 := (c-1)\left[\mathcal{P}(x^0) - \mathcal{P}^\star\right] + \frac{c}{2}\mathcal{R}_0^2(x^\star, y^\star)$, $\mathcal{R}_0$ is defined in (3.42)*, and $M_H \in [0,\infty]$ is the Lipschitz constant of H.*

As a result, Algorithm 1 for solving (P) *has* $\min\left\{\mathcal{O}\left(\frac{1}{k}\right), \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ *non-ergodic convergence rate on the primal objective residual.*

Proof. By the definition of $\mathcal{B}_k$ in (3.55) and the projection step of $\tilde{y}^k$, we have $\|\tilde{y}^k\| \leq \rho_{k-1}R_y$. Similar to (3.53), by definition of y_*, we can show that

$$\begin{aligned} L_k &\leq L_g\left(2\|y_*\| + \|\tilde{y}^k + \rho_k g(\hat{x}^k)\|\right) \leq L_g(2\|y_*\| + \rho_{k-1}R_y + \rho_k B_g) \\ &\leq L_g[2\|y_*\| + \rho_k(R_y + B_g)], \end{aligned} \tag{3.57}$$

Thus, by the update of β_k and η_k in (3.55), one can show that

$$\begin{aligned} \tfrac{1}{\beta_k} - L_k - L_f - \tfrac{\rho_k^2 M_g^2}{\rho_k - \eta_k} \geq &\left[L_f + 2L_g\|y_*\| + \rho_k L_g(R_y + B_g) + \tfrac{\rho_k M_g^2}{\gamma}\right] \\ &- L_g[2\|y_*\| + \rho_k(R_y + B_g)] - L_f - \tfrac{\rho_k M_g^2}{\gamma} = 0. \end{aligned}$$

Using this inequality and the update rules from (3.55) onto (3.29) of Lemma 3.2, we derive

$$\begin{aligned} \mathcal{L}(x^{k+1}, s^{k+1}, y) - &\mathcal{L}(x, s, \breve{y}^{k+1}) + \tfrac{k+c}{c} \cdot \tfrac{\rho_0}{2}\|g(x^{k+1}) + s^{k+1}\|^2 \\ &\leq \tfrac{k}{k+c}\left[\mathcal{L}(x^k, s^k, y) - \mathcal{L}(x, s, \breve{y}^k) + \tfrac{k+c-1}{c} \cdot \tfrac{\rho_0}{2}\|g(x^k) + s^k\|^2\right] \\ &\quad + \tfrac{c}{k+c} \cdot \tfrac{\tau_k}{2\beta_k}(\|\tilde{x}^k - x\|^2 - \|\tilde{x}^{k+1} - x\|^2) \\ &\quad + \tfrac{c}{k+c} \cdot \tfrac{1}{2\eta_0}(\|\tilde{y}^k - y\|^2 - \|\tilde{y}^{k+1} - y\|^2) - \tfrac{(c-1)k}{c(k+c)} \cdot \tfrac{\rho_0}{2}\|g(x^k) + s^k\|^2. \end{aligned} \tag{3.58}$$

Note that $y^\star \in \mathcal{B}_0 \subseteq \mathcal{B}_k$ by (3.54) and (3.55). Thus, we can substitute $(x, s, y) := (x^\star, s^\star, y^\star)$ into (3.58) while introducing the following notations:

$$\begin{cases} a_k^2 := \frac{\rho_0}{2}\|g(x^k) + s^k\|^2, \quad b_k^2 := \frac{\tau_k}{2\beta_k}\|\tilde{x}^k - x^\star\|^2 + \frac{1}{2\eta_0}\|\tilde{y}^k - y^\star\|^2, \quad \text{and} \\ \widetilde{G}_k := \mathcal{L}(x^k, s^k, y^\star) - \mathcal{L}(x^\star, s^\star, \breve{y}^k) = \mathcal{L}(x^k, s^k, y^\star) - \mathcal{P}^\star \geq 0. \end{cases} \tag{3.59}$$

Then, we can simplify (3.58) as:

$$\begin{aligned} \widetilde{G}_{k+1} + \tfrac{k+c}{c}a_{k+1}^2 &\leq \tfrac{k}{k+c}\left(\widetilde{G}_k + \tfrac{k+c-1}{c}a_k^2\right) - \tfrac{(c-1)k}{c(k+c)}a_k^2 \\ &\quad + \tfrac{c}{k+c}\left[b_k^2 - b_{k+1}^2 + \tfrac{c}{2\beta_k}\left(\tfrac{1}{k+c+1} - \tfrac{1}{k+c}\right)\|\tilde{x}^{k+1} - x^\star\|^2\right] \\ &\leq \tfrac{k}{k+c}\left(\widetilde{G}_k + \tfrac{k+c-1}{c}a_k^2\right) - \tfrac{(c-1)k}{c(k+c)}a_k^2 + \tfrac{c}{k+c}(b_k^2 - b_{k+1}^2). \end{aligned}$$

Multiplying both sides of the last inequality by $k + c$ and rearranging the result, we get

$$\begin{aligned} (c-1)\left(\widetilde{G}_k + \tfrac{k+c-1}{c}a_k^2\right) &\leq (c-1)\left(\widetilde{G}_k + \tfrac{2k+c-1}{c}a_k^2\right) \\ &\leq \left[(k+c-1)\widetilde{G}_k + \tfrac{(k+c-1)^2}{c}a_k^2 + cb_k^2\right] - \left[(k+c)\widetilde{G}_{k+1} + \tfrac{(k+c)^2}{c}a_{k+1}^2 + cb_{k+1}^2\right]. \end{aligned} \tag{3.60}$$

Since $c > 1$, $\widetilde{G}_k \geq 0$, and $a_k^2 \geq 0$, the inequality (3.60) implies that

$$(k+c)\widetilde{G}_{k+1} + \frac{(k+c)^2}{c}a_{k+1}^2 + cb_{k+1}^2 \leq (k+c-1)\widetilde{G}_k + \frac{(k+c-1)^2}{c}a_k^2 + cb_k^2.$$

By induction, we can show that

$$\begin{aligned} (k+c-1)\widetilde{G}_k + \tfrac{(k+c-1)^2}{c}a_k^2 + cb_k^2 &\leq (c-1)\widetilde{G}_0 + \tfrac{(c-1)^2}{c}a_0^2 + cb_0^2 \\ &= (c-1)[\mathcal{P}(x^0) - \mathcal{P}^\star] + \tfrac{c\mathcal{R}^2(x^\star, y^\star)}{2} = \Delta_0^2, \end{aligned}$$

where in the second line we have used $\|g(x^0) + s^0\| = \|\Theta_0\| = 0$ as initialized in Step 2 of Algorithm 1. As a result,

$$\mathcal{L}(x^k, s^k, y^\star) - \mathcal{P}^\star = \widetilde{G}_k \leq \frac{\Delta_0^2}{k+c-1} \quad \text{and} \quad \|g(x^k) + s^k\| = a_k\sqrt{\frac{2}{\rho_0}} \leq \frac{\sqrt{2c/\rho_0}\Delta_0}{k+c-1}. \tag{3.61}$$

Consequently, if H is M_H-Lipschitz continuous, then we can show that

$$\begin{aligned}\mathcal{P}(x^k)-\mathcal{P}^\star &= F(x^k)+H\left(g(x^k)\right)-\mathcal{P}^\star \le F(x^k)+H(-s^k)+M_H\|g(x^k)+s^k\|-\mathcal{P}^\star\\ &\le \mathcal{L}(x^k,s^k,y^\star)-\mathcal{P}^\star+\left(\|y^\star\|+M_H\right)\|g(x^k)+s^k\| \qquad (3.62)\\ &\overset{(3.61)}{\le} \tfrac{\Delta_0^2}{k+c-1}+\tfrac{\sqrt{2c/\rho_0}\Delta_0(\|y^\star\|+M_H)}{k+c-1},\end{aligned}$$

which is the first assertion of (3.56).

Next, summing up (3.60) from $j := 0$ to $j := k$, we get

$$\begin{aligned}(c-1)\textstyle\sum_{j=0}^k\left[\widetilde{G}_j+\frac{j+c-1}{c}a_j^2\right] \le &\left[(c-1)\widetilde{G}_0+\tfrac{(c-1)^2}{c}a_0^2+cb_0^2\right]\\ &-\left[(k+c)\widetilde{G}_{k+1}+\tfrac{(k+c)^2}{c}a_{k+1}^2+cb_{k+1}^2\right]\le\Delta_0^2.\end{aligned}$$

Since $c>1$ and $\widetilde{G}_j\ge 0$, we can apply Lemma B.1(2) to show that

$$\liminf_{k\to\infty}(k\log k)\left(\widetilde{G}_k+\frac{ka_k^2}{c}\right)=0, \qquad (3.63)$$

Combining this limit with (3.61) and (3.62), and applying Lemma B.1(3a), we can easily prove the second assertion of (3.56). □

3.4 Our Second Primal-Dual Algorithm: Strongly Convex-Concave Case

Recall that $F := f + h$ as defined in Assumption 3.2, where f is L_f-smooth, and h is not necessary smooth. In this section, we additionally impose the following assumption:

Assumption 3.3. *The function h in Assumption 3.2(1) is μ_h-strongly convex with $\mu_h > 0$.*

Note that even if h is not strongly convex, but f is μ_f-strongly convex with $\mu_f > 0$, then we can let $\hat{h}(x) := h(x) + \frac{\mu_f}{2}\|x\|^2$, and $\hat{f}(x) := f(x) - \frac{\mu_f}{2}\|x\|^2$. In this way, we have $\mu_{\hat{h}} = \mu_f$. Hence, Assumption 3.3 still holds for $\hat{h}$, and we can apply the algorithms in this section to solve (SP) with the same objective term $F(x) = \hat{f}(x) + \hat{h}(x)$.

3.4.1 The derivation and the complete algorithm

Under Assumptions 3.1, 3.2, and 3.3, we modify the scheme (3.27) by replacing Nesterov's acceleration steps by Tseng's steps [132], i.e., the x^{k+1}-update in (3.23) is broken into the following two lines of updates:

$$\begin{cases} \tilde{x}^{k+1} := \operatorname{prox}_{(\beta_k/\tau_k)h}\left(\tilde{x}^k - \frac{\beta_k}{\tau_k}[\nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k)]\right), \\ x^{k+1} := \operatorname{prox}_{\alpha_k h}\left(\hat{x}^k - \alpha_k[\nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k)]\right), \end{cases}$$

in order to achieve the faster $\mathcal{O}\left(\frac{1}{k^2}\right)$ and $\underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)$ non-ergodic convergence rates. The slack variable s^{k+1} is still defined as in (3.22). Meanwhile, the y^{k+1}-, $\tilde{y}^{k+1}$-, and Θ_{k+1}-updates, as well as the relation $\hat{x}^k = (1-\tau_k)x^k + \tau_k \tilde{x}^k$ in (3.28), are the same as before.

Using the expression of partial derivative $\nabla_x \phi$ in (3.17), we present the resulting algorithm as Algorithm 2.

Algorithm 2 Our second primal-dual algorithm: Strongly convex-concave case

1: **Initialization:** Choose an initial primal-dual point $(x^0, y^0) \in \mathbb{R}^p \times \mathbb{R}^m$.
2: Set $\tilde{x}^0 := \hat{x}^0 := x^0$, $\tilde{y}^0 := y^0$, and $\Theta_0 := 0$.
3: Choose appropriate initial parameters, according to (3.72), (3.79), or (3.86).
4: **For** $k = 0$ **to** $k_{\max}$
5: Update parameters according to (3.73), (3.80), or (3.87), consistent with Step 3.
6: Update $(\tilde{x}^k, x^k, \hat{x}^k, y^k, \tilde{y}^k)$ as follows:

$$\begin{cases} y^{k+1} := \operatorname{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right), \\ \tilde{x}^{k+1} := \operatorname{prox}_{(\beta_k/\tau_k)h}\left(\tilde{x}^k - \frac{\beta_k}{\tau_k}\left(\nabla f(\hat{x}^k) + [\nabla g(\hat{x}^k)]^\top y^{k+1}\right)\right), \\ x^{k+1} := \operatorname{prox}_{\alpha_k h}\left(\hat{x}^k - \alpha_k\left(\nabla f(\hat{x}^k) + [\nabla g(\hat{x}^k)]^\top y^{k+1}\right)\right), \\ \hat{x}^{k+1} := (1-\tau_{k+1})x^{k+1} + \tau_{k+1}\tilde{x}^{k+1}, \\ \Theta_{k+1} := g(x^{k+1}) - g(\hat{x}^k) + \frac{1}{\rho_k}(y^{k+1} - \tilde{y}^k), \\ \tilde{y}^{k+1} := \operatorname{proj}_{\mathcal{B}_k}\left(\tilde{y}^k + \eta_k\left[\Theta_{k+1} - (1-\tau_k)\Theta_k\right]\right). \end{cases} \tag{3.64}$$

7: **EndFor**

Per-iteration complexity. The per-iteration complexity of Algorithm 2 is the same as that of Algorithm 1, except for one additional proximal operator of h at line 2 of Step 6.

3.4.2 Convergence rate analysis

Parallel to Subsection 3.3.2, let us first present a key recursive estimate to analyze the convergence of Algorithm 2.

Lemma 3.3. *Define $\mathcal{L}$ as in* (3.11)*, $\mathcal{L}_\rho$ as in* (3.15)*, and L_f, M_g, and $\mathbf{L}_g$ as in Assumption 3.2. Let $\{(x^k, \hat{x}^k, \tilde{x}^k, y^k, \tilde{y}^k)\}$ be generated by* (3.64) *with $\tau_k \in [0,1]$, $\rho_k > \eta_k$, and $\alpha_k > \beta_k$. Furthermore, define $\{s^k\}$ as in* (3.22)*, and define L_k and $\{\breve{y}^k\}$ as in* (3.28)*. Then, for all $k \in \mathbb{N}$ and any*

$(x, s, y) \in \mathbb{R}^p \times \mathbb{R}^m \times \mathcal{B}_k$, *it holds that:*

$$\begin{aligned}\mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, y) - \mathcal{L}(x, s, \breve{y}^{k+1}) \leq\ & (1-\tau_k)[\mathcal{L}_{\rho_{k-1}}(x^k, s^k, y) - \mathcal{L}(x, s, \breve{y}^k)] \\ & + \tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^k - x\|^2 - \tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1} - x\|^2 \\ & + \tfrac{1}{2\eta_k}(\|\tilde{y}^k - y\|^2 - \|\tilde{y}^{k+1} - y\|^2) \\ & - \tfrac{(1-\tau_k)[\rho_{k-1} - (1-\tau_k)\rho_k]}{2}\|g(x^k) + s^k\|^2 \\ & - \tfrac{1}{2}\left[\tfrac{1}{\alpha_k}\left(1 - \tfrac{\beta_k}{\alpha_k}\right) + \tfrac{1}{\alpha_k} - L_k - L_f - \tfrac{\rho_k^2 M_g^2}{\rho_k - \eta_k}\right]\|x^{k+1} - \hat{x}^k\|^2.\end{aligned} \tag{3.65}$$

Proof. Define $\breve{x}^{k+1} := (1-\tau_k)x^k + \tau_k\tilde{x}^{k+1}$. For readability, we first state two claims: for all $k \in \mathbb{N}$ and any $(x, s, y) \in \mathbb{R}^p \times \mathbb{R}^m \times \mathcal{B}_k$, we have

$$\begin{aligned}F(x^{k+1}) + H(-s^{k+1}) \leq\ & (1-\tau_k)[F(x^k) + H(-s^k)] + \tau_k[F(x) + H(-s)] \\ & + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ (1-\tau_k)x^k + \tau_k x - x^{k+1}\rangle \\ & + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ (1-\tau_k)s^k + \tau_k s - s^{k+1}\rangle \\ & + \tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^k - x\|^2 - \tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1} - x\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\alpha_k} - L_f\right)\|x^{k+1} - \hat{x}^k\|^2 \\ & - \tfrac{1}{2}\left(\tfrac{1}{\alpha_k} + \mu_h\right)\|\breve{x}^{k+1} - x^{k+1}\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\beta_k} - \tfrac{1}{\alpha_k}\right)\|\breve{x}^{k+1} - \hat{x}^k\|^2,\end{aligned} \tag{3.66}$$

and

$$\begin{aligned}\phi_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \leq\ & (1-\tau_k)\phi_{\rho_k}(x^k, s^k, \tilde{y}^k) + \tau_k\phi_{\rho_k}(x, s, \tilde{y}^k) + \tfrac{L_k + \rho_k M_g^2}{2}\|x^{k+1} - \hat{x}^k\|^2 \\ & + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^{k+1} - (1-\tau_k)x^k - \tau_k x\rangle \\ & + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ s^{k+1} - (1-\tau_k)s^k - \tau_k s\rangle \\ & - \tfrac{(1-\tau_k)\rho_k}{2}\|[g(x^k) + s^k] - [g(\hat{x}^k) + s^{k+1}]\|^2 \\ & - \tfrac{\tau_k\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2,\end{aligned} \tag{3.67}$$

the proofs of which are deferred as Lemma B.5 and Lemma B.6 in Appendix B.3.

Summing up the estimates (3.66) and (3.67), we get

$$\begin{aligned}\mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) &\overset{(3.15)}{=} F(x^{k+1}) + H(-s^{k+1}) + \phi_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \\ &\overset{(3.66)(3.67)}{\leq} (1-\tau_k)\mathcal{L}_{\rho_k}(x^k, s^k, \tilde{y}^k) + \tau_k\mathcal{L}_{\rho_k}(x, s, \tilde{y}^k) \\ &\quad + \tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^k - x\|^2 - \tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1} - x\|^2 \\ &\quad - \tfrac{1}{2}\left(\tfrac{1}{\alpha_k} - L_k - L_f - \rho_k M_g^2\right)\|x^{k+1} - \hat{x}^k\|^2 \\ &\quad - \tfrac{1}{2}\left(\tfrac{1}{\beta_k} - \tfrac{1}{\alpha_k}\right)\|\breve{x}^{k+1} - \hat{x}^k\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\alpha_k} + \mu_h\right)\|\breve{x}^{k+1} - x^{k+1}\|^2 \\ &\quad - \tfrac{(1-\tau_k)\rho_k}{2}\|[g(x^k) + s^k] - [g(\hat{x}^k) + s^{k+1}]\|^2 \\ &\quad - \tfrac{\tau_k\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2.\end{aligned} \tag{3.68}$$

Since $y \in \mathcal{B}_k$, by the same analysis as the proof for Lemma 3.2, one can easily check that (3.32)-(3.35) and (3.37)-(3.38) still hold. Substituting them into (3.68), we can further expand it as

$$\begin{aligned}\mathcal{L}_{\rho_k}(x^{k+1},s^{k+1},y) - &\mathcal{L}(x,s,\breve{y}^{k+1}) \le (1-\tau_k)[\mathcal{L}_{\rho_{k-1}}(x^k,s^k,y) - \mathcal{L}(x,s,\breve{y}^k)] \\ &+\tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^k - x\|^2 - \tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1}-x\|^2 \\ &-\tfrac{1}{2}\left(\tfrac{1}{\beta_k}-\tfrac{1}{\alpha_k}\right)\|\breve{x}^{k+1}-\hat{x}^k\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\alpha_k}+\mu_h\right)\|\breve{x}^{k+1}-x^{k+1}\|^2 \\ &+\tfrac{1}{2\eta_k}\left(\|\tilde{y}^k-y\|^2 - \|\tilde{y}^{k+1}-y\|^2\right) \\ &-\tfrac{1}{2}\left(\tfrac{1}{\alpha_k}-L_k-L_f-\rho_k M_g^2\right)\|x^{k+1}-\hat{x}^k\|^2 + \mathcal{T}_1,\end{aligned} \tag{3.69}$$

where

$$\begin{aligned}\mathcal{T}_1 \;:=\; &\tfrac{\eta_k}{2}\|[g(x^{k+1})+s^{k+1}] - (1-\tau_k)[g(x^k)+s^k]\|^2 + \tfrac{(1-\tau_k)(\rho_k-\rho_{k-1})}{2}\|g(x^k)+s^k\|^2 \\ &-\tfrac{(1-\tau_k)\rho_k}{2}\|[g(x^k)+s^k] - [g(\hat{x}^k)+s^{k+1}]\|^2 - \tfrac{\tau_k\rho_k}{2}\|g(\hat{x}^k)+s^{k+1}\|^2 \\ \overset{(3.41)}{\le}\; &\tfrac{\rho_k\eta_k M_g^2}{2(\rho_k-\eta_k)}\|x^{k+1}-\hat{x}^k\|^2 - \tfrac{(1-\tau_k)[\rho_{k-1}-(1-\tau_k)\rho_k]}{2}\|g(x^k)+s^k\|^2.\end{aligned} \tag{3.70}$$

Moreover, applying Lemma B.1(1) with $t_1 := \frac{1}{2}\left(\frac{1}{\beta_k}-\frac{1}{\alpha_k}\right) > 0$ and $t_2 := \frac{1}{2}\left(\frac{1}{\alpha_k}+\mu_h\right)$, we can easily show that

$$\begin{aligned}-\tfrac{1}{2}\left(\tfrac{1}{\beta_k}-\tfrac{1}{\alpha_k}\right)\|\breve{x}^{k+1}-\hat{x}^k\|^2 - &\tfrac{1}{2}\left(\tfrac{1}{\alpha_k}+\mu_h\right)\|\breve{x}^{k+1}-x^{k+1}\|^2 \le -\tfrac{t_1t_2}{t_1+t_2}\|x^{k+1}-\hat{x}^k\|^2 \\ &= -\tfrac{1}{2(1/\beta_k+\mu_h)}\left(\tfrac{1}{\beta_k}-\tfrac{1}{\alpha_k}\right)\left(\tfrac{1}{\alpha_k}+\mu_h\right)\|x^{k+1}-\hat{x}^k\|^2 \\ &\le -\tfrac{1}{2\alpha_k}\left(1-\tfrac{\beta_k}{\alpha_k}\right)\|x^{k+1}-\hat{x}^k\|^2,\end{aligned} \tag{3.71}$$

where in the last inequality we used $\alpha_k > \beta_k$. Substituting (3.70) and (3.71) into (3.69), we eventually obtain (3.65). □

Now, we establish three types of convergence rates for Algorithm 2. Each type of convergence rate is obtained by specifying the initialization and update rule for the parameters such as τ_k, ρ_k, and $\mathcal{B}_k$.

3.4.2.1 The $\mathcal{O}\left(\frac{1}{k^2}\right)$ ergodic convergence rate

We first prove in Theorem 3.4 that Algorithm 2 enjoys $\mathcal{O}\left(\frac{1}{k^2}\right)$ ergodic rate without assuming the boundedness of g or $\mathcal{B}_k$.

Theorem 3.4. *Suppose Assumptions 3.1, 3.2 and 3.3 hold for* (SP). *Let* $\{(x^k, y^k)\}_{k\geq 0}$ *be generated by Algorithm 2 with the following parameter configurations:*

- Initialization: *Choose* $\rho_0,\ \beta_0,\ \eta_0,\ \hat{M} > 0$, *and* $\gamma,\ \Gamma \in (0,\ 1)$ *such that*

$$\begin{cases} \beta_0 := \frac{\Gamma}{L_f + \rho_0 \hat{M}^2}, \quad \eta_0 := (1-\gamma)\rho_0, \quad \textit{and} \\ L_g[\|y^\star\| + (\sqrt{\eta_0} + \rho_0\sqrt{\beta_0} M_g)\mathcal{R}_0(x^\star, y^\star)] \leq \rho_0 \left[(2-\Gamma)\hat{M}^2 - \frac{M_g^2}{\gamma}\right]. \end{cases} \tag{3.72}$$

- Update: *For all* $k \in \mathbb{N}$, *set* $\mathcal{B}_k \equiv \mathbb{R}^m$, *and update*

$$\begin{cases} \tau_k \equiv 1, \quad \alpha_k := \frac{1}{L_f + \rho_k \hat{M}^2}, \quad \eta_k := (1-\gamma)\rho_k, \\ \theta_{k+1} := \frac{1}{\sqrt{1+\mu_h \beta_k}}, \quad \rho_{k+1} := \frac{\rho_k}{\theta_{k+1}}, \quad \textit{and} \quad \beta_{k+1} := \theta_{k+1}\beta_k. \end{cases} \tag{3.73}$$

Let $\{(\bar{x}^k, \bar{y}^k)\}_{k\geq 1}$ *be an ergodic sequence defined as*

$$(\bar{x}^k, \bar{y}^k) := \frac{1}{\Sigma_k}\sum_{j=0}^{k-1} \rho_j (x^{j+1}, y^{j+1}), \quad \textit{where} \quad \Sigma_k := \sum_{j=0}^{k-1} \rho_j. \tag{3.74}$$

Then, for any $k \geq 2$, *the following bounds hold:*

$$\begin{cases} \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(\bar{x}^k, \bar{y}^k) & \leq \frac{1}{(\sqrt{1+\mu_h\beta_0}-1)k(k-1)} \sup\limits_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \mathcal{R}_0^2(x,y), \\ \mathcal{P}(\bar{x}^k) - \mathcal{P}^\star & \leq \frac{1}{(\sqrt{1+\mu_h\beta_0}-1)k(k-1)} \left[\frac{\|x^0 - x^\star\|^2}{\beta_0} + \frac{(\|y^0\| + M_H)^2}{\eta_0}\right], \\ \mathcal{D}^\star - \mathcal{D}(\bar{y}^k) & \leq \frac{1}{(\sqrt{1+\mu_h\beta_0}-1)k(k-1)} \left[\frac{(\|x^0\| + M_{F^*})^2}{\beta_0} + \frac{\|y^0 - y^\star\|^2}{\eta_0}\right], \\ \mathcal{P}(\bar{x}^k) - \mathcal{D}(\bar{y}^k) & \leq \frac{1}{(\sqrt{1+\mu_h\beta_0}-1)k(k-1)} \left[\mathcal{R}_0^2(x^\star, y^\star) + \frac{(\|x^0\| + M_{F^*})^2}{\beta_0} + \frac{(\|y^0\| + M_H)^2}{\eta_0}\right], \end{cases} \tag{3.75}$$

where $\mathcal{R}_0$ *is defined in* (3.42), *and* $M_H,\ M_{F^*} \in [0, \infty]$ *are the Lipschitz constants of* H *and* F^*, *respectively.*

As a result, Algorithm 2 for solving (SP) *has* $\mathcal{O}\left(\frac{1}{k^2}\right)$ *ergodic convergence rate on the primal objective residual, the dual objective residual, and the primal-dual gaps.*

Proof. We firstly need two claims as below:

$$\begin{cases} \beta_k \leq \frac{\Gamma}{L_f + \rho_k \hat{M}^2}, \quad \rho_k \geq \rho_0 + (\sqrt{1+\mu_h\beta_0} - 1)\rho_0 k, \\ \frac{1+\mu_h\beta_k}{\beta_k} = \frac{1}{\theta_{k+1}\beta_{k+1}}, \quad \frac{1}{\rho_k} = \frac{1}{\theta_{k+1}\rho_{k+1}}, \quad \text{and} \quad \frac{1}{\eta_k} = \frac{1}{\theta_{k+1}\eta_{k+1}}, \end{cases} \tag{3.76}$$

and

$$L_g(\|y^\star\| + \|\tilde{y}^k - y^\star\| + \rho_k M_g \|\tilde{x}^k - x^\star\|) \le \rho_k \left[(2-\Gamma)\hat{M}^2 - \frac{M_g^2}{\gamma} \right]. \tag{3.77}$$

The proofs of these two claims are deferred as Lemma B.7 and Lemma B.8, resp., in Appendix B.3. Note that the proof of (3.77) has utilized (3.76).

Since (3.77) holds for all $k \in \mathbb{N}$, we can use the same lines from (B.30) to (B.32) to get for all $j \in \mathbb{N}$ and any $(x, s, y) \in \mathbb{R}^p \times \mathbb{R}^m \times \mathbb{R}^m$ that

$$\begin{aligned} \mathcal{L}(x^{j+1}, s^{j+1}, y) - \mathcal{L}(x, s, y^{j+1}) &\overset{\tau_k \equiv 1}{=} \mathcal{L}(x^{j+1}, s^{j+1}, y) - \mathcal{L}(x, s, \breve{y}^{j+1}) \\ &\le \left(\tfrac{1}{2\beta_j} \|\tilde{x}^j - x\|^2 + \tfrac{1}{2\eta_j} \|\tilde{y}^j - y\|^2 \right) \\ &\quad - \tfrac{1}{\theta_{j+1}} \left(\tfrac{1}{2\beta_{j+1}} \|\tilde{x}^{j+1} - x\|^2 + \tfrac{1}{2\eta_{j+1}} \|\tilde{y}^{j+1} - y\|^2 \right). \end{aligned}$$

Multiplying this inequality by $2\rho_j$ and noticing that $\frac{\rho_j}{\theta_{j+1}} = \rho_{j+1}$, we have

$$\begin{aligned} 2\rho_j [\mathcal{L}(x^{j+1}, s^{j+1}, y) - \mathcal{L}(x, s, y^{j+1})] &\le \rho_j \left(\tfrac{1}{\beta_j} \|\tilde{x}^j - x\|^2 + \tfrac{1}{\eta_j} \|\tilde{y}^j - y\|^2 \right) \\ &\quad - \rho_{j+1} \left(\tfrac{1}{\beta_{j+1}} \|\tilde{x}^{j+1} - x\|^2 + \tfrac{1}{\eta_{j+1}} \|\tilde{y}^{j+1} - y\|^2 \right). \end{aligned}$$

Summing up this inequality from $j := 0$ to $j := k-1$, we obtain

$$\sum_{j=0}^{k-1} \rho_j [\mathcal{L}(x^{j+1}, s^{j+1}, y) - \mathcal{L}(x, s, y^{j+1})] \le \frac{\rho_0}{2} \left(\frac{1}{\beta_0} \|x^0 - x\|^2 + \frac{1}{\eta_0} \|y^0 - y\|^2 \right) = \frac{\rho_0 \mathcal{R}_0^2(x, y)}{2}.$$

Dividing it by $\sum_{j=0}^{k-1} \rho_j$, and using the convexity of $\mathcal{L}$ in x and s, the concavity in y, the $\{(\bar{x}^k, \bar{y}^k)\}$ defined by (3.74), and $\bar{s}^k := \left(\sum_{j=0}^{k-1} \rho_j\right)^{-1} \sum_{j=0}^{k-1} \rho_j s^{j+1}$, we get

$$\mathcal{L}(\bar{x}^k, \bar{s}^k, y) - \mathcal{L}(x, s, \bar{y}^k) \overset{(3.74)}{\le} \frac{1}{\sum_{j=0}^{k-1} \rho_j} \sum_{j=0}^{k-1} \rho_j [\mathcal{L}(x^{j+1}, s^{j+1}, y) - \mathcal{L}(x, s, y^{j+1})] \le \frac{\rho_0 \mathcal{R}_0^2(x, y)}{2 \sum_{j=0}^{k-1} \rho_j}.$$

By the second inequality in (3.76), we have

$$\sum_{j=0}^{k-1} \rho_j \ge k\rho_0 + \frac{1}{2}(\sqrt{1 + \mu_h \beta_0} - 1)\rho_0 k(k-1) \ge \frac{1}{2}(\sqrt{1 + \mu_h \beta_0} - 1)\rho_0 k(k-1).$$

Combining the above two inequalities, we eventually get

$$\mathcal{L}(\bar{x}^k, \bar{s}^k, y) - \mathcal{L}(x, s, \bar{y}^k) \leq \frac{\mathcal{R}_0^2(x, y)}{(\sqrt{1+\mu_h\beta_0}-1)k(k-1)}.$$

Therefore, we can use the same lines as in the proof of Theorem 3.1 to prove (3.75). □

Remark 3.5 (Initial parameters in (3.72))**.** *As shown in Appendix B.3, the following parameter values are feasible to* (3.72)*:*

$$\begin{cases} \rho_0 := 1, \quad \gamma := \Gamma := \frac{1}{2}, \quad \textit{and} \\ \hat{M}^2 := \max\left\{L_f + 1, \ \frac{8L_g^2D^2}{9} + \frac{4(2M_g^2 + L_gD + \sqrt{2}L_gDM_g)}{3}\right\}, \end{cases} \tag{3.78}$$

where D is defined in Remark 3.1. This bound is relatively loose in pursuit of a simple expression, thus, one can choose tighter values for these parameters that satisfy (3.72) *in order to achieve better practical performance.* □

Remark 3.6 (Optimal rate)**.** *As shown in [142, Theorem 2], the $\mathcal{O}\left(\frac{1}{k^2}\right)$ convergence rate of Algorithm 2 is optimal in the sense of Remark 3.2.* □

Remark 3.7. *Alternatively, if we let $\tau_k \equiv 1$ and $\alpha_k \equiv \beta_k$ (i.e., $\Gamma := 1$), then scheme* (3.64) *is simplified as*

$$\begin{cases} y^{k+1} := \operatorname{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(x^k)\right), \\ x^{k+1} := \operatorname{prox}_{\beta_k h}\left(x^k - \beta_k\left(\nabla f(x^k) + [\nabla g(x^k)]^\top y^{k+1}\right)\right), \\ \tilde{y}^{k+1} := \tilde{y}^k + \eta_k\left[g(x^{k+1}) - g(x^k) + \frac{1}{\rho_k}(y^{k+1} - \tilde{y}^k)\right], \end{cases}$$

with only one proximal operation. Now, if we combine the initialization condition (3.43) *(with (ρ, β, η) there replaced by $(\rho_0, \beta_0, \eta_0)$) and the update rule* (3.73) *(except for the absence of α_k), then we would still achieve the same $\mathcal{O}\left(\frac{1}{k^2}\right)$ ergodic convergence rates. This can be proved using similar lines as the proofs of Lemma 3.3 and Theorem 3.4.* □

3.4.2.2 The $\mathcal{O}\left(\frac{1}{k^2}\right)$ semi-ergodic convergence rate

Next, we analyze the semi-ergodic convergence rate of Algorithm 2 via Theorem 3.5.

Theorem 3.5. *Suppose Assumptions 3.1, 3.2 and 3.3 hold for* (SP)*, and assume*

1. *$\|g(x)\| \leq B_g$ for all $x \in \operatorname{dom} g \cap \operatorname{dom} F$ for some $B_g \in [0, \infty]$ such that $L_g B_g < +\infty$.*

2. *There exists* $s_* \in -\partial H^*(0)$, *and there exists* $y_* \in \partial H(0)$ *such that* $\|y_*\| \leq \frac{(1-\Gamma)L_f}{2L_g}$, *where* Γ *will be chosen below.*

Let $\{(x^k, y^k)\}_{k\geq 1}$ *be generated by Algorithm 2 with* $y^0 := 0$ *and the following configurations:*

- Initialization: *Set* $\tau_0 := 1$. *Choose* $\rho_0,\ \hat{M} > 0$, *and* $\gamma,\ \Gamma \in (0,\ 1)$ *such that*

$$\frac{M_g^2 + L_g[(2-\gamma)B_g + 2(1-\gamma)\|s_*\|]}{\gamma(2-\Gamma)} \leq \hat{M}^2 \leq \frac{\Gamma\mu_h}{2\rho_0}. \tag{3.79}$$

- Update: *For* $k \in \mathbb{N}$, *fix* $\mathcal{B}_k \equiv \mathbb{R}^m$, *and update*

$$\begin{cases} \rho_k := \frac{\rho_0}{\tau_k^2}, \quad \alpha_k := \frac{1}{L_f + \rho_k \hat{M}^2}, \quad \beta_k := \Gamma\alpha_k, \\ \eta_k := (1-\gamma)\rho_k, \quad \text{and} \quad \tau_{k+1} := \frac{\tau_k}{2}\left(\sqrt{\tau_k^2 + 4} - \tau_k\right). \end{cases} \tag{3.80}$$

Let $\{\bar{y}^k\}_{k\geq 0}$ *be the ergodic sequence defined as* $\bar{y}^{k+1} := (1-\tau_k)\bar{y}^k + \tau_k y^{k+1}$. *Then, for all* $k \geq 0$, *the following bounds hold:*

$$\begin{cases} \mathcal{G}_{\mathcal{X}\times\mathcal{Y}}(x^k, \bar{y}^k) & \leq \frac{2}{(k+1)^2} \sup\limits_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \mathcal{R}_0^2(x,y), \\ \mathcal{P}(x^k) - \mathcal{P}^\star & \leq \frac{2}{(k+1)^2}\left[\frac{\|x^0 - x^\star\|^2}{\beta_0} + \frac{\left(\|y^0\| + M_H\right)^2}{\eta_0}\right], \\ \mathcal{D}^\star - \mathcal{D}(\bar{y}^k) & \leq \frac{2}{(k+1)^2}\left[\frac{\left(\|x^0\| + M_{F^*}\right)^2}{\beta_0} + \frac{\|y^0 - y^\star\|^2}{\eta_0}\right], \\ \mathcal{P}(x^k) - \mathcal{D}(\bar{y}^k) & \leq \frac{2}{(k+1)^2}\left[\mathcal{R}_0^2(x^\star, y^\star) + \frac{\left(\|x^0\| + M_{F^*}\right)^2}{\beta_0} + \frac{\left(\|y\|^0 + M_H\right)^2}{\eta_0}\right], \end{cases} \tag{3.81}$$

where $\mathcal{R}_0$ *is as defined in* (3.42), *and the* $M_H,\ M_{F^*} \in [0, \infty]$ *are the Lipschitz constants of* H *and* F^*, *respectively.*

As a result, Algorithm 2 for solving (SP) *has* $\mathcal{O}\left(\frac{1}{k^2}\right)$ *non-ergodic rate on the primal objective residual, ergodic rate on the dual objective residual, and semi-ergodic rate on the primal-dual gap.*

Proof. Firstly, the update of $\{\tau_k\}$ in (3.80) leads to

$$\tau_k^2 = (1-\tau_k)\tau_{k-1}^2 \quad \text{and} \quad \frac{1}{k+1} \leq \tau_k \leq \frac{2}{k+2}. \tag{3.82}$$

Thus $\rho_{k-1} = \frac{\rho_0}{\tau_{k-1}^2} = \frac{(1-\tau_k)\rho_0}{\tau_k^2} = (1-\tau_k)\rho_k$, which implies

$$\rho_k > \eta_k, \quad \rho_k \geq \rho_0, \quad \frac{1}{2\eta_k} = \frac{1-\tau_k}{2\eta_{k-1}}, \quad \text{and} \quad \rho_{k-1} - (1-\tau_k)\rho_k = 0. \tag{3.83}$$

By the equality in (3.79), the updates in (3.80), and (3.82), for $k \geq 1$, we have

$$\begin{aligned}
\tfrac{\tau_k^2}{2\beta_k} &\overset{(3.80)}{=} \tfrac{\tau_k^2(L_f+\rho_k\hat{M}^2)}{2\Gamma} \overset{(3.83)}{=} \tfrac{\tau_k^2[L_f+(\rho_{k-1}+\tau_k\rho_k)\hat{M}^2]}{2\Gamma} = \tfrac{\tau_k^2(L_f+\rho_{k-1}\hat{M}^2)}{2\Gamma} + \tfrac{\tau_k^3\rho_k\hat{M}^2}{2\Gamma} \\
&\overset{(3.80)}{=} \tfrac{\tau_k^2}{2\beta_{k-1}} + \tfrac{\tau_k\rho_0\hat{M}^2}{2\Gamma} \overset{(3.79)}{=} \tfrac{\tau_k^2}{2\beta_{k-1}} + \tfrac{\tau_k\mu_h}{4} \overset{(3.82)}{\leq} \tfrac{(1-\tau_k)\tau_{k-1}^2}{2\beta_{k-1}} + \tfrac{(1-\tau_k)\tau_{k-1}\mu_h}{2} \\
&= \tfrac{(1-\tau_k)\tau_{k-1}(\tau_{k-1}+\beta_{k-1}\mu_h)}{2\beta_{k-1}}.
\end{aligned} \tag{3.84}$$

Moreover, by definitions of s_* and y_*, it is easily shown that (3.53) of Lemma B.4 still holds. Therefore, by the first inequality of (3.79) and $y^0 := 0$, we can derive that

$$\begin{aligned}
&\tfrac{1}{\alpha_k}\left(1-\tfrac{\beta_k}{\alpha_k}\right) + \tfrac{1}{\alpha_k} - L_k - L_f - \tfrac{\rho_k^2 M_g^2}{\rho_k - \eta_k} \\
&\overset{(3.80)(3.53)}{\geq} (1-\Gamma)(L_f+\rho_k\hat{M}^2) + \rho_k\hat{M}^2 - \tfrac{\rho_k M_g^2}{\gamma} \\
&\quad -L_g\left(2\|y_*\| + \tfrac{\rho_k}{\gamma}[(2-\gamma)B_g + 2(1-\gamma)\|s_*\|]\right) \\
&= \rho_k\left((2-\Gamma)\hat{M}^2 - \tfrac{1}{\gamma}[M_g^2 + (2-\gamma)B_g L_g + 2(1-\gamma)\|s_*\|L_g]\right) \\
&\quad +(1-\Gamma)L_f - 2L_g\|y_*\| \overset{(3.79)}{\geq} 0,
\end{aligned} \tag{3.85}$$

Combining (3.83), (3.84) and (3.85), we can simplify the relation (3.65) in Lemma 3.3 as

$$\begin{aligned}
\mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, y) - \mathcal{L}(x, s, \bar{y}^{k+1}) + \tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1} - x\|^2 + \tfrac{1}{2\eta_k}\|\tilde{y}^{k+1} - y\|^2 \\
\leq (1-\tau_k)\left[\mathcal{L}_{\rho_{k-1}}(x^k, s^k, y) - \mathcal{L}(x, s, \bar{y}^k)\right. \\
\left. + \tfrac{\tau_{k-1}(\tau_{k-1}+\beta_{k-1}\mu_h)}{2\beta_{k-1}}\|\tilde{x}^k - x\|^2 + \tfrac{1}{2\eta_{k-1}}\|\tilde{y}^k - y\|^2\right].
\end{aligned}$$

Here, we have used the fact that the $\breve{y}^k$ defined in (3.28) is equal to the ergodic iterate $\bar{y}^k$ defined in the statement of Theorem 3.5, thus we replace $\breve{y}^k$ by $\bar{y}^k$. By induction, this inequality implies

$$\begin{aligned}
&\mathcal{L}(x^k, s^k, y) - \mathcal{L}(x, s, \bar{y}^k) \leq \mathcal{L}_{\rho_{k-1}}(x^k, s^k, y) - \mathcal{L}(x, s, \bar{y}^k) \\
&\leq \left[\textstyle\prod_{j=1}^{k-1}(1-\tau_j)\right]\left[\mathcal{L}_{\rho_0}(x^1, s^1, y) - \mathcal{L}(x, s, \bar{y}^1) + \tfrac{\tau_0(\tau_0+\beta_0\mu_h)}{2\beta_0}\|\tilde{x}^1 - x\|^2 + \tfrac{1}{2\eta_0}\|\tilde{y}^1 - y\|^2\right] \\
&\overset{(3.65)(3.82)}{=} \left(\textstyle\prod_{j=1}^{k-1}\frac{\tau_j^2}{\tau_{j-1}^2}\right)\left(\tfrac{\tau_0^2}{2\beta_0}\|\tilde{x}^0 - x\|^2 + \tfrac{1}{2\eta_0}\|\tilde{y}^0 - y\|^2\right) \\
&\overset{(3.82)}{\leq} \tfrac{2}{(k+1)^2}\left(\tfrac{1}{\beta_0}\|x^0 - x\|^2 + \tfrac{1}{\eta_0}\|y^0 - y\|^2\right) = \tfrac{2\mathcal{R}_0^2(x,y)}{(k+1)^2}.
\end{aligned}$$

Using the last estimate we can prove (3.81) in a similar manner as in the proof of Theorem 3.2.

We therefore omit the details here. □

3.4.2.3 The $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$ non-ergodic convergence rate

Finally, using a different update rule for parameters, we establish a potentially faster $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$ convergence rate of Algorithm 2 in Theorem 3.6 below.

Theorem 3.6. *Suppose that Assumptions 3.1, 3.2, and 3.3 hold for* (SP)*, and assume*

1. $\|g(x)\| \leq B_g$ *for all* $x \in \operatorname{dom} g \cap \operatorname{dom} F$ *for some* $B_g \in [0,\ \infty]$ *such that* $L_g B_g < +\infty$.
2. *There exists* $y_* \in \partial H(0)$ *such that* $\|y_*\| \leq \frac{(1-\Gamma)L_f}{2L_g}$*, where* Γ *will be chosen below.*

Let $\{(x^k, y^k)\}_{k\geq 0}$ *be generated by Algorithm 2 using the following parameter configurations:*

- Initialization: *Choose* $\rho_0,\ \hat{M},\ R_y > 0$, $c > 2$, *and* $\gamma,\ \Gamma \in (0,\ 1)$, *such that*

$$\frac{M_g^2 + \gamma L_g(R_y + B_g)}{\gamma(2-\Gamma)} \leq \hat{M}^2 \leq \frac{c^2\Gamma\mu_h}{(2c+1)\rho_0} \quad \text{and} \quad \rho_0 R_y \geq \|y^\star\|. \tag{3.86}$$

- Update: *For all* $k \in \mathbb{N}$*, update*

$$\begin{cases} \tau_k := \frac{c}{k+c}, \quad \rho_k := \frac{\rho_0}{\tau_k^2}, \quad \alpha_k := \frac{1}{L_f + \rho_k \hat{M}^2}, \quad \beta_k := \Gamma\alpha_k, \\ \eta_k := (1-\gamma)\rho_k, \quad \text{and} \quad \mathcal{B}_k := \{y \mid \|y\| \leq \rho_{k-1} R_y\}. \end{cases} \tag{3.87}$$

Then,

$$\mathcal{P}(x^k) - \mathcal{P}^\star \leq \frac{R_p^2}{(k+c-1)^2} \ \text{ for } \ \forall k \geq 0, \quad \text{and} \quad \liminf_{k\to\infty} k^2\sqrt{\log k}[\mathcal{P}(x^k) - \mathcal{P}^\star] = 0, \tag{3.88}$$

where $R_P^2 := \Delta_0^2 + c\sqrt{2/\rho_0}(\|y^\star\| + M_H)\Delta_0$, *where* $\Delta_0^2 := (c-1)^2[\mathcal{P}(x^0) - \mathcal{P}^\star] + \left(c - 1 + \frac{c\mu_h\beta_0}{2}\right)\frac{c-1}{2\beta_0}\|x^0 - x^\star\|^2 + \frac{c^2}{2\eta_0}\|y^0 - y^\star\|^2$, *and* $M_H \in [0, \infty]$ *is the Lipschitz continuous constant of* H.

As a result, Algorithm 2 for solving (P) *has* $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$ *non-ergodic convergence rate on the primal objective residual.*

Proof. Since $\tilde{y}^k$ is projected onto $\mathcal{B}_{k-1}$, we can use the definition of y_* and similar arguments as (3.57) to show that $L_k \leq L_g[2\|y_*\| + \rho_k(R_y + B_g)]$. Now, by the first inequality in (3.86), and the

update of α_k and β_k in (3.87), we can show that

$$\begin{aligned}
&\tfrac{1}{\alpha_k}\left(1-\tfrac{\beta_k}{\alpha_k}\right)+\tfrac{1}{\alpha_k}-L_k-L_f-\tfrac{\rho_k^2M_g^2}{\rho_k-\eta_k}\\
&\quad\overset{(3.87)}{\geq}(1-\Gamma)(L_f+\rho_k\hat{M}^2)+\rho_k\hat{M}^2-L_g\left[2\|y_*\|+\rho_k(R_y+B_g)\right]-\tfrac{\rho_kM_g^2}{\gamma}\\
&\quad=\ \rho_k\left[(2-\Gamma)\hat{M}^2-\tfrac{M_g^2}{\gamma}-L_g(R_y+B_g)\right]+(1-\Gamma)L_f-2L_g\|y_*\|\overset{(3.86)}{\geq}0.
\end{aligned}$$

Utilizing this inequality and the update rules (3.87) into (3.65) of Lemma 3.3, we can derive

$$\begin{aligned}
&\mathcal{L}(x^{k+1},s^{k+1},y)-\mathcal{L}(x,s,\breve{y}^{k+1})+\left(\tfrac{k+c}{c}\right)^2\cdot\tfrac{\rho_0}{2}\|g(x^{k+1})+s^{k+1}\|^2\\
&\quad\leq\tfrac{k}{k+c}\left[\mathcal{L}(x^k,s^k,y)-\mathcal{L}(x,s,\breve{y}^k)+\left(\tfrac{k+c-1}{c}\right)^2\cdot\tfrac{\rho_0}{2}\|g(x^k)+s^k\|^2\right]\\
&\qquad+\tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^k-x\|^2-\tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2}\|\tilde{x}^{k+1}-x\|^2\\
&\qquad+\left(\tfrac{c}{k+c}\right)^2\cdot\tfrac{1}{2\eta_0}(\|\tilde{y}^k-y\|^2-\|\tilde{y}^{k+1}-y\|^2)-\tfrac{k[(k+c-1)^2-k(k+c)]}{c^2(k+c)}\cdot\tfrac{\rho_0}{2}\|g(x^k)+s^k\|^2.
\end{aligned}\tag{3.89}$$

Since $y^\star\leq\rho_0R_y$, we have $y^\star\in\mathcal{B}_0\subseteq\mathcal{B}_k$. We can then substitute $(x,s,y):=(x^\star,s^\star,y^\star)$ into (3.89), and then abbreviate

$$\begin{cases}
a_k^2:=\tfrac{\rho_0}{2}\|g(x^k)+s^k\|^2,\quad b_k^2:=\tfrac{c^2}{2\beta_k}\|\tilde{x}^k-x^\star\|^2,\quad d_k^2:=\tfrac{1}{2\eta_0}\|\tilde{y}^k-y^\star\|^2,\quad\text{and}\\
\widetilde{G}_k:=\mathcal{L}(x^k,s^k,y^\star)-\mathcal{L}(x^\star,s^\star,\breve{y}^k)=\mathcal{L}(x^k,s^k,y^\star)-\mathcal{P}^\star\geq0.
\end{cases}\tag{3.90}$$

Then, we can simplify (3.89) as:

$$\begin{aligned}
\widetilde{G}_{k+1}+\left(\tfrac{k+c}{c}\right)^2a_{k+1}^2&\leq\tfrac{k}{k+c}\left[\widetilde{G}_k+\left(\tfrac{k+c-1}{c}\right)^2a_k^2\right]+\tfrac{b_k^2}{(k+c)^2}-\tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1}-x^\star\|^2\\
&\quad+\left(\tfrac{c}{k+c}\right)^2(d_k^2-d_{k+1}^2)-\tfrac{k[(k+c-1)^2-k(k+c)]}{c^2(k+c)}a_k^2\\
&\leq\tfrac{k}{k+c}\widetilde{G}_k+\left(\tfrac{k}{c}\right)^2a_k^2+\tfrac{1}{(k+c)^2}(b_k^2-b_{k+1}^2)+\left(\tfrac{c}{k+c}\right)^2(d_k^2-d_{k+1}^2),
\end{aligned}\tag{3.91}$$

where in the second inequality we have used the definition of b_{k+1}^2 in (3.90) and that

$$\begin{aligned}
\tfrac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}-\tfrac{c^2}{2(k+c)^2\beta_{k+1}}&=\tfrac{c\mu_h}{2(k+c)}+\tfrac{c^2}{2(k+c)^2}\left(\tfrac{1}{\beta_k}-\tfrac{1}{\beta_{k+1}}\right)\overset{(3.87)}{=}\tfrac{c\mu_h}{2(k+c)}-\tfrac{[2(k+c)+1]\rho_0\hat{M}^2}{2(k+c)^2\Gamma}\\
&\geq\tfrac{1}{2(k+c)}\left(c\mu_h-\tfrac{(2c+1)\rho_0\hat{M}^2}{c\Gamma}\right)\overset{(3.86)}{\geq}0,
\end{aligned}$$

which implies that $\frac{\tau_k(\tau_k+\beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1}-x^\star\|^2\geq\frac{1}{(k+c)^2}b_{k+1}^2$. Multiplying both sides of (3.91) by $(k+c)^2$ and rearranging the result, we get

$$
\begin{aligned}
\Delta_{k+1}^2 &:= (k+c)^2\widetilde{G}_{k+1} + \tfrac{(k+c)^4}{c^2}a_{k+1}^2 + b_{k+1}^2 + c^2 d_{k+1}^2 \\
&\le k(k+c)\widetilde{G}_k + \tfrac{k^2(k+c)^2}{c^2}a_k^2 + b_k^2 + c^2 d_k^2 \\
&\le \left[(k+c-1)^2 - (c-2)(k+c-1)\right]\widetilde{G}_k \\
&\quad + \left(\tfrac{(k+c-1)^4}{c^2} - \tfrac{(c-2)(k+c-1)^3}{c^2}\right)a_k^2 + b_k^2 + c^2 d_k^2 \\
&= \Delta_k^2 - (c-2)\left[(k+c-1)\widetilde{G}_k + \tfrac{(k+c-1)^3}{c^2}a_k^2\right].
\end{aligned} \tag{3.92}
$$

where we have used $c > 2$ and the following elementary facts:

$$
\begin{cases}
k(k+c) & \le (k+c-1)^2 - (c-2)(k+c-1), \\
k^2(k+c)^2 & \le (k+c-1)^4 - (c-2)(k+c-1)^3.
\end{cases}
$$

Using (3.92), and by induction, we can deduce $(k+c-1)^2\widetilde{G}_k + \frac{(k+c-1)^4}{c^2}a_k^2 \le \Delta_k^2 \le \Delta_0^2$. In particular, we obtain

$$
\mathcal{L}(x^k, s^k, y^\star) - \mathcal{P}^\star = \widetilde{G}_k \le \tfrac{\Delta_0^2}{(k+c-1)^2} \quad \text{and} \quad \|g(x^k) + s^k\| = a_k\sqrt{\tfrac{2}{\rho_0}} \le \tfrac{c\sqrt{2/\rho_0}\Delta_0}{(k+c-1)^2}. \tag{3.93}
$$

Furthermore, summing up (3.92) from $j := 0$ to $j := k$, we also get

$$
(c-2)\sum_{j=0}^{k}\left[(j+c-1)\widetilde{G}_j + \frac{(j+c-1)^3}{c^2}a_j^2\right] \le \sum_{j=0}^{k}(\Delta_j^2 - \Delta_{j+1}^2) \le \Delta_0^2 < +\infty.
$$

Since $c > 2$ and $\widetilde{G}_j \ge 0$, if we apply Lemma B.1(2), then we can easily show that

$$
\liminf_{k\to\infty}(k^2\log k)\left[\widetilde{G}_k + \left(\frac{ka_k}{c}\right)^2\right] = 0. \tag{3.94}
$$

Since (3.93) and (3.94) are the parallel counterparts of (3.61) and (3.63) in proof Theorem 3.3. Therefore, the remaining proof of Theorem 3.6 is similar to the one in Theorem 3.3, and we omit the verbatim here. □

Remark 3.8 (Initial parameters in (3.79) and (3.86))**.** *The initializations in Theorems 3.5 and 3.6 are both feasible. For simplicity, one may set* $\gamma := \Gamma := \frac{1}{2}$*, then choose* $\hat{M}^2$ *such that the first inequality in* (3.79) *or* (3.86) *is tight, and then choose* ρ_0 *according to the second inequality. In order to fulfill the second inequality in* (3.86)*, one can easily solve a quadratic equation for*

$c > 2$, after setting the values for $\hat{M}^2$ and ρ_0. However, the user may choose other feasible initial parameters for better practical performance. □

3.5 Application to Cone-Constrained Convex Optimization

One important special case of (SP) is the cone-constrained convex optimization problem (3.1). In this section, we specify our algorithms and their convergence results to handle (3.1).

For the convenience of reference, let us recall (3.1) as follows:

$$\min_{x\in\mathbb{R}^p} \{F(x) := f(x) + h(x) \quad \text{s.t.} \quad g(x) \in -\mathcal{K}\}. \tag{CP}$$

By Assumption 3.2(3), since $\langle g(x), y\rangle$ is convex in x for any $y \in \mathcal{K}^*$, g is $\mathcal{K}$-convex, i.e., for all $x, x' \in \mathrm{dom} g$ and $t \in [0,1]$, it holds that $(1-t)g(x) + tg(x') - g\left((1-t)x + tx'\right) \in \mathcal{K}$. Thus, the constraint in (CP) is convex. Some important special cases of (CP) have been listed in Example 1.1 of Section 3.1.

To develop special variants of Algorithms 1 and 2 for solving (CP) and to establish their convergence guarantees, we redefine the associated Lagrange function as

$$\mathcal{L}(x, s, y) := F(x) + \langle y,\ g(x) + s\rangle, \tag{3.95}$$

where $s \in \mathcal{K}$ is again the slack variable, and $y \in \mathcal{K}^*$ is the Lagrange multiplier.

The following theorem tailors both Algorithms 1 and 2 to (CP), and provides their convergence rate guarantees on both the primal objective value and the feasibility gap.

Theorem 3.7. *To solve* (CP)*, let us*

$$\textit{specify the update} \quad y^{k+1} := \mathrm{prox}_{\rho_k H^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right) \quad \textit{as} \quad y^{k+1} := \mathrm{proj}_{\mathcal{K}^*}\left(\tilde{y}^k + \rho_k g(\hat{x}^k)\right)$$

in Step 6 of Algorithms 1 and 2, and define

$$\mathcal{E}(x) := \max\left\{|F(x) - F^\star|,\ \mathrm{dist}_{-\mathcal{K}} g(x)\right\} \quad \textit{and} \quad \mathcal{R}_1^2 := \frac{\|x^0 - x^\star\|^2}{\beta_0} + \frac{\left(\|y^0\| + \|y^\star\| + 1\right)^2}{\eta_0}, \tag{3.96}$$

where $\mathcal{E}(x)$ denotes the combined primal objective residual and primal feasibility violation at x. Then, for all $k \geq 1$, the following statements hold:

1. *Under the conditions of Theorem 3.1, we have $\mathcal{E}(\bar{x}^k) \leq \frac{\mathcal{R}_1^2}{2k}$.*
2. *Under the conditions of Theorem 3.2, we have $\mathcal{E}(x^k) \leq \frac{\mathcal{R}_1^2}{2k}$.*
3. *Under the conditions of Theorem 3.3, we have*

$$\mathcal{E}(x^k) \leq \frac{\Delta_0^2 + \sqrt{2c/\rho_0}\|y^\star\|\Delta_0}{k+c-1} \quad \text{and} \quad \liminf_{k\to\infty} k\sqrt{\log k} \cdot \mathcal{E}(x^k) = 0,$$

   where $\Delta_0 := (c-1)[F(x^0) - F^\star] + \frac{c}{2}\mathcal{R}_0^2(x^\star, y^\star)$.
4. *Under the conditions of Theorem 3.4, we have $\mathcal{E}(\bar{x}^k) \leq \frac{\mathcal{R}_1^2}{(\sqrt{1+\mu_h\beta_0}-1)(k-1)k}$.*
5. *Under the conditions of Theorem 3.5, we have $\mathcal{E}(x^k) \leq \frac{2\mathcal{R}_1^2}{(k+1)^2}$.*
6. *Under the conditions of Theorem 3.6, we have*

$$\mathcal{E}(x^k) \leq \frac{\Delta_0^2 + \sqrt{2c/\rho_0}\|y^\star\|\Delta_0}{(k+c-1)^2} \quad \text{and} \quad \liminf_{k\to\infty} k^2\sqrt{\log k} \cdot \mathcal{E}(x^k) = 0,$$

   where $\Delta_0 := (c-1)^2[F(x^0) - F^\star] + \frac{c^2}{2}\mathcal{R}_0^2(x^\star, y^\star)$.

As a result, Algorithm 1 for solving (CP) *is convergent on both the objective residual and the feasibility violation, with convergence rate $\mathcal{O}\left(\frac{1}{k}\right)$ in ergodic sense, and rate $\min\left\{\mathcal{O}\left(\frac{1}{k}\right), \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ in non-ergodic sense. Algorithm 2 for solving* (CP) *boosts these rates to $\mathcal{O}\left(\frac{1}{k^2}\right)$ and $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right), \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$, respectively.*

Proof. Using the same lines as in the proof of Theorem 3.1, we can see that (3.48) still holds with $\mathcal{L}$ defined in (3.95). Substituting $(x, s) := (x^\star, s^\star)$ into (3.48), we get

$$F(\bar{x}^k) + \langle y,\ g(\bar{x}^k) + \bar{s}^k\rangle - F^\star \leq \frac{\mathcal{R}_0^2(x^\star, y)}{2k}.$$

Let $\mathcal{R}_0(y) := \mathcal{R}_0^2(x^\star, y)$, then for any fixed $r > 0$, we have

$$F(\bar{x}^k) - F^\star \leq F(\bar{x}^k) - F^\star + r\|g(\bar{x}^k) + \bar{s}^k\| \leq \frac{1}{2k}\sup\{\mathcal{R}_0(y) \mid \|y\| \leq r\}. \tag{3.97}$$

On the other hand, by the saddle-point relation (3.14), we have

$$F(\bar{x}^k) + \langle y^\star,\ g(\bar{x}^k) + \bar{s}^k\rangle = \mathcal{L}(\bar{x}^k, \bar{s}^k, y^\star) \geq \mathcal{L}(x^\star, s^\star, y^\star) = F^\star.$$

By the Cauchy-Schwarz inequality, the last estimate leads to

$$F(\bar{x}^k) - F^\star \geq -\langle y^\star,\ g(\bar{x}^k) + \bar{s}^k\rangle \geq -\|y^\star\|\|g(\bar{x}^k) + \bar{s}^k\|. \tag{3.98}$$

Substituting (3.98) into (3.97), we get

$$(r - \|y^\star\|)\|g(\bar{x}^k) + \bar{s}^k\| \leq \frac{1}{2k}\sup\{\mathcal{R}_0^2(y) \mid \|y\| \leq r\}.$$

Let us choose $r := \|y^\star\| + 1$. Notice that $\bar{s}^k \in \mathcal{K}$ due to (3.22), $\mathrm{dom}H = -\mathcal{K}$, and that $\mathcal{K}$ is convex. Therefore, the last inequality becomes

$$\begin{array}{rcl} \mathrm{dist}_{-\mathcal{K}} g(\bar{x}^k) & = & \inf\limits_{s\in\mathcal{K}} \|g(\bar{x}^k) + s\| \leq \|g(\bar{x}^k) + \bar{s}^k\| \leq \dfrac{1}{2k}\sup\{\mathcal{R}_0^2(y) \mid \|y\| \leq \|y^\star\| + 1\} \\ & \overset{(3.42)}{\leq} & \frac{1}{2k}\left[\frac{1}{\beta_0}\|x^0 - x^\star\|^2 + \frac{1}{\eta_0}(\|y^0\| + \|y^\star\| + 1)^2\right], \end{array} \tag{3.99}$$

Combining (3.97), (3.98), and (3.99), we have proved Statement 1 of Theorem 3.7.

Statement 2 can be proved in a similar way as above, thus we omit the verbatim.

The first part of Statement 3 follows from (3.61). For the second part, notice that (3.63) still holds. Applying Lemma B.1(3b) with $u_k := \widetilde{G}_k$, $v_k := a_k$, $t_1 := \frac{1}{c}$, and $t_2 := \|y^\star\| + 1$, we get

$$\begin{array}{l} \liminf_{k\to\infty} k\sqrt{\log k}[|F(x^k) - F^\star| + \|g(x^k) + s^k\|] \\ \qquad \leq \liminf_{k\to\infty} k\sqrt{\log k}[\mathcal{L}(x^k, s^k, y^\star) - F^\star + (\|y^\star\| + 1)\|g(x^k) + s^k\|] = 0, \end{array} \tag{3.100}$$

which is exactly the second part of Statement 3.

The last three statements: Statements 4, 5, and 6, can be proved the same way as the first three statements. We therefore omit the details. □

3.6 Numerical Experiments

In this section, we aim at testing our algorithms on four numerical examples. The first one is a special case of quadratically constrained quadratic programming (QCQP) in Subsection 3.6.1. We use this example to verify the theoretical convergence rates of our algorithms. The second example

is a convex-concave min-max game in Subsection 3.6.2. The other two examples focus on the case where g is linear. In Subsection 3.6.3, we consider a bilinear min-max game; in Subsection 3.6.4, we use our algorithms to conduct image rocessing and reconstruct noisy, blurry, or lost image data.

We suggest the following tips when implementing our algorithms in order to obtain faster performance. These tips are guided by our theoretical results.

- As briefly discussed in Remark 3.3, when H^* is separable (or block-separable) in y, which is often the case, such as QCQP, instead of using the product such as $L_g M_g$ in (3.43), we can tighten it as $\sum_{i=1}^m L_{g_i} M_{g_i}$. In this case, Theorem 3.1 still holds. Similarly, the product $L_g B_g$ can be replaced by $\sum_{i=1}^m L_{g_i} B_{g_i}$ in the expressions of parameter initialization updates, e.g., in (3.51) and (3.79), and the theorems still hold true. Therefore, it is useful to use such replacements in implementation.
- One can tune the initial parameters, such as ρ_0 and β_0, in order to improve the performance. These parameters trade-off the dependence of the right-hand-side convergence bounds on the primal and dual initial points x^0 and y^0, resp.; see the definition of $\mathcal{R}_0^2$ in (3.42).
- We can directly use use $L_k := \mathbf{L}_g(y^{k+1})$, and adaptively update the parameter

$$\beta_k := \frac{1}{L_f + L_k + \frac{\rho_k M_g^2}{\gamma}}$$

in Algorithm 1. In this case, the last term in (3.29) of Lemma 3.2 diminishes with the largest possible β_k, which often improves the algorithm's practical performance by taking more aggressive primal steps. Similarly, in Algorithm 2, we can let

$$\alpha_k := \frac{1}{L_f + \frac{1}{2-\Gamma}\left(L_k + \frac{\rho_k M_g^2}{\gamma}\right)}.$$

- Restarting the parameters by periodically setting, e.g., $x^0 := x^k$, and $\tau_k := 1$, in the context of Theorems 3.2, 3.3, 3.5, and 3.6. In this way, we can avoid the primal stepsizes β_k and α_k from becoming too small after many iterations. While restarting technique can significantly boost the algorithms' performance [43, 96], we did not implement it for this section due to the lack of theoretical guarantee.

In this section, in order to address our six algorithmic variants, we use Algorithm 1 (v1) to denote the variant combining Algorithm 1 and parameter initialization/update rules specified in (3.43)-(3.44) in Theorem 3.1. Similarly, we call the other two variants Algorithm 1 (v2) and Algorithm 1 (v3), respectively. The three variants of Algorithm 2 are named accordingly.

3.6.1 Verifying theoretical guarantees via a special case of QCQP

We consider the following problem of computing the square distance from a given point a_0 to the intersection of m given balls centered at a_i of radius r_i, where $i = 1, \ldots, m$:

$$\begin{cases} \min\limits_{x \in \mathbb{R}^p} \ \|x - a_0\|^2, \\ \ \ \text{s.t.} \ \|x - a_i\|^2 \leq r_i^2, \quad i = 1, \ldots, m, \end{cases} \tag{3.101}$$

where $a_i \in \mathbb{R}^p$ for $i = 0, 1, \ldots, m$, and $r_i > 0$ is a scalar for each $i = 1, \ldots, m$. Problem (3.101) fits the special case (CP) of our template with $f(x) := 0$, $h(x) := \|x - a_0\|^2$, $g_i(x) := \|x - a_i\|^2 - r_i^2$, and $\mathcal{K} := \mathbb{R}^m_+$. Here, h is strongly convex with $\mu_h = 2$.

We first fix the problem size as $p := 400$ and $m = 1000$. Next, we generate problem instances of (3.101) by drawing all entries of a_i's from uniform distribution in $(-1, 1)$, where $i = 0, 1, \ldots, m$. Then we define $r_i^2 := \|a_i\|^2 + \varepsilon_i$, where $\varepsilon_i > 0$ is a scalar drawn from uniform distribution in $(0, 1)$. Clearly, 0 is a strictly feasible solution to (3.101).

To test our algorithms, we generate 30 random problem instances of the same size. For each instance, we run all six algorithmic variants up to 10^4 iterations. Without over-tuning, we simply set $\rho_0 := 5 \times 10^{-4}$ for all three variants of Algorithm 1, as well as Algorithm 2 (v1); we set $\rho_0 := 5 \times 10^{-5}$, and $\hat{M} := 10^3$ for Algorithm 2 (v1) and (v2). Furthermore, we set $c := 2$ for Algorithm 1 (v3), and $c := 4$ for Algorithm 2 (v3).

The performance of six algorithmic variants is shown in Figure 3.1, where the relative objective residual and the relative feasibility residual, defined by

$$\frac{|F(x) - F^\star|}{\max\{1, \ |F^\star|\}} \quad \text{and} \quad \frac{\|g(x)_+\|}{\max\{1, \ \|g(x^\star)_+\|\}},$$

are shown on the left and right, respectively, in log-scale. Here, $x^\star$ and value $F^\star$ appearing in the

figure is computed by CVX [55, 56] with the MOSEK solver [84] at the highest precision. For each algorithmic variant, we plot its theoretically convergent sequence:

- For Algorithm 1 (v1), the blue curve is based on the ergodic (averaging) sequence $\{\bar{x}^k\}$ defined by (3.45) in Theorem 3.1.
- For Algorithm 1 (v2), the red curve is based on the non-ergodic (last-iterate) sequence $\{x^k\}$.
- For Algorithm 1 (v3), the green curve is based on the so-called "best-iterate" sequence $\{\underline{x}^k\}$, defined as the minimizer of the quantity $F(x^j) + \frac{1}{2}\|[g(x^j)]_+\|$ over $0 \leq j \leq k$, guided by definition of $\mathcal{E}$ in (3.96).
- The curves (black, pink, and yellow) of Algorithm 2 are similarly based on their respective iterate sequences.

Since we generate 30 different random problem instances, we use the thick line to indicate the mean value, and use the shaded area to describe the range over all instances.

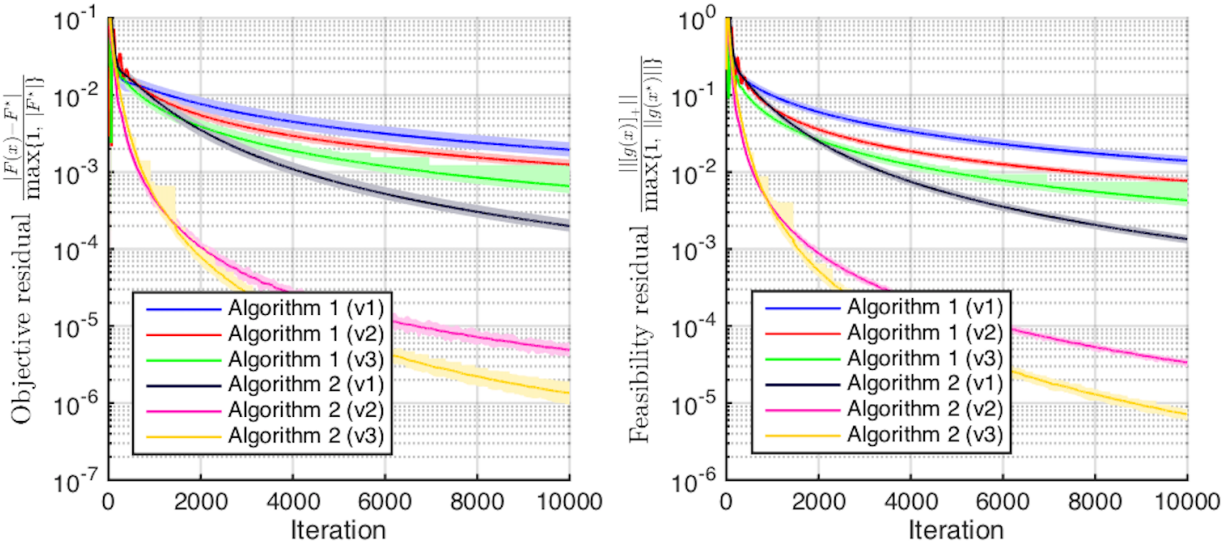

Figure 3.1: Average performance of our six algorithmic variants on 30 instances of QCQP (3.101)

From Figure 3.1, we observe that Algorithm 1 indeed behaves with $\mathcal{O}\left(\frac{1}{k}\right)$ convergence rate, in terms of both the objective value and the cone constraint violation. Among the three variants, Algorithm 1 (v3), with theoretical $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ convergence rate, is the fastest. On the other hand, Algorithm 1 (v1), whose theoretical rate is based on the averaging iterate, has

the worst performance.[5] Moreover, since problem (3.101) is strongly convex, the three variants of Algorithm 2 indeed took advantages of this property, and boosted the performance to $\mathcal{O}\left(\frac{1}{k^2}\right)$. Again, as theoretically predicted, the yellow curve for the "best-iterate" sequence is the best, which exhibits an empirically faster $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$ rate.

3.6.2 Convex-concave min-max game

In this subsection, we consider a convex-concave min-max game between two players, where Player 1 chooses strategy $x \in \Delta_p := \{x \in \mathbb{R}^p \mid \sum_{j=1}^p x_j = 1\}$ to minimize cost function $F(x)$, and simultaneously, Player 2 chooses strategy $y \in \Delta_m := \{y \in \mathbb{R}^m \mid \sum_{i=1}^m y_i = 1\}$ to minimize cost function H^*. In addition, Player 1 has to pay "loss function" $\langle g(x), y\rangle$ to Player 2.

Let $p = m$, and define the following function:

$$\begin{cases} F(x) := f(x) + h(x), & f(x) := \sum_{j=1}^n \log(1 + e^{a_j^\top x}), \quad h(x) := \delta_{\Delta_m}(x), \\ g_i(x) := \frac{b_i}{1+x_i}, & g(x) := (g_1(x), \ldots, g_m(x))^\top, \quad H^*(y) := \delta_{\Delta_m}(y), \end{cases} \tag{3.102}$$

where $A = (a_1, \ldots, a_n) \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. We can model the two persons min-max game model into the following problem, which fits well our template (SP):

$$\min_{x\in\Delta_m} \max_{y\in\Delta_m} \left\{ \sum_{j=1}^n \log(1 + e^{a_j^\top x}) + \sum_{i=1}^m \frac{b_i y_i}{1 + x_i} \right\}. \tag{3.103}$$

This problem is similar to the one in [23, Section 4.3], but our coupling term is linear in y. It is easy to compute that $L_f = \frac{\|A\|^2}{4}$, and $L_{g_i} = 2|b_i|$, $M_{g_i} = B_{g_i} = |b_i|$ for each $i \in \{1, \ldots, m\}$.

Since f in (3.102) is not strongly convex, we solve (3.103) using three variants of Algorithm 1: Algorithm 1 (v1) and (v2) both have $\mathcal{O}\left(\frac{1}{k}\right)$ convergence guarantees on the primal-dual gap; although Algorithm 1 (v3) does not have dual convergence guarantee, it does have a potentially faster $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ primal convergence rate. For `(v1)`, the theoretically convergent gap is based on the averaging sequences; for (v2), the convergence of gap $\mathcal{P}(x^k) - \mathcal{D}(\bar{y}^k)$ is based on primal last-iterate sequence and the dual averaging sequence; for `(v3)`, however, the primal

[5] We remark here that first-order methods with such constant stepsizes usually perform well empirically using the non-ergodic (last-iterate) sequence. However, only the ergodic (averaging) sequence possesses theoretical convergence guarantees as proved in existing literature.

residual $\mathcal{P}(x^k) - \mathcal{P}^\star$ is based on the last-iterate sequence, and there is a diminishing subsequence of $\{k\sqrt{\log k}[\mathcal{P}(x^k) - \mathcal{P}^\star]\}$, which indicates a potentially faster rate on the primal.

We compare our algorithmic variants with two existing algorithms: the Accelerated Primal-Dual (APD) algorithm proposed by [57], and the Mirror descent method in [88]. Similar to Algorithm 1 (v1), they both have the $\mathcal{O}\left(\frac{1}{k}\right)$ rate on duality gap based on averaging sequences. Note that APD does not write $F(x)$ as two separate functions as in (3.102), thus it has to solve a non-trivial subproblem at each iteration k to update x^k:

$$x^{k+1} := \arg\min_{x\in\Delta_m} \left\{ \beta \sum_{j=1}^{n} \log(1 + e^{a_j^\top x}) + \frac{1}{2} \left\| x - \beta \left(x^k - [\nabla g(x^k)]^\top y^{k+1} \right) \right\|^2 \right\}. \tag{3.104}$$

We have implemented restarted FISTA [121] to solve this problem with stopping criterion: $\|x^k_{j+1} - x^k_j\| < \varepsilon \max\{1, \|x^k_j\|\}$, where $\{x^k_j\}_{j=0}^\infty$ is the iterates for the subproblem (3.104), and we set $\varepsilon := 10^{-6}$. On the other hand, note that Mirror descent is double-loop, and in each inner iteration, it solves two subproblems that are slightly easier than (3.104), where we again employ a restarted FISTA routine. To compute the projection onto the simplex, we use the method in [140] for all algorithmic variants.

To generate problem instances, we set $p = m := 1000$, and $n := 500$, and simply draw all entries of A and b from standard Gaussian distribution, and A is sparse with 20% nonzero entries. For APD, we set the primal stepsize as $\beta := \frac{1}{L_g + M_g^2}$, and the dual stepsize as $\rho := 1$, as suggested in [57, Remarks 2.3 and 2.4]. For Mirror descent, we set the primal-dual stepsizes as $(\beta, \rho) := \left(\frac{1}{\sqrt{2}M_g}, \frac{1}{\sqrt{2}(L_f + L_g)}\right)$, as suggested in [88, eqn. (3.2)]. For both of our variants Algorithm 1 (v1) and (v2), we simply set $\gamma := \frac{1}{2}$ and $\rho_0 := 1$, without over-tuning. For Algorithm 1 (v3), we set $\rho_0 = 2$ and $c = 2$.

We generate 30 problem instances; for each instance, we run each algorithm up to 500 iterations. The performance is shown in Figure 3.2: on the left, we plot the duality gap against the number of iterations; and on the right, we plot the duality gap against time. The curves are all based on each method's theoretical iterations, i.e., they are based on ergodic (averaging) iterates $\{\bar{x}^k\}$ and $\{\bar{y}^k\}$ for Algorithm 1 (v1), APD, and Mirror descent. However, for Algorithm 1 (v2), we use the last (non-ergodic) iterates $\{x^k\}$. For Algorithm 1 (v3), given the theory, the most suitable sequence to

plot is the smallest gap $\min_{1\le j\le k}[\mathcal{P}(x^j) - \mathcal{D}(y^j)]$ up until the current iteration k.[6]

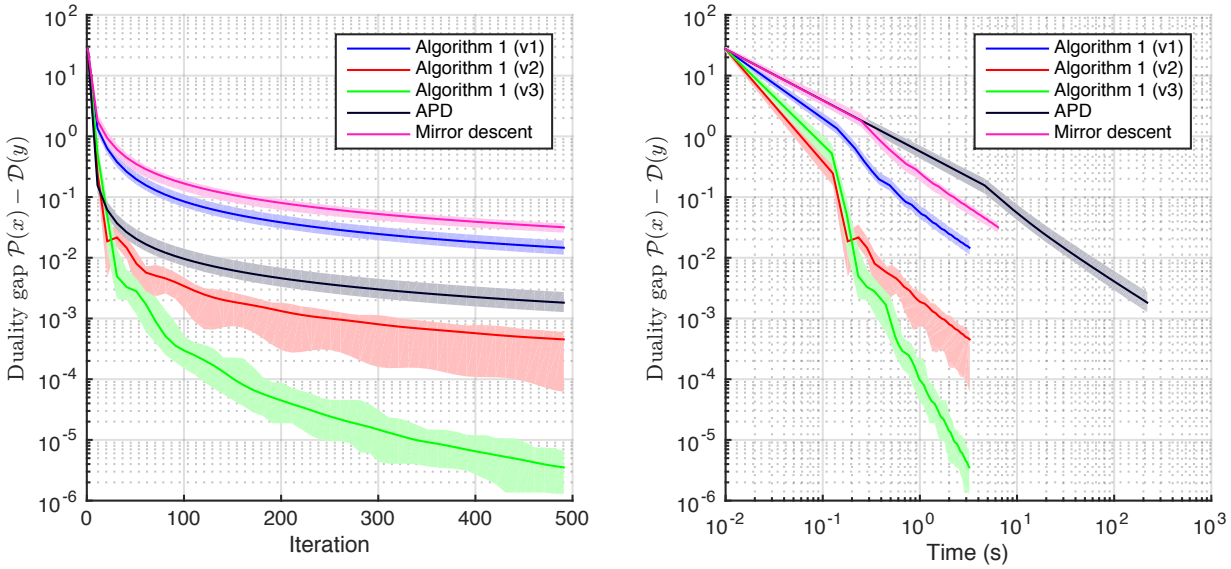

Figure 3.2: Average performance of five methods on 30 instances of min-max game (3.103) with problem size $(m, n) = (1000, 500)$

As in Subsection 3.6.1, we take the mean over all 30 instances to plot as thick curves, and take the range of duality gaps over all 30 instances to plot as shaded areas. We make some comments:

- Algorithm 1 (v1) and Mirror descent have relatively similar behavior, while our method, Algorithm 1 (v1), is still slightly faster.
- Algorithm 1 (v2) converges faster than Algorithm 1 (v1), Mirror descent, and APD, since it is the only method in these four that reduces the duality gap below 10^{-3}. However, it exhibits the most oscillation, shown through both the mean curve and the shaded range area. This is a normal behavior since it uses the last-iterate sequence, and thus is less smooth than other curves, which use an averaging sequence.
- Algorithm 1 (v3) is the fastest. The mean curve over the 30 random instances shows a higher than 10^{-5} accuracy, more than twice as good as other methods (except for the second fastest Algorithm 1 (v2)). It suggests that this variant's theoretical $\min\{\mathcal{O}(\cdot), \underline{o}(\cdot)\}$ rate may

[6]Recall Theorem 3.3 that for Algorithm 1 (v3), the convergence guarantee is $\mathcal{P}(x^k) - \mathcal{P}^\star = \min\{\mathcal{O}(1/k),\ \underline{o}\left(1/(k\sqrt{\log k})\right)\}$. By definition, this rate also holds for $\mathcal{P}(\underline{x}^k) - \mathcal{P}(x^\star)$; and by the definition of $\underline{o}(\cdot)$ rate in (3.3), using sequence $\{\underline{x}^k\}$ fits the theory better than simply $\{x^k\}$. For (v3), we do not have convergence rate on the dual, and thus for simplicity we use the primal-dual sequence in terms of the best gap. We remark that if g is linear, then our algorithm applied to the dual would have the convergence rate on the dual residual: $\mathcal{D}^\star - \mathcal{D}(\underline{y}^k) = \min\{\mathcal{O}(1/k),\ \underline{o}\left(1/(k\sqrt{\log k})\right)\}$.

practically be the faster $o(\cdot)$ rate.

- APD takes the longest time to run, since it has to solve the expensive subproblem (3.104). It is approximately 100 times slower than Algorithm 1 variants, as can be seen on the time axis of the right plot of Figure 3.2.

In order to further solidify our conclusions, we also conducted experiment on another 30 problem instances with larger size $(m, n) := (1500, 750)$. Indeed, the resulting performance shown in Figure 3.3 verified the fast speed of our proposed methods in terms of both number of iterations and the CPU time seen in Figure 3.2.

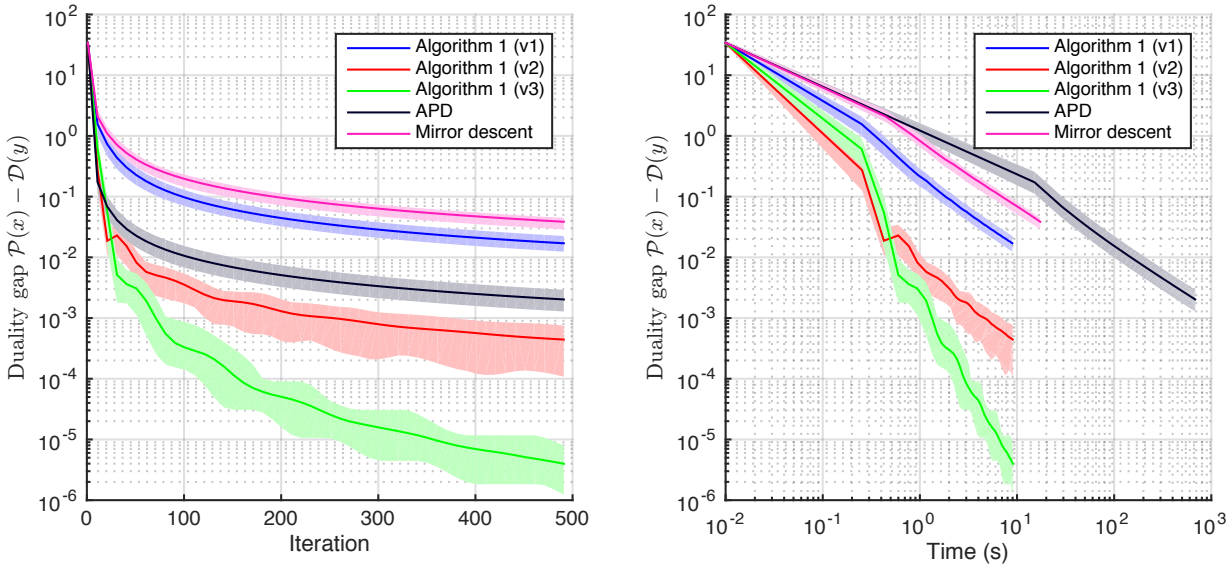

Figure 3.3: Average performance of five methods on 30 instances of min-max game (3.103) with problem size $(m, n) = (1500, 750)$

3.6.3 Bilinear min-max game

Consider the following matrix min-max game problem studied in [91]:

$$\min_{x \in \mathbb{R}^p} \max_{y \in \mathbb{R}^m} \langle Ax, y \rangle. \tag{3.105}$$

where $A \in \mathbb{R}^{m \times p}$. It is a special case of problem (3.103) (except that here $p \neq m$), where $f(x) \equiv 0$, and $g(x) := Ax$ is linear.

We will compare the performance of Algorithm 1 and Nesterov's smoothing algorithm in [91]

(Smoothing) on problem (3.105). Note that Smoothing would not work on problem (3.103), because it is specified for problems with bilinear coupling term. For the three variants of Algorithm 1, we compare the same types of iterates as in Subsection 3.6.2. For Smoothing, we use the semi-ergodic iterates, i.e., non-ergodic on the primal, and ergodic on the dual, as they are the iterates that the convergence guarantee is based on; see [91].

We set $(p, m) = (2000, 1000)$, and generate matrix $A \in \mathbb{R}^{m \times p}$ uniformly from interval $(-1, 1)$, then normalize it such that $\|A\| = 1$. We set four values for accuracy $\varepsilon = 10^{-1}, 10^{-2}, 10^{-3}$ and 10^{-4}, and accordingly set the numbers of iterations $k_{\max} = 4 \times 10^1,\ 4 \times 10^2,\ 4 \times 10^3$ and 4×10^4, resp.; see [91, (4.8)]. We then set the following configurations for the two algorithms:

- For Algorithm 1 (v1) and (v2), we simply choose $\gamma := \frac{1}{2}$ and $\rho_0 := 1$. For (v3), we set $\rho_0 := \frac{1}{4}$ and $c := 2$.
- For Smoothing, we follow the same configurations as in [91] for Euclidean distance smoothing. In particular, for a fixed accuracy $\varepsilon > 0$, the recommended smoothing parameter is $\rho_* = \frac{2(1-1/m)}{\varepsilon}$. On top of this value we test the variants where $\rho = \frac{\rho_*}{5}$ and $\rho = 5\rho_*$ in order observe the dependency of Smoothing on the smoothing parameter.

The performance of Smoothing is measured by the semi-ergodic duality gap $\mathcal{P}(x^k) - \mathcal{D}(\bar{y}^k)$ as in the theory [91], and the performance of Algorithm 1 is measured by the same types of iterates as in Subsection 3.6.2. In Figure 3.4, we observe that overall Algorithm 1 behaves faster than Smoothing. We can see that Smoothing depends heavily on the smoothing parameter: it works best when $\rho = \rho_*$ is set by theory, but not as good with greater or smaller values. Even if ρ is set by theory, the performance of Smoothing (pink curve) is still losing to variants of Algorithm 1. On the other hand, our Algorithm 1 (v3) gets significantly faster compared with other methods, when the number of iterations goes up. It suggests that this variant's $\min\{\mathcal{O}(\cdot),\ \underline{o}(\cdot)\}$ rate may actually be the faster $o(\cdot)$ rate. This result is consistent with what we have seen in Subsection 3.6.2.

3.6.4 Image reconstruction using TV-norm

In this subsection, we apply our algorithms on image processing problems, which is to reconstruct noisy, blurry or lost image data. This type of problems can be modeled as:

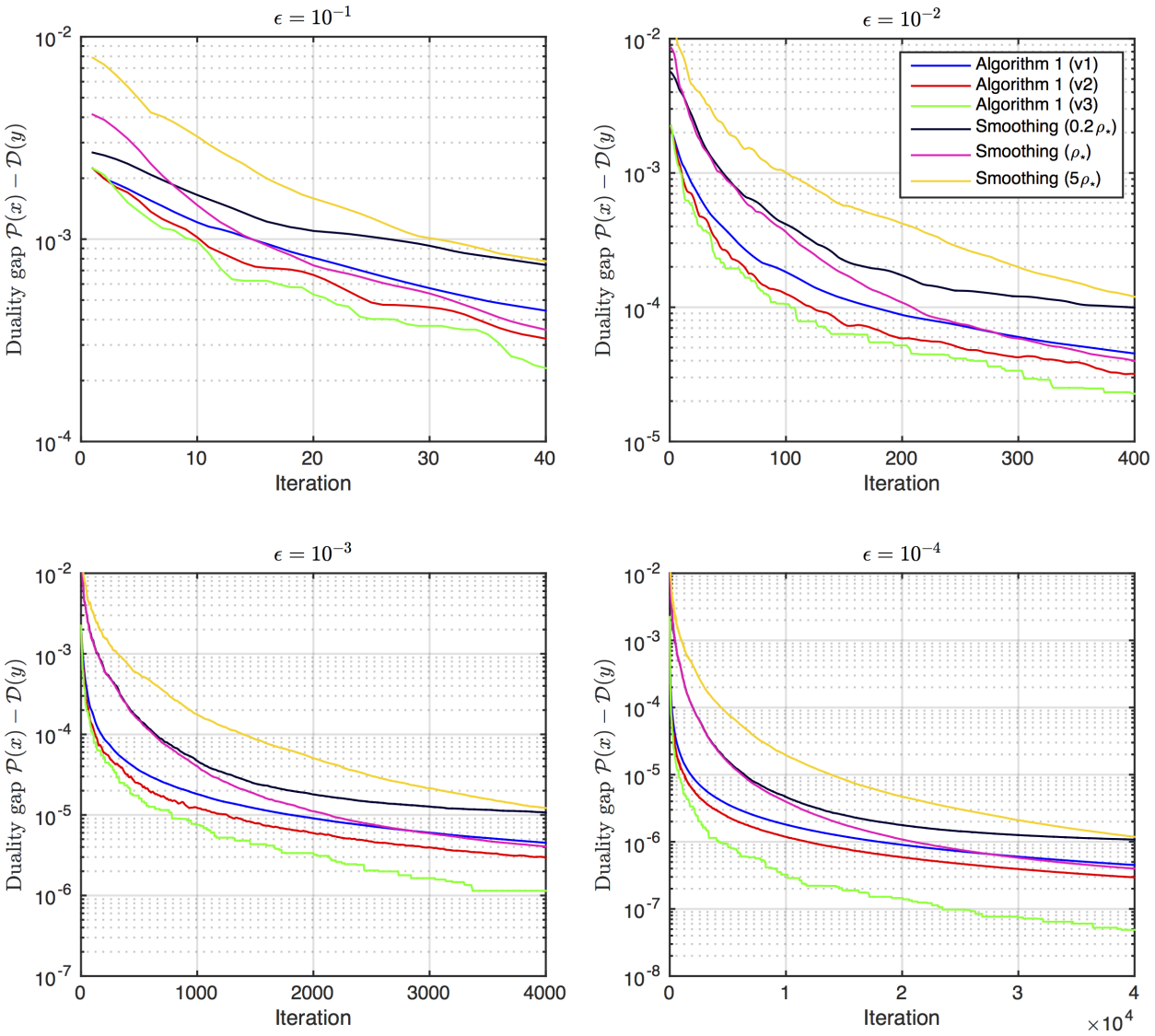

Figure 3.4: Performance of Algorithm 1 and Smoothing on a bilinear min-max game problem with four different configurations

$$\mathcal{P}^\star := \min_{x \in \mathcal{X}} \{\mathcal{P}(x) := f(x) + \|x\|_{\mathrm{TV}}\}, \tag{3.106}$$

where the primal space $\mathcal{X} := [0,1]^{N_1 \times N_2 \times N_3}$ contains the image data, where $N_1 \times N_2$ is the number of pixels (length and height, resp.), and N_3 is the number of channels, e.g.,

$$N_3 = \begin{cases} 1, & \text{if the image is black and white,} \\ 3, & \text{if the image is in RGB color.} \end{cases}$$

The function f in (3.106) is a convex data fidelity function, enforcing that the difference between the solution image and the deteriorated image is small. The problem (3.106) fits our template (P) with $H\left(g(x)\right) := \|x\|_{\mathrm{TV}}$, the isotropic total variation (TV) norm, where $g(x) = (g^{(1)}, g^{(2)}) := Ax \in \mathbb{R}^{N_1 \times N_2 \times N_3 \times 2}$ is the discrete gradient vector defined as

$$g^{(1)}_{i,j,k} := \begin{cases} x_{i+1,j,k} - x_{i,j,k}, & \text{if } i < N_1, \\ 0, & \text{if } i = N_1, \end{cases} \quad \text{and} \quad g^{(2)}_{i,j,k} := \begin{cases} x_{i,j+1,k} - x_{i,j,k}, & \text{if } j < N_2, \\ 0, & \text{if } j = N_2, \end{cases}$$

which is essentially the horizontal and vertical differences, resp., between two neighboring pixels. It is shown in [19, Theorem 3.1] that $M_g = \|A\| = \sqrt{8}$. Now, writing function H with argument g, the isotropic TV-norm in (3.106) is defined as

$$H(g) = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} \sum_{k=1}^{N_3} \sqrt{\left(g^{(1)}_{i,j,k}\right)^2 + \left(g^{(2)}_{i,j,k}\right)^2}.$$

Notice the handling of multiple channels is to use channel-summation instead of channel-coupling, see [83], i.e., we assume that different channels are independent; if we instead use channel-coupling, then the resulting image would tend to be black and white. It is easy to see that the TV-norm $\|x\|_{\mathrm{TV}} := H\left(Ax\right)$ enforces the similarity between pixels that are next to each other, which is a common requirement for successful image reconstruction.

We will run Algorithm 1 or Algorithm 2, depending on whether f is strongly convex. To compare, we will also run the first-order primal-dual algorithms CP [20, Algorithm 1] or CP-scvx [20, Algorithm 2], depending on whether f is strongly convex.[7] Finally, we run ADMM [15, eqn. (3.2)-(3.4)] on three types of image processing problems: de-noising, de-blurring and in-painting. We will consider the ergodic sequence $\{\bar{x}^k\}$ of Algorithm 1 (v1), Algorithm 2 (v1), CP and ADMM, since these algorithms all have ergodic convergence rate; for the same reason, we consider the non-ergodic sequence $\left\{x^k\right\}$ of our variant v2 and the best-iterate sequence $\{\underline{x}^k\}$ of our variant (v3), where $\underline{x}^k$ is such that $\mathcal{P}(\underline{x}^k) = \min_{1 \le j \le k} \mathcal{P}(x^j)$.

For each type of image processing problem, we run each algorithm on four standard benchmark images: "house", "lena", "mandril", and "peppers"; all of them has resolution $N_1 = N_2 = 512$.

[7] The ergodic convergence rate of [20, Algorithm 2] in terms of objective value is not shown until in a followup work [21].

Since they are RGB images, $N_3 = 3$. We run each algorithm for 500 iterations. The three criteria we consider are: objective value of (3.106) corresponding to the output of the algorithm, peak signal-to-noise ratio (PSNR) of the output image of the algorithm, and the running time of the algorithm. An algorithm with better performance would have small objective value, high PSNR, and short running time.

3.6.4.1 Image denoising

Suppose that $u \in \mathcal{X}$ is a given noisy image. If we let

$$f(x) := \frac{\lambda}{2} \|x - u\|_F^2 ,$$

where $\|\cdot\|_F$ is the tensor Frobenius norm, then solving (3.106) would reconstruct a less noisy image close to the given data, i.e., with high fidelity. The regularizer λ determines the desired fidelity level. This model is called the ROF model pioneered by Rudin, Osher and Fatemi [117].

We corrupt each clean image with 20% salt and pepper noise, and set $\lambda := 4$. For best performance of all algorithms, we do a slight tuning, and simply set the following algorithm parameter configurations for all problems:

- For Algorithm 2, set $\gamma := 0.5$, $\Gamma := 0.99$ and $\rho_0 := 0.5$; for (v3), set $c := 3$;
- For CP-scvx, set $\rho := 0.125$ as recommended in [20, Section 6.2].
- For ADMM, we use the trick in [22] to split the problem into three variables to avoid the expensive subproblem, and we solve the underlying linear system with at most either 20 conjugate gradient iterations or up to 10^{-5} accuracy. We find that ADMM works best with $\rho := 10$.

Table 3.2 shows the performance of the algorithms. We observe that

- The objective values of the three variants of Algorithm 2 are comparable to CP-scvx, and Algorithm 2 (v2) has objective values slightly better than those of the others; ADMM performs the worst in this aspect.
- The PSNR performance of the five algorithm variants are very similar, and CP-scvx is slightly better than the others.

- In terms of time, CP-scvx is the fastest, since it has the simplest scheme. Algorithm 2 is slower since it has an additional proximal operation. ADMM is the slowest as expected due to its computationally expensive subproblem.

Figure 3.5 show the effect of running Algorithm 2 (v3) to reconstruct noisy images. Here, each column represents an image tested; the first row shows clean images, the second row shows noisy images, and the last row shows the effect of denoising. The title above each figure shows the PSNR. We can see that by running Algorithm 2 we have indeed largely recovered the quality of figures.

Figure 3.5: Visual outcome of Algorithm 2 (v3) on four noisy images

3.6.4.2 Image deblurring

Suppose that $u \in \mathcal{X}$ is a given blurry image, which can be obtained from linear operator $\mathcal{A}$, the convolution with the point spread function (PSF), as studied in [19]. Let

Table 3.2: Results of five algorithmic variants on four image denoising problems after 500 iterations

	house			lena			mandril			peppers		
Method	P^{500}	PSNR	time	P^{500}	PSNR	time	P^{500}	PSNR	time	P^{500}	PSNR	time
None (noisy image)	257905.00	12.1220	-	252613.44	12.1917	-	289711.91	12.1950	-	257001.01	11.9094	-
Algorithm 2 (v1)	100341.97	21.1444	81.98	95646.65	22.8552	74.21	104154.29	18.9234	74.42	101304.03	21.8648	69.06
Algorithm 2 (v2)	100334.81	21.1263	77.87	95638.79	22.8405	72.29	104147.49	18.9066	73.59	101297.13	21.8555	72.17
Algorithm 2 (v3)	100333.46	21.1353	80.06	95637.58	22.8433	88.63	104146.09	18.9138	76.20	101295.78	21.8555	75.20
CP-scvx	100335.20	21.1500	46.24	95639.28	22.8558	46.17	104147.90	18.9278	44.23	101297.38	21.8639	43.02
ADMM	100841.39	21.1093	200.25	96178.30	22.7831	210.81	104737.84	18.9339	186.90	101767.97	21.7835	190.58

Table 3.3: Results of five algorithmic variants on four image deblurring problems after 500 iterations

	house			lena			mandril			peppers		
Method	P^{500}	PSNR	time	P^{500}	PSNR	time	P^{500}	PSNR	time	P^{500}	PSNR	time
None (blurry image)	285670.60	12.0990	-	308783.76	8.2030	-	308898.00	8.2030	-	306471.05	7.7162	-
Algorithm 2 (v1)	83846.19	20.5032	98.70	82957.63	19.0928	100.36	83121.85	19.0939	106.61	73079.93	22.6796	106.08
Algorithm 2 (v2)	46501.97	24.2845	99.11	53901.95	20.6946	102.69	53972.06	20.6947	113.86	41721.43	27.4607	114.44
Algorithm 2 (v3)	46275.05	24.4673	106.08	53779.62	20.7193	131.17	53849.68	20.7193	112.99	41534.96	27.7043	109.54
CP-scvx	48844.71	23.7578	68.87	56273.96	20.8717	72.81	56342.41	20.8725	71.73	43748.20	26.3107	75.68
ADMM	45949.41	25.0617	221.61	53657.08	20.7877	212.08	53728.18	20.7877	217.51	41334.22	28.3067	243.53

Table 3.4: Results of five algorithmic variants on four image inpainting problems after 500 iterations

	house			lena			mandril			peppers		
Method	P^{500}	PSNR	time	P^{500}	PSNR	time	P^{500}	PSNR	time	P^{500}	PSNR	time
None (lines missing)	44577.08	4.6010	-	52194.49	6.1782	-	55688.58	6.6664	-	63373.92	6.9335	-
Algorithm 1 (v1)	12128.26	19.6703	69.14	9929.64	25.0911	77.46	22124.51	19.4687	61.22	11077.70	23.4720	54.07
Algorithm 1 (v2)	11583.61	19.0654	75.87	9403.19	25.0201	102.95	21716.40	19.4181	73.71	10425.04	23.1166	68.14
Algorithm 1 (v3)	11590.92	18.9390	75.37	9421.88	24.9966	104.69	21726.84	19.4192	78.60	10444.24	23.0575	71.54
CP	11623.67	19.1180	44.40	9468.96	25.1469	49.88	21773.12	19.4632	48.77	10499.13	23.2726	44.71
ADMM	11676.93	19.1173	280.52	9522.37	24.9877	276.11	21750.00	19.4176	229.59	10624.78	23.1157	250.58

$$f(x) = \frac{\lambda}{2} \|\mathcal{A}(x) - u\|_F^2 , \tag{3.107}$$

where $\|\cdot\|_F$ is the tensor Frobenius norm, then the solution to (3.106) would be a less blurry image. Due to the special structure of $\mathcal{A}$, the proximal operator $\text{prox}_{\gamma f}(x)$ can be computed in a closed form using fast Fourier transform (FFT) [19, 22]. For each image, we apply a convolution filter to approximate the linear motion of a camera with length 30 pixels and angle 45° (counter-clockwise), followed by Gaussian noise with standard deviation 0.01. We set $\lambda := 720$. The configuration is similar to that in Subsubsection 3.6.4.1, except that we set $\rho_0 := 1$ for CP-scvx and all three variants of Algorithm 2.

Table 3.3 shows the performance of the algorithms. We observe that

Figure 3.6: Visual outcome of Algorithm 2 (v3) on four blurry images

- The objective value of Algorithm 2 (v1) is the largest, followed by CP-scvx. Algorithm 2 (v2), (v3), and ADMM has the smallest objective values.
- The PSNR performance of Algorithm 2 (v1) is the worst, and there is not much difference for the other algorithmic variants in this aspects.
- As in Subsubsection 3.6.4.1, CP-scvx is the fastest in CPU time, and ADMM is the slowest. The reason is the same as in the denoising example in Subsubsection 3.6.4.1.

We include Figure 3.6 to present the visual outcome of Algorithm 2 (v3). It shows that the images have been reconstructed to a great extent.

3.6.4.3 Image inpainting

Suppose that $u \in \mathcal{X}$ is a given deteriorated image with lost pixels at index set $I \subseteq \{1, 2, ..., N_1\} \times \{1, 2, ..., N_2\}$. If we let

$$f(x) := \frac{\lambda}{2} \sum_{i,j \notin I} \sum_{k} (x_{i,j,k} - u_{i,j,k})^2,$$

then solving (3.106) would reconstruct an image with the un-deteriorated pixels close to the given un-deteriorated data.

We set $\lambda := 32$ in f with 80% missing lines. Notice that f is *not* strongly convex – it does not involve the entries where $(i, j) \in I$. Therefore, we apply Algorithm 1, CP from [20, Algorithm 1], and ADMM. The configuration is again similar to that in Subsubsection 3.6.4.1, except that $\rho_0 := 1$ for CP and all three variants of Algorithm 1.

Table 3.4 shows the performance of the algorithms. We observe that

- The objective values of Algorithm 1 (v1) is the largest. The other algorithmic variants have similar performance in this aspect.
- The performance on PSNR is very similar for all methods, but Algorithm 1 (v1) is slightly better than the rest.
- In terms of CPU time, ADMM is the slowest, and CP is the fastest. It takes ADMM around four times longer time to run than CP.

We include Figure 3.7 to present the performance of Algorithm 1 (v3).

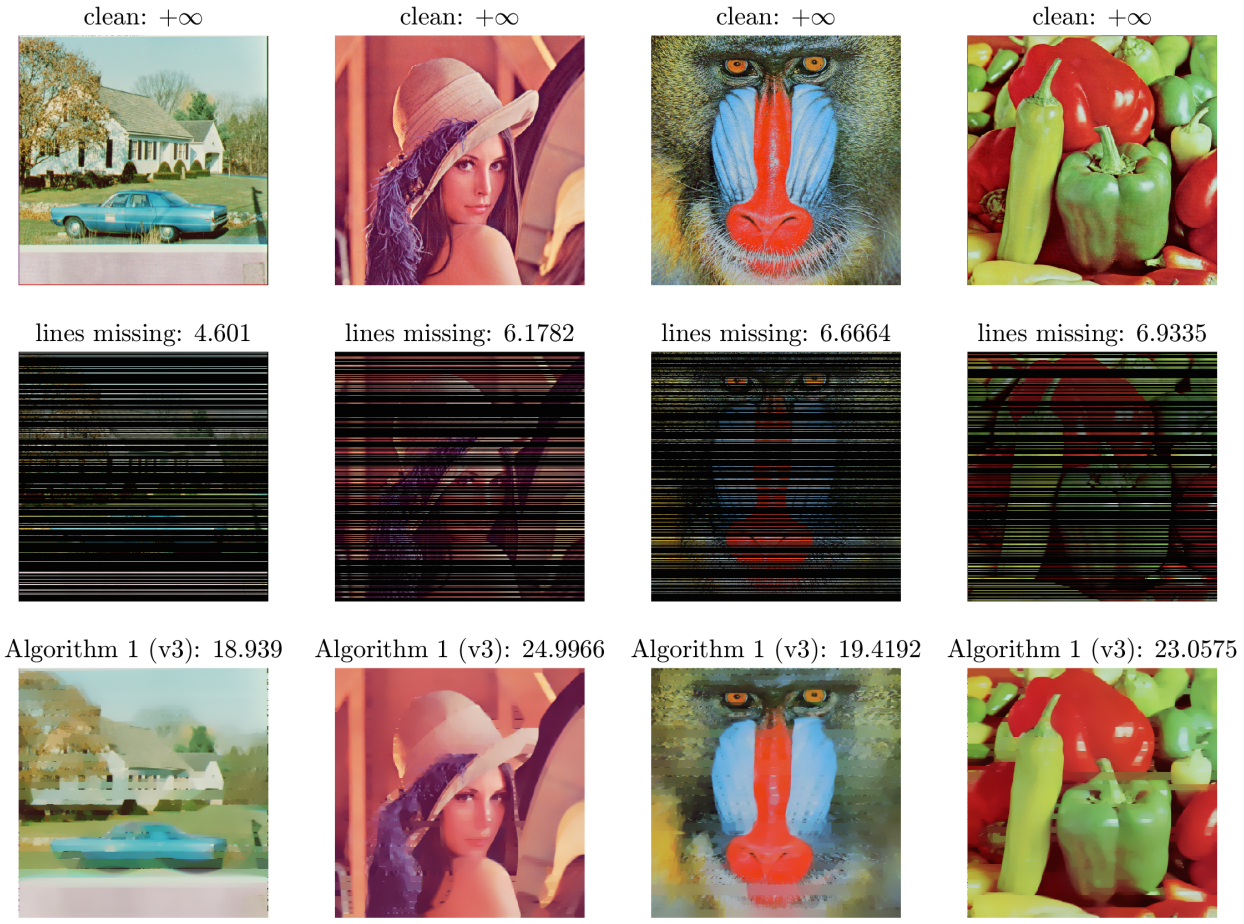

Figure 3.7: Visual outcome of Algorithm 1 (v3) on four images with 80% lines missing

3.7 Conclusions

In this chapter, we have studied a class of convex-concave saddle-point problems (SP) involving non-bilinear coupling function. We have developed two novel primal-dual algorithms to solve (SP) and its primal-dual pair reformulation (P)-(D). Our algorithms have single-loop, where all the parameters are updated with explicit formulas. The first algorithm, Algorithm 1, achieves both ergodic and semi-ergodic optimal $\mathcal{O}\left(\frac{1}{k}\right)$ convergence rates on the duality gap, and can be boosted up to $\min\left\{\mathcal{O}\left(\frac{1}{k}\right), \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ non-ergodic primal convergence rate. Under strong convexity of F, our second algorithm, Algorithm 2, can be accelerated to have $\mathcal{O}\left(\frac{1}{k^2}\right)$ and $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right), \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$ convergence rates. To the best of our knowledge, these are the first algorithms that achieve such fast rates for non-bilinear saddle-point problems.

CHAPTER 4
CONCLUSIONS AND OUTLOOK

4.1 Conclusions

In this thesis, we have addressed two research questions brought up in Section 1.1:

1. To developing a preprocessing algorithm for SDPs (1.3);
2. To developing first-order primal-dual algorithms to solve the composite convex program (P), or its saddle-point form (SP).

For the first research question, we proposed a facial reduction algorithm, Sieve-SDP, to preprocess SDPs. It is extremely simple, works in machine precision, and does not rely on any optimization solver. We have developed Sieve-SDP as a software package in MATLAB, and thus it is ready to use and to be integrated into solvers that can be called from MATLAB. Finally, we present extensive computational results on general SDPs to show it competitiveness compared to existing methods.

To answer the second research question, we developed two novel primal-dual algorithms. They have mild assumptions, low per-iteration complexity, and all the parameters are updated with explicit formulas. The first algorithm achieves both ergodic and semi-ergodic optimal $\mathcal{O}\left(\frac{1}{k}\right)$ convergence rates on the duality gap, and can be boosted up to $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ non-ergodic primal convergence rate. Under strong convexity, our second algorithm can be accelerated to have $\mathcal{O}\left(\frac{1}{k^2}\right)$ and $\min\left\{\mathcal{O}\left(\frac{1}{k^2}\right),\ \underline{o}\left(\frac{1}{k^2\sqrt{\log k}}\right)\right\}$ convergence rates. To the best of our knowledge, these are the first algorithms that achieve such fast rates for non-bilinear saddle-point problems.

4.2 Outlook

We first address possible future works on Sieve-SDP. Since the first version of [153] was available online, we have been reached out by researchers on questions arisen when they apply Sieve-SDP software to preprocess SDPs in their research. Apparently, it is beneficial to maintain the software

and upgrade it to cater non-standard datatypes. Furthermore, it is interesting to look into more efficient and effective ways to recover dual solution, and whether there is a certificate when dual solution recovery is impossible.

As for developing first-order primal-dual algorithms for the saddle-point problem

$$\min_x \max_y \quad F(x) + \langle g(x), y\rangle - H^*(y), \tag{SP}$$

here are several directions worth exploring.

- Is it possible to generalize our algorithms and their convergence guarantees for problems with the coupling term $\langle g(x), y\rangle$ in (SP) replaced by more general function $\Phi(x, y)$, which is convex in x and concave in y?
- Currently, our semi-ergodic and non-ergodic convergence guarantees assume linearity or boundedness of g. Is it possible to relax this boundedness condition?
- Our first algorithm enjoys $\min\left\{\mathcal{O}\left(\frac{1}{k}\right),\ \underline{o}\left(\frac{1}{k\sqrt{\log k}}\right)\right\}$ non-ergodic convergence rate. Is it possible to improve this rate to at least $o\left(\frac{1}{k}\right)$? Currently, the most related work has proved $o\left(\frac{1}{\sqrt{k}}\right)$ rate when g is linear [28–30].
- Stochastic or coordinate descent techniques can be very efficient when the problem size is very large. Therefore, we believe that our algorithms' performance can be further boosted using such techniques.
- With restarting techniques, the empirical performance of our algorithms would be much better. The theoretical guarantees could be derived by combining our theory with the techniques in [125].

APPENDIX A

SUPPLEMENTAL CONTENT FOR CHAPTER 2

A.1 Very Detailed Results

We now give very detailed computational results on problems from five datasets. We only give results on problems that were reduced by at least one of the five preprocessors.

In all tables the first column gives the number of the SDP, the second gives the name, and the third gives the names of the preprocessing methods.

The next two columns describe the size of the problem. The entry "f; l; s" describes the size of the variables of the problem, where

- the number "f" is the number of *free* variables;
- the number "l" is the number of *linear nonnegative* variables;
- the number, or numbers "s" describes the size of the *PSD* variable blocks, possibly with multiplicity.

For example, the tuple $3; 5; 6, 5_3$ means that a problem has 3 free variables; 5 linear nonnegative variables; and four PSD matrix variable blocks, which are of order $6, 5, 5, 5$, respectively. The number m is the number of constraints.

In the next three columns, we put information about the preprocessors. In the column "Red." we put 1 if a preprocessor reduced a problem, and 0 if it did not. In this column under Sieve-SDP, we put the same entries, except if Sieve-SDP actually proved infeasibility, then we entered "infeas" there. The number $\mathrm{t_{prep}}$ is the time spent on preprocessing, and the number $\mathrm{t_{conv}}$ is the time spent on converting from MOSEK format to SeDuMi format and back (for the methods pd1, pd2, dd1, dd2, as they preprocessing using SeDuMi format).

In the next four columns we show how MOSEK performed. In the column "Infeas" we have a 1 if MOSEK detected infeasibility, and 0 if it did not. The column "Obj (P, D)" shows the objective values (primal and dual, respectively). The column DIMACS contains the *greatest* absolute value of the DIMACS errors.

In the last column we show *help codes*, which show whether a preprocessor helped or hurt to

solve an SDP. Although the help codes can be deduced from the previous columns, they still help to quickly evaluate the preprocessors. A positive help code means that a preprocessor helped, and a negative one means that it hurt. In detail, let us recall from Section 2.2 that $\text{DIMACS}_{\text{before}}$ ($\text{DIMACS}_{\text{after}}$) is the greatest absolute value of the DIMACS error before (after) preprocessing. Furthermore, we let $\text{obj}_{\text{before}}$ ($\text{obj}_{\text{after}}$) be the primal objective values before (after) preprocessing. Given this notation, the help code is

- Code 1, if
  - Sieve-SDP detects infeasibility, or
  - MOSEK does not detect infeasibility *before* preprocessing, but does detect infeasibility *after* preprocessing;
- Code -1, if MOSEK detects infeasibility *before* preprocessing, but does not detect infeasibility *after* preprocessing;
- Code 2, if it is not ± 1 and preprocessing improved the DIMACS error, i.e.,

$$\text{DIMACS}_{\text{before}} > 10^{-6} \quad \text{and} \quad \frac{\text{DIMACS}_{\text{after}}}{\text{DIMACS}_{\text{before}}} < \frac{1}{10};$$

- Code -2, if it is not ± 1 and preprocessing worsened the DIMACS error, i.e.,

$$\text{DIMACS}_{\text{after}} > 10^{-6} \quad \text{and} \quad \frac{\text{DIMACS}_{\text{after}}}{\text{DIMACS}_{\text{before}}} > 10;$$

- Code 3, if preprocessing shifted the objective function, i.e., if help codes ± 1 and -2 do not apply, and

$$\frac{|\text{obj}_{\text{before}} - \text{obj}_{\text{after}}|}{1 + |\text{obj}_{\text{before}}|} > 10^{-6};$$

- Code "MM", if a code ran out of memory or crashed.

A.1.1 Detailed results on the Permenter-Parrilo dataset

This dataset has 68 problems, 59 of which were reduced by at least one of the five preprocessing methods. There is one problem where pd2 crashed.

Table A.1: Detailed results on PP dataset, part 1 of 5

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
1	CompactDim2R1	none	$0; 3; 3$	5				0	3.79e+06, 4.20e+06	2.22e+01	3.02	
		pd1	$0; 3; 1$	3	1	0.05	0.00	1	0.00e+00, 1.00e+00	7.07e-01	0.64	1
		pd2	$0; 3; 1$	3	1	0.04	0.00	1	0.00e+00, 1.00e+00	7.07e-01	0.69	1
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			infeas	0.01					0.00	1
2	CompactDim2R2	none	$0; 0; 6, 3_3$	14				0	6.41e-10, 6.81e-10	7.07e-01	3.16	
		pd1	$0; 0; 1_3$	2	1	0.11	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.13	1
		pd2	$0; 0; 1_3$	2	1	0.09	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.04	1
		dd1			0	0.03	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.01					0.00	1
3	CompactDim2R3	none	$0; 0; 10, 6_3$	27				0	1.50e+00, 1.50e+00	1.15e-07	2.03	
		pd1	$0; 0; 1_3$	2	1	0.14	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.07	1
		pd2	$0; 0; 1_3$	2	1	0.13	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.09	1
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			infeas	0.01					0.00	1
4	CompactDim2R4	none	$0; 0; 15, 10_3$	44				0	1.50e+00, 1.50e+00	1.13e-07	2.07	
		pd1	$0; 0; 1_3$	2	1	0.20	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.09	1
		pd2	$0; 0; 1_3$	2	1	0.17	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.09	1
		dd1			0	0.03	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.02					0.00	1
5	CompactDim2R5	none	$0; 0; 21, 15_3$	65				0	1.50e+00, 1.50e+00	1.83e-07	2.05	
		pd1	$0; 0; 1_3$	2	1	0.25	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.06	1
		pd2	$0; 0; 1_3$	2	1	0.27	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.07	1
		dd1			0	0.02	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP			infeas	0.03					0.00	1
6	CompactDim2R6	none	$0; 0; 28, 21_3$	90				0	1.50e+00, 1.50e+00	2.70e-07	2.06	
		pd1	$0; 0; 1_3$	2	1	0.32	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.10	1
		pd2	$0; 0; 1_3$	2	1	0.38	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.03	1
		dd1			0	0.05	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP			infeas	0.04					0.00	1
7	CompactDim2R7	none	$0; 0; 36, 28_3$	119				0	1.50e+00, 1.50e+00	3.66e-07	2.13	
		pd1	$0; 0; 1_3$	2	1	0.41	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.08	1
		pd2	$0; 0; 1_3$	2	1	0.59	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.06	1
		dd1			0	0.03	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP			infeas	0.06					0.00	1
8	CompactDim2R8	none	$0; 0; 45, 36_3$	152				0	1.50e+00, 1.50e+00	5.61e-07	2.07	
		pd1	$0; 0; 1_3$	2	1	0.56	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.02	1
		pd2	$0; 0; 1_3$	2	1	0.86	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.05	1
		dd1			0	0.03	0.00					
		dd2			0	0.09	0.00					
		Sieve-SDP			infeas	0.08					0.00	1
9	CompactDim2R9	none	$0; 0; 55, 45_3$	189				0	1.50e+00, 1.50e+00	6.27e-07	2.11	
		pd1	$0; 0; 1_3$	2	1	0.71	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.08	1
		pd2	$0; 0; 1_3$	2	1	1.28	0.00	1	1.00e+00, 2.00e+00	7.07e-01	1.05	1
		dd1			0	0.03	0.00					
		dd2			0	0.14	0.00					
		Sieve-SDP			infeas	0.11					0.00	1
10	CompactDim2R10	none	$0; 0; 66, 55_3$	230				0	1.50e+00, 1.50e+00	5.17e-07	2.28	
		pd1	$0; 0; 1_3$	2	1	0.86	0.01	1	1.00e+00, 2.00e+00	7.07e-01	1.17	1
		pd2	$0; 0; 1_3$	2	1	1.90	0.01	1	1.00e+00, 2.00e+00	7.07e-01	1.09	1
		dd1			0	0.04	0.01					
		dd2			0	0.18	0.01					
		Sieve-SDP			infeas	0.15					0.00	1
11	Example1	none	$0; 0; 3$	2				0	0.00e+00, 0.00e+00	0.00e+00	1.72	
		pd1	$0; 0; 2$	1	1	0.06	0.01	0	0.00e+00, 0.00e+00	0.00e+00	1.05	
		pd2	$0; 0; 2$	1	1	0.05	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.96	
		dd1	$5; 0; 1$	2	1	0.07	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.96	
		dd2	$5; 0; 1$	2	1	0.06	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.96	
		Sieve-SDP	$0; 0; 2$	1	1	0.03		0	0.00e+00, 0.00e+00	0.00e+00	1.63	
12	Example2	none	$0; 0; 3$	2				0	3.33e-01, 3.33e-01	5.05e-02	1.73	
		pd1	$0; 0; 2$	1	1	0.05	0.01	0	1.00e+00, 1.00e+00	0.00e+00	0.94	2,3
		pd2	$0; 0; 2$	1	1	0.05	0.00	0	1.00e+00, 1.00e+00	0.00e+00	0.97	2,3
		dd1	$3; 0; 2$	2	1	0.05	0.00	0	4.73e-15, 1.82e-14	2.75e-14	1.01	2,3
		dd2	$3; 0; 2$	2	1	0.02	0.00	0	4.73e-15, 1.82e-14	2.75e-14	1.01	2,3
		Sieve-SDP	$0; 0; 2$	1	1	0.01		0	1.00e+00, 1.00e+00	0.00e+00	2.61	2,3

Table A.2: Detailed results on PP dataset, part 2 of 5

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
13	Example3	none	0; 0; 3	4				0	3.33e-01, 3.33e-01	6.90e-02	1.79	
		pd1	0; 0; 2	1	1	0.03	0.00	0	1.17e-07, 1.69e-07	5.14e-08	1.44	2,3
		pd2	0; 0; 2	1	1	0.03	0.00	0	1.17e-07, 1.69e-07	5.14e-08	1.48	2,3
		dd1	3; 0; 2	4	1	0.02	0.01	0	4.73e-15, 1.82e-14	2.75e-14	1.00	2,3
		dd2	3; 0; 2	4	1	0.03	0.00	0	4.73e-15, 1.82e-14	2.75e-14	0.99	2,3
		Sieve-SDP	0; 0; 2	1	1	0.01		0	1.17e-07, 1.69e-07	5.14e-08	1.59	2,3
14	Example4	none	0; 0; 3	3				1	0.00e+00, 3.74e-07	5.00e-01	1.43	
		pd1	0; 0; 1	1	1	0.03	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.64	
		pd2	0; 0; 1	1	1	0.03	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.63	
		dd1	5; 0; 1	3	1	0.03	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.96	-1
		dd2	5; 0; 1	3	1	0.04	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.99	-1
		Sieve-SDP			infeas	0.00					0.00	1
15	Example6	none	0; 0; 8	8				0	1.00e+00, 1.00e+00	1.95e-08	0.66	
		pd1	0; 0; 5	4	1	0.04	0.00	0	1.00e+00, 1.00e+00	0.00e+00	0.99	
		pd2	0; 0; 5	4	1	0.04	0.00	0	1.00e+00, 1.00e+00	0.00e+00	0.98	
		dd1	26; 0; 4	8	1	0.02	0.00	0	1.00e+00, 1.00e+00	9.75e-09	1.02	
		dd2	26; 0; 4	8	1	0.02	0.00	0	1.00e+00, 1.00e+00	9.75e-09	1.19	
		Sieve-SDP	0; 0; 5	4	1	0.01		0	1.00e+00, 1.00e+00	0.00e+00	0.56	
16	Example7	none	0; 0; 5	3				0	0.00e+00, 0.00e+00	0.00e+00	0.60	
		pd1	0; 0; 4	2	1	0.02	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.96	
		pd2	0; 0; 4	2	1	0.03	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.97	
		dd1	14; 0; 1	3	1	0.03	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.98	
		dd2	14; 0; 1	3	1	0.03	0.00	0	0.00e+00, 0.00e+00	0.00e+00	1.00	
		Sieve-SDP	0; 0; 4	2	1	0.00		0	0.00e+00, 0.00e+00	0.00e+00	0.54	
17	Example9size20	none	0; 0; 20	20				1	0.00e+00, 3.39e-01	5.00e-01	2.58	
		pd1	0; 0; 1	1	1	0.06	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.63	
		pd2	0; 0; 1	1	1	0.04	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.62	
		dd1	209; 0; 1	20	1	0.19	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.95	-1
		dd2	209; 0; 1	20	1	0.24	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.97	-1
		Sieve-SDP			infeas	0.00					0.00	1
18	Example9size100	none	0; 0; 100	100				1	0.00e+00, 3.43e-01	5.00e-01	0.83	
		pd1	0; 0; 1	1	1	0.04	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.64	
		pd2	0; 0; 1	1	1	0.19	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.63	
		dd1	5049; 0; 1	100	1	1.33	0.00	0	0.00e+00, 0.00e+00	0.00e+00	1.01	-1
		dd2	5049; 0; 1	100	1	3.50	0.00	0	0.00e+00, 0.00e+00	0.00e+00	1.00	-1
		Sieve-SDP			infeas	0.00					0.00	1
19	RandGen6	none	0; 0; 320	140				0	3.95e-06, 3.24e-06	2.29e-05	24.07	
		pd1			0	3.64	1.00					
		pd2			0	16.39	1.00					
		dd1			0	0.75	1.00					
		dd2	19985; 0; 250	140	1	37.13	2.14	0	1.68e-07, 1.26e-11	8.00e-07	5.88	2,3
		Sieve-SDP	0; 0; 120	70	1	2.10		0	3.73e-06, 3.04e-06	9.17e-06	2.32	
20	RandGen7	none	0; 0; 40	27				0	9.42e-07, 4.22e-07	4.69e-06	0.67	
		pd1			0	0.03	0.01					
		pd2	0; 0; 28	14	1	0.10	0.02	0	9.85e-07, 4.53e-07	3.27e-06	1.04	
		dd1			0	0.02	0.01					
		dd2	649; 0; 18	27	1	0.11	0.01	0	2.65e-11, 4.69e-16	7.21e-11	1.08	2
		Sieve-SDP	0; 0; 28	14	1	0.02		0	9.85e-07, 4.53e-07	3.27e-06	0.72	
21	RandGen8	none	0; 0; 60	40				0	5.41e-09, 2.44e-09	9.31e-08	0.83	
		pd1			0	0.04	0.01					
		pd2			0	0.22	0.01					
		dd1			0	0.02	0.01					
		dd2	1269; 0; 33	40	1	0.33	0.02	0	2.15e-15, 2.78e-19	6.90e-14	1.05	
		Sieve-SDP	0; 0; 30	20	1	0.03		0	1.52e-09, 6.33e-10	9.04e-09	0.69	
22	copos_1	none	0; 0; 35	210				0	0.00e+00, 1.11e-08	4.40e-07	0.66	
		pd1			0	0.02	0.00					
		pd2	0; 0; 25	160	1	0.06	0.02	0	0.00e+00, -3.86e-10	2.12e-08	1.01	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP			0	0.02						
23	copos_2	none	0; 0; 120	1716				0	0.00e+00, 5.76e-11	1.69e-08	1.83	
		pd1			0	0.03	0.00					
		pd2	0; 0; 96	1524	1	0.57	0.11	0	0.00e+00, -2.31e-13	6.38e-11	1.72	
		dd1			0	0.02	0.00					
		dd2			0	0.13	0.00					
		Sieve-SDP			0	0.09						
24	copos_3	none	0; 0; 286	8008				0	0.00e+00, -4.93e-10	1.59e-07	44.68	
		pd1			0	0.10	0.01					
		pd2	0; 0; 242	7524	1	37.41	0.57	0	0.00e+00, -4.51e-11	1.26e-08	30.28	
		dd1			0	0.06	0.01					
		dd2			0	0.85	0.01					
		Sieve-SDP			0	0.46						

Table A.3: Detailed results on PP dataset, part 3 of 5

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
25	copos_4	none	0; 0; 560	27132				0	0.00e+00, -9.00e-11	7.21e-08	1526.50	
		pd1			0	0.46	0.06					
		pd2	0; 0; 490	26152	1	26.16	1.98	0	0.00e+00, -1.70e-10	6.56e-08	1139.18	
		dd1			0	0.36	0.06					
		dd2			0	5.09	0.06					
		Sieve-SDP			0	1.80						
26	cprank_1	none	$9; 0; 1_9, 10, 9$	46				0	-3.00e+00, -3.00e+00	3.50e-08	1.32	
		pd1			0	0.08	0.00					
		pd2			0	0.03	0.00					
		dd1	$30; 0; 1_7, 8, 9$	46	1	0.07	0.01	0	-3.00e+00, -3.00e+00	4.62e-08	1.16	
		dd2	$30; 0; 1_7, 8, 9$	46	1	0.06	0.01	0	-3.00e+00, -3.00e+00	3.88e-08	1.17	
		Sieve-SDP			0	0.01						
27	cprank_2	none	$1296; 0; 1_{81}, 82, 81$	3322				0	-9.00e+00, -9.00e+00	6.62e-08	15.40	
		pd1			0	0.08	0.00					
		pd2			0	0.18	0.00					
		dd1	$3456; 0; 1_{49}, 50, 81$	3322	1	0.14	0.31	0	-9.00e+00, -9.00e+00	6.64e-09	10.50	
		dd2	$3456; 0; 1_{49}, 50, 81$	3322	1	0.50	0.32	0	-9.00e+00, -9.00e+00	1.51e-09	9.75	
		Sieve-SDP			0	0.06						
28	hinf12	none	$0; 0; 6_2, 12$	43				0	-1.45e-13, -1.17e-13	1.80e+00	1.38	
		pd1			0	0.03	0.00					
		pd2	0; 0; 6, 2, 6	22	1	0.04	0.00	0	-2.64e-15, -1.77e-15	1.79e+00	1.69	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP			0	0.01						
29	horn2	none	0; 0; 4	7				0	0.00e+00, 6.69e-13	9.06e-13	2.03	
		pd1			0	0.02	0.00					
		pd2	0; 0; 2	3	1	0.06	0.00	0	0.00e+00, 0.00e+00	1.57e-16	0.99	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			0	0.01						
30	horn3	none	0; 0; 10	28				0	0.00e+00, 1.46e-07	8.62e-07	2.00	
		pd1			0	0.02	0.00					
		pd2	0; 0; 6	16	1	0.05	0.00	0	0.00e+00, 3.53e-09	2.65e-08	0.99	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			0	0.00						
31	horn4	none	0; 0; 20	84				0	0.00e+00, 1.13e-07	1.90e-06	2.14	
		pd1			0	0.02	0.00					
		pd2	0; 0; 14	60	1	0.07	0.01	0	0.00e+00, 7.11e-09	7.44e-08	1.07	2
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			0	0.01						
32	horn5	none	0; 0; 35	210				0	0.00e+00, 1.07e-08	2.69e-07	2.05	
		pd1			0	0.02	0.00					
		pd2	0; 0; 25	160	1	0.08	0.01	0	0.00e+00, -2.28e-09	2.35e-07	0.99	
		dd1			0	0.03	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			0	0.03						
33	hornD2	none	0; 0; 4	3				0	-5.25e-08, 0.00e+00	5.25e-08	2.04	
		pd1			0	0.02	0.00					
		pd2			0	0.03	0.00					
		dd1			0	0.03	0.00					
		dd2	7; 0; 2	3	1	0.05	0.00	0	-1.88e-16, 0.00e+00	1.78e-15	1.00	
		Sieve-SDP			0	0.00						
34	hornD3	none	0; 0; 10	27				0	-5.58e-08, 0.00e+00	8.62e-07	2.01	
		pd1			0	0.03	0.00					
		pd2			0	0.02	0.00					
		dd1			0	0.03	0.00					
		dd2	34; 0; 6	27	1	0.05	0.00	0	-8.68e-10, 0.00e+00	1.88e-08	1.13	
		Sieve-SDP			0	0.01						
35	hornD4	none	0; 0; 20	126				0	1.77e-07, 0.00e+00	1.02e-06	2.04	
		pd1			0	0.02	0.00					
		pd2			0	0.04	0.00					
		dd1			0	0.02	0.00					
		dd2	105; 0; 14	126	1	0.06	0.01	0	7.49e-08, 0.00e+00	2.38e-07	1.12	
		Sieve-SDP			0	0.02						
36	hornD5	none	0; 0; 35	420				0	2.32e-08, 0.00e+00	1.83e-07	2.06	
		pd1			0	0.03	0.00					
		pd2			0	0.04	0.00					
		dd1			0	0.03	0.00					
		dd2	305; 0; 25	420	1	0.09	0.03	0	5.58e-10, 0.00e+00	2.00e-09	1.26	
		Sieve-SDP			0	0.04						

Table A.4: Detailed results on PP dataset, part 4 of 5

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
37	hybridLyap	none	$860; 0; 6, 108, 11_{10}$	3093				0	0.00e+00, 7.29e-07	2.11e-04	7.85	
		pd1	$860; 0; 6, 56, 11, 1_2, 11, 1_2, 11_2$	1607	1	0.16	0.09	0	0.00e+00, 3.48e-07	6.61e-05	1.49	
		pd2	$860; 0; 6, 34, 8, 1_2, 8, 1_2, 9, 7$	1173	1	1.02	0.05	0	0.00e+00, 4.24e-09	4.86e-07	1.23	2
		dd1			0	0.05	0.00					
		dd2			0	0.14	0.00					
		Sieve-SDP			0	0.05						
38	leverage_limit	none	$0; 18100; 151_{100}, 30_{100}$	68195				0	-8.75e+01, -8.75e+01	1.53e-05	278.60	
		pd1			0	2.10	0.17					
		pd2	$0; 18100; 151_{99}, 121, 30_{100}$	67700	1	120.98	7.87	0	-8.75e+01, -8.75e+01	5.63e-06	150.78	3
		dd1	$958500; 18100; 61_{100}, 30_{100}$	68195	1	3.87	7.20	0	-8.75e+01, -8.75e+01	2.45e-05	250.27	
		dd2	$1193505; 18100; 1_{99}, 31$	68195	1	291.58	1.39	-1	-3.35e+00, 0.00e+00	1.03e+01	1.97	-2
		Sieve-SDP	$0; 18100; 143_{97}, 141_3, 26_{98}, 25_2$	56196	1	253.43		0	-8.74e+01, -8.74e+01	1.73e-05	179.26	3
39	long_only	none	$0; 9000; 91_{100}, 30_{100}$	59095				0	-4.13e+01, -4.13e+01	5.23e-06	373.38	
		pd1			0	1.18	0.17					
		pd2	$0; 9000; 91_{99}, 61, 30_{100}$	58600	1	24.33	6.91	0	-4.13e+01, -4.13e+01	4.64e-07	205.50	2,3
		dd1	$229500; 9000; 61_{100}, 30_{100}$	59095	1	1.77	6.96	0	-4.13e+01, -4.13e+01	6.47e-06	246.03	3
		dd2	$229500; 9000; 61_{100}, 30_{100}$	59095	1	531.60	6.94	0	-4.13e+01, -4.13e+01	2.80e-06	315.18	3
		Sieve-SDP	$0; 8573; 83_{97}, 81_3, 26_{98}, 25_2$	46670	1	190.92		0	-4.13e+01, -4.13e+01	1.64e-06	94.12	3
40	sector_neutral	none	$0; 12000; 121_{100}, 30_{100}$	62392				0	-1.21e+02, -1.21e+02	8.35e-05	152.27	
		pd1			0	1.84	0.26					
		pd2	$0; 12000; 121_{99}, 91, 30_{100}$	61897	1	183.96	7.17	0	-1.21e+02, -1.21e+02	2.79e-04	150.19	3
		dd1	$549000; 12000; 61_{100}, 30_{100}$	62392	1	2.79	7.05	0	-1.21e+02, -1.21e+02	8.23e-05	154.78	3
		dd2	$549000; 12000; 61_{100}, 30_{100}$	62392	1	217.62	7.13	0	-1.21e+02, -1.21e+02	1.24e-04	140.83	3
		Sieve-SDP	$0; 12000; 121_{99}, 111, 30_{100}$	62247	1	52.25		0	-1.21e+02, -1.21e+02	1.76e-04	151.52	3
41	unconstrained	none	$0; 12000; 121_{100}, 30_{100}$	62095				0	-1.33e+02, -1.33e+02	7.89e-05	279.82	
		pd1			0	1.52	0.15					
		pd2	$0; 12000; 121_{99}, 91, 30_{100}$	61600	1	38.70	6.95	0	-1.33e+02, -1.33e+02	1.42e-05	282.87	3
		dd1	$549000; 12000; 61_{100}, 30_{100}$	62095	1	2.63	6.74	0	-1.33e+02, -1.33e+02	3.34e-05	258.89	3
		dd2	$549000; 12000; 61_{100}, 30_{100}$	62095	1	505.61	6.70	0	-1.33e+02, -1.33e+02	3.64e-05	260.11	3
		Sieve-SDP	$0; 12000; 113_{97}, 111_3, 26_{98}, 25_2$	50097	1	213.04		0	-1.28e+02, -1.28e+02	1.64e-05	185.98	3
42	unboundDim1R1	none	$0; 2; 2$	2				0	1.33e-09, -7.05e-10	4.38e-09	2.89	
		pd1			0	0.05	0.01					
		pd2			0	0.03	0.01					
		dd1			0	0.03	0.01					
		dd2			0	0.02	0.01					
		Sieve-SDP	$0; 1; 1$	1	1	0.01		0	0.00e+00, 0.00e+00	0.00e+00	2.25	
43	unboundDim1R2	none	$0; 0; 3, 2_2$	4				0	-8.91e-15, -8.01e-15	7.07e-01	4.52	
		pd1	$0; 0; 1_2$	1	1	0.11	0.01	0	0.00e+00, 0.00e+00	0.00e+00	0.43	2
		pd2	$0; 0; 1_2$	1	1	0.10	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.47	2
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.00		0	0.00e+00, 0.00e+00	0.00e+00	2.39	2
44	unboundDim1R3	none	$0; 0; 4, 3_2$	6				0	-2.04e-11, -2.02e-11	7.07e-01	4.17	
		pd1	$0; 0; 1_2$	1	1	0.12	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.50	2
		pd2	$0; 0; 1_2$	1	1	0.11	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.47	2
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.01		0	0.00e+00, 0.00e+00	0.00e+00	2.31	2
45	unboundDim1R4	none	$0; 0; 5, 4_2$	8				0	-2.34e-10, -2.32e-10	7.07e-01	3.79	
		pd1	$0; 0; 1_2$	1	1	0.14	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.43	2
		pd2	$0; 0; 1_2$	1	1	0.14	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.43	2
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.01		0	0.00e+00, 0.00e+00	0.00e+00	2.41	2
46	unboundDim1R5	none	$0; 0; 6, 5_2$	10				0	-1.00e+00, -1.00e+00	9.88e-08	2.74	
		pd1	$0; 0; 1_2$	1	1	0.16	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.43	3
		pd2	$0; 0; 1_2$	1	1	0.20	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.42	3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.01		0	0.00e+00, 0.00e+00	0.00e+00	2.32	3
47	unboundDim1R6	none	$0; 0; 7, 6_2$	12				0	-1.00e+00, -1.00e+00	2.15e-07	2.78	
		pd1	$0; 0; 1_2$	1	1	0.20	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.50	3
		pd2	$0; 0; 1_2$	1	1	0.22	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.49	3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.01		0	0.00e+00, 0.00e+00	0.00e+00	2.24	3
48	unboundDim1R7	none	$0; 0; 8, 7_2$	14				0	-1.00e+00, -1.00e+00	5.11e-08	2.82	
		pd1	$0; 0; 1_2$	1	1	0.21	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.45	3
		pd2	$0; 0; 1_2$	1	1	0.23	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.44	3
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.02		0	0.00e+00, 0.00e+00	0.00e+00	2.32	3

Table A.5: Detailed results on PP dataset, part 5 of 5

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
49	unboundDim1R8	none	$0; 0; 9, 8_2$	16				0	-1.00e+00, -1.00e+00	5.43e-08	2.29	
		pd1	$0; 0; 1_2$	1	1	0.52	0.02	0	0.00e+00, 0.00e+00	0.00e+00	0.44	3
		pd2	$0; 0; 1_2$	1	1	0.53	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.45	3
		dd1			0	0.07	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.13		0	0.00e+00, 0.00e+00	0.00e+00	1.74	3
50	unboundDim1R9	none	$0; 0; 10, 9_2$	18				0	-1.00e+00, -1.00e+00	6.50e-08	2.09	
		pd1	$0; 0; 1_2$	1	1	0.31	0.01	0	0.00e+00, 0.00e+00	0.00e+00	0.43	3
		pd2	$0; 0; 1_2$	1	1	0.30	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.64	3
		dd1			0	0.04	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.05		0	0.00e+00, 0.00e+00	0.00e+00	1.60	3
51	unboundDim1R10	none	$0; 0; 11, 10_2$	20				0	-1.00e+00, -1.00e+00	1.41e-07	2.76	
		pd1	$0; 0; 1_2$	1	1	0.28	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.44	3
		pd2	$0; 0; 1_2$	1	1	0.32	0.00	0	0.00e+00, 0.00e+00	0.00e+00	0.45	3
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 1_2$	1	1	0.04		0	0.00e+00, 0.00e+00	0.00e+00	2.33	3
52	vamos_5_34	none	0; 0; 52	721				0	0.00e+00, -4.18e-09	5.21e-08	2.10	
		pd1			0	0.07	0.00					
		pd2				MM						MM
		dd1			0	0.05	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP			0	0.06						
53	wei_wagner_F7_minus_4	none	0; 0; 8	31				0	0.00e+00, -9.60e-13	1.11e-11	1.87	
		pd1			0	0.02	0.00					
		pd2	0; 0; 5	14	1	0.08	0.01	0	0.00e+00, -5.80e-11	2.12e-10	0.99	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			0	0.01						
54	wei_wagner_P7	none	0; 0; 8	32				0	0.00e+00, -1.46e-08	9.09e-08	1.99	
		pd1			0	0.02	0.00					
		pd2	0; 0; 4	10	1	0.05	0.00	0	0.00e+00, -3.02e-10	1.31e-09	1.04	
		dd1			0	0.03	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			0	0.01						
55	wei_wagner_W3Plus	none	0; 0; 8	31				0	0.00e+00, -6.06e-09	5.47e-08	1.95	
		pd1			0	0.02	0.00					
		pd2	0; 0; 3	6	1	0.05	0.00	0	0.00e+00, -4.77e-09	1.11e-08	1.01	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP			0	0.00						
56	wei_wagner_W3_PlusE	none	0; 0; 9	38				0	0.00e+00, -9.18e-09	5.53e-08	1.98	
		pd1			0	0.02	0.00					
		pd2	0; 0; 5	15	1	0.06	0.00	0	0.00e+00, -7.21e-09	3.21e-08	1.02	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			0	0.00						
57	wei_wagner_nP_minus_1_24	none	0; 0; 12	64				0	0.00e+00, -5.50e-09	8.80e-08	2.03	
		pd1			0	0.02	0.00					
		pd2	0; 0; 6	21	1	0.07	0.00	0	0.00e+00, -1.08e-11	5.60e-11	1.02	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			0	0.01						
58	wei_wagner_nP_minus_9_12	none	0; 0; 12	64				0	0.00e+00, -3.92e-09	4.87e-08	1.98	
		pd1			0	0.02	0.00					
		pd2	0; 0; 5	15	1	0.05	0.00	0	0.00e+00, -4.11e-15	2.34e-14	1.02	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP			0	0.01						
59	wei_wagner_vamos_12	none	0; 0; 16	103				0	0.00e+00, -1.59e-08	1.38e-07	2.10	
		pd1			0	0.02	0.00					
		pd2	0; 0; 13	74	1	0.06	0.01	0	0.00e+00, -2.54e-10	1.62e-09	1.03	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP			0	0.01						

A.1.2 Detailed results on the Mittelmann dataset

This dataset has 31 problems, 8 of which were reduced by at least one of the five preprocessing methods. There were 5 problems where pd2 or dd2 ran out of memory/crashed.

Table A.6: Detailed results on Mittelmann dataset, part 1 of 2

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
1	diamond_patch	none	0; 0; 5477	5478				0	1.63e+01, 1.63e+01	3.56e-04	10854.97	
		pd1			0	31.05	0.06					
		pd2				MM						MM
		dd1			0	27.94	0.06					
		dd2			0	3008.86	0.06					
		Sieve-SDP			0	1.12						
2	e_moment_stable_17_0.5_2_2	none	$0; 342; 171, 18_{17}$	5984				0	-1.98e-01, -1.98e-01	1.14e-05	38.53	
		pd1	$0; 342; 18_{18}$	1139	1	0.71	0.16	0	-1.98e-01, -1.98e-01	8.44e-06	1.64	
		pd2	$0; 342; 18_{18}$	1139	1	0.86	0.13	0	-1.98e-01, -1.98e-01	8.44e-06	1.66	
		dd1			0	0.08	0.01					
		dd2			0	0.34	0.01					
		Sieve-SDP	$0; 342; 18_{18}$	1139	1	0.52		0	-1.98e-01, -1.98e-01	8.44e-06	1.63	
3	ice_2.0	none	0; 0; 8113	8113				0	6.81e+03, 6.81e+03	4.58e-07	17680.82	
		pd1			0	65.58	0.01					
		pd2				MM						MM
		dd1			0	64.03	0.01					
		dd2				MM						MM
		Sieve-SDP			0	0.80						
4	G60_mb	none	0; 0; 7000	7001				0	1.93e+03, 1.93e+03	6.64e-05	29138.79	
		pd1			0	107.66	10.87					
		pd2				MM						MM
		dd1			0	72.76	10.87					
		dd2				MM						MM
		Sieve-SDP			0	22.42						
5	maxG60	none	0; 0; 7000	7000				0	-1.52e+04, -1.52e+04	6.73e-07	5217.88	
		pd1			0	47.47	0.01					
		pd2				MM						MM
		dd1			0	45.65	0.01					
		dd2				MM						MM
		Sieve-SDP			0	0.47						
6	neu3	none	0; 2; 418	7364				0	7.10e-08, 1.12e-08	2.01e-06	153.03	
		pd1	0; 2; 87	1152	1	0.94	0.11	0	4.69e-08, 3.50e-08	1.94e-07	3.01	2
		pd2	0; 2; 87	1152	1	5.41	0.10	0	4.69e-08, 3.50e-08	1.94e-07	2.97	2
		dd1			0	0.16	0.02					
		dd2			0	2.29	0.02					
		Sieve-SDP	0; 2; 87	1152	1	2.34		0	4.69e-08, 3.50e-08	1.94e-07	2.99	2
7	neu3g	none	0; 0; 462	8007				0	4.58e-08, -2.89e-09	8.67e-07	151.22	
		pd1	0; 0; 87	1151	1	1.32	0.11	0	8.91e-08, 5.65e-08	2.91e-07	3.00	
		pd2	0; 0; 87	1151	1	10.68	0.11	0	8.91e-08, 5.65e-08	2.91e-07	3.09	
		dd1			0	0.19	0.03					
		dd2			0	2.66	0.03					
		Sieve-SDP	0; 0; 87	1151	1	2.26		0	8.91e-08, 5.65e-08	2.91e-07	3.03	
8	p_auss2_3.0	none	0; 0; 9115	9115				0	8.62e+03, 8.62e+03	2.36e-07	25651.19	
		pd1			0	93.91	0.02					
		pd2				MM						MM
		dd1			0	97.11	0.02					
		dd2				MM						MM
		Sieve-SDP			0	0.76						
9	rose13	none	0; 0; 105	2379				0	1.20e+01, 1.20e+01	1.65e-06	7.63	
		pd1	0; 0; 92	1911	1	0.11	0.14	0	1.20e+01, 1.20e+01	4.86e-07	5.26	
		pd2	0; 0; 80	1523	1	0.51	0.11	0	1.20e+01, 1.20e+01	1.98e-07	2.94	
		dd1			0	0.05	0.01					
		dd2			0	0.12	0.01					
		Sieve-SDP	0; 0; 92	1911	1	0.39		0	1.20e+01, 1.20e+01	4.86e-07	5.28	
10	rose15	none	0; 2; 135	3860				0	-3.11e-06, -2.94e-06	1.83e-05	19.47	
		pd1	0; 2; 121	3181	1	0.08	0.24	0	-3.52e-07, -1.52e-07	5.07e-05	11.73	3
		pd2	0; 2; 107	2593	1	0.66	0.19	0	-1.59e-09, -1.57e-09	1.10e-08	5.74	2,3
		dd1			0	0.07	0.00					
		dd2			0	0.18	0.00					
		Sieve-SDP	0; 2; 121	3181	1	0.52		0	-3.52e-07, -1.52e-07	5.07e-05	11.71	3
11	taha1a	none	$0; 0; 252, 56_3, 126_{10}$	3002				0	-1.00e+00, -1.00e+00	9.39e-07	37.54	
		pd1	$0; 0; 126, 56_3, 126_{10}$	2001	1	10.57	0.72	0	-1.00e+00, -1.00e+00	1.20e-07	21.55	
		pd2	$0; 0; 126, 56_3, 126_{10}$	2001	1	18.98	0.75	0	-1.00e+00, -1.00e+00	1.20e-07	21.50	
		dd1			0	0.21	0.06					
		dd2			0	21.47	0.06					
		Sieve-SDP	$0; 0; 126, 56_3, 126_{10}$	2001	1	1.75		0	-1.00e+00, -1.00e+00	1.20e-07	21.70	

Table A.7: Detailed results on Mittelmann dataset, part 2 of 2

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
12	taha1b	none	$0; 3; 286, 66_{20}$	8007				0	-7.73e-01, -7.73e-01	1.59e-07	148.99	
		pd1	$0; 3; 66_{21}$	3002	1	13.97	0.87	0	-7.73e-01, -7.73e-01	1.32e-07	34.29	
		pd2	$0; 3; 66_{21}$	3002	1	18.37	0.85	0	-7.73e-01, -7.73e-01	1.32e-07	33.03	
		dd1			0	0.16	0.04					
		dd2			0	1.82	0.04					
		Sieve-SDP	$0; 3; 66_{21}$	3002	1	1.97		0	-7.73e-01, -7.73e-01	1.32e-07	32.97	
13	taha1c	none	$0; 0; 462, 126_3, 252_{10}$	6187				0	-1.00e+00, -1.00e+00	3.12e-07	314.61	
		pd1	$0; 0; 252, 126_3, 252_{10}$	4367	1	148.36	2.11	0	-1.00e+00, -1.00e+00	4.37e-07	178.22	
		pd2	$0; 0; 252, 126_3, 252_{10}$	4367	1	187.99	2.01	0	-1.00e+00, -1.00e+00	4.37e-07	177.80	
		dd1			0	0.75	0.25					
		dd2			0	156.72	0.25					
		Sieve-SDP	$0; 0; 252, 126_3, 252_{10}$	4367	1	10.85		0	-1.00e+00, -1.00e+00	4.37e-07	182.86	

A.1.3 Detailed results on the Dressler-Illiman-de Wolff dataset

This is a collection of 155 SDP relaxations from polynomial optimization generated by GloptiPoly 3 [62] based on paper [33].

Table A.8: Detailed results on DIW dataset, part 1 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
1	ex3.3_order4	none	0; 1; 15	44				0	3.54e-10, 3.56e-10	6.87e-01	1.69	
		pd1	0; 1; 2	3	1	0.10	0.00	1	0.00e+00, 5.00e-01	5.27e-01	0.63	1
		pd2	0; 1; 2	3	1	0.35	0.00	1	0.00e+00, 5.00e-01	5.27e-01	0.73	1
		dd1			0	0.04	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.03					0.00	1
2	ex3.3_order5	none	0; 0; 21, 3	65				0	6.16e-02, 6.16e-02	1.18e-06	1.25	
		pd1	0; 0; 2, 1	3	1	0.15	0.00	1	0.00e+00, 5.00e-01	5.27e-01	0.64	1
		pd2	0; 0; 2, 1	3	1	0.15	0.00	1	0.00e+00, 5.00e-01	5.27e-01	0.75	1
		dd1			0	0.04	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.03					0.00	1
3	ex3.3_order6	none	0; 0; 28, 6	90				0	6.16e-02, 6.16e-02	1.60e-06	1.00	
		pd1	0; 0; 8, 2	22	1	0.10	0.00	0	6.16e-02, 6.16e-02	4.23e-08	1.08	2
		pd2	0; 0; 8, 2	22	1	0.16	0.00	0	6.16e-02, 6.16e-02	4.23e-08	0.69	2
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 8, 2	22	1	0.03		0	6.16e-02, 6.16e-02	4.23e-08	0.69	2
4	ex3.3_order7	none	0; 0; 36, 10	119				0	6.16e-02, 6.16e-02	3.07e-06	0.65	
		pd1	0; 0; 10, 4	31	1	0.13	0.00	0	6.16e-02, 6.16e-02	6.03e-08	0.59	2,3
		pd2	0; 0; 10, 4	31	1	0.24	0.00	0	6.16e-02, 6.16e-02	6.03e-08	0.65	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 10, 4	31	1	0.05		0	6.16e-02, 6.16e-02	6.03e-08	0.59	2,3
5	ex3.3_order8	none	0; 0; 45, 15	152				0	6.16e-02, 6.16e-02	2.45e-06	1.46	
		pd1	0; 0; 12, 6	38	1	0.25	0.01	0	6.16e-02, 6.16e-02	5.23e-08	0.65	2,3
		pd2	0; 0; 12, 6	38	1	0.37	0.01	0	6.16e-02, 6.16e-02	5.23e-08	0.67	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	0; 0; 12, 6	38	1	0.07		0	6.16e-02, 6.16e-02	5.23e-08	0.58	2,3
6	ex3.3_order9	none	0; 0; 55, 21	189				0	6.16e-02, 6.16e-02	4.63e-06	0.94	
		pd1	0; 0; 13, 7	41	1	0.18	0.01	0	6.16e-02, 6.16e-02	5.54e-08	0.60	2,3
		pd2	0; 0; 13, 7	41	1	0.36	0.01	0	6.16e-02, 6.16e-02	5.54e-08	0.61	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP	0; 0; 13, 7	41	1	0.09		0	6.16e-02, 6.16e-02	5.54e-08	0.69	2,3
7	ex3.3_order10	none	0; 0; 66, 28	230				0	6.16e-02, 6.16e-02	4.33e-06	1.29	
		pd1	0; 0; 14, 8	44	1	0.33	0.03	0	6.16e-02, 6.16e-02	6.73e-08	1.06	2,3
		pd2	0; 0; 14, 8	44	1	0.56	0.01	0	6.16e-02, 6.16e-02	6.73e-08	1.00	2,3
		dd1			0	0.06	0.00					
		dd2			0	0.08	0.00					
		Sieve-SDP	0; 0; 14, 8	44	1	0.15		0	6.16e-02, 6.16e-02	6.73e-08	1.24	2,3

Table A.9: Detailed results on DIW dataset, part 2 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
8	ex3.3_order11	none	0; 0; 78, 36	275				0	6.16e-02, 6.16e-02	1.12e-05	1.38	
		pd1	0; 0; 16, 10	53	1	0.31	0.01	0	6.16e-02, 6.16e-02	9.05e-08	1.06	2,3
		pd2	0; 0; 16, 10	53	1	0.70	0.01	0	6.16e-02, 6.16e-02	9.05e-08	1.06	2,3
		dd1			0	0.04	0.00					
		dd2			0	0.10	0.00					
		Sieve-SDP	0; 0; 16, 10	53	1	0.18		0	6.16e-02, 6.16e-02	9.05e-08	1.22	2,3
9	ex3.3_order12	none	0; 0; 91, 45	324				0	6.16e-02, 6.16e-02	2.34e-05	1.50	
		pd1	0; 0; 18, 12	60	1	0.38	0.01	0	6.16e-02, 6.16e-02	1.12e-07	1.38	2,3
		pd2	0; 0; 18, 12	60	1	1.17	0.01	0	6.16e-02, 6.16e-02	1.12e-07	0.63	2,3
		dd1			0	0.04	0.00					
		dd2			0	0.11	0.00					
		Sieve-SDP	0; 0; 18, 12	60	1	0.22		0	6.16e-02, 6.16e-02	1.12e-07	0.68	2,3
10	ex3.3_order13	none	0; 0; 105, 55	377				0	6.15e-02, 6.15e-02	3.47e-05	1.38	
		pd1	0; 0; 19, 13	63	1	0.45	0.01	0	6.16e-02, 6.16e-02	1.21e-07	0.61	2,3
		pd2	0; 0; 19, 13	63	1	1.50	0.01	0	6.16e-02, 6.16e-02	1.21e-07	0.59	2,3
		dd1			0	0.03	0.01					
		dd2			0	0.15	0.01					
		Sieve-SDP	0; 0; 19, 13	63	1	0.28		0	6.16e-02, 6.16e-02	1.21e-07	0.57	2,3
11	ex3.3_order14	none	0; 0; 120, 66	434				0	6.16e-02, 6.16e-02	1.41e-05	1.50	
		pd1	0; 0; 20, 14	66	1	0.58	0.01	0	6.16e-02, 6.16e-02	1.31e-07	0.57	2,3
		pd2	0; 0; 20, 14	66	1	2.07	0.01	0	6.16e-02, 6.16e-02	1.31e-07	1.39	2,3
		dd1			0	0.05	0.00					
		dd2			0	0.22	0.00					
		Sieve-SDP	0; 0; 20, 14	66	1	0.38		0	6.16e-02, 6.16e-02	1.31e-07	1.27	2,3
12	ex3.3_order15	none	0; 0; 136, 78	495				0	6.16e-02, 6.16e-02	1.06e-05	2.72	
		pd1	0; 0; 22, 16	75	1	0.73	0.01	0	6.16e-02, 6.16e-02	1.47e-07	1.06	2,3
		pd2	0; 0; 22, 16	75	1	2.89	0.02	0	6.16e-02, 6.16e-02	1.47e-07	1.06	2,3
		dd1			0	0.05	0.00					
		dd2			0	0.24	0.00					
		Sieve-SDP	0; 0; 22, 16	75	1	0.47		0	6.16e-02, 6.16e-02	1.47e-07	1.10	2,3
13	ex3.3_order16	none	0; 0; 153, 91	560				0	6.16e-02, 6.16e-02	1.18e-05	3.64	
		pd1	0; 0; 24, 18	82	1	0.92	0.01	0	6.16e-02, 6.16e-02	1.79e-07	0.95	2,3
		pd2	0; 0; 24, 18	82	1	3.88	0.01	0	6.16e-02, 6.16e-02	1.79e-07	0.92	2,3
		dd1			0	0.05	0.00					
		dd2			0	0.29	0.00					
		Sieve-SDP	0; 0; 24, 18	82	1	0.58		0	6.16e-02, 6.16e-02	1.79e-07	0.93	2,3
14	ex3.3_order17	none	0; 0; 171, 105	629				0	6.15e-02, 6.15e-02	2.84e-05	4.62	
		pd1	0; 0; 25, 19	85	1	1.16	0.02	0	6.16e-02, 6.16e-02	1.89e-07	0.94	2,3
		pd2	0; 0; 25, 19	85	1	5.37	0.02	0	6.16e-02, 6.16e-02	1.89e-07	0.94	2,3
		dd1			0	0.06	0.01					
		dd2			0	0.41	0.01					
		Sieve-SDP	0; 0; 25, 19	85	1	0.73		0	6.16e-02, 6.16e-02	1.89e-07	0.92	2,3
15	ex3.3_order18	none	0; 0; 190, 120	702				0	6.15e-02, 6.15e-02	6.51e-05	6.29	
		pd1	0; 0; 26, 20	88	1	1.55	0.02	0	6.16e-02, 6.16e-02	1.98e-07	0.93	2,3
		pd2	0; 0; 26, 20	88	1	7.78	0.02	0	6.16e-02, 6.16e-02	1.98e-07	0.58	2,3
		dd1			0	0.05	0.01					
		dd2			0	0.50	0.01					
		Sieve-SDP	0; 0; 26, 20	88	1	0.94		0	6.16e-02, 6.16e-02	1.98e-07	0.61	2,3
16	ex3.3_order19	none	0; 0; 210, 136	779				0	6.15e-02, 6.15e-02	2.92e-04	10.01	
		pd1	0; 0; 28, 22	97	1	2.18	0.02	0	6.16e-02, 6.16e-02	2.10e-07	1.22	2,3
		pd2	0; 0; 28, 22	97	1	10.34	0.03	0	6.16e-02, 6.16e-02	2.10e-07	0.69	2,3
		dd1			0	0.06	0.01					
		dd2			0	0.60	0.01					
		Sieve-SDP	0; 0; 28, 22	97	1	1.31		0	6.16e-02, 6.16e-02	2.10e-07	0.67	2,3
17	ex3.3_order20	none	0; 0; 231, 153	860				0	6.15e-02, 6.15e-02	2.95e-04	21.36	
		pd1	0; 0; 30, 24	104	1	2.68	0.03	0	6.16e-02, 6.16e-02	2.16e-07	1.10	2,3
		pd2	0; 0; 30, 24	104	1	13.41	0.03	0	6.16e-02, 6.16e-02	2.16e-07	1.09	2,3
		dd1			0	0.09	0.02					
		dd2			0	0.86	0.02					
		Sieve-SDP	0; 0; 30, 24	104	1	1.35		0	6.16e-02, 6.16e-02	2.16e-07	1.31	2,3
18	ex4.1_order3	none	0; 1; 10	27				0	1.24e-09, 1.30e-09	8.29e-01	1.20	
		pd1	0; 1; 1	2	1	0.07	0.00	1	1.18e-12, 4.80e+00	8.28e-01	0.79	1
		pd2	0; 1; 1	2	1	0.06	0.00	1	1.18e-12, 4.80e+00	8.28e-01	0.62	1
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP			infeas	0.01					0.00	1
19	ex4.1_order4	none	0; 0; 15, 3	44				0	2.55e-09, 2.60e-09	8.29e-01	0.84	
		pd1	0; 0; 1	1	1	0.10	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.35	1
		pd2	0; 0; 1	1	1	0.08	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.66	1
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			infeas	0.02					0.00	1

Table A.10: Detailed results on DIW dataset, part 3 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
20	ex4.1_order5	none	0; 0; 21, 6	65				0	3.08e-09, 3.12e-09	8.29e-01	0.94	
		pd1	0; 0; 1	1	1	0.10	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.39	1
		pd2	0; 0; 1	1	1	0.11	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.34	1
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			infeas	0.02					0.00	1
21	ex4.1_order6	none	0; 0; 28, 10	90				0	1.00e+00, 1.00e+00	6.31e-07	0.85	
		pd1	0; 0; 1	1	1	0.14	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.39	1
		pd2	0; 0; 1	1	1	0.15	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.36	1
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			infeas	0.02					0.00	1
22	ex4.1_order7	none	0; 0; 36, 15	119				0	1.00e+00, 1.00e+00	1.00e-06	0.61	
		pd1	0; 0; 1	1	1	0.17	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.34	1
		pd2	0; 0; 1	1	1	0.21	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.61	1
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			infeas	0.04					0.00	1
23	ex4.1_order8	none	0; 0; 45, 21	152				0	1.00e+00, 1.00e+00	1.42e-06	0.68	
		pd1	0; 0; 1	1	1	0.19	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.39	1
		pd2	0; 0; 1	1	1	0.27	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.35	1
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.05					0.00	1
24	ex4.1_order9	none	0; 0; 55, 28	189				0	1.00e+00, 1.00e+00	1.10e-06	0.87	
		pd1	0; 0; 1	1	1	0.26	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.39	1
		pd2	0; 0; 1	1	1	0.45	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.39	1
		dd1			0	0.02	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP			infeas	0.07					0.00	1
25	ex4.1_order10	none	0; 0; 66, 36	230				0	1.00e+00, 1.00e+00	9.94e-07	0.69	
		pd1	0; 0; 1	1	1	0.32	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.39	1
		pd2	0; 0; 1	1	1	0.50	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.33	1
		dd1			0	0.02	0.00					
		dd2			0	0.06	0.00					
		Sieve-SDP			infeas	0.09					0.00	1
26	ex4.1_order11	none	0; 0; 78, 45	275				0	1.00e+00, 1.00e+00	2.60e-06	0.93	
		pd1	0; 0; 1	1	1	0.33	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.38	1
		pd2	0; 0; 1	1	1	0.71	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.34	1
		dd1			0	0.02	0.00					
		dd2			0	0.09	0.00					
		Sieve-SDP			infeas	0.13					0.00	1
27	ex4.1_order12	none	0; 0; 91, 55	324				0	1.00e+00, 1.00e+00	2.29e-06	1.06	
		pd1	0; 0; 1	1	1	0.50	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.34	1
		pd2	0; 0; 1	1	1	1.02	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.41	1
		dd1			0	0.02	0.00					
		dd2			0	0.11	0.00					
		Sieve-SDP			infeas	0.16					0.00	1
28	ex4.1_order13	none	0; 0; 105, 66	377				0	1.00e+00, 1.00e+00	6.83e-06	1.00	
		pd1	0; 0; 1	1	1	0.52	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.77	1
		pd2	0; 0; 1	1	1	1.55	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.37	1
		dd1			0	0.03	0.00					
		dd2			0	0.14	0.00					
		Sieve-SDP			infeas	0.19					0.00	1
29	ex4.1_order14	none	0; 0; 120, 78	434				0	1.00e+00, 1.00e+00	2.19e-06	1.59	
		pd1	0; 0; 1	1	1	0.81	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.43	1
		pd2	0; 0; 1	1	1	2.20	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.38	1
		dd1			0	0.03	0.00					
		dd2			0	0.20	0.00					
		Sieve-SDP			infeas	0.25					0.00	1
30	ex4.1_order15	none	0; 0; 136, 91	495				0	1.00e+00, 1.00e+00	4.32e-06	1.66	
		pd1	0; 0; 1	1	1	0.87	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.36	1
		pd2	0; 0; 1	1	1	2.79	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.35	1
		dd1			0	0.04	0.00					
		dd2			0	0.24	0.00					
		Sieve-SDP			infeas	0.29					0.00	1
31	ex4.1_order16	none	0; 0; 153, 105	560				0	1.00e+00, 1.00e+00	7.99e-07	2.41	
		pd1	0; 0; 1	1	1	1.10	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.36	1
		pd2	0; 0; 1	1	1	4.06	0.00	1	0.00e+00, 3.00e+00	7.50e-01	0.53	1
		dd1			0	0.04	0.00					
		dd2			0	0.31	0.00					
		Sieve-SDP			infeas	0.37					0.00	1

Table A.11: Detailed results on DIW dataset, part 4 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
32	ex4.1_order17	none	0; 0; 171, 120	629				0	1.00e+00, 1.00e+00	1.45e-06	3.23	
		pd1	0; 0; 1	1	1	1.27	0.01	1	0.00e+00, 3.00e+00	7.50e-01	0.34	1
		pd2	0; 0; 1	1	1	5.29	0.01	1	0.00e+00, 3.00e+00	7.50e-01	0.37	1
		dd1			0	0.05	0.01					
		dd2			0	0.44	0.01					
		Sieve-SDP			infeas	0.47					0.00	1
33	ex4.1_order18	none	0; 0; 190, 136	702				0	1.00e+00, 1.00e+00	2.43e-06	4.52	
		pd1	0; 0; 1	1	1	1.72	0.01	1	0.00e+00, 3.00e+00	7.50e-01	0.37	1
		pd2	0; 0; 1	1	1	6.99	0.01	1	0.00e+00, 3.00e+00	7.50e-01	0.46	1
		dd1			0	0.08	0.01					
		dd2			0	0.52	0.01					
		Sieve-SDP			infeas	0.67					0.00	1
34	ex4.1_order19	none	0; 0; 210, 153	779				0	1.00e+00, 1.00e+00	2.76e-06	6.04	
		pd1	0; 0; 1	1	1	2.01	0.01	1	0.00e+00, 3.00e+00	7.50e-01	0.37	1
		pd2	0; 0; 1	1	1	9.39	0.01	1	0.00e+00, 3.00e+00	7.50e-01	0.37	1
		dd1			0	0.06	0.01					
		dd2			0	0.69	0.01					
		Sieve-SDP			infeas	0.87					0.00	1
35	ex4.1_order20	none	0; 0; 231, 171	860				0	1.00e+00, 1.00e+00	5.65e-06	8.86	
		pd1	0; 0; 1	1	1	2.68	0.02	1	0.00e+00, 3.00e+00	7.50e-01	0.35	1
		pd2	0; 0; 1	1	1	12.71	0.02	1	0.00e+00, 3.00e+00	7.50e-01	0.34	1
		dd1			0	0.07	0.02					
		dd2			0	0.87	0.02					
		Sieve-SDP			infeas	0.88					0.00	1
36	ex4.2_order4	none	0; 1; 15	44				0	1.00e-09, 1.01e-09	7.07e-01	0.81	
		pd1	0; 1; 1	2	1	0.08	0.00	0	1.00e+00, 1.00e+00	5.00e-01	0.57	3
		pd2	0; 1; 1	2	1	0.09	0.00	0	1.00e+00, 1.00e+00	5.00e-01	0.61	3
		dd1			0	0.04	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP			infeas	0.03					0.00	1
37	ex4.2_order5	none	0; 0; 21, 3	65				0	5.53e-01, 5.53e-01	9.19e-08	1.85	
		pd1	0; 0; 1	1	1	0.15	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.65	1
		pd2	0; 0; 1	1	1	0.14	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.44	1
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.03					0.00	1
38	ex4.2_order6	none	0; 0; 28, 6	90				0	5.53e-01, 5.53e-01	3.24e-07	0.65	
		pd1	0; 0; 8, 2	22	1	0.09	0.00	0	5.53e-01, 5.53e-01	7.64e-09	0.63	
		pd2	0; 0; 8, 2	22	1	0.14	0.00	0	5.53e-01, 5.53e-01	7.64e-09	0.62	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 8, 2	22	1	0.03		0	5.53e-01, 5.53e-01	7.64e-09	0.74	
39	ex4.2_order7	none	0; 0; 36, 10	119				0	5.53e-01, 5.53e-01	2.87e-07	0.65	
		pd1	0; 0; 9, 3	25	1	0.13	0.00	0	5.53e-01, 5.53e-01	4.89e-09	0.57	
		pd2	0; 0; 9, 3	25	1	0.16	0.00	0	5.53e-01, 5.53e-01	4.89e-09	0.55	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 9, 3	25	1	0.04		0	5.53e-01, 5.53e-01	4.89e-09	0.68	
40	ex4.2_order8	none	0; 0; 45, 15	152				0	5.53e-01, 5.53e-01	1.08e-06	0.64	
		pd1	0; 0; 10, 4	28	1	0.17	0.00	0	5.53e-01, 5.53e-01	3.90e-08	0.58	2
		pd2	0; 0; 10, 4	28	1	0.23	0.00	0	5.53e-01, 5.53e-01	3.90e-08	0.57	2
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	0; 0; 10, 4	28	1	0.06		0	5.53e-01, 5.53e-01	3.90e-08	0.70	2
41	ex4.2_order9	none	0; 0; 55, 21	189				0	5.53e-01, 5.53e-01	1.10e-06	0.67	
		pd1	0; 0; 10, 4	28	1	0.22	0.00	0	5.53e-01, 5.53e-01	3.90e-08	0.58	2
		pd2	0; 0; 10, 4	28	1	0.33	0.00	0	5.53e-01, 5.53e-01	3.90e-08	0.57	2
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	0; 0; 10, 4	28	1	0.08		0	5.53e-01, 5.53e-01	3.90e-08	0.71	2
42	ex4.2_order10	none	0; 0; 66, 28	230				0	5.53e-01, 5.53e-01	1.42e-06	0.74	
		pd1	0; 0; 10, 4	28	1	0.39	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.71	2,3
		pd2	0; 0; 10, 4	28	1	0.56	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.60	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.06	0.00					
		Sieve-SDP	0; 0; 10, 4	28	1	0.13		0	5.53e-01, 5.53e-01	3.90e-08	0.76	2,3
43	ex4.2_order11	none	0; 0; 78, 36	275				0	5.53e-01, 5.53e-01	1.61e-06	1.66	
		pd1	0; 0; 10, 4	28	1	0.49	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.80	2,3
		pd2	0; 0; 10, 4	28	1	0.82	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.73	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.10	0.00					
		Sieve-SDP	0; 0; 10, 4	28	1	0.16		0	5.53e-01, 5.53e-01	3.90e-08	0.70	2,3

Table A.12: Detailed results on DIW dataset, part 5 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
44	ex4.2_order12	none	0; 0; 91, 45	324				0	5.53e-01, 5.53e-01	2.77e-06	1.23	
		pd1	0; 0; 10, 4	28	1	0.51	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.68	2,3
		pd2	0; 0; 10, 4	28	1	1.06	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.57	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.12	0.00					
		Sieve-SDP	0; 0; 10, 4	28	1	0.19		0	5.53e-01, 5.53e-01	3.90e-08	0.70	2,3
45	ex4.2_order13	none	0; 0; 105, 55	377				0	5.53e-01, 5.53e-01	4.58e-06	1.53	
		pd1	0; 0; 10, 4	28	1	0.51	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.68	2,3
		pd2	0; 0; 10, 4	28	1	1.54	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.62	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.15	0.00					
		Sieve-SDP	0; 0; 10, 4	28	1	0.25		0	5.53e-01, 5.53e-01	3.90e-08	0.73	2,3
46	ex4.2_order14	none	0; 0; 120, 66	434				0	5.53e-01, 5.53e-01	1.61e-06	2.24	
		pd1	0; 0; 10, 4	28	1	0.82	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.73	2,3
		pd2	0; 0; 10, 4	28	1	2.36	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.76	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.26	0.00					
		Sieve-SDP	0; 0; 10, 4	28	1	0.32		0	5.53e-01, 5.53e-01	3.90e-08	0.61	2,3
47	ex4.2_order15	none	0; 0; 136, 78	495				0	5.53e-01, 5.53e-01	3.52e-06	2.90	
		pd1	0; 0; 10, 4	28	1	1.08	0.01	0	5.53e-01, 5.53e-01	3.90e-08	1.50	2,3
		pd2	0; 0; 10, 4	28	1	3.81	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.86	2,3
		dd1			0	0.04	0.01					
		dd2			0	0.34	0.01					
		Sieve-SDP	0; 0; 10, 4	28	1	0.59		0	5.53e-01, 5.53e-01	3.90e-08	0.57	2,3
48	ex4.2_order16	none	0; 0; 153, 91	560				0	5.53e-01, 5.53e-01	2.54e-06	4.55	
		pd1	0; 0; 10, 4	28	1	1.50	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.67	2,3
		pd2	0; 0; 10, 4	28	1	4.37	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.60	2,3
		dd1			0	0.04	0.01					
		dd2			0	0.37	0.01					
		Sieve-SDP	0; 0; 10, 4	28	1	0.50		0	5.53e-01, 5.53e-01	3.90e-08	0.62	2,3
49	ex4.2_order17	none	0; 0; 171, 105	629				0	5.52e-01, 5.52e-01	1.06e-04	4.96	
		pd1	0; 0; 10, 4	28	1	1.52	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.57	2,3
		pd2	0; 0; 10, 4	28	1	5.71	0.01	0	5.53e-01, 5.53e-01	3.90e-08	0.63	2,3
		dd1			0	0.05	0.01					
		dd2			0	0.50	0.01					
		Sieve-SDP	0; 0; 10, 4	28	1	0.59		0	5.53e-01, 5.53e-01	3.90e-08	0.62	2,3
50	ex4.2_order18	none	0; 0; 190, 120	702				0	5.52e-01, 5.52e-01	1.70e-04	9.65	
		pd1	0; 0; 10, 4	28	1	2.01	0.02	0	5.53e-01, 5.53e-01	3.90e-08	0.62	2,3
		pd2	0; 0; 10, 4	28	1	8.06	0.02	0	5.53e-01, 5.53e-01	3.90e-08	0.65	2,3
		dd1			0	0.06	0.01					
		dd2			0	0.65	0.01					
		Sieve-SDP	0; 0; 10, 4	28	1	0.78		0	5.53e-01, 5.53e-01	3.90e-08	0.62	2,3
51	ex4.2_order19	none	0; 0; 210, 136	779				0	5.52e-01, 5.52e-01	9.26e-04	11.00	
		pd1	0; 0; 10, 4	28	1	2.40	0.02	0	5.53e-01, 5.53e-01	3.90e-08	0.59	2,3
		pd2	0; 0; 10, 4	28	1	9.99	0.02	0	5.53e-01, 5.53e-01	3.90e-08	0.60	2,3
		dd1			0	0.06	0.02					
		dd2			0	0.80	0.02					
		Sieve-SDP	0; 0; 10, 4	28	1	0.93		0	5.53e-01, 5.53e-01	3.90e-08	0.75	2,3
52	ex4.2_order20	none	0; 0; 231, 153	860				0	5.49e-01, 5.49e-01	4.36e-03	17.65	
		pd1	0; 0; 10, 4	28	1	2.93	0.03	0	5.53e-01, 5.53e-01	3.90e-08	0.58	2,3
		pd2	0; 0; 10, 4	28	1	12.77	0.03	0	5.53e-01, 5.53e-01	3.90e-08	0.57	2,3
		dd1			0	0.07	0.02					
		dd2			0	0.94	0.02					
		Sieve-SDP	0; 0; 10, 4	28	1	1.11		0	5.53e-01, 5.53e-01	3.90e-08	0.68	2,3
53	ex4.3_order2	none	0; 1; 10	34				0	1.73e-07, 1.83e-07	9.09e-01	4.39	
		pd1	0; 1; 4	10	1	0.14	0.03	1	0.00e+00, 8.00e+00	9.09e-01	1.24	1
		pd2	0; 1; 4	10	1	0.34	0.00	1	0.00e+00, 8.00e+00	9.09e-01	1.30	1
		dd1			0	0.08	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.52					0.00	1
54	ex4.3_order3	none	0; 0; 20, 4	83				0	6.22e-09, 6.48e-09	9.09e-01	3.78	
		pd1	0; 0; 1	1	1	0.19	0.01	1	0.00e+00, 8.00e+00	8.89e-01	1.29	1
		pd2	0; 0; 1	1	1	0.13	0.00	1	0.00e+00, 8.00e+00	8.89e-01	1.28	1
		dd1			0	0.05	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP			infeas	0.05					0.00	1
55	ex4.3_order4	none	0; 0; 35, 10	164				0	2.50e-09, 2.53e-09	9.09e-01	3.88	
		pd1	0; 0; 1	1	1	0.11	0.00	1	0.00e+00, 8.00e+00	8.89e-01	1.31	1
		pd2	0; 0; 1	1	1	0.13	0.00	1	0.00e+00, 8.00e+00	8.89e-01	1.29	1
		dd1			0	0.03	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP			infeas	0.04					0.00	1

Table A.13: Detailed results on DIW dataset, part 6 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
56	ex4.3_order5	none	0; 0; 56, 20	285				0	5.18e-09, 5.22e-09	9.09e-01	3.30	
		pd1	0; 0; 1	1	1	0.15	0.00	1	0.00e+00, 8.00e+00	8.89e-01	1.24	1
		pd2	0; 0; 1	1	1	0.20	0.00	1	0.00e+00, 8.00e+00	8.89e-01	1.22	1
		dd1			0	0.04	0.00					
		dd2			0	0.08	0.00					
		Sieve-SDP			infeas	0.06					0.00	1
57	ex4.3_order6	none	0; 0; 84, 35	454				0	1.60e+01, 1.60e+01	3.34e-06	3.20	
		pd1	0; 0; 1	1	1	0.21	0.00	1	0.00e+00, 8.00e+00	8.89e-01	1.16	1
		pd2	0; 0; 1	1	1	0.41	0.00	1	0.00e+00, 8.00e+00	8.89e-01	1.20	1
		dd1			0	0.05	0.00					
		dd2			0	0.13	0.00					
		Sieve-SDP			infeas	0.10					0.00	1
58	ex4.3_order7	none	0; 0; 120, 56	679				0	1.60e+01, 1.60e+01	5.26e-06	4.45	
		pd1	0; 0; 1	1	1	0.30	0.01	1	0.00e+00, 8.00e+00	8.89e-01	1.23	1
		pd2	0; 0; 1	1	1	0.83	0.01	1	0.00e+00, 8.00e+00	8.89e-01	1.19	1
		dd1			0	0.06	0.01					
		dd2			0	0.23	0.01					
		Sieve-SDP			infeas	0.17					0.00	1
59	ex4.3_order8	none	0; 0; 165, 84	968				0	1.60e+01, 1.60e+01	5.17e-06	9.40	
		pd1	0; 0; 1	1	1	0.48	0.01	1	0.00e+00, 8.00e+00	8.89e-01	0.65	1
		pd2	0; 0; 1	1	1	2.05	0.01	1	0.00e+00, 8.00e+00	8.89e-01	0.71	1
		dd1			0	0.05	0.01					
		dd2			0	0.45	0.01					
		Sieve-SDP			infeas	0.27					0.00	1
60	ex4.3_order9	none	0; 0; 220, 120	1329				0	1.60e+01, 1.60e+01	5.88e-06	17.62	
		pd1	0; 0; 1	1	1	0.85	0.02	1	0.00e+00, 8.00e+00	8.89e-01	0.56	1
		pd2	0; 0; 1	1	1	4.46	0.02	1	0.00e+00, 8.00e+00	8.89e-01	0.59	1
		dd1			0	0.07	0.02					
		dd2			0	0.88	0.02					
		Sieve-SDP			infeas	0.53					0.00	1
61	ex4.3_order10	none	0; 0; 286, 165	1770				0	1.60e+01, 1.60e+01	3.92e-05	42.57	
		pd1	0; 0; 1	1	1	1.66	0.04	1	0.00e+00, 8.00e+00	8.89e-01	0.35	1
		pd2	0; 0; 1	1	1	9.01	0.04	1	0.00e+00, 8.00e+00	8.89e-01	0.34	1
		dd1			0	0.09	0.04					
		dd2			0	1.79	0.04					
		Sieve-SDP			infeas	0.87					0.00	1
62	ex4.3_order11	none	0; 0; 364, 220	2299				0	7.85e-06, 7.85e-06	9.09e-01	116.27	
		pd1	0; 0; 1	1	1	3.12	0.07	1	0.00e+00, 8.00e+00	8.89e-01	0.34	1
		pd2	0; 0; 1	1	1	17.61	0.07	1	0.00e+00, 8.00e+00	8.89e-01	0.34	1
		dd1			0	0.16	0.07					
		dd2			0	18.38	0.07					
		Sieve-SDP			infeas	1.51					0.00	1
63	ex4.3_order12	none	0; 0; 455, 286	2924				0	2.62e-06, 2.62e-06	9.09e-01	330.94	
		pd1	0; 0; 1	1	1	5.71	0.11	1	0.00e+00, 8.00e+00	8.89e-01	0.52	1
		pd2	0; 0; 1	1	1	34.17	0.11	1	0.00e+00, 8.00e+00	8.89e-01	0.36	1
		dd1			0	0.25	0.11					
		dd2			0	39.81	0.11					
		Sieve-SDP			infeas	2.90					0.00	1
64	ex4.3_order13	none	0; 0; 560, 364	3653				0	4.85e-07, 4.85e-07	9.09e-01	814.60	
		pd1	0; 0; 1	1	1	10.54	0.18	1	0.00e+00, 8.00e+00	8.89e-01	0.34	1
		pd2	0; 0; 1	1	1	61.06	0.18	1	0.00e+00, 8.00e+00	8.89e-01	0.35	1
		dd1			0	0.40	0.18					
		dd2			0	74.87	0.18					
		Sieve-SDP			infeas	5.29					0.00	1
65	ex4.3_order14	none	0; 0; 680, 455	4494				0	1.01e+01, 1.01e+01	9.40e-02	1178.45	
		pd1	0; 0; 1	1	1	17.38	0.27	1	0.00e+00, 8.00e+00	8.89e-01	0.34	1
		pd2	0; 0; 1	1	1	109.31	0.27	1	0.00e+00, 8.00e+00	8.89e-01	0.47	1
		dd1			0	0.63	0.27					
		dd2			0	146.45	0.27					
		Sieve-SDP			infeas	9.73					0.00	1
66	ex4.3_order15	none	0; 0; 816, 560	5455				0	8.94e+00, 8.94e+00	1.76e-01	2010.36	
		pd1	0; 0; 1	1	1	33.32	0.41	1	0.00e+00, 8.00e+00	8.89e-01	0.50	1
		pd2	0; 0; 1	1	1	192.22	0.41	1	0.00e+00, 8.00e+00	8.89e-01	0.70	1
		dd1			0	0.99	0.41					
		dd2			0	303.20	0.41					
		Sieve-SDP			infeas	16.20					0.00	1
67	ex4.3_order16	none	0; 0; 969, 680	6544				0	7.95e+00, 7.95e+00	2.23e-01	3158.88	
		pd1	0; 0; 1	1	1	46.39	0.55	1	0.00e+00, 8.00e+00	8.89e-01	1.43	1
		pd2	0; 0; 1	1	1	295.70	0.55	1	0.00e+00, 8.00e+00	8.89e-01	1.33	1
		dd1			0	1.49	0.55					
		dd2			0	485.30	0.55					
		Sieve-SDP			infeas	29.21					0.00	1

Table A.14: Detailed results on DIW dataset, part 7 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
68	ex4.3_order17	none	0; 0; 1140, 816	7769				0	7.45e+00, 7.45e+00	2.00e-01	5618.65	
		pd1	0; 0; 1	1	1	71.92	0.81	1	0.00e+00, 8.00e+00	8.89e-01	1.34	1
		pd2	0; 0; 1	1	1	472.15	0.81	1	0.00e+00, 8.00e+00	8.89e-01	1.33	1
		dd1			0	2.13	0.81					
		dd2			0	949.49	0.81					
		Sieve-SDP			infeas	49.63					0.00	1
69	ex4.3_order18	none	0; 0; 1330, 969	9138				0	7.16e+00, 7.16e+00	2.14e-01	11769.31	
		pd1	0; 0; 1	1	1	112.46	1.13	1	0.00e+00, 8.00e+00	8.89e-01	1.40	1
		pd2	0; 0; 1	1	1	753.21	1.13	1	0.00e+00, 8.00e+00	8.89e-01	1.47	1
		dd1			0	3.05	1.13					
		dd2			0	1624.34	1.13					
		Sieve-SDP			infeas	81.60					0.00	1
70	ex4.3_order19	none	0; 0; 1540, 1140	10659				0	6.82e+00, 6.82e+00	2.63e-01	22830.51	
		pd1	0; 0; 1	1	1	171.19	1.62	1	0.00e+00, 8.00e+00	8.89e-01	1.26	1
		pd2	0; 0; 1	1	1	1177.71	1.60	1	0.00e+00, 8.00e+00	8.89e-01	1.23	1
		dd1			0	4.53	1.60					
		dd2			0	2852.27	1.60					
		Sieve-SDP			infeas	134.13					0.00	1
71	ex4.3_order20	none	0; 0; 1771, 1330	12340				0	6.52e+00, 6.52e+00	3.66e-01	38786.88	
		pd1	0; 0; 1	1	1	375.53	2.81	1	0.00e+00, 8.00e+00	8.89e-01	1.09	1
		pd2	0; 0; 1	1	1	2479.13	2.79	1	0.00e+00, 8.00e+00	8.89e-01	0.64	1
		dd1			0	6.68	2.79					
		dd2			0	6408.87	2.79					
		Sieve-SDP			infeas	260.68					0.00	1
72	ex4.4_order3	none	0; 0; 20, 10	83				1	9.88e-02, 1.18e-01	8.66e-01	1.72	
		pd1	0; 0; 1	1	1	0.09	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.60	
		pd2	0; 0; 1	1	1	0.10	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.53	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP			infeas	0.01					0.00	1
73	ex4.4_order4	none	0; 0; 35, 20	164				1	1.73e-05, 1.84e-05	8.66e-01	2.17	
		pd1	0; 0; 1	1	1	0.12	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.54	
		pd2	0; 0; 1	1	1	0.14	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.60	
		dd1			0	0.02	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP			infeas	0.03					0.00	1
74	ex4.4_order5	none	0; 0; 56, 35	285				0	5.66e-08, 5.85e-08	8.66e-01	2.11	
		pd1	0; 0; 1	1	1	0.14	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.60	1
		pd2	0; 0; 1	1	1	0.25	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.59	1
		dd1			0	0.03	0.00					
		dd2			0	0.14	0.00					
		Sieve-SDP			infeas	0.04					0.00	1
75	ex4.4_order6	none	0; 0; 84, 56	454				0	1.49e-08, 1.50e-08	8.66e-01	3.11	
		pd1	0; 0; 1	1	1	0.20	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.53	1
		pd2	0; 0; 1	1	1	0.47	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.55	1
		dd1			0	0.03	0.00					
		dd2			0	0.43	0.00					
		Sieve-SDP			infeas	0.08					0.00	1
76	ex4.4_order7	none	0; 0; 120, 84	679				0	6.64e-09, 6.68e-09	8.66e-01	4.44	
		pd1	0; 0; 1	1	1	0.30	0.01	1	0.00e+00, 1.00e+00	5.00e-01	0.58	1
		pd2	0; 0; 1	1	1	0.96	0.01	1	0.00e+00, 1.00e+00	5.00e-01	0.57	1
		dd1			0	0.04	0.01					
		dd2			0	1.12	0.01					
		Sieve-SDP			infeas	0.17					0.00	1
77	ex4.4_order8	none	0; 0; 165, 120	968				0	1.80e-09, 1.80e-09	8.66e-01	12.00	
		pd1	0; 0; 1	1	1	0.61	0.01	1	0.00e+00, 1.00e+00	5.00e-01	0.61	1
		pd2	0; 0; 1	1	1	2.09	0.01	1	0.00e+00, 1.00e+00	5.00e-01	0.55	1
		dd1			0	0.05	0.01					
		dd2			0	3.86	0.01					
		Sieve-SDP			infeas	0.32					0.00	1
78	ex4.4_order9	none	0; 0; 220, 165	1329				0	4.13e-10, 4.14e-10	8.66e-01	31.08	
		pd1	0; 0; 1	1	1	1.18	0.03	1	0.00e+00, 1.00e+00	5.00e-01	0.80	1
		pd2	0; 0; 1	1	1	5.85	0.03	1	0.00e+00, 1.00e+00	5.00e-01	0.76	1
		dd1			0	0.12	0.03					
		dd2			0	8.18	0.03					
		Sieve-SDP			infeas	0.53					0.00	1
79	ex4.4_order10	none	0; 0; 286, 220	1770				0	2.65e-10, 2.65e-10	8.66e-01	71.26	
		pd1	0; 0; 1	1	1	2.24	0.07	1	0.00e+00, 1.00e+00	5.00e-01	0.69	1
		pd2	0; 0; 1	1	1	10.10	0.06	1	0.00e+00, 1.00e+00	5.00e-01	0.56	1
		dd1			0	0.12	0.06					
		dd2			0	13.87	0.06					
		Sieve-SDP			infeas	0.89					0.00	1

Table A.15: Detailed results on DIW dataset, part 8 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
80	ex4.4_order11	none	0; 0; 364, 286	2299				0	1.60e-05, 1.61e-05	4.44e-05	64.15	
		pd1	0; 0; 1	1	1	4.21	0.10	1	0.00e+00, 1.00e+00	5.00e-01	0.62	1
		pd2	0; 0; 1	1	1	22.13	0.08	1	0.00e+00, 1.00e+00	5.00e-01	0.63	1
		dd1			0	0.20	0.08					
		dd2			0	30.60	0.08					
		Sieve-SDP			infeas	1.77					0.00	1
81	ex4.4_order12	none	0; 0; 455, 364	2924				0	1.93e-07, 1.78e-07	2.52e-06	115.81	
		pd1	0; 0; 1	1	1	5.82	0.13	1	0.00e+00, 1.00e+00	5.00e-01	1.05	1
		pd2	0; 0; 1	1	1	32.88	0.13	1	0.00e+00, 1.00e+00	5.00e-01	1.07	1
		dd1			0	0.31	0.12					
		dd2			0	53.13	0.12					
		Sieve-SDP			infeas	2.99					0.00	1
82	ex4.4_order13	none	0; 0; 560, 455	3653				0	1.45e-08, 2.31e-09	1.10e-06	238.42	
		pd1	0; 0; 1	1	1	9.81	0.19	1	0.00e+00, 1.00e+00	5.00e-01	1.18	1
		pd2	0; 0; 1	1	1	61.06	0.19	1	0.00e+00, 1.00e+00	5.00e-01	1.02	1
		dd1			0	0.49	0.19					
		dd2			0	100.91	0.19					
		Sieve-SDP			infeas	5.35					0.00	1
83	ex4.4_order14	none	0; 0; 680, 560	4494				0	-1.06e-08, -3.67e-08	1.87e-06	455.22	
		pd1	0; 0; 1	1	1	16.86	0.30	1	0.00e+00, 1.00e+00	5.00e-01	0.46	1
		pd2	0; 0; 1	1	1	107.61	0.30	1	0.00e+00, 1.00e+00	5.00e-01	0.45	1
		dd1			0	0.68	0.30					
		dd2			0	169.13	0.30					
		Sieve-SDP			infeas	9.68					0.00	1
84	ex4.4_order15	none	0; 0; 816, 680	5455				0	-1.65e-08, -4.13e-08	1.79e-06	923.64	
		pd1	0; 0; 1	1	1	28.95	0.45	1	0.00e+00, 1.00e+00	5.00e-01	0.50	1
		pd2	0; 0; 1	1	1	182.62	0.45	1	0.00e+00, 1.00e+00	5.00e-01	0.44	1
		dd1			0	1.13	0.45					
		dd2			0	284.93	0.45					
		Sieve-SDP			infeas	17.79					0.00	1
85	ex4.4_order16	none	0; 0; 969, 816	6544				0	-2.21e-08, -4.65e-08	1.75e-06	1906.07	
		pd1	0; 0; 1	1	1	45.76	0.63	1	0.00e+00, 1.00e+00	5.00e-01	0.53	1
		pd2	0; 0; 1	1	1	301.76	0.63	1	0.00e+00, 1.00e+00	5.00e-01	0.46	1
		dd1			0	1.70	0.63					
		dd2			0	476.46	0.63					
		Sieve-SDP			infeas	30.87					0.00	1
86	ex4.4_order17	none	0; 0; 1140, 969	7769				0	-7.69e-09, -1.50e-08	5.48e-07	3654.29	
		pd1	0; 0; 1	1	1	69.43	0.88	1	0.00e+00, 1.00e+00	5.00e-01	1.18	1
		pd2	0; 0; 1	1	1	474.62	0.89	1	0.00e+00, 1.00e+00	5.00e-01	1.22	1
		dd1			0	2.52	0.88					
		dd2			0	823.92	0.88					
		Sieve-SDP			infeas	51.79					0.00	1
87	ex4.4_order18	none	0; 0; 1330, 1140	9138				0	-9.27e-09, -1.60e-08	4.84e-07	7063.56	
		pd1	0; 0; 1	1	1	103.53	1.24	1	0.00e+00, 1.00e+00	5.00e-01	1.55	1
		pd2	0; 0; 1	1	1	723.31	1.23	1	0.00e+00, 1.00e+00	5.00e-01	1.69	1
		dd1			0	3.71	1.23					
		dd2			0	1328.75	1.23					
		Sieve-SDP			infeas	86.64					0.00	1
88	ex4.4_order19	none	0; 0; 1540, 1330	10659				0	-2.12e-08, -3.68e-08	1.22e-06	12500.09	
		pd1	0; 0; 1	1	1	150.44	1.70	1	0.00e+00, 1.00e+00	5.00e-01	1.47	1
		pd2	0; 0; 1	1	1	1128.87	1.70	1	0.00e+00, 1.00e+00	5.00e-01	1.64	1
		dd1			0	5.13	1.69					
		dd2			0	2112.63	1.69					
		Sieve-SDP			infeas	140.48					0.00	1
89	ex4.4_order20	none	0; 0; 1771, 1540	12340				0	-3.07e-08, -5.42e-08	2.04e-06	25422.27	
		pd1	0; 0; 1	1	1	249.05	2.70	1	0.00e+00, 1.00e+00	5.00e-01	0.61	1
		pd2	0; 0; 1	1	1	1889.89	2.69	1	0.00e+00, 1.00e+00	5.00e-01	0.63	1
		dd1			0	7.15	2.69					
		dd2			0	3796.27	2.69					
		Sieve-SDP			infeas	256.35					0.00	1
90	ex5.4_order5	none	0; 0; 21	65				0	2.28e+00, 2.28e+00	5.89e-07	0.90	
		pd1	0; 0; 10	31	1	0.07	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.86	
		pd2	0; 0; 10	31	1	0.09	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.78	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.02		0	2.28e+00, 2.28e+00	6.97e-08	0.77	
91	ex5.4_order6	none	0; 0; 28	90				0	2.28e+00, 2.28e+00	1.89e-06	0.95	
		pd1	0; 0; 10	31	1	0.09	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.78	2,3
		pd2	0; 0; 10	31	1	0.11	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.77	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.03		0	2.28e+00, 2.28e+00	6.97e-08	0.84	2,3

Table A.16: Detailed results on DIW dataset, part 9 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
92	ex5.4_order7	none	0; 0; 36	119				0	2.28e+00, 2.28e+00	2.55e-06	0.98	
		pd1	0; 0; 10	31	1	0.10	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.78	2,3
		pd2	0; 0; 10	31	1	0.13	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.81	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.04		0	2.28e+00, 2.28e+00	6.97e-08	0.82	2,3
93	ex5.4_order8	none	0; 0; 45	152				0	2.28e+00, 2.28e+00	2.86e-06	0.96	
		pd1	0; 0; 10	31	1	0.11	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.77	2,3
		pd2	0; 0; 10	31	1	0.17	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.78	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.04		0	2.28e+00, 2.28e+00	6.97e-08	0.85	2,3
94	ex5.4_order9	none	0; 0; 55	189				0	2.28e+00, 2.28e+00	4.47e-06	0.95	
		pd1	0; 0; 10	31	1	0.13	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.78	2,3
		pd2	0; 0; 10	31	1	0.23	0.00	0	2.28e+00, 2.28e+00	6.97e-08	0.85	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.05		0	2.28e+00, 2.28e+00	6.97e-08	0.77	2,3
95	ex5.4_order10	none	0; 0; 66	230				0	2.28e+00, 2.28e+00	2.08e-06	1.98	
		pd1	0; 0; 10	31	1	0.21	0.01	0	2.28e+00, 2.28e+00	6.97e-08	1.59	2,3
		pd2	0; 0; 10	31	1	0.33	0.00	0	2.28e+00, 2.28e+00	6.97e-08	1.62	2,3
		dd1			0	0.05	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.10		0	2.28e+00, 2.28e+00	6.97e-08	1.62	2,3
96	ex5.4_order11	none	0; 0; 78	275				0	2.28e+00, 2.28e+00	9.31e-06	3.15	
		pd1	0; 0; 10	31	1	0.23	0.00	0	2.28e+00, 2.28e+00	6.97e-08	3.16	2,3
		pd2	0; 0; 10	31	1	0.43	0.00	0	2.28e+00, 2.28e+00	6.97e-08	2.25	2,3
		dd1			0	0.04	0.00					
		dd2			0	0.09	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.12		0	2.28e+00, 2.28e+00	6.97e-08	1.91	2,3
97	ex5.4_order12	none	0; 0; 91	324				0	2.28e+00, 2.28e+00	1.09e-05	2.52	
		pd1	0; 0; 10	31	1	0.29	0.01	0	2.28e+00, 2.28e+00	6.97e-08	1.72	2,3
		pd2	0; 0; 10	31	1	0.54	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.82	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.16	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.20		0	2.28e+00, 2.28e+00	6.97e-08	1.20	2,3
98	ex5.4_order13	none	0; 0; 105	377				0	2.28e+00, 2.28e+00	4.58e-06	2.22	
		pd1	0; 0; 10	31	1	0.42	0.00	0	2.28e+00, 2.28e+00	6.97e-08	1.13	2,3
		pd2	0; 0; 10	31	1	0.76	0.00	0	2.28e+00, 2.28e+00	6.97e-08	1.22	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.10	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.13		0	2.28e+00, 2.28e+00	6.97e-08	1.23	2,3
99	ex5.4_order14	none	0; 0; 120	434				0	2.28e+00, 2.28e+00	1.14e-05	1.78	
		pd1	0; 0; 10	31	1	0.31	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.87	2,3
		pd2	0; 0; 10	31	1	0.98	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.84	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.12	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.15		0	2.28e+00, 2.28e+00	6.97e-08	0.77	2,3
100	ex5.4_order15	none	0; 0; 136	495				0	2.28e+00, 2.28e+00	6.12e-06	2.11	
		pd1	0; 0; 10	31	1	0.43	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.79	2,3
		pd2	0; 0; 10	31	1	1.34	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.75	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.16	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.18		0	2.28e+00, 2.28e+00	6.97e-08	0.74	2,3
101	ex5.4_order16	none	0; 0; 153	560				0	2.28e+00, 2.28e+00	6.71e-06	3.38	
		pd1	0; 0; 10	31	1	0.53	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.83	2,3
		pd2	0; 0; 10	31	1	1.84	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.85	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.20	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.23		0	2.28e+00, 2.28e+00	6.97e-08	0.80	2,3
102	ex5.4_order17	none	0; 0; 171	629				0	2.28e+00, 2.28e+00	4.57e-06	4.32	
		pd1	0; 0; 10	31	1	0.62	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.77	2,3
		pd2	0; 0; 10	31	1	2.50	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.95	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.26	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.27		0	2.28e+00, 2.28e+00	6.97e-08	1.28	2,3
103	ex5.4_order18	none	0; 0; 190	702				0	2.27e+00, 2.27e+00	3.25e-03	7.00	
		pd1	0; 0; 10	31	1	0.93	0.01	0	2.28e+00, 2.28e+00	6.97e-08	1.14	2,3
		pd2	0; 0; 10	31	1	4.02	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.81	2,3
		dd1			0	0.08	0.01					
		dd2			0	0.33	0.01					
		Sieve-SDP	0; 0; 10	31	1	0.33		0	2.28e+00, 2.28e+00	6.97e-08	0.79	2,3

Table A.17: Detailed results on DIW dataset, part 10 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
104	ex5.4_order19	none	0; 0; 210	779				0	2.23e+00, 2.23e+00	2.09e-02	8.09	
		pd1	0; 0; 10	31	1	0.98	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.77	2,3
		pd2	0; 0; 10	31	1	4.38	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.77	2,3
		dd1			0	0.04	0.00					
		dd2			0	0.61	0.00					
		Sieve-SDP	0; 0; 10	31	1	0.67		0	2.28e+00, 2.28e+00	6.97e-08	1.66	2,3
105	ex5.4_order20	none	0; 0; 231	860				0	2.14e+00, 2.15e+00	5.08e-02	10.64	
		pd1	0; 0; 10	31	1	1.19	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.75	2,3
		pd2	0; 0; 10	31	1	6.78	0.01	0	2.28e+00, 2.28e+00	6.97e-08	0.99	2,3
		dd1			0	0.08	0.01					
		dd2			0	0.65	0.01					
		Sieve-SDP	0; 0; 10	31	1	0.48		0	2.28e+00, 2.28e+00	6.97e-08	0.75	2,3
106	ex5.5_order4	none	0; 0; 15	44				0	1.62e-01, 1.62e-01	2.66e-06	0.88	
		pd1	0; 0; 7	20	1	0.05	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.81	2,3
		pd2	0; 0; 7	20	1	0.07	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.02		0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
107	ex5.5_order5	none	0; 0; 21	65				0	1.62e-01, 1.62e-01	1.73e-06	0.83	
		pd1	0; 0; 7	20	1	0.07	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		pd2	0; 0; 7	20	1	0.08	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.02		0	3.07e-01, 3.07e-01	1.08e-07	0.92	2,3
108	ex5.5_order6	none	0; 0; 28	90				0	1.62e-01, 1.62e-01	2.59e-06	0.88	
		pd1	0; 0; 7	20	1	0.08	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		pd2	0; 0; 7	20	1	0.10	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.02		0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
109	ex5.5_order7	none	0; 0; 36	119				0	1.62e-01, 1.62e-01	4.17e-06	0.90	
		pd1	0; 0; 7	20	1	0.09	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		pd2	0; 0; 7	20	1	0.13	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.03		0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
110	ex5.5_order8	none	0; 0; 45	152				0	1.62e-01, 1.62e-01	1.97e-06	0.96	
		pd1	0; 0; 7	20	1	0.12	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		pd2	0; 0; 7	20	1	0.18	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.77	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.04		0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
111	ex5.5_order9	none	0; 0; 55	189				0	1.62e-01, 1.62e-01	1.33e-06	0.92	
		pd1	0; 0; 7	20	1	0.14	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.77	2,3
		pd2	0; 0; 7	20	1	0.22	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.05		0	3.07e-01, 3.07e-01	1.08e-07	0.74	2,3
112	ex5.5_order10	none	0; 0; 66	230				0	1.62e-01, 1.62e-01	1.21e-05	0.94	
		pd1	0; 0; 7	20	1	0.17	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.86	2,3
		pd2	0; 0; 7	20	1	0.29	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.06		0	3.07e-01, 3.07e-01	1.08e-07	0.74	2,3
113	ex5.5_order11	none	0; 0; 78	275				0	1.62e-01, 1.62e-01	5.86e-06	1.06	
		pd1	0; 0; 7	20	1	0.19	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.92	2,3
		pd2	0; 0; 7	20	1	0.39	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.81	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.06	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.08		0	3.07e-01, 3.07e-01	1.08e-07	1.13	2,3
114	ex5.5_order12	none	0; 0; 91	324				0	1.62e-01, 1.62e-01	5.58e-06	1.49	
		pd1	0; 0; 7	20	1	0.35	0.00	0	3.07e-01, 3.07e-01	1.08e-07	1.03	2,3
		pd2	0; 0; 7	20	1	0.55	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.82	2,3
		dd1			0	0.02	0.00					
		dd2			0	0.08	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.09		0	3.07e-01, 3.07e-01	1.08e-07	0.85	2,3
115	ex5.5_order13	none	0; 0; 105	377				0	1.62e-01, 1.62e-01	1.01e-05	1.24	
		pd1	0; 0; 7	20	1	0.29	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.88	2,3
		pd2	0; 0; 7	20	1	0.76	0.00	0	3.07e-01, 3.07e-01	1.08e-07	1.26	2,3
		dd1			0	0.04	0.00					
		dd2			0	0.17	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.15		0	3.07e-01, 3.07e-01	1.08e-07	0.77	2,3

Table A.18: Detailed results on DIW dataset, part 11 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
116	ex5.5_order14	none	0; 0; 120	434				0	1.62e-01, 1.62e-01	2.12e-05	1.54	
		pd1	0; 0; 7	20	1	0.34	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.83	2,3
		pd2	0; 0; 7	20	1	1.15	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.12	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.15		0	3.07e-01, 3.07e-01	1.08e-07	0.96	2,3
117	ex5.5_order15	none	0; 0; 136	495				0	1.62e-01, 1.62e-01	4.22e-05	1.85	
		pd1	0; 0; 7	20	1	0.44	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.77	2,3
		pd2	0; 0; 7	20	1	1.39	0.00	0	3.07e-01, 3.07e-01	1.08e-07	0.79	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.17	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.19		0	3.07e-01, 3.07e-01	1.08e-07	0.88	2,3
118	ex5.5_order16	none	0; 0; 153	560				0	1.62e-01, 1.62e-01	4.38e-05	2.43	
		pd1	0; 0; 7	20	1	0.52	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		pd2	0; 0; 7	20	1	1.93	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		dd1			0	0.03	0.01					
		dd2			0	0.22	0.01					
		Sieve-SDP	0; 0; 7	20	1	0.22		0	3.07e-01, 3.07e-01	1.08e-07	0.77	2,3
119	ex5.5_order17	none	0; 0; 171	629				0	1.62e-01, 1.62e-01	6.83e-06	3.24	
		pd1	0; 0; 7	20	1	0.63	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.74	2,3
		pd2	0; 0; 7	20	1	2.52	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.75	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.26	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.28		0	3.07e-01, 3.07e-01	1.08e-07	0.79	2,3
120	ex5.5_order18	none	0; 0; 190	702				0	1.62e-01, 1.62e-01	4.05e-06	4.51	
		pd1	0; 0; 7	20	1	0.92	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.89	2,3
		pd2	0; 0; 7	20	1	3.45	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.89	2,3
		dd1			0	0.04	0.00					
		dd2			0	0.34	0.00					
		Sieve-SDP	0; 0; 7	20	1	0.33		0	3.07e-01, 3.07e-01	1.08e-07	0.80	2,3
121	ex5.5_order19	none	0; 0; 210	779				0	1.62e-01, 1.62e-01	4.64e-05	7.12	
		pd1	0; 0; 7	20	1	1.41	0.01	0	3.07e-01, 3.07e-01	1.08e-07	1.23	2,3
		pd2	0; 0; 7	20	1	5.53	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.82	2,3
		dd1			0	0.04	0.01					
		dd2			0	0.43	0.01					
		Sieve-SDP	0; 0; 7	20	1	0.39		0	3.07e-01, 3.07e-01	1.08e-07	0.80	2,3
122	ex5.5_order20	none	0; 0; 231	860				0	1.62e-01, 1.62e-01	6.34e-05	8.90	
		pd1	0; 0; 7	20	1	1.22	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		pd2	0; 0; 7	20	1	5.88	0.01	0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
		dd1			0	0.05	0.01					
		dd2			0	0.52	0.01					
		Sieve-SDP	0; 0; 7	20	1	0.46		0	3.07e-01, 3.07e-01	1.08e-07	0.76	2,3
123	ex5.6_order4	none	0; 0; 15	44				0	3.04e-01, 3.04e-01	1.15e-07	1.01	
		pd1	0; 0; 8	21	1	0.07	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.94	
		pd2	0; 0; 8	21	1	0.08	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.84	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.02		0	3.04e-01, 3.04e-01	1.01e-08	1.02	
124	ex5.6_order5	none	0; 0; 21	65				0	3.04e-01, 3.04e-01	1.74e-07	1.10	
		pd1	0; 0; 8	21	1	0.07	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.84	
		pd2	0; 0; 8	21	1	0.07	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.90	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.02		0	3.04e-01, 3.04e-01	1.01e-08	0.92	
125	ex5.6_order6	none	0; 0; 28	90				0	3.04e-01, 3.04e-01	3.45e-07	0.89	
		pd1	0; 0; 8	21	1	0.07	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.78	
		pd2	0; 0; 8	21	1	0.09	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.86	
		dd1			0	0.03	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.05		0	3.04e-01, 3.04e-01	1.01e-08	2.02	
126	ex5.6_order7	none	0; 0; 36	119				0	3.04e-01, 3.04e-01	3.21e-07	1.17	
		pd1	0; 0; 8	21	1	0.12	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.96	
		pd2	0; 0; 8	21	1	0.17	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.85	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.03		0	3.04e-01, 3.04e-01	1.01e-08	1.39	
127	ex5.6_order8	none	0; 0; 45	152				0	3.04e-01, 3.04e-01	4.51e-07	0.91	
		pd1	0; 0; 8	21	1	0.12	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.79	
		pd2	0; 0; 8	21	1	0.16	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.94	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.04		0	3.04e-01, 3.04e-01	1.01e-08	0.74	

Table A.19: Detailed results on DIW dataset, part 12 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
128	ex5.6_order9	none	0; 0; 55	189				0	3.04e-01, 3.04e-01	7.42e-07	0.93	
		pd1	0; 0; 8	21	1	0.14	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.79	
		pd2	0; 0; 8	21	1	0.24	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.82	
		dd1			0	0.03	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.06		0	3.04e-01, 3.04e-01	1.01e-08	0.82	
129	ex5.6_order10	none	0; 0; 66	230				0	3.04e-01, 3.04e-01	7.57e-07	0.87	
		pd1	0; 0; 8	21	1	0.16	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.77	
		pd2	0; 0; 8	21	1	0.28	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.74	
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.06		0	3.04e-01, 3.04e-01	1.01e-08	0.75	
130	ex5.6_order11	none	0; 0; 78	275				0	3.04e-01, 3.04e-01	6.11e-07	0.90	
		pd1	0; 0; 8	21	1	0.18	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.74	
		pd2	0; 0; 8	21	1	0.40	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.76	
		dd1			0	0.02	0.00					
		dd2			0	0.06	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.08		0	3.04e-01, 3.04e-01	1.01e-08	0.76	
131	ex5.6_order12	none	0; 0; 91	324				0	3.04e-01, 3.04e-01	9.01e-07	1.02	
		pd1	0; 0; 8	21	1	0.23	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.75	
		pd2	0; 0; 8	21	1	0.54	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.75	
		dd1			0	0.02	0.00					
		dd2			0	0.08	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.10		0	3.04e-01, 3.04e-01	1.01e-08	0.74	
132	ex5.6_order13	none	0; 0; 105	377				0	3.04e-01, 3.04e-01	7.94e-07	1.20	
		pd1	0; 0; 8	21	1	0.28	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.77	
		pd2	0; 0; 8	21	1	0.71	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.84	
		dd1			0	0.03	0.00					
		dd2			0	0.11	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.13		0	3.04e-01, 3.04e-01	1.01e-08	0.79	
133	ex5.6_order14	none	0; 0; 120	434				0	3.04e-01, 3.04e-01	1.19e-06	1.31	
		pd1	0; 0; 8	21	1	0.34	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.82	2
		pd2	0; 0; 8	21	1	1.01	0.00	0	3.04e-01, 3.04e-01	1.01e-08	0.74	2
		dd1			0	0.02	0.00					
		dd2			0	0.13	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.15		0	3.04e-01, 3.04e-01	1.01e-08	0.79	2
134	ex5.6_order15	none	0; 0; 136	495				0	3.04e-01, 3.04e-01	8.76e-07	1.62	
		pd1	0; 0; 8	21	1	0.43	0.00	0	3.04e-01, 3.04e-01	1.01e-08	1.27	
		pd2	0; 0; 8	21	1	1.45	0.01	0	3.04e-01, 3.04e-01	1.01e-08	1.21	
		dd1			0	0.04	0.00					
		dd2			0	0.16	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.19		0	3.04e-01, 3.04e-01	1.01e-08	0.76	
135	ex5.6_order16	none	0; 0; 153	560				0	3.04e-01, 3.04e-01	1.79e-06	1.81	
		pd1	0; 0; 8	21	1	0.52	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.75	2
		pd2	0; 0; 8	21	1	1.91	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.74	2
		dd1			0	0.03	0.00					
		dd2			0	0.21	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.22		0	3.04e-01, 3.04e-01	1.01e-08	0.76	2
136	ex5.6_order17	none	0; 0; 171	629				0	3.04e-01, 3.04e-01	1.78e-06	2.29	
		pd1	0; 0; 8	21	1	0.60	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.83	2
		pd2	0; 0; 8	21	1	2.47	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.84	2
		dd1			0	0.03	0.00					
		dd2			0	0.26	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.28		0	3.04e-01, 3.04e-01	1.01e-08	0.74	2
137	ex5.6_order18	none	0; 0; 190	702				0	3.04e-01, 3.04e-01	2.53e-06	2.68	
		pd1	0; 0; 8	21	1	0.75	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.76	2
		pd2	0; 0; 8	21	1	3.24	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.83	2
		dd1			0	0.04	0.00					
		dd2			0	0.31	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.33		0	3.04e-01, 3.04e-01	1.01e-08	0.74	2
138	ex5.6_order19	none	0; 0; 210	779				0	3.04e-01, 3.04e-01	2.54e-06	3.34	
		pd1	0; 0; 8	21	1	1.06	0.01	0	3.04e-01, 3.04e-01	1.01e-08	1.52	2
		pd2	0; 0; 8	21	1	4.87	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.89	2
		dd1			0	0.05	0.00					
		dd2			0	0.45	0.00					
		Sieve-SDP	0; 0; 8	21	1	0.50		0	3.04e-01, 3.04e-01	1.01e-08	0.92	2
139	ex5.6_order20	none	0; 0; 231	860				0	3.04e-01, 3.04e-01	3.35e-06	5.55	
		pd1	0; 0; 8	21	1	1.48	0.01	0	3.04e-01, 3.04e-01	1.01e-08	1.02	2
		pd2	0; 0; 8	21	1	6.57	0.01	0	3.04e-01, 3.04e-01	1.01e-08	0.91	2
		dd1			0	0.07	0.01					
		dd2			0	0.58	0.01					
		Sieve-SDP	0; 0; 8	21	1	0.53		0	3.04e-01, 3.04e-01	1.01e-08	1.02	2

Table A.20: Detailed results on DIW dataset, part 13 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
140	ex5.7_order5	none	0; 1; 21	65				0	8.52e-09, -5.99e-09	7.10e-08	0.93	
		pd1	0; 1; 8	22	1	0.08	0.00	0	1.81e-08, -2.87e-09	5.26e-08	1.20	
		pd2	0; 1; 8	22	1	0.14	0.00	0	1.81e-08, -2.87e-09	5.26e-08	1.12	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP	0; 1; 8	22	1	0.03		0	1.81e-08, -2.87e-09	5.26e-08	1.17	
141	ex5.7_order6	none	0; 0; 28, 3	90				0	2.28e-08, -2.32e-09	7.80e-08	0.97	
		pd1	0; 0; 10, 2	28	1	0.09	0.00	0	7.23e-09, -4.48e-09	2.43e-08	1.07	
		pd2	0; 0; 10, 2	28	1	0.18	0.00	0	7.23e-09, -4.48e-09	2.43e-08	0.93	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 10, 2	28	1	0.03		0	7.23e-09, -4.48e-09	2.43e-08	1.02	
142	ex5.7_order7	none	0; 0; 36, 6	119				0	2.01e-08, -9.32e-09	1.19e-07	0.89	
		pd1	0; 0; 11, 3	33	1	0.14	0.00	0	7.70e-09, -4.95e-09	2.93e-08	0.89	
		pd2	0; 0; 11, 3	33	1	0.18	0.00	0	7.70e-09, -4.95e-09	2.93e-08	0.91	
		dd1			0	0.02	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	0; 0; 11, 3	33	1	0.04		0	7.70e-09, -4.95e-09	2.93e-08	0.95	
143	ex5.7_order8	none	0; 0; 45, 10	152				0	1.43e-08, -1.11e-08	1.17e-07	0.93	
		pd1	0; 0; 12, 4	36	1	0.16	0.00	0	2.24e-08, -2.72e-09	7.97e-08	0.91	
		pd2	0; 0; 12, 4	36	1	0.26	0.00	0	2.24e-08, -2.72e-09	7.97e-08	1.02	
		dd1			0	0.02	0.00					
		dd2			0	0.04	0.00					
		Sieve-SDP	0; 0; 12, 4	36	1	0.07		0	2.24e-08, -2.72e-09	7.97e-08	1.00	
144	ex5.7_order9	none	0; 0; 55, 15	189				0	6.00e-09, -4.83e-09	5.61e-08	1.24	
		pd1	0; 0; 13, 5	41	1	0.24	0.01	0	2.15e-08, -3.69e-09	8.53e-08	1.19	
		pd2	0; 0; 13, 5	41	1	0.39	0.01	0	2.15e-08, -3.69e-09	8.53e-08	0.89	
		dd1			0	0.02	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP	0; 0; 13, 5	41	1	0.10		0	2.15e-08, -3.69e-09	8.53e-08	0.88	
145	ex5.7_order10	none	0; 0; 66, 21	230				0	1.30e-08, -1.43e-08	1.69e-07	0.91	
		pd1	0; 0; 13, 5	41	1	0.31	0.01	0	2.15e-08, -3.69e-09	8.53e-08	0.95	
		pd2	0; 0; 13, 5	41	1	0.53	0.01	0	2.15e-08, -3.69e-09	8.53e-08	1.58	
		dd1			0	0.03	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP	0; 0; 13, 5	41	1	0.15		0	2.15e-08, -3.69e-09	8.53e-08	0.94	
146	ex5.7_order11	none	0; 0; 78, 28	275				0	1.71e-08, -1.50e-08	2.11e-07	1.11	
		pd1	0; 0; 14, 6	45	1	0.36	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.91	
		pd2	0; 0; 14, 6	45	1	0.75	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.93	
		dd1			0	0.02	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP	0; 0; 14, 6	45	1	0.17		0	1.73e-08, -3.62e-09	8.09e-08	0.95	
147	ex5.7_order12	none	0; 0; 91, 36	324				0	5.85e-09, -6.21e-09	7.87e-08	1.04	
		pd1	0; 0; 14, 6	45	1	0.44	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.91	
		pd2	0; 0; 14, 6	45	1	1.04	0.01	0	1.73e-08, -3.62e-09	8.09e-08	1.03	
		dd1			0	0.03	0.00					
		dd2			0	0.12	0.00					
		Sieve-SDP	0; 0; 14, 6	45	1	0.24		0	1.73e-08, -3.62e-09	8.09e-08	1.50	
148	ex5.7_order13	none	0; 0; 105, 45	377				0	2.13e-08, -2.19e-08	3.23e-07	1.15	
		pd1	0; 0; 14, 6	45	1	0.49	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.90	
		pd2	0; 0; 14, 6	45	1	1.56	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.95	
		dd1			0	0.03	0.00					
		dd2			0	0.14	0.00					
		Sieve-SDP	0; 0; 14, 6	45	1	0.31		0	1.73e-08, -3.62e-09	8.09e-08	1.05	
149	ex5.7_order14	none	0; 0; 120, 55	434				0	2.24e-08, -2.28e-08	3.72e-07	1.69	
		pd1	0; 0; 14, 6	45	1	0.77	0.01	0	1.73e-08, -3.62e-09	8.09e-08	1.07	
		pd2	0; 0; 14, 6	45	1	2.39	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.94	
		dd1			0	0.03	0.01					
		dd2			0	0.22	0.01					
		Sieve-SDP	0; 0; 14, 6	45	1	0.37		0	1.73e-08, -3.62e-09	8.09e-08	0.88	
150	ex5.7_order15	none	0; 0; 136, 66	495				0	2.09e-08, -2.22e-08	3.87e-07	1.56	
		pd1	0; 0; 14, 6	45	1	0.82	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.84	
		pd2	0; 0; 14, 6	45	1	2.80	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.86	
		dd1			0	0.03	0.01					
		dd2			0	0.26	0.01					
		Sieve-SDP	0; 0; 14, 6	45	1	0.45		0	1.73e-08, -3.62e-09	8.09e-08	1.38	
151	ex5.7_order16	none	0; 0; 153, 78	560				0	1.87e-08, -1.91e-08	3.70e-07	2.13	
		pd1	0; 0; 14, 6	45	1	1.13	0.01	0	1.73e-08, -3.62e-09	8.09e-08	0.97	
		pd2	0; 0; 14, 6	45	1	3.91	0.01	0	1.73e-08, -3.62e-09	8.09e-08	1.41	
		dd1			0	0.04	0.01					
		dd2			0	0.34	0.01					
		Sieve-SDP	0; 0; 14, 6	45	1	0.60		0	1.73e-08, -3.62e-09	8.09e-08	1.43	

Table A.21: Detailed results on DIW dataset, part 14 of 14

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
152	ex5.7_order17	none	0; 0; 171, 91	629				0	1.84e-08, -1.83e-08	3.89e-07	2.56	
		pd1	0; 0; 14, 6	45	1	1.47	0.02	0	1.73e-08, -3.62e-09	8.09e-08	0.92	
		pd2	0; 0; 14, 6	45	1	5.58	0.02	0	1.73e-08, -3.62e-09	8.09e-08	1.13	
		dd1			0	0.05	0.01					
		dd2			0	0.46	0.01					
		Sieve-SDP	0; 0; 14, 6	45	1	0.95		0	1.73e-08, -3.62e-09	8.09e-08	1.47	
153	ex5.7_order18	none	0; 0; 190, 105	702				0	1.76e-08, -1.75e-08	4.12e-07	3.01	
		pd1	0; 0; 14, 6	45	1	1.76	0.02	0	1.73e-08, -3.62e-09	8.09e-08	0.84	
		pd2	0; 0; 14, 6	45	1	7.46	0.02	0	1.73e-08, -3.62e-09	8.09e-08	1.15	
		dd1			0	0.07	0.01					
		dd2			0	0.58	0.01					
		Sieve-SDP	0; 0; 14, 6	45	1	0.94		0	1.73e-08, -3.62e-09	8.09e-08	1.36	
154	ex5.7_order19	none	0; 0; 210, 120	779				0	1.68e-08, -1.67e-08	4.25e-07	4.46	
		pd1	0; 0; 14, 6	45	1	2.49	0.02	0	1.73e-08, -3.62e-09	8.09e-08	1.09	
		pd2	0; 0; 14, 6	45	1	10.47	0.02	0	1.73e-08, -3.62e-09	8.09e-08	0.89	
		dd1			0	0.05	0.02					
		dd2			0	0.84	0.02					
		Sieve-SDP	0; 0; 14, 6	45	1	1.11		0	1.73e-08, -3.62e-09	8.09e-08	1.01	
155	ex5.7_order20	none	0; 0; 231, 136	860				0	1.54e-08, -1.69e-08	4.40e-07	5.84	
		pd1	0; 0; 14, 6	45	1	3.28	0.03	0	1.73e-08, -3.62e-09	8.09e-08	1.02	
		pd2	0; 0; 14, 6	45	1	13.67	0.03	0	1.73e-08, -3.62e-09	8.09e-08	1.17	
		dd1			0	0.08	0.02					
		dd2			0	1.04	0.02					
		Sieve-SDP	0; 0; 14, 6	45	1	1.71		0	1.73e-08, -3.62e-09	8.09e-08	0.86	

A.1.4 Detailed results on the Henrion-Toh dataset

This dataset has 98 problems from polynomial optimization, 18 of which were reduced by at least one of the five preprocessing methods.

Table A.22: Detailed results on Henrion dataset, part 1 of 2

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
1	sedumi-brown	none	925; 0; 56	461				0	-7.34e-09, 0.00e+00	3.75e-07	0.93	
		pd1	925; 0; 21	251	1	0.15	0.05	0	-9.33e-11, 0.00e+00	6.25e-09	0.80	
		pd2	925; 0; 21	251	1	0.23	0.02	0	-9.33e-11, 0.00e+00	6.25e-09	0.78	
		dd1			0	0.03	0.00					
		dd2			0	0.06	0.00					
		Sieve-SDP	925; 0; 21	251	1	0.04		0	-9.33e-11, 0.00e+00	6.25e-09	0.79	
2	sedumi-conform3	none	630; 0; 56	285				0	2.05e-08, 0.00e+00	4.54e-07	0.66	
		pd1	630; 0; 53	273	1	0.04	0.02	0	2.51e-08, 0.00e+00	4.90e-07	0.71	
		pd2	630; 0; 53	273	1	0.11	0.03	0	2.51e-08, 0.00e+00	4.90e-07	0.73	
		dd1			0	0.03	0.00					
		dd2			0	0.06	0.00					
		Sieve-SDP	630; 0; 53	273	1	0.02		0	2.51e-08, 0.00e+00	4.90e-07	0.68	
3	sedumi-conform4	none	1890; 0; 84	454				0	-2.51e-08, 0.00e+00	5.57e-06	0.84	
		pd1	1890; 0; 81	442	1	0.07	0.04	0	-6.47e-09, 0.00e+00	1.74e-06	0.81	
		pd2	1890; 0; 81	442	1	0.26	0.04	0	-6.47e-09, 0.00e+00	1.74e-06	0.88	
		dd1			0	0.02	0.00					
		dd2			0	0.08	0.00					
		Sieve-SDP	1890; 0; 81	442	1	0.02		0	-6.47e-09, 0.00e+00	1.74e-06	0.77	
4	sedumi-fp23	none	$0; 0; 28, 7_{13}$	209				0	2.13e+02, 2.13e+02	3.97e-06	1.50	
		pd1	$0; 0; 7_{14}$	83	1	0.09	0.02	0	2.13e+02, 2.13e+02	9.96e-07	1.60	3
		pd2	$0; 0; 7_{14}$	83	1	0.13	0.02	0	2.13e+02, 2.13e+02	9.96e-07	1.44	3
		dd1			0	0.05	0.00					
		dd2			0	0.06	0.00					
		Sieve-SDP	$0; 0; 7_{14}$	83	1	0.04		0	2.13e+02, 2.13e+02	9.96e-07	1.33	3
5	sedumi-fp24	none	$0; 0; 105, 14_{35}$	2379				0	1.95e+02, 1.95e+02	9.68e-08	6.36	
		pd1	$0; 0; 14_{36}$	559	1	0.32	0.21	0	1.95e+02, 1.95e+02	1.74e-10	1.77	
		pd2	$0; 0; 14_{36}$	559	1	0.82	0.20	0	1.95e+02, 1.95e+02	1.74e-10	1.90	
		dd1			0	0.06	0.00					
		dd2			0	0.25	0.00					
		Sieve-SDP	$0; 0; 14_{36}$	559	1	0.23		0	1.95e+02, 1.95e+02	1.74e-10	1.78	

Table A.23: Detailed results on Henrion dataset, part 2 of 2

No.	Name	Method	f; l; s	m	Red.	t_{prep}	t_{conv}	Infeas	Obj (P, D)	DIMACS	t_{sol}	Help
6	sedumi-fp25	none	$0; 0; 28, 7_{15}$	209				0	1.10e+01, 1.10e+01	6.63e-06	1.39	
		pd1	$0; 0; 7_{16}$	83	1	0.12	0.03	0	1.10e+01, 1.10e+01	1.39e-07	1.46	2,3
		pd2	$0; 0; 7_{16}$	83	1	0.17	0.05	0	1.10e+01, 1.10e+01	1.39e-07	1.28	2,3
		dd1			0	0.05	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP	$0; 0; 7_{16}$	83	1	0.03		0	1.10e+01, 1.10e+01	1.39e-07	1.30	2,3
7	sedumi-fp26	none	$0; 0; 66, 11_{31}$	1000				0	2.68e+02, 2.68e+02	3.74e-08	2.11	
		pd1	$0; 0; 11_{32}$	285	1	0.17	0.16	0	2.68e+02, 2.68e+02	1.18e-07	1.46	
		pd2	$0; 0; 11_{32}$	285	1	0.53	0.15	0	2.68e+02, 2.68e+02	1.18e-07	1.48	
		dd1			0	0.08	0.00					
		dd2			0	0.46	0.00					
		Sieve-SDP	$0; 0; 11_{32}$	285	1	0.11		0	2.68e+02, 2.68e+02	1.18e-07	1.53	
8	sedumi-fp27	none	$0; 0; 66, 11_{25}$	1000				0	3.90e+01, 3.90e+01	1.96e-10	2.50	
		pd1	$0; 0; 11_{26}$	285	1	0.16	0.10	0	3.90e+01, 3.90e+01	3.98e-09	1.50	
		pd2	$0; 0; 11_{26}$	285	1	0.39	0.11	0	3.90e+01, 3.90e+01	3.98e-09	1.45	
		dd1			0	0.05	0.01					
		dd2			0	0.30	0.01					
		Sieve-SDP	$0; 0; 11_{26}$	285	1	0.11		0	3.90e+01, 3.90e+01	3.98e-09	1.46	
9	sedumi-fp32	none	$0; 0; 165, 45_{22}$	3002				0	-7.05e+00, -7.05e+00	2.79e-07	47.43	
		pd1	$0; 0; 45_4, 9_3, 45_{16}$	1286	1	2.17	0.56	0	-7.05e+00, -7.05e+00	2.50e-06	8.58	3
		pd2	$0; 0; 45_4, 9_3, 45_{16}$	1286	1	4.55	0.57	0	-7.05e+00, -7.05e+00	2.50e-06	9.26	3
		dd1			0	0.11	0.02					
		dd2			0	9.78	0.02					
		Sieve-SDP	$0; 0; 45_4, 9_3, 45_{16}$	1286	1	1.21		0	-7.05e+00, -7.05e+00	2.50e-06	12.14	3
10	sedumi-fp33	none	$0; 0; 21, 6_{16}$	125				0	-1.01e+04, -1.01e+04	3.36e-07	0.75	
		pd1	$0; 0; 13, 6_{16}$	105	1	0.10	0.04	0	-1.01e+04, -1.01e+04	3.01e-07	0.94	
		pd2	$0; 0; 13, 6_{16}$	105	1	0.20	0.03	0	-1.01e+04, -1.01e+04	3.01e-07	1.32	
		dd1			0	0.04	0.00					
		dd2			0	0.12	0.00					
		Sieve-SDP	$0; 0; 14, 6_{16}$	111	1	0.03		0	-1.18e+04, -1.18e+04	9.28e-02	1.05	-2
11	sedumi-fp34	none	$0; 0; 28, 7_{16}$	209				0	1.72e+02, 1.72e+02	8.10e-07	0.94	
		pd1	$0; 0; 7, 1_2, 7_{14}$	83	1	0.14	0.04	0	1.72e+02, 1.72e+02	3.11e-07	0.88	
		pd2	$0; 0; 7, 1_2, 7_{14}$	83	1	0.11	0.03	0	1.72e+02, 1.72e+02	3.11e-07	0.73	
		dd1			0	0.03	0.00					
		dd2			0	0.07	0.00					
		Sieve-SDP	$0; 0; 7, 1_2, 7_{14}$	83	1	0.02		0	1.72e+02, 1.72e+02	3.11e-07	0.78	
12	sedumi-fp35	none	$0; 0; 35, 20_8$	164				0	4.00e+00, 4.00e+00	5.76e-06	0.86	
		pd1	$0; 0; 20_8, 10$	119	1	0.16	0.04	0	4.00e+00, 4.00e+00	5.66e-07	0.80	2,3
		pd2	$0; 0; 20_8, 10$	119	1	0.36	0.05	0	4.00e+00, 4.00e+00	5.66e-07	0.85	2,3
		dd1			0	0.03	0.00					
		dd2			0	0.28	0.00					
		Sieve-SDP	$0; 0; 20_8, 10$	119	1	0.05		0	4.00e+00, 4.00e+00	5.66e-07	0.89	2,3
13	sedumi-fp44	none	$0; 0; 4, 3_2$	6				0	4.44e+02, 4.44e+02	4.67e-08	1.02	
		pd1	$0; 0; 3_3$	5	1	0.04	0.00	0	4.44e+02, 4.44e+02	1.55e-08	0.87	
		pd2	$0; 0; 3_3$	5	1	0.04	0.00	0	4.44e+02, 4.44e+02	1.55e-08	0.82	
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 3_3$	5	1	0.01		0	4.44e+02, 4.44e+02	1.55e-08	0.77	
14	sedumi-fp46	none	$0; 0; 10, 6_2$	27				0	6.70e-08, -2.54e-07	4.78e-07	0.89	
		pd1	$0; 0; 5, 3, 2$	11	1	0.13	0.00	0	1.54e-07, -2.20e-08	2.22e-07	0.76	
		pd2	$0; 0; 5, 3, 2$	11	1	0.14	0.00	0	1.54e-07, -2.20e-08	2.22e-07	0.69	
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$0; 0; 5, 3, 2$	11	1	0.01		0	1.54e-07, -2.20e-08	2.22e-07	0.76	
15	sedumi-fp49	none	$1; 0; 6, 3_4$	14				0	1.67e+01, 1.67e+01	5.98e-08	1.04	
		pd1	$1; 0; 4, 3_4$	10	1	0.05	0.00	0	1.67e+01, 1.67e+01	1.40e-08	0.62	
		pd2	$1; 0; 4, 3_4$	10	1	0.06	0.00	0	1.67e+01, 1.67e+01	1.40e-08	0.58	
		dd1			0	0.02	0.00					
		dd2			0	0.02	0.00					
		Sieve-SDP	$1; 0; 4, 3_4$	10	1	0.01		0	1.67e+01, 1.67e+01	1.40e-08	0.60	
16	sedumi-fp210	none	$66; 0; 66, 11_{10}$	1000				0	3.75e-01, 3.75e-01	2.15e-07	1.59	
		pd1	$66; 0; 11_{11}$	285	1	0.12	0.05	0	3.75e-01, 3.75e-01	2.55e-08	1.35	
		pd2	$66; 0; 11_{11}$	285	1	0.17	0.03	0	3.75e-01, 3.75e-01	2.55e-08	1.28	
		dd1			0	0.05	0.00					
		dd2			0	0.09	0.00					
		Sieve-SDP	$66; 0; 11_{11}$	285	1	0.05		0	3.75e-01, 3.75e-01	2.55e-08	1.31	
17	sedumi-fp410	none	$1; 0; 6, 3_4$	14				0	1.67e+01, 1.67e+01	5.98e-08	0.87	
		pd1	$1; 0; 4, 3_4$	10	1	0.08	0.01	0	1.67e+01, 1.67e+01	1.40e-08	0.81	
		pd2	$1; 0; 4, 3_4$	10	1	0.08	0.00	0	1.67e+01, 1.67e+01	1.40e-08	0.90	
		dd1			0	0.03	0.00					
		dd2			0	0.03	0.00					
		Sieve-SDP	$1; 0; 4, 3_4$	10	1	0.02		0	1.67e+01, 1.67e+01	1.40e-08	0.81	
18	sedumi-l4	none	$0; 0; 45$	152				0	3.70e-02, 3.70e-02	7.10e-08	0.83	
		pd1	$0; 0; 1$	1	1	0.15	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.44	1
		pd2	$0; 0; 1$	1	1	0.19	0.00	1	0.00e+00, 1.00e+00	5.00e-01	0.42	1
		dd1			0	0.02	0.00					
		dd2			0	0.05	0.00					
		Sieve-SDP			infeas	0.04					0.00	1

A.2 Core MATLAB Code

In this section, we provide our core MATLAB code of Sieve-SDP (excluding input, output, and dual solution recovery) with some comments. In our code, we physically delete rows and columns of the A_i and of C only at the very end. During the execution of the algorithm we only *mark* such rows, columns and constraints as deleted.

We use two arrays to keep track of what has been marked deleted:

1. The m-vector `undeleted`, whose i-th entry is 1 if constraint i *has not* been deleted, and 0 if it *has* been deleted.
2. The sparse array $\mathtt{I} \in \{0,1\}^{n\times(m+1)}$ with entries defined as follows.
   - For all i and for $1 \le j \le m$,

$$I(i,j) = \begin{cases} 0, & \text{if in } A_j \text{ the } i\text{-th row and column are all zero or have been deleted;} \\ 1, & \text{otherwise.} \end{cases}$$

   - For all i,

$$I(i,\ m+1) = \begin{cases} 0, & \text{if in all } A_j\text{'s the } i\text{-th row and column have been deleted;} \\ 1, & \text{otherwise.} \end{cases}$$

```
function[Ar, br, Cr, info] = SieveSDP(A, b, C, EPS)
    % Inputs:
    %  A: n-by-n*m sparse matrix (m symmetric n-by-n matrices side by side)
    %  b: the vector of rhs in R^m, and b <= 0;
    %  C: the objective coefficient n-by-n matrix;
    %  EPS: accuracy for safe mode, with default value eps
    % Outputs:
    %  Ar, br, cr: Reduced data after preprocessing
    %  info: A structure containing preprocessing info

    if nargin < 4, EPS = eps; end
    sqrtEPS = sqrt(EPS);
    Ar = []; br = []; Cr = []; n = size(C, 1); m = length(b);
```

```
14      I = true(n, m + 1);         % initial nonzero indices
15      for i = 1:m, I(:, i) = any(A(:, (n*(i - 1) + 1):(n*i)), 2); end
16
17      not_done    = 1;            % 1 means preprocessing not done
18      undeleted   = ones(m, 1); % keep track of deleted constraints
19      constr_ind  = (1:m);        % indices or undeleted constraints
20      mr          = m;            % reduced number of constraints
21      info.infeas = 0;            % infeasibility detected?
22      info.red    = 0;            % any reduction?
23      bn = -sqrtEPS*max(1, norm(b, inf));
24                      % b < 0 if b < -sqrt(epsilon)*max{1, ||b||}
25      bz = bn*sqrtEPS; % b = 0 if -epsilon*max{1, ||b||} < b <= 0
26
27      % Preprocessing
28      while not_done
29          not_done = 0;
30          for ii = 1:mr
31              i  = constr_ind(ii);
32              Ii = I(:, i); % indicates undeleted vars in matrix i
33              Ai = A(Ii, n*(i - 1) + find(Ii)); % nonzero submatrix
34              Iaux = any(Ai, 2);
35              if find(Iaux == false, 1),
36                  I(Ii, i) = Iaux; Ii = I(:, i); Ai = Ai(Iaux, Iaux);
37              end
38              if isempty(Ai)
39                  if b(i) < bn, info.infeas = 1; return; end
40                      % Ai = 0 and bi < 0 => infeasible
41                  if b(i) > bz, undeleted(i) = 0; continue; end
42                      % Ai = 0 and bi = 0 => reduce
43              end
44              if b(i) < bn
45                  [~, pd_check] = chol(Ai);
46                  if pd_check == 0, info.infeas = 1; return; end
47                      % Ai PD and bi < 0 => infeasible
48              else
49                  if b(i) > bz
```

```
                    [~, pd_check] = chol(Ai);
                    if pd_check == 0
                        I(Ii, :) = false; undeleted(i) = 0; not_done = 1;
                        % Ai PD and bi = 0 => reduce
                    else
                        [~, nd_check] = chol(-Ai);
                        if nd_check == 0
                           I(Ii, :) = false; undeleted(i) = 0; not_done = 1;
                           % Ai ND and bi = 0 => reduce
                        end
                    end
                end
            end
        end
        constr_ind = find(undeleted); mr = length(constr_ind);
    end

    % Undeleted rows/columns marked in I(:, m + 1); now physically delete
    if mr == m, Ar = A; br = b; Cr = C; info.red = 0; return; end
    info.red = 1;
    I_nonzero = I(:, m + 1); nr = nnz(I_nonzero); Ar = sparse(nr, nr*mr);
    for ii = 1:mr
        i = constr_ind(ii);
        Ar(:, (nr*(ii - 1) + 1):(nr*ii)) ...
            = A(I_nonzero, n*(i - 1) + find(I_nonzero));
    end
    br = b(constr_ind); Cr = C(I_nonzero, I_nonzero);

end
```

A.3 The DIMACS Errors

For completeness, we describe the DIMACS errors, which are commonly used to measure the accuracy of approximate solutions X of (P) and y of (D).

Define the operator $\mathcal{A}: \mathbb{R}^m \to \mathcal{S}^n$ and its adjoint as

$$\mathcal{A}(X) = (A_1 \cdot X, \ldots, A_m \cdot X)^\top \quad \text{and} \quad \mathcal{A}^*(y) = \sum_{i=1}^{m} y_i A_i.$$

Suppose that we are given an approximate primal solution X of (P) and an approximate dual solution y of (D). For brevity, define the slack variable $Z := C - \mathcal{A}^*(y)$. Then the DIMACS error measures are defined as follows:

$$\begin{cases} \text{err}_1 = \frac{\|\mathcal{A}(X)-b\|_2}{1+\|b\|_\infty}, & \text{err}_2 = \max\left\{0,\ \frac{-\lambda_{\min}(X)}{1+\|b\|_\infty}\right\}, & \text{err}_3 = \frac{\|\mathcal{A}^*(y)+Z-C\|_F}{1+\|C\|_\infty}, \\ \text{err}_4 = \max\left\{0,\ \frac{-\lambda_{\min}(Z)}{1+\|C\|_\infty}\right\}, & \text{err}_5 = \frac{b^\top y - C\cdot X}{1+|C\cdot X|+|b^\top y|}, & \text{err}_6 = \frac{Z\cdot X}{1+|C\cdot X|+|b^\top y|}. \end{cases}$$

Here, $\|\cdot\|_F$ is the Frobenius norm, $\|\cdot\|_\infty$ is the infinity norm, and $\lambda_{\min}(\cdot)$ denotes the smallest eigenvalue of a square matrix.

A.4 Dual Solution Recovery

In this section we address the following question: suppose we preprocessed the problem (P) by Sieve-SDP, then computed an optimal solution of the reduced SDP, (P_r) and of its dual, (D_r). Can we compute an optimal solution of the original primal (P) and of its dual (D)? The answer to the first question (primal solution recovery) is easy, while the issue of dual solution recovery is much more subtle.

First let us look at primal solution recovery. Since Sieve-SDP deletes rows and columns from the variable matrix X that are zeros anyway, if X^r is an optimal solution of (P-R), then by simply padding X^r with zeroes we obtain an optimal solution of (P).

Next we discuss dual solution recovery. For simplicity assume that Sieve-SDP performed just one iteration. Further, let us also assume that in the Basic Step in Figure 2.2 it eliminated the constraint $A_1 \cdot X = 0$, where

$$A_1 = \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix},$$

where $D \succ 0$, and let r be the order of D. After this one iteration of preprocessing, the reduced dual problem becomes:

$$\begin{array}{ll} \max\limits_{y\in\mathbb{R}^{m-1}} & \sum_{i=2}^m b_i y_i \\ \text{s.t.} & \begin{pmatrix} Z_{11} & Z_{21}^\top \\ Z_{21} & Z_{22} \end{pmatrix} = Z := C - \sum_{i=2}^m y_i A_i \in \begin{pmatrix} \times & \times \\ \times & \oplus \end{pmatrix}, \end{array} \tag{$\mathrm{D_r}$}$$

where the notation $\oplus$ means that the lower right $(n-r)\times(n-r)$ principal block of slack variable Z is PSD, and the rest is arbitrary. Thus clearly

$$\mathcal{D}^\star \leq \mathcal{D}_{\mathrm{r}}^\star, \tag{A.1}$$

since ($\mathrm{D_r}$) has a feasible region that is at least as large as that of (D) (and usually it is larger). Assume that $y^{\mathrm{r}} = (y_2^{\mathrm{r}}, \ldots, y_m^{\mathrm{r}})$ and Z^{r} is an optimal solution of ($\mathrm{D_r}$). Our recovery procedure, which we call Basic-Recovery, fixes y^{r} and seeks y_1 such that (y_1, y^{r}) is feasible in (D), i.e.,

$$Z^{\mathrm{r}} - y_1 A_1 \succeq 0. \tag{A.2}$$

We do this by a very basic line-search: we first try the values $y_1 = 0$, -1 and -2. If these all fail, then we try $y_1 = -100$. If we fail with $y_1 = -100$, we stop; otherwise we test $y_1 = -3, -4, \ldots$, and find the y_1 with the smallest magnitude such that (A.2) holds, where we test whether $Z^{\mathrm{r}} - y_1 A_1 + 10^{-6} I \succ 0$ holds by Cholesky factorization.

When Sieve-SDP deletes multiple constraints, we run Basic-Recovery to find the corresponding y_i's sequentially. For simplicity, assume that Sieve-SDP deleted constraints $1, 2, \ldots, k$ and we found an optimal primal and dual solution of the resulting SDP (using MOSEK). We then attempt to find an optimal dual solution of the SDP obtained by deleting only constraints $1, \ldots, k-1$; then to the SDP obtained by deleting only constraints $1, \ldots, k-2$; and so on.

Basic-Recovery is inspired by the dual solution recovery procedure in [104], which builds on the ideas in [101], and it assumes that the dual problem (D) is the one that is reduced.[1]

A.4.1 Conditions to ensure successful dual solution recovery

The procedure Basic-Recovery may fail. To see why, first assume that it succeeds, i.e.,

[1] See Remark 2.1 about how the primal and dual are defined in [104].

it computes a feasible solution of (D). Since y_1 has zero objective coefficient in (D), this solution has objective value $\mathcal{D}_{\rm r}^\star$, hence by inequality (A.1), it is optimal in (D), thus $\mathcal{D}^\star = \mathcal{D}_{\rm r}^\star$. Conversely, if $\mathcal{D}^\star < \mathcal{D}_{\rm r}^\star$, then Basic-Recovery *must* fail.

Example A.1 (Example 2.3 continued)**.** *When we apply Sieve-SDP to the SDP* (2.4)*, it deletes the first row and first column in all matrices and it also deletes the first constraint. Let us write out* ($\rm D_r$) *again for this problem (i.e., repeat the SDP* (2.7)*):*

$$\begin{array}{ll} \max\limits_{y} & y_2 \\ s.t. & y_2 \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \preceq \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \end{array} \tag{A.3}$$

whose optimal solution is $y_2^r = 1$. Thus, Basic-Recovery seeks y_1 such that

$$y_1 \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \preceq \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

and clearly there is no such y_1. □

For completeness, we state two conditions, given by [104], that ensure successful dual solution recovery for one iteration. While [104] states the conditions in the dual direction, we here use the notation and the block structure of ($\rm D_r$). Condition A.1 is equivalent to successful dual recovery, and Condition A.2 is a strong sufficient condition.

Condition A.1. $R(Z_{21}) \subseteq R(Z_{22})$.

Condition A.2. *Any X of the form*

$$X = \begin{pmatrix} 0 & X_{21}^\top \\ X_{21} & X_{22} \end{pmatrix},$$

that satisfies

$$A_i \cdot X = b_i, \quad i = 2, \ldots, m$$

has $X_{21} = 0$.

We remark here that if Condition A.2 holds, and (D_r) has optimal solution (y^r, Z^r), where $Z^r_{22} \succeq 0$, then there exists an optimal solution of (D_r) of the form

$$y^{r,\,\text{new}} := y^r + \hat{y} \quad \text{and} \quad Z^{r,\,\text{new}} = \begin{pmatrix} Z^r_{11} + \hat{Z}_{11} & 0 \\ 0 & Z^r_{22} \end{pmatrix}.$$

Therefore, in view of Condition A.1, dual solution recovery succeeds.

However, Condition A.2 is very strong, and none of the reduced problems in our datasets satisfies it. Thus, it may only have conceptual value.

A.4.2 Computational results on dual solution recovery

As discussed in the last Subsection (and Subsection A.4.3), we point out that dual solution recovery is much more difficult in SDP than in LP. We thus implemented an "ideal" recovery procedure, which we call Ideal-Recovery. It works as follows. Suppose $y^r = (y^r_{k+1}, \ldots, y^r_m)$ is an optimal dual solution of the SDP obtained by deleting constraints $1, \ldots, k$. Ideal-Recovery fixes y^r, then calls MOSEK to find a feasible solution $(y_1, \ldots, y_k)$ of the semidefinite feasibility problem

$$\sum_{i=1}^{k} y_i A_i + \sum_{i=k+1}^{m} y^r_i A_i \preceq C. \tag{A.4}$$

Table A.24 shows on how many instances methods pd1+Basic-Recovery, pd2+Basic-Recovery, Sieve-SDP+Basic-Recovery and Sieve-SDP+Ideal-Recovery succeeded. (Note that these three preprocessors succeeded on overlapping, but different problem sets, as a preprocessor may reduce an SDP, while another preprocessor may not reduce the same SDP. We do not report results with dd1 and dd2, since they preprocess from the dual direction and only reduced a very small proportion of the problems.)

Table A.24: Dual solution recovery by four methods

Method	# Reduced feasible	# Success	# Failure	Success rate	Time (s)
pd1 + Basic-Recovery	137	23	114	16.8%	154.75
pd2 + Basic-Recovery	158	39	119	24.7%	172.13
Sieve-SDP + Basic-Recovery	143	25	118	17.5%	12.62
Sieve-SDP + Ideal-Recovery	143	103	40	72.0%	1313.57

For pd1+Basic-Recovery and pd2+Basic-Recovery, success means that their dual solution recovery code reported success. For Sieve-SDP+Basic-Recovery, success means that it succeeded in every iteration: it computed the y_i for every deleted constraint. For Sieve-SDP+Ideal-Recovery it means that MOSEK did *not* report that (A.4) is infeasible.

Next, we make the criterion of "success" more rigorous: we redefined "success" as returning a pair of primal-dual optimal solutions whose greatest DIMACS error in absolute value is at most 10^{-6}. Table A.25 shows the results: now Sieve-SDP+Basic-Recovery is the winner, as it beats the supposedly perfect Sieve-SDP+Ideal-Recovery procedure. The success rates of pd1+Basic-Recovery and pd2+Basic-Recovery have also dropped.

Table A.25: Dual solution recovery assuming the tightest standard for "success"

Method	# Reduced feasible	# Success	# Failure	Success rate	Time (s)
pd1 + Basic-Recovery	137	19	118	13.9%	154.75
pd2 + Basic-Recovery	158	34	124	21.5%	172.13
Sieve-SDP + Basic-Recovery	143	25	118	17.5%	12.62
Sieve-SDP + Ideal-Recovery	143	17	126	11.9%	1313.57

Nevertheless, none of the methods do very well, and dual solution recovery in facial reduction remains a challenge, and is an interesting area for further research.

A.4.3 Case study: Failure of dual solution recovery on "unbound" problems

These 10 "unbound" problems, originally from [139], are part of the PP dataset and were discussed in Subsection 2.3.2. During preprocessing, pd1 and pd2 reduced the last 9 out of 10 problems, and Sieve-SDP reduced all 10 problems. However, none of these methods were able to recover the dual solution for any of the last 9 problems using Basic-Recovery or Ideal-Recovery introduced in the last Subsection. Therefore, we take a closer look at these problems.

For each $r \in \{1, \ldots, 10\}$, the r-th problem in this dataset is

$$\min_{X \in \mathcal{S}_+^{3r+1}} C \cdot X, \quad \text{s.t.} \quad A_1 \cdot X = 1, \quad A_i \cdot X = 0, \quad i = 2, 3, \ldots, 2r,$$

where C and A_i are the following diagonal block matrices:

$$C = \begin{pmatrix} E_0 & & \\ & O_r & \\ & & -F_0 \end{pmatrix}, \quad A_i = \begin{pmatrix} E_i & & \\ & F_{i-1} & \\ & & F_{i-2} - F_i \end{pmatrix}, \qquad i = 2, 3, \ldots, 2r-2,$$

$$A_1 = \begin{pmatrix} E_1 & & \\ & F_0 & \\ & & -F_1 \end{pmatrix}, \quad A_{2r-1} = \begin{pmatrix} E_{2r-1} & & \\ & F_{2r-2} & \\ & & F_{2r-3} \end{pmatrix}, \quad A_{2r} = \begin{pmatrix} E_{2r} & & \\ & O_r & \\ & & F_{2r-2} \end{pmatrix},$$

where all the empty blocks mean zeros, and E_j and F_j are defined as:

$$(E_j)_{\alpha,\beta} = \begin{cases} 1, & \text{if } \alpha + \beta = j + 2, \\ 0, & \text{o/w}, \end{cases} \quad \text{and} \quad (F_j)_{\alpha,\beta} = \begin{cases} 1, & \text{if } \alpha + \beta = j + 2, \\ 0, & \text{o/w}, \end{cases} \tag{A.5}$$

where $1 \le \alpha, \beta \le r+1$ for $E_j \in \mathcal{S}^{r+1}$ $(j = 0, 1, \ldots, 2r)$, and $1 \le \alpha, \beta \le r$ for F_j $(j = 0, 1, \ldots, 2r-2)$.

During Sieve-SDP preprocessing, by the structure shown in (A.5), in the k-th iteration, A_{2r-k+1} is the reducing certificate, $k = 1, 2, \ldots, 2r-1$. If k is odd, the $\left(\frac{2r-k+3}{2}, \frac{2r-k+3}{2}\right)$-entry of the first diagonal block and the $\left(\frac{2r-k+1}{2}, \frac{2r-k+1}{2}\right)$-entry of the third diagonal block are deleted, i.e., the corresponding rows and columns of X become 0; if k is even, the $\left(\frac{2r-k+2}{2}, \frac{2r-k+2}{2}\right)$-entry of the second diagonal block is deleted. In the end, the only constraint left is $A_1 \cdot X = 1$, and the fully reduced primal problem is

$$\min_{X \in \mathcal{S}^{3r+1}} \left(\begin{array}{c|c|c} 1 & & \\ & O_r & \\ \hline & O_r & \\ \hline & & -1 \\ & & \qquad O_{r-1} \end{array} \right) \cdot X,$$

$$
\text{s.t.} \quad \left(\begin{array}{cc|c|cc}
0 & 1 & & & \\
1 & 0 & & & \\
\multicolumn{2}{c|}{O_{r-1}} & & & \\
\hline
\multicolumn{2}{c|}{} & 1 & & \\
\multicolumn{2}{c|}{} & \multicolumn{2}{c}{O_{r-1}} & \\
\hline
\multicolumn{3}{c|}{} & 0 & -1 \\
\multicolumn{3}{c|}{} & -1 & 0 \\
\multicolumn{3}{c|}{} & \multicolumn{2}{c}{O_{r-2}}
\end{array}\right) \cdot X = 1,
$$

$$
X = \left(\begin{array}{cc|c|c}
x_{11} & & x_{21} & & \\
\multicolumn{2}{c|}{O_r} & & & \\
\hline
x_{21} & & x_{22} & & \\
\multicolumn{2}{c|}{} & \multicolumn{2}{c}{O_{r-1}} & \\
\hline
\multicolumn{3}{c|}{} & \multicolumn{2}{c}{O_r}
\end{array}\right),
$$

$$
\left(\begin{array}{cc} x_{11} & x_{21} \\ x_{21} & x_{22} \end{array}\right) \succeq 0,
$$

where we have explicitly partitioned the block structure, and x_{11}, x_{21}, and x_{22} are scalars. The fully reduced dual problem is

$$
\max_{y \in \mathbb{R}^{2r},\ Z \in \mathcal{S}^{3r+1}} \quad y_1
$$

$$
\text{s.t.} \quad Z = \left(\begin{array}{cc|c|cc}
1 & -y_1 & & & \\
-y_1 & 0 & & & \\
\multicolumn{2}{c|}{} & O_{r-1} & & \\
\hline
\multicolumn{2}{c|}{} & & -y_1 & \\
\multicolumn{2}{c|}{} & & \multicolumn{2}{c}{O_{r-1}} \\
\hline
\multicolumn{3}{c|}{} & -1 & y_1 \\
\multicolumn{3}{c|}{} & y_1 & O_{r-1}
\end{array}\right), \quad \left(\begin{array}{c|c} 1 & \\ \hline & -y_1 \end{array}\right) \succeq 0,
$$

with optimal solution

$$y_1^{\mathrm{r}} = 0, \quad Z^{\mathrm{r}} = \left(\begin{array}{c|c|cc} 1 & & & \\ & O_r & & \\ \hline & O_r & & \\ \hline & & -1 & \\ & & & O_{r-1} \end{array}\right).$$

In the first iteration of dual solution recovery, the reducing certificate is A_2, and the "deleted" rows (or columns) are the second row of the first diagonal block and the first row of the third diagonal block. In this iteration, the slack matrix is

$$Z^{\mathrm{r}} - y_2 A_2 = \left(\begin{array}{ccc|cc|ccc} 1 & 0 & -y_2 & & & & & \\ 0 & -y_2 & 0 & & & & & \\ -y_2 & 0 & 0 & & & & & \\ & & & O_{r-2} & & & & \\ \hline & & & & 0 & -y_2 & & \\ & & & & -y_2 & 0 & & \\ & & & & & & O_{r-2} & \\ \hline & & & & & & -1-y_2 & 0 & y_2 \\ & & & & & & 0 & y_2 & 0 \\ & & & & & & y_2 & 0 & 0 \\ & & & & & & & & & O_{r-3} \end{array}\right),$$

and we seek y_2 is such that

$$\left(\begin{array}{cc|c|c} 1 & 0 & & \\ 0 & -y_2 & & \\ \hline & & 0 & \\ \hline & & & -1-y_2 \end{array}\right) \succeq 0,$$

i.e., $y_2 \leq -1$.

In the second iteration of dual solution recovery, the reducing certificate is A_3, and the "deleted" row (column) is the second row of the second diagonal block. The resulting slack variable is shown

as (A.6), where we seek $y_2 \leq -1$ and y_3 such that

$$\left(\begin{array}{cc|cc|c} 1 & 0 & & & \\ 0 & -y_2 & & & \\ \hline & & 0 & -y_2 & \\ & & -y_2 & -y_3 & \\ \hline & & & & -1-y_2 \end{array}\right) \succeq 0.$$

However, it is impossible, since the second block being PSD requires $y_2 = 0$, contradicting with our result $y_2 \leq 1$ from the previous iteration. Thus we have shown that dual solution recovery always fails for "unbound" problems with $r \geq 2$.

By [139, Theorem 1 and Proposition 1], we know that this set of problems indeed have dual objective value 0, but they do not have strong duality when $r \geq 2$, and in this case the optimal objective value of the dual is not attained. Therefore, we were not able to recover the original dual solution since we started with a reduced dual solution with an attained objective value 0.

$$Z^{\mathrm{r}} - y_2 A_2 - y_3 A_3 = \left(\begin{array}{cccc|ccc|ccccc} 0 & 0 & -y_2 & -y_3 & & & & & & & & \\ 0 & -y_2 & -y_3 & 0 & & & & & & & & \\ -y_2 & -y_3 & 0 & 0 & & & & & & & & \\ -y_3 & 0 & 0 & 0 & & & & & & & & \\ & & & & O_{r-3} & & & & & & & \\ \hline & & & & & 0 & -y_2 & -y_3 & & & & \\ & & & & & -y_2 & -y_3 & 0 & & & & \\ & & & & & -y_3 & 0 & 0 & & & & \\ & & & & & & & & O_{r-3} & & & \\ \hline & & & & & & & & -1-y_2 & -y_3 & y_2 & y_3 \\ & & & & & & & & -y_3 & y_2 & y_3 & 0 \\ & & & & & & & & y_2 & y_3 & 0 & 0 \\ & & & & & & & & y_3 & 0 & 0 & 0 \\ & & & & & & & & & & & & O_{r-4} \end{array}\right) \tag{A.6}$$

APPENDIX B

SUPPLEMENTAL CONTENT FOR CHAPTER 3

B.1 A Preliminary Lemma

This appendix provides a lemma with some elementary results, which are useful in our analysis in Chapter 3.

Lemma B.1. *The following statements hold:*

1. *For any $u, v, w \in \mathbb{R}^p$ and $t_1, t_2 \in \mathbb{R}$ with $t_1 + t_2 \neq 0$, it holds that*

$$t_1\|u-w\|^2 + t_2\|v-w\|^2 = (t_1+t_2)\left\|w - \frac{t_1u+t_2v}{t_1+t_2}\right\|^2 + \frac{t_1t_2}{t_1+t_2}\|u-v\|^2.$$

2. *Let $\{u_k\}$ be a nonnegative sequence. If $\sum_{k=1}^{\infty} u_k < \infty$, then $\liminf_{k\to\infty}(k\log k)u_k = 0$.*
3. *Let $\{u_k\}$ and $\{v_k\}$ be two nonnegative sequences and $t_1, t_2 > 0$ be two constants.*
   (a) *If $\liminf_{k\to\infty}(k\log k)(u_k + t_1 k v_k^2) = 0$, then $\liminf_{k\to\infty} k\sqrt{\log k}(u_k + t_2 v_k) = 0$.*
   (b) *If $\liminf_{k\to\infty}(k^2\log k)(u_k + t_1 k^2 v_k^2) = 0$, then $\liminf_{k\to\infty} k^2\sqrt{\log k}(u_k + t_2 v_k) = 0$.*

Proof. 1. It is elementary, and we skip its proof.

2. Since $u_k \geq 0$, and the $\liminf$ of a lower bounded sequence always exists, we set $\bar{u} := \liminf_{k\to\infty}(k\log k)u_k \geq 0$. Assume that $\bar{u} > 0$. Then, by definition, for any $\varepsilon > 0$ such that $\bar{u} - \varepsilon > 0$, there exists integer $k_\varepsilon \geq 2$ such that for any $k \geq k_\varepsilon$ it holds that $(k\log k)u_k \geq \bar{u} - \varepsilon$. This leads to

$$+\infty > \sum_{k=1}^{\infty} u_k \geq \sum_{k=k_\varepsilon}^{\infty} u_k \geq \sum_{k=k_\varepsilon}^{\infty} \frac{\bar{u}-\varepsilon}{k\log k} = (\bar{u}-\varepsilon)\sum_{k=k_\varepsilon}^{\infty}\frac{1}{k\log k} = +\infty,$$

   which is a contradiction. Hence, we must have $\bar{u} = 0$, which proves that $\liminf_{k\to\infty}(k\log k)u_k = 0$.

3. For the first part (a) of item 3, since $\liminf_{k\to\infty}(k\log k)(u_k + t_1 k v_k^2) = 0$, there exists a subsequence $\{(k_j\log k_j)(u_{k_j} + t_1 k_j v_{k_j}^2)\}_{j\geq 0}$ that converges to 0, i.e., for any $\varepsilon > 0$, there exists $j_\varepsilon \geq 0$ such that for any $j \geq j_\varepsilon$, it holds that

$$(k_j \log k_j)(u_{k_j} + t_1 k_j v_{k_j}^2) < \min\left\{\frac{\varepsilon}{2},\ \frac{t_1\varepsilon^2}{4t_2^2}\right\}.$$

Thus, we have

$$\begin{cases} k_j\sqrt{\log k_j}u_{k_j} \le (k_j \log k_j)u_{k_j} < \frac{\varepsilon}{2}, \quad \text{and} \\ t_2 k_j\sqrt{\log k_j}v_{k_j} = \frac{t_2}{\sqrt{t_1}}\sqrt{t_1(k_j^2 \log k_j)v_{k_j}^2} < \frac{t_2}{\sqrt{t_1}}\sqrt{\frac{t_1\varepsilon^2}{4t_2^2}} = \frac{\varepsilon}{2}, \end{cases}$$

which implies that $k_j\sqrt{\log k_j}(u_{k_j} + t_2 v_{k_j}) < \frac{\varepsilon}{2} + \frac{\varepsilon}{2} = \varepsilon$, which proves the first part (a) of item 3. The second part (b) of item 3 can be proved analogously. □

B.2 Supplemental Proofs in Section 3.3: General Convex-Concave Case

This appendix provides the supplemental proofs of technical results in Section 3.3. Firstly, we prove the following claim used in Lemma 3.2 in Subsection 3.3.2 for one-iteration analysis of Algorithm 1.

Lemma B.2. *Let $\{(x^k, \tilde{y}^k)\}$ be generated by scheme (3.27), and $\{s^k\}$ be given by (3.22). Then, for any $(x, s) \in \mathbb{R}^p \times \mathbb{R}^m$, it holds that*

$$\begin{aligned} \mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \le\ & \mathcal{L}_{\rho_k}(x, s, \tilde{y}^k) + \tfrac{1}{\beta_k}\langle x^{k+1} - \hat{x}^k,\ x - x^{k+1}\rangle \\ & - \tfrac{\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2 + \tfrac{L_k + L_f + \rho_k M_g^2}{2}\|x^{k+1} - \hat{x}^k\|^2. \end{aligned} \tag{B.1}$$

Proof. First, the optimality condition of the x^{k+1}-subproblem in the second line of (3.27), which is equivalent to (3.23), can be written as

$$0 = \beta_k \nabla h(x^{k+1}) + \beta_k \nabla f(\hat{x}^k) + \beta_k \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) + x^{k+1} - \hat{x}^k, \tag{B.2}$$

for some $\nabla h(x^{k+1}) \in \partial h(x^{k+1})$. Next, by convexity of h and L_f-smoothness of f, for any $x \in \mathrm{dom}\mathcal{P}$ we have

$$\begin{cases} h(x^{k+1}) \le h(x) + \langle \nabla h(x^{k+1}),\ x^{k+1} - x\rangle, \\ f(x^{k+1}) \le f(x) + \langle \nabla f(\hat{x}^k),\ x^{k+1} - x\rangle + \frac{L_f}{2}\|x^{k+1} - \hat{x}^k\|^2. \end{cases}$$

Combining these two inequalities and then using (B.2) and $F := f + h$, we can derive

$$
\begin{aligned}
F(x^{k+1}) \leq\ & F(x) - \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^{k+1} - x\rangle + \tfrac{L_f}{2}\|x^{k+1} - \hat{x}^k\|^2 \\
& - \tfrac{1}{\beta_k}\langle x^{k+1} - \hat{x}^k,\ x^{k+1} - x\rangle.
\end{aligned} \tag{B.3}
$$

Similarly, by the s^{k+1}-subproblem (3.22) and the convexity of H, we have

$$
H(-s^{k+1}) \leq H(-s) + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ s - s^{k+1}\rangle. \tag{B.4}
$$

Furthermore, combining (3.26) and the definition of L_k in (3.28), we have $L_k = \mathbf{L}_g\left(\tilde{y}^k + \rho_k[g(\hat{x}^k) + s^{k+1}]\right)$. Thus, we can use (3.19) in Lemma 3.1, for any $(x, s) \in \mathbb{R}^p \times \mathbb{R}^m$, to get

$$
\begin{cases}
\phi_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \leq\ \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^{k+1} - \hat{x}^k\rangle \\
\qquad\qquad\qquad\qquad + \frac{\rho_k}{2}\|g(x^{k+1}) - g(\hat{x}^k)\|^2 + \frac{L_k}{2}\|x^{k+1} - \hat{x}^k\|^2, \\
\phi_{\rho_k}(x, s, \tilde{y}^k) \qquad\quad \geq\ \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x - \hat{x}^k\rangle \\
\qquad\qquad\qquad\qquad + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ s - s^{k+1}\rangle + \frac{\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2.
\end{cases} \tag{B.5}
$$

By (3.6), the above two inequalities imply

$$
\begin{aligned}
\phi_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \leq\ & \phi_{\rho_k}(x, s, \tilde{y}^k) + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^{k+1} - x\rangle \\
& + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ s^{k+1} - s\rangle + \tfrac{L_k + \rho_k M_g^2}{2}\|x^{k+1} - \hat{x}^k\|^2 \\
& - \tfrac{\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2.
\end{aligned} \tag{B.6}
$$

Now, combining (B.3)-(B.6), we can derive

$$
\begin{aligned}
\mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) =\ & F(x^{k+1}) + H(-s^{k+1}) + \phi_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \\
\leq\ & F(x) + H(-s) + \phi_{\rho_k}(x, s, \tilde{y}^k) + \tfrac{1}{\beta_k}\langle x^{k+1} - \hat{x}^k,\ x - x^{k+1}\rangle \\
& - \tfrac{\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2 + \tfrac{L_k + L_f + \rho_k M_g^2}{2}\|x^{k+1} - \hat{x}^k\|^2,
\end{aligned}
$$

which proves (B.1). □

The following lemma shows the boundedness of $\{\|\tilde{y}^k - y^\star\|\}$ and $\{\|x^k - x^\star\|\}$. It is used in the proof of Theorem 3.1 in Subsubsection 3.3.2.1 for proving the $\mathcal{O}\left(\frac{1}{k}\right)$ ergodic rate of Algorithm 1.

Lemma B.3. *Let $\{(x^k, \tilde{y}^k)\}$ be generated by Algorithm 1, where the parameters, including ρ and*

C, are set as in (3.43) *and* (3.44)*. Then, for all $k \in \mathbb{N}$, we have*

$$L_g(\|y^\star\| + \|\tilde{y}^k - y^\star\| + \rho M_g \|x^k - x^\star\|) \le \rho C. \tag{B.7}$$

Proof. We prove (B.7) by induction. For $k = 0$, (B.7) holds due to the choice of C in (3.43). Suppose that (B.7) holds for all $k \in \{0, 1, \ldots, K\}$ for some $K \ge 0$, i.e., $L_g(\|y^\star\|+\|\tilde{y}^k-y^\star\|+\rho M_g\|x^k-x^\star\|) \le \rho C$, we now prove that (B.7) also holds for $K+1$. Indeed, using $y^\star = \operatorname{prox}_{\rho H^*}(y^\star + \rho g(x^\star))$ from (3.12), for $0 \le k \le K$ we have

$$\begin{aligned}
L_k &\overset{(3.28)}{=} \mathbf{L}_g(y^{k+1}) \overset{(3.27)}{=} \mathbf{L}_g\left(\operatorname{prox}_{\rho H^*}\left(\tilde{y}^k + \rho g(x^k)\right)\right) \\
&\overset{(3.12)}{=} \mathbf{L}_g\left(\operatorname{prox}_{\rho H^*}\left(\tilde{y}^k + \rho g(x^k)\right) - \operatorname{prox}_{\rho H^*}\left(y^\star + \rho g(x^\star)\right) + y^\star\right) \\
&\le L_g\left[\|\operatorname{prox}_{\rho H^*}\left(\tilde{y}^k + \rho g(x^k)\right) - \operatorname{prox}_{\rho H^*}\left(y^\star + \rho g(x^\star)\right)\| + \|y^\star\|\right] \\
&\le L_g\left(\|[\tilde{y}^k + \rho g(x^k)] - [y^\star + \rho g(x^\star)]\| + \|y^\star\|\right) \\
&\le L_g\left(\|y^\star\| + \|\tilde{y}^k - y^\star\| + \rho M_g\|x^k - x^\star\|\right) \le \rho C,
\end{aligned} \tag{B.8}$$

where in the third line we applied Assumption 3.2(3), in the fourth line we used the non-expansiveness of proximal operators, and the last inequality is due to induction assumption. Now, by definitions of β and η in (3.43), for $0 \le k \le K$, we have

$$\frac{1}{\beta} - L_k - L_f - \frac{\rho^2 M_g^2}{\rho - \eta} \overset{(3.43)(B.8)}{\ge} \frac{\gamma L_f + \rho(\gamma C + M_g^2)}{\gamma} - \rho C - L_f - \frac{\rho^2 M_g^2}{\gamma\rho} = 0. \tag{B.9}$$

Using this estimate, we substitute $\tau_k := 1$ and $\tilde{x}^k := x^k$ into (3.29) of Lemma 3.2 to obtain for any $(x, s, y) \in \mathbb{R}^p \times \mathbb{R}^m \times \mathbb{R}^m$ that

$$\begin{aligned}
\mathcal{L}_\rho(x^{k+1}, s^{k+1}, y) - \mathcal{L}(x, s, y^{k+1}) \le\ & \tfrac{1}{2\beta}\left(\|x^k - x\|^2 - \|x^{k+1} - x\|^2\right) \\
& + \tfrac{1}{2\eta}\left(\|\tilde{y}^k - y\|^2 - \|\tilde{y}^{k+1} - y\|^2\right).
\end{aligned} \tag{B.10}$$

By (3.14), we have $\mathcal{L}_\rho(x^{k+1}, s^{k+1}, y^\star) - \mathcal{L}(x^\star, s^\star, y^k) \ge 0$. Hence,(B.10) implies that

$$\frac{1}{\beta}\|x^{k+1} - x^\star\|^2 + \frac{1}{\eta}\|\tilde{y}^{k+1} - y^\star\|^2 \le \frac{1}{\beta}\|x^k - x^\star\|^2 + \frac{1}{\eta}\|\tilde{y}^k - y^\star\|^2.$$

Since the above inequality holds for all $0 \le k \le K$, we can show that

$$\begin{aligned}\tfrac{1}{\beta}\|x^{K+1}-x^\star\|^2+\tfrac{1}{\eta}\|\tilde{y}^{K+1}-y^\star\|^2 &\le \tfrac{1}{\beta}\|x^{K}-x^\star\|^2+\tfrac{1}{\eta}\|\tilde{y}^{K}-y^\star\|^2\\ &\le \tfrac{1}{\beta}\|x^{0}-x^\star\|^2+\tfrac{1}{\eta}\|y^{0}-y^\star\|^2 \overset{(3.42)}{=} \mathcal{R}_0^2(x^\star,y^\star).\end{aligned}$$

The last inequality leads to $\|x^{K+1}-x^\star\| \le \sqrt{\beta}\mathcal{R}_0(x^\star,y^\star)$ and $\|\tilde{y}^{K+1}-y^\star\| \le \sqrt{\eta}\mathcal{R}_0(x^\star,y^\star)$. Using these bounds and (3.43), we can derive

$$L_g(\|y^\star\|+\|\tilde{y}^{K+1}-y^\star\|+\rho M_g\|x^{K+1}-x^\star\|) \le L_g[\|y^\star\|+(\sqrt{\eta}+\rho\sqrt{\beta}M_g)\mathcal{R}_0(x^\star,y^\star)] \overset{(3.43)}{\le} \rho C.$$

Hence, we prove that (B.7) also holds for $K+1$. By induction, it holds for all $k \in \mathbb{N}$. □

The following proof shows that the parameter initialization (3.49) in Remark 3.1 indeed satisfies the condition (3.43) in Theorem 3.1.

Proof for Remark 3.1. For simplicity, we set $\rho := 1$ and $\gamma := \frac{1}{2}$. Substituting them into the expression of β, η, and $\mathcal{R}_0^2(x^\star,y^\star)$, we get

$$\beta = \frac{1}{L_f+C+2M_g^2}, \quad \eta = \frac{1}{2}, \quad \text{and} \quad \mathcal{R}_0^2(x^\star,y^\star) \le (L_f+C+2M_g^2+2)D^2, \tag{B.11}$$

where $D \ge \max\{\|x^0-x^\star\|,\ \|y^0-y^\star\|,\ \|y^\star\|\}$ is defined in Remark 3.1. Substituting the above expressions for ρ, β, η, γ, and $\mathcal{R}_0^2(x^\star,y^\star)$ into the second line of (3.43), we only need the following inequality in order for (3.43) to hold:

$$\begin{aligned}&1+\sqrt{\tfrac{L_f+C}{2}+M_g^2+1}+M_g\sqrt{1+\tfrac{2}{L_f+C+2M_g^2}}\\ &=1+\left(\tfrac{1}{\sqrt{2}}+\tfrac{M_g}{\sqrt{L_f+C+2M_g^2}}\right)\sqrt{L_f+C+2M_g^2+2} \le \tfrac{C}{L_gD}.\end{aligned} \tag{B.12}$$

Let

$$C \ge L_f+2M_g^2+2, \tag{B.13}$$

then $\frac{L_f+C}{2}+M_g^2+1 \le C$ and $\frac{2}{L_f+C+2M_g^2} \le 3$. Substituting them into the left-hand-side in (B.12), we only need the following inequality in order for (B.12) to hold:

$$1+\sqrt{C}+2M_g \le \frac{C}{L_gD},$$

which can be solved as

$$\sqrt{C} \geq \sqrt{L_g D(L_g D + 4M_g + 2)}, \tag{B.14}$$

where we have used $\frac{t_1+t_2}{2} \leq \sqrt{\frac{t_1^2+t_2^2}{2}}$ to simplify the expression. Combining (B.13) and (B.14), we finally get (3.49). □

The lemma below bounds the term $\left\{\frac{L_k}{\rho_k}\right\}$. It is used in the proof of Theorem 3.2 in Subsubsection 3.3.2.2 for proving the $\mathcal{O}\left(\frac{1}{k}\right)$ semi-ergodic rate of Algorithm 1.

Lemma B.4. *Let $\{\tilde{y}^k\}$ be generated by Algorithm 1, where the parameters, including ρ_k and γ, are defined in* (3.50) *and* (3.51)*. Let B_g and s_* be defined in Theorem 3.2. Then for $k \in \mathbb{N}$,*

$$\frac{\|\tilde{y}^k\|}{\rho_k} \leq \frac{1}{\gamma}\left[\frac{\|y^0\|}{\rho_0} + 2(1-\gamma)(B_g + \|s_*\|)\right]. \tag{B.15}$$

Proof. Since $\mathcal{B}_k \equiv \mathbb{R}^m$, i.e., there is no projection, the $\tilde{y}^{k+1}$-update in Algorithm 1 becomes $\tilde{y}^{k+1} := \tilde{y}^k + \eta_k[\Theta_{k+1} - (1-\tau_k)\Theta_k]$. Thus

$$\tilde{y}^{k+1} - \eta_k\Theta_{k+1} = \tilde{y}^k - (1-\tau_k)\eta_k\Theta_k \overset{(3.51)}{=} \tilde{y}^k - \eta_{k-1}\Theta_k.$$

By induction, for all $k \in \mathbb{N}$, we obtain

$$\tilde{y}^{k+1} - \eta_k\Theta_{k+1} = \tilde{y}^1 - \eta_0\Theta_1 = \tilde{y}^0 - (1-\tau_0)\eta_0\Theta_0 = y^0. \tag{B.16}$$

Next, by definition of s_*, we have $-s_* = \mathrm{prox}_{H/\rho_k}(-s_*)$ for any $\rho_k > 0$. By the update of s^{k+1} in (3.22), the definition of B_g, and the non-expansiveness of prox_{H/ρ_k}, we have

$$\begin{aligned}\|s^{k+1}\| &= \left\|\mathrm{prox}_{H/\rho_k}\left(\tfrac{\tilde{y}^k}{\rho_k} + g(\hat{x}^k)\right) - \mathrm{prox}_{H/\rho_k}(-s_*) - s_*\right\| \\ &\leq \left\|\tfrac{\tilde{y}^k}{\rho_k} + g(\hat{x}^k) + s_*\right\| + \|s_*\| \leq \tfrac{\|\tilde{y}^k\|}{\rho_k} + B_g + 2\|s_*\|.\end{aligned}$$

Furthermore, by the Θ_{k+1}-update in (3.27) and the connection between y^{k+1} and s^{k+1} described by (3.26), we have

$$\|\Theta_{k+1}\| = \|g(x^{k+1}) + s^{k+1}\| \leq B_g + \|s^{k+1}\| \leq 2B_g + 2\|s_*\| + \frac{\|\tilde{y}^k\|}{\rho_k}. \tag{B.17}$$

Now, we can prove (B.15) by induction. For $k = 0$, it is true since $\gamma \in (0, 1)$ and $\tilde{y}^0 = y^0$. Suppose that (B.15) holds for some $K \geq 0$. We prove that it also holds for $K + 1$. Indeed, using (B.16), (B.17), $\rho_{K+1} \geq \rho_0$, and the induction hypothesis, we have

$$\begin{aligned}
\frac{\|\tilde{y}^{K+1}\|}{\rho_{K+1}} &\overset{(B.16)}{=} \frac{1}{\rho_{K+1}}\|y^0 + \eta_K \Theta_{K+1}\| \overset{(3.51)}{\leq} \frac{\|y^0\|}{\rho_0} + (1-\gamma)\|\Theta_{K+1}\| \\
&\overset{(B.17)}{\leq} \frac{\|y^0\|}{\rho_0} + (1-\gamma)\left(2B_g + 2\|s_*\| + \frac{\|\tilde{y}^K\|}{\rho_K}\right) \\
&\leq \frac{\|y^0\|}{\rho_0} + (1-\gamma)\left(2B_g + 2\|s_*\| + \frac{1}{\gamma}\left[\frac{\|y^0\|}{\rho_0} + 2(1-\gamma)(B_g + \|s_*\|)\right]\right) \\
&= \frac{1}{\gamma}\left[\frac{\|y^0\|}{\rho_0} + 2(1-\gamma)(B_g + \|s_*\|)\right].
\end{aligned}$$

This shows that (B.15) also holds for $K + 1$. Therefore, by induction, we conclude that (B.15) holds for all $k \in \mathbb{N}$. □

B.3 Supplemental Proofs in Section 3.4: Strongly Convex-Concave Case

This section provides the supplemental proofs of technical results in Section 3.4. Firstly, we prove the following two lemmas used in Lemma 3.3 in Subsection 3.4.2 for one-iteration analysis of Algorithm 2.

Lemma B.5. *Let $\{(x^k, \tilde{x}^k, \hat{x}^k, \tilde{y}^k)\}$ be generated by (3.64) with $\tau_k \in [0, 1]$, and $\{s^k\}$ be defined in (3.22). Let us define*

$$\breve{x}^{k+1} := (1 - \tau_k)x^k + \tau_k \tilde{x}^{k+1}. \tag{B.18}$$

Then, for any $(x, s) \in \mathbb{R}^p \times \mathbb{R}^m$,

$$\begin{aligned}
F(x^{k+1}) + H(-s^{k+1}) \leq\ & (1-\tau_k)[F(x^k) + H(-s^k)] + \tau_k[F(x) + H(-s)] \\
&+ \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ (1-\tau_k)x^k + \tau_k x - x^{k+1}\rangle \\
&+ \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ (1-\tau_k)s^k + \tau_k s - s^{k+1}\rangle \\
&+ \tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^k - x\|^2 - \tfrac{\tau_k(\tau_k + \beta_k\mu_h)}{2\beta_k}\|\tilde{x}^{k+1} - x\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\alpha_k} - L_f\right)\|x^{k+1} - \hat{x}^k\|^2 \\
&- \tfrac{1}{2}\left(\tfrac{1}{\alpha_k} + \mu_h\right)\|\breve{x}^{k+1} - x^{k+1}\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\beta_k} - \tfrac{1}{\alpha_k}\right)\|\breve{x}^{k+1} - \hat{x}^k\|^2.
\end{aligned} \tag{B.19}$$

Proof. Firstly, from the optimality condition of the $\tilde{x}^{k+1}$-subproblem in the second line of (3.64), there exists $\nabla h(\tilde{x}^{k+1}) \in \partial h(\tilde{x}^{k+1})$ such that

$$\nabla h(\tilde{x}^{k+1}) = -\frac{\tau_k}{\beta_k}(\tilde{x}^{k+1} - \tilde{x}^k) - [\nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k)],$$

where we have used the expression of $\nabla_x \phi$ in (3.17). Combining this expression and (B.18), and using the μ_h-strong convexity of h, we can derive

$$\begin{aligned} h(\breve{x}^{k+1}) \leq\ & (1-\tau_k)h(x^k) + \tau_k h(x) + \tau_k \langle \nabla h(\tilde{x}^{k+1}),\ \tilde{x}^{k+1} - x\rangle \\ & - \tfrac{\tau_k \mu_h}{2}\|\tilde{x}^{k+1} - x\|^2 - \tfrac{\tau_k(1-\tau_k)\mu_h}{2}\|\tilde{x}^{k+1} - x^k\|^2 \\ =\ & (1-\tau_k)h(x^k) + \tau_k h(x) - \tau_k \langle \nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ \tilde{x}^{k+1} - x\rangle \\ & - \tfrac{\tau_k^2}{\beta_k}\langle \tilde{x}^{k+1} - \tilde{x}^k,\ \tilde{x}^{k+1} - x\rangle - \tfrac{\tau_k \mu_h}{2}\|\tilde{x}^{k+1} - x\|^2 - \tfrac{\tau_k(1-\tau_k)\mu_h}{2}\|\tilde{x}^{k+1} - x^k\|^2. \end{aligned} \tag{B.20}$$

Next, by the x^{k+1}-subproblem in the third line of (3.64) and the μ_h-strong convexity of h, we can show that

$$\begin{aligned} & h(x^{k+1}) + \tfrac{1}{2\alpha_k}\|x^{k+1} - \hat{x}^k\|^2 + \langle \nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^{k+1} - \hat{x}^k\rangle \\ & \leq h(\breve{x}^{k+1}) + \tfrac{1}{2\alpha_k}\|\breve{x}^{k+1} - \hat{x}^k\|^2 \\ & \quad + \langle \nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ \breve{x}^{k+1} - \hat{x}^k\rangle - \left(\tfrac{1}{2\alpha_k} + \tfrac{\mu_h}{2}\right)\|\breve{x}^{k+1} - x^{k+1}\|^2. \end{aligned} \tag{B.21}$$

Combining (B.18), (B.20), and (B.21), and using $\breve{x}^{k+1} - \hat{x}^k = \tau_k(\tilde{x}^{k+1} - \tilde{x}^k)$, we further derive

$$\begin{aligned} h(x^{k+1}) \overset{\text{(B.21)}}{\leq}\ & h(\breve{x}^{k+1}) + \langle \nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ \breve{x}^{k+1} - x^{k+1}\rangle \\ & + \tfrac{1}{2\alpha_k}(\|\breve{x}^{k+1} - \hat{x}^k\|^2 - \|x^{k+1} - \hat{x}^k\|^2 - \|\breve{x}^{k+1} - x^{k+1}\|^2) - \tfrac{\mu_h}{2}\|\breve{x}^{k+1} - x^{k+1}\|^2 \\ \overset{\text{(B.20)}}{\leq}\ & (1-\tau_k)h(x^k) + \tau_k h(x) - \tfrac{\tau_k^2}{\beta_k}\langle \tilde{x}^{k+1} - \tilde{x}^k,\ \tilde{x}^{k+1} - x\rangle - \tfrac{\tau_k \mu_h}{2}\|\tilde{x}^{k+1} - x\|^2 \\ & + \langle \nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ \breve{x}^{k+1} - x^{k+1} - \tau_k(\tilde{x}^{k+1} - x)\rangle \\ & + \tfrac{1}{2\alpha_k}(\|\breve{x}^{k+1} - \hat{x}^k\|^2 - \|x^{k+1} - \hat{x}^k\|^2 - \|\breve{x}^{k+1} - x^{k+1}\|^2) \\ & - \tfrac{\tau_k(1-\tau_k)\mu_h}{2}\|\tilde{x}^{k+1} - x^k\|^2 - \tfrac{\mu_h}{2}\|\breve{x}^{k+1} - x^{k+1}\|^2 \\ \overset{\text{(B.18)}}{\leq}\ & (1-\tau_k)h(x^k) + \tau_k h(x) + \tfrac{\tau_k^2}{2\beta_k}\|\tilde{x}^k - x\|^2 - \tfrac{\tau_k(\tau_k + \beta_k \mu_h)}{2\beta_k}\|\tilde{x}^{k+1} - x\|^2 \\ & + \langle \nabla f(\hat{x}^k) + \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ (1-\tau_k)x^k + \tau_k x - x^{k+1}\rangle \\ & - \tfrac{1}{2}\left(\tfrac{1}{\beta_k} - \tfrac{1}{\alpha_k}\right)\|\breve{x}^{k+1} - \hat{x}^k\|^2 - \tfrac{1}{2\alpha_k}\|x^{k+1} - \hat{x}^k\|^2 - \tfrac{1}{2}\left(\tfrac{1}{\alpha_k} + \mu_h\right)\|\breve{x}^{k+1} - x^{k+1}\|^2. \end{aligned} \tag{B.22}$$

On the other hand, by the L_f-smoothness and the convexity of f, one can show that

$$\begin{aligned} f(x^{k+1}) \leq\ & f(\hat{x}^k) + \langle \nabla f(\hat{x}^k),\ x^{k+1} - \hat{x}^k\rangle + \tfrac{L_f}{2}\|x^{k+1} - \hat{x}^k\|^2 \\ \leq\ & (1-\tau_k) f(x^k) + \tau_k f(x) + \langle \nabla f(\hat{x}^k),\ x^{k+1} - (1-\tau_k)x^k - \tau_k x\rangle \\ & - \tfrac{(1-\tau_k)\tau_k \mu_f}{2}\|x^k - x\|^2 + \tfrac{L_f}{2}\|x^{k+1} - \hat{x}^k\|^2. \end{aligned} \tag{B.23}$$

Moreover, by the s^{k+1}-subproblem in (3.22), we get exactly (B.4) again, which implies

$$\begin{aligned} H(-s^{k+1}) \leq\ & (1-\tau_k) H(-s^k) + \tau_k H(-s) \\ & + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ (1-\tau_k)s^k + \tau_k s - s^{k+1}\rangle. \end{aligned} \tag{B.24}$$

Finally, combining (B.22)-(B.24), we obtain (B.19). □

Lemma B.6. *Let $\{(x^k, y^k)\}$ be generated by (3.64) with $\tau_k \in [0,1]$, and $\{s^k\}$ be defined in (3.22), then, for any $(x,s) \in \mathbb{R}^p \times \mathbb{R}^m$, we have*

$$\begin{aligned} \phi_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \leq\ & (1-\tau_k)\phi_{\rho_k}(x^k, s^k, \tilde{y}^k) + \tau_k \phi_{\rho_k}(x, s, \tilde{y}^k) + \tfrac{L_k + \rho_k M_g^2}{2}\|x^{k+1} - \hat{x}^k\|^2 \\ & + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^{k+1} - (1-\tau_k)x^k - \tau_k x\rangle \\ & + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ s^{k+1} - (1-\tau_k)s^k - \tau_k s\rangle \\ & - \tfrac{(1-\tau_k)\rho_k}{2}\|[g(x^k) + s^k] - [g(\hat{x}^k) + s^{k+1}]\|^2 \\ & - \tfrac{\tau_k \rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2. \end{aligned} \tag{B.25}$$

Proof. Combining (3.26) and the definition of L_k in (3.28), we have $L_k = \mathbf{L}_g\left(\tilde{y}^k + \rho_k[g(\hat{x}^k) + s^k]\right)$. Thus we can use (3.19) in Lemma 3.1 and the $\mathbf{M}_g$-Lipschitz continuity of g to get

$$\begin{cases} \phi_{\rho_k}(x^{k+1}, s^{k+1}, \tilde{y}^k) \leq\ \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^{k+1} - \hat{x}^k\rangle \\ \qquad\qquad + \frac{L_k + \rho_k M_g^2}{2}\|x^{k+1} - \hat{x}^k\|^2, \\ \phi_{\rho_k}(x, s, \tilde{y}^k) \geq\ \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x - \hat{x}^k\rangle \\ \qquad\qquad + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k), s - s^{k+1}\rangle + \frac{\rho_k}{2}\|[g(x) + s] - [g(\hat{x}^k) + s^{k+1}]\|^2. \end{cases} \tag{B.26}$$

Letting $(x,s) := (x^k, s^k)$ in the second inequality of (B.26), we get

$$\begin{aligned} \phi_{\rho_k}(x^k, s^k, \tilde{y}^k) \geq\ & \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k) + \langle \nabla_x \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ x^k - \hat{x}^k\rangle \\ & + \langle \nabla_s \phi_{\rho_k}(\hat{x}^k, s^{k+1}, \tilde{y}^k),\ s^k - s^{k+1}\rangle \\ & + \tfrac{\rho_k}{2}\|[g(x^k) + s^k] - [g(\hat{x}^k) + s^{k+1}]\|^2. \end{aligned} \tag{B.27}$$

Multiplying the second inequality of (B.26) by τ_k, multiplying (B.27) by $1-\tau_k$, and then adding them to the first inequality of (B.26), we arrive at (B.25). □

To prove Theorem 3.4 for deriving the $\mathcal{O}\left(\frac{1}{k^2}\right)$ ergodic rate of Algorithm 2, we need the following two technical lemmas.

Lemma B.7. *Let ρ_k, β_k, and η_k be defined by* (3.72) *and* (3.73) *of Theorem 3.4. Then, for all $k \in \mathbb{N}$, we have*

$$\begin{cases} \beta_k \leq \frac{\Gamma}{L_f+\rho_k\hat{M}^2}, \quad \rho_k \geq \rho_0 + (\sqrt{1+\mu_h\beta_0}-1)\rho_0 k, \\ \frac{1+\mu_h\beta_k}{\beta_k} = \frac{1}{\theta_{k+1}\beta_{k+1}}, \quad \frac{1}{\rho_k} = \frac{1}{\theta_{k+1}\rho_{k+1}}, \quad \textit{and} \quad \frac{1}{\eta_k} = \frac{1}{\theta_{k+1}\eta_{k+1}}. \end{cases} \tag{B.28}$$

Proof. We prove the first inequality in (B.28) by induction. First, it holds with equality for $k=0$ due to the definition of β_0. Furthermore, if it holds for $k \geq 0$, then

$$\beta_{k+1} \overset{(3.73)}{=} \theta_{k+1}\beta_k \leq \frac{\theta_{k+1}\Gamma}{L_f+\rho_k\hat{M}^2} = \frac{\Gamma}{\frac{L_f}{\theta_{k+1}}+\frac{\rho_k}{\theta_{k+1}}\hat{M}^2} \overset{(3.73)}{\leq} \frac{\Gamma}{L_f+\rho_{k+1}\hat{M}^2}.$$

Thus the first inequality of (B.28) is also true for $k+1$. By induction, the first inequality in (B.28) holds for all $k \in \mathbb{N}$.

To prove the second inequality of the first line in (B.28), we notice that (3.73) implies

$$\begin{aligned} \rho_{k+1} &\overset{(3.73)}{=} \rho_k\sqrt{1+\mu_h\beta_k} = \rho_k\left(1+\tfrac{\mu_h\beta_k}{1+\sqrt{1+\mu_h\beta_k}}\right) \overset{(3.73)}{=} \rho_k + \tfrac{\mu_h\beta_0\rho_0}{1+\sqrt{1+\mu_h\beta_k}} \\ &\overset{\beta_k\leq\beta_0}{\geq} \rho_k + \tfrac{\mu_h\beta_0\rho_0}{1+\sqrt{1+\mu_h\beta_0}} = \rho_k + (\sqrt{1+\mu_h\beta_0}-1)\rho_0. \end{aligned}$$

The desired inequality is then achieved via induction.

The first statement of the second line in (B.28) holds since

$$\frac{1+\mu_h\beta_k}{\beta_k} \overset{(3.73)}{=} \frac{1}{\theta_{k+1}^2\beta_k} \overset{(3.73)}{=} \frac{1}{\theta_{k+1}\beta_{k+1}}.$$

The last two equations of (B.28) directly follow from the update of η_k and ρ_k in (3.73). □

Lemma B.8. *Let ρ_k be defined by* (3.72) *and* (3.73) *of Theorem 3.4. Then, for all $k \in \mathbb{N}$,*

$$L_g(\|y^\star\| + \|\tilde{y}^k - y^\star\| + \rho_k M_g\|\tilde{x}^k - x^\star\|) \leq \rho_k\left[(2-\Gamma)\hat{M}^2 - \frac{M_g^2}{\gamma}\right]. \tag{B.29}$$

Proof. We prove (B.29) by induction. For $k = 0$, the inequality (B.29) holds due to the second line in (3.72). Suppose (B.29) holds for all $k \in \{0, 1, \ldots, K\}$ for some $K \in \mathbb{N}$. Then by the definition of L_k in Lemma 3.3 and the same lines as (B.8), for all $k \in \{0, 1, \ldots, K\}$, we can show that

$$L_k \overset{\text{(B.8)}}{\leq} L_g(\|y^\star\| + \|\tilde{y}^k - y^\star\| + \rho_k M_g \|\tilde{x}^k - x^\star\|) \leq \rho_k \left[(2-\Gamma)\hat{M}^2 - \frac{M_g^2}{\gamma}\right]. \tag{B.30}$$

By the update of α_k, η_k and β_k in (3.73), the first inequality of (B.28), and (B.30), we have

$$\begin{aligned} &\tfrac{1}{\alpha_k}\left(1 - \tfrac{\beta_k}{\alpha_k}\right) + \tfrac{1}{\alpha_k} - L_k - L_f - \tfrac{\rho_k^2 M_g^2}{\rho_k - \eta_k} \\ \overset{\text{(3.73)(B.28)}}{\geq}\ & (1-\Gamma)(L_f + \rho_k \hat{M}^2) + \rho_k \hat{M}^2 - L_k - \tfrac{\rho_k M_g^2}{\gamma} \\ \overset{\text{(B.30)}}{\geq}\ & (2-\Gamma)\rho_k \hat{M}^2 - \tfrac{\rho_k M_g^2}{\gamma} - \rho_k \left[(2-\Gamma)\hat{M}^2 - \tfrac{M_g^2}{\gamma}\right] = 0. \end{aligned} \tag{B.31}$$

Using this inequality, $\tau_k := 1$, and $(x, s, y) := (x^\star, s^\star, y^\star)$ into (3.65) of Lemma 3.3, we get

$$\begin{aligned} 0 \quad \leq\quad & \mathcal{L}_{\rho_k}(x^{k+1}, s^{k+1}, y^\star) - \mathcal{P}^\star \\ \overset{\text{(3.65)(B.31)}}{\leq}\ & \tfrac{1}{2\beta_k}\|\tilde{x}^k - x^\star\|^2 - \tfrac{1+\beta_k\mu_h}{2\beta_k}\|\tilde{x}^{k+1} - x^\star\|^2 \\ & + \tfrac{1}{2\eta_k}(\|\tilde{y}^k - y^\star\|^2 - \|\tilde{y}^{k+1} - y^\star\|^2) \\ \overset{\text{(B.28)}}{=}\ & \left(\tfrac{1}{2\beta_k}\|\tilde{x}^k - x^\star\|^2 + \tfrac{1}{2\eta_k}\|\tilde{y}^k - y^\star\|^2\right) \\ & - \tfrac{1}{\theta_{k+1}}\left(\tfrac{1}{2\beta_{k+1}}\|\tilde{x}^{k+1} - x^\star\|^2 + \tfrac{1}{2\eta_{k+1}}\|\tilde{y}^{k+1} - y^\star\|^2\right). \end{aligned} \tag{B.32}$$

Multiplying (B.32) by $2\rho_k$, and noticing that $\rho_k \leq \frac{\rho_k}{\theta_{k+1}} = \rho_{k+1}$, we get

$$\begin{aligned} \rho_k\left(\tfrac{1}{\beta_{k+1}}\|\tilde{x}^{k+1} - x^\star\|^2 + \tfrac{1}{\eta_{k+1}}\|\tilde{y}^{k+1} - y^\star\|^2\right) &\leq \rho_{k+1}\left(\tfrac{1}{\beta_{k+1}}\|\tilde{x}^{k+1} - x^\star\|^2 + \tfrac{1}{\eta_{k+1}}\|\tilde{y}^{k+1} - y^\star\|^2\right) \\ &\leq \rho_k\left(\tfrac{1}{\beta_k}\|\tilde{x}^k - x^\star\|^2 + \tfrac{1}{\eta_k}\|\tilde{y}^k - y^\star\|^2\right). \end{aligned}$$

By induction, the above holds for $k \in \{0, 1, \ldots, K\}$. Consequently, one has

$$\frac{1}{\beta_{k+1}}\|\tilde{x}^{k+1} - x^\star\|^2 + \frac{1}{\eta_{k+1}}\|\tilde{y}^{k+1} - y^\star\|^2 \leq \frac{1}{\beta_0}\|x^0 - x^\star\|^2 + \frac{1}{\eta_0}\|y^0 - y^\star\|^2 = \mathcal{R}_0^2(x^\star, y^\star),$$

which implies that $\|\tilde{x}^{k+1} - x^\star\| \leq \sqrt{\beta_{k+1}}\mathcal{R}_0(x^\star, y^\star)$ and $\|\tilde{y}^{k+1} - y^\star\| \leq \sqrt{\eta_{k+1}}\mathcal{R}_0(x^\star, y^\star)$. Finally, using the above estimates, we can easily deduce

$$
\begin{aligned}
&\tfrac{1}{\rho_{K+1}}\left(\|y^\star\| + \|\tilde{y}^{K+1} - y^\star\| + \rho_{K+1} M_g \|\tilde{x}^{K+1} - x^\star\|\right) \\
&\leq \tfrac{1}{\rho_{K+1}}\|y^\star\| + \left(\tfrac{\sqrt{\eta_{K+1}}}{\rho_{K+1}} + M_g\sqrt{\beta_{K+1}}\right)\mathcal{R}_0(x^\star, y^\star) \\
&\overset{(3.73)}{\leq} \tfrac{1}{\rho_0}\|y^\star\| + \left(\tfrac{\sqrt{\eta_0}}{\rho_0} + M_g\sqrt{\beta_0}\right)\mathcal{R}_0(x^\star, y^\star) \overset{(3.72)}{\leq} \tfrac{1}{L_g}\left[(2-\Gamma)\hat{M}^2 - \tfrac{M_g^2}{\gamma}\right].
\end{aligned}
$$

This inequality shows that (B.29) also holds for $K+1$. By induction, we have thus proved that (B.29) holds for all $k \in \mathbb{N}$. □

The following proof shows that the parameter initialization (3.78) in Remark 3.5 indeed satisfies the condition (3.72) in Theorem 3.4.

Proof for Remark 3.5. For simplicity, we set $\rho_0 := 1$, and $\gamma := \Gamma := \frac{1}{2}$. Using the same lines as (B.11) and (B.12) in the proof for Remark 3.1, it is clear that we only need the following inequality for (3.72) to hold:

$$
L_g D\left(1 + \sqrt{L_f + \hat{M}^2 + 1} + M_g\sqrt{1 + \frac{1}{L_f + \hat{M}^2}}\right) \leq \frac{3\hat{M}^2}{2} - 2M_g^2. \tag{B.33}
$$

Let

$$
\hat{M}^2 \geq L_f + 1, \tag{B.34}
$$

then $\sqrt{L_f + \hat{M}^2 + 1} \leq \sqrt{2}\hat{M}$ and $\frac{1}{L_f + \hat{M}^2} \leq 1$. Substituting these into (B.33), we can see that (B.33) holds if

$$
L_g D(1 + \sqrt{2}\hat{M} + \sqrt{2}M_g) \leq \frac{3\hat{M}^2}{2} - 2M_g^2,
$$

which can be solved as

$$
\hat{M} \geq \frac{2}{3}\sqrt{2L_g^2 D^2 + 3(2M_g^2 + L_g D + \sqrt{2}L_g D M_g)}. \tag{B.35}
$$

Combining (B.34) and (B.35), we finally get (3.78). □

BIBLIOGRAPHY

[1] Ravindra K Ahuja, Thomas L Magnanti, and James B Orlin. *Network Flows*. Cambridge, Mass.: Alfred P. Sloan School of Management, Massachusetts, 1988. 5

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 214–223, 2017. 2

[3] Hedy Attouch and Juan Peypouquet. The rate of convergence of Nesterov's accelerated forward-backward method is actually faster than $1/k^2$. *SIAM Journal on Optimization*, 26(3):1824–1834, 2016. 37

[4] Victor Baston. Extreme copositive quadratic forms. *Acta Arithmetica*, 15(3):319–327, 1969. 17

[5] Heinz H Bauschke and Patrick L Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, volume 408. Springer, 2011. 3, 5

[6] Amir Beck, Aharon Ben-Tal, Nili Guttmann-Beck, and Luba Tetruashvili. The CoMirror algorithm for solving nonsmooth constrained convex problems. *Operations Research Letters*, 38(6):493–498, 2010. 40

[7] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*, volume 28. Princeton University Press, 2009. 2

[8] Dimitri P Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Academic press, 2014. 5, 7, 37

[9] Dimitris Bertsimas and John N Tsitsiklis. *Introduction to Linear Optimization*, volume 6. Athena Scientific Belmont, MA, 1997. 5

[10] Digvijay Boob, Qi Deng, and Guanghui Lan. Proximal point methods for optimization with nonconvex functional constraints. *arXiv preprint arXiv:1908.02734*, 2019. 2

[11] Jonathan M Borwein and Henry Wolkowicz. Facial reduction for a cone-convex programming problem. *Journal of the Australian Mathematical Society*, 30(3):369–380, 1981. 5, 14

[12] Jonathan M Borwein and Henry Wolkowicz. Regularizing the abstract convex program. *Journal of Mathematical Analysis and Applications*, 83(2):495–530, 1981. 5, 11, 14

[13] Radu Ioan Boţ, Ernö Robert Csetnek, André Heinrich, and Christopher Hendrich. On the convergence rate improvement of a primal-dual splitting algorithm for solving monotone inclusion problems. *Mathematical Programming*, 150(2):251–279, 2015. 41

[14] Stephen Boyd, Mark T Mueller, Brendan O'Donoghue, and Yang Wang. Performance bounds and suboptimal policies for multi-period investment. *Foundations and Trends® in Optimization*, 1(1):1–72, 2014. vii, 17, 26

[15] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011. 6, 83

[16] Stephen Boyd and Lieven Vandenberghe. Semidefinite programming relaxations of non-convex problems in control and combinatorial optimization. In *Communications, Computation, Control, and Signal Processing*, pages 279–287. Springer, 1997. 4

[17] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge university press, 2004. 5

[18] Sam Burton, Cynthia Vinzant, and Yewon Youm. A real stable extension of the vamos matroid polynomial. *arXiv preprint arXiv:1411.2038*, 2014. 17

[19] Antonin Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1-2):89–97, 2004. 83, 85, 87

[20] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011. xiii, 3, 6, 41, 46, 83, 84, 88

[21] Antonin Chambolle and Thomas Pock. On the ergodic convergence rates of a first-order primal–dual algorithm. *Mathematical Programming*, 159(1-2):253–287, 2016. 6, 83

[22] Stanley H Chan, Ramsin Khoshabeh, Kristofor B Gibson, Philip E Gill, and Truong Q Nguyen. An augmented Lagrangian method for total variation video restoration. *IEEE Transactions on Image Processing*, 20(11):3097–3111, 2011. 84, 87

[23] Yunmei Chen, Guanghui Lan, and Yuyuan Ouyang. Accelerated schemes for a class of variational inequalities. *Mathematical Programming*, 165(1):113–149, 2017. 77

[24] Yuen-Lam Cheung, Simon Schurr, and Henry Wolkowicz. Preprocessing and regularization for degenerate semidefinite programs. In *Computational and Analytical Mathematics*, pages 251–303. Springer, 2013. vii, 17, 22, 25

[25] Patrick L Combettes and Jean-Christophe Pesquet. Primal-dual splitting algorithm for solving inclusions with mixtures of composite, Lipschitzian, and parallel-sum type monotone operators. *Set-Valued and Variational Analysis*, 20(2):307–330, 2012. 6

[26] Patrick L Combettes and Valérie R Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005. 3

[27] Laurent Condat. A primal–dual splitting method for convex optimization involving lipschitzian, proximable and linear composite terms. *Journal of Optimization Theory and Applications*, 158(2):460–479, 2013. 6

[28] Damek Davis. Convergence rate analysis of primal-dual splitting schemes. *SIAM Journal on Optimization*, 25(3):1912–1943, 2015. 6, 41, 91

[29] Damek Davis. Convergence rate analysis of the forward-Douglas-Rachford splitting scheme. *SIAM Journal on Optimization*, 25(3):1760–1786, 2015.

[30] Damek Davis and Wotao Yin. Convergence rate analysis of several splitting schemes. In *Splitting Methods in Communication, Imaging, Science, and Engineering*, pages 115–163. Springer, 2016. 91

[31] Palahenedi Hewage Diananda. On non-negative forms in real variables some or all of which are non-negative. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 58, pages 17–25. Cambridge University Press, 1962. 17

[32] Jim Douglas and Henry H Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American Mathematical Society*, 82(2):421–439, 1956. 6

[33] Mareike Dressler, Sadik Iliman, and Timo de Wolff. An approach to constrained polynomial optimization via nonnegative circuit polynomials and geometric programming. *Journal of Symbolic Computation*, 91:149–172, 2019. x, xiii, 17, 27, 100

[34] Dmitriy Drusvyatskiy, Nathan Krislock, Yuen-Lam Voronin, and Henry Wolkowicz. Noisy Euclidean distance realization: Robust facial reduction and the Pareto frontier. *SIAM Journal on Optimization*, 27(4):2301–2331, 2017. 5, 11, 14

[35] Dmitriy Drusvyatskiy, Gábor Pataki, and Henry Wolkowicz. Coordinate shadows of semidefinite and Euclidean distance matrices. *SIAM Journal on Optimization*, 25(2):1160–1178, 2015. 5, 11, 14

[36] Celestine Dünner, Simone Forte, Martin Takác, and Martin Jaggi. Primal-dual rates and certificates. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 783–792, 2016. 3

[37] Jonathan Eckstein. Parallel alternating direction multiplier decomposition of convex programs. *Journal of Optimization Theory and Applications*, 80(1):39–62, 1994. 6

[38] John Ernest Esser. *Primal Dual Algorithms for Convex Models and Applications to Image Restoration, Registration and Nonlocal Inpainting.* University of California, Los Angeles, 2010. 3, 6, 46

[39] John Ernest Esser, Xiaoqun Zhang, and Tony F Chan. A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science. *SIAM Journal on Imaging Sciences*, 3(4):1015–1046, 2010. 6

[40] Francisco Facchinei and Jong-Shi Pang. *Finite-Dimensional Variational Inequalities and Complementarity Problems.* Springer Science & Business Media, 2007. 2

[41] Farzan Farnia and David Tse. A convex duality framework for GANs. In *Advances in Neural Information Processing Systems 31 (NIPS 2018)*, pages 5248–5258, 2018. 2

[42] Hamza Fawzi and Pablo Parrilo. Self-scaled bounds for atomic cone ranks: applications to nonnegative rank and cp-rank. *Mathematical Programming*, 158(1-2):417–465, 2016. 17

[43] Olivier Fercoq and Zheng Qu. Restarting accelerated gradient methods with a rough strong convexity estimate. *arXiv preprint arXiv:1609.07358*, 2016. 74

[44] Anthony V Fiacco and Garth P McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* SIAM, 1990. 5

[45] Roger Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, 2013. 5

[46] Henrik A Friberg. Facial reduction heuristics and the motivational example of mixed-integer conic optimization. *Optimization Online*, 2016. 5, 14

[47] Katsuki Fujisawa, Mituhiro Fukuda, Kazuhiro Kobayashi, Masakazu Kojima, Kazuhide Nakata, Maho Nakata, and Makoto Yamashita. SDPA (SemiDefinite Programming Algorithm) and SDPA-GMP user's manual - version 7.1.1. Technical report, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2008. 4, 11, 14, 24, 29

[48] Katsuki Fujisawa, Masakazu Kojima, Kazuhide Nakata, and Makoto Yamashita. SDPA (SemiDefinite Programming Algorithm) users manual - version 6.2.0. Technical report, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2002. 4, 11

[49] Pontus Giselsson, Minh Dang Doan, Tamás Keviczky, Bart De Schutter, and Anders Rantzer. Accelerated gradient methods and dual decomposition in distributed model predictive control. *Automatica*, 49(3):829–833, 2013. 6

[50] Roland Glowinski, Stanley J Osher, and Wotao Yin. *Splitting Methods in Communication, Imaging, Science, and Engineering*. Springer, 2017. 3, 6

[51] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the Association for Computing Machinery*, 42(6):1115–1145, 1995. 4

[52] Tom Goldstein, Min Li, and Xiaoming Yuan. Adaptive primal-dual splitting methods for statistical learning and image processing. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, pages 2089–2097, 2015. 3, 6

[53] Tom Goldstein, Brendan O'Donoghue, Simon Setzer, and Richard Baraniuk. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3):1588–1623, 2014. 6

[54] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pages 2672–2680, 2014. 2

[55] Michael Grant and Stephen Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008. `http://stanford.edu/~boyd/graph_dcp.html`. 76

[56] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, 2014. 76

[57] Erfan Yazdandoost Hamedani and Necdet Serhat Aybat. A primal-dual algorithm for general convex-concave saddle point problems. *arXiv preprint arXiv:1803.01401*, 2019. xiii, 2, 6, 7, 36, 40, 47, 78

[58] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2009. 3

[59] Bingsheng He and Xiaoming Yuan. Convergence analysis of primal-dual algorithms for a saddle-point problem: from contraction perspective. *SIAM Journal on Imaging Sciences*, 5(1):119–149, 2012. 6

[60] Yunlong He and Renato DC Monteiro. An accelerated HPE-type algorithm for a class of composite convex-concave saddle-point problems. *SIAM Journal on Optimization*, 26(1):29–56, 2016. 6

[61] Didier Henrion and Jean-Bernard Lasserre. Detecting global optimality and extracting solutions in GloptiPoly. In *Positive Polynomials in Control*, pages 293–310. Springer, 2005. 28

[62] Didier Henrion, Jean-Bernard Lasserre, and Johan Löfberg. GloptiPoly 3: Moments, optimization and semidefinite programming. *Optimization Methods & Software*, 24(4-5):761–779, 2009. 27, 100

[63] Didier Henrion, Simone Naldi, and Mohab Safey El Din. Exact algorithms for linear matrix inequalities. *SIAM Journal on Optimization*, 26(4):2512–2539, 2016. 14, 23

[64] Le Thi Khanh Hien, Renbo Zhao, and William B Haskell. An inexact primal-dual smoothing framework for large-scale non-bilinear saddle point problems. *arXiv preprint arXiv:1711.03669*, 2020. 6, 7, 36, 40, 47

[65] Seung-Jean Kim and Stephen Boyd. A minimax theorem with applications to machine learning, signal processing, and finance. *SIAM Journal on Optimization*, 19(3):1344–1367, 2008. 2

[66] Michal Kočvara and Michael Stingl. PENNON: A code for convex nonlinear and semidefinite programming. *Optimization Methods & Software*, 18(3):317–333, 2003. 4, 11

[67] Nathan Krislock and Henry Wolkowicz. Explicit sensor network localization using semidefinite representations and facial reductions. *SIAM Journal on Optimization*, 20(5):2679–2708, 2010. 5, 11, 14

[68] Guanghui Lan, Zhaosong Lu, and Renato DC Monteiro. Primal-dual first-order methods with $\mathcal{O}(1/\epsilon)$ iteration-complexity for cone programming. *Mathematical Programming*, 126(1):1–29, 2011. 7

[69] Guanghui Lan and Renato DC Monteiro. Iteration-complexity of first-order penalty methods for convex programming. *Mathematical Programming*, 138(1-2):115–139, 2013. 6

[70] Guanghui Lan and Renato DC Monteiro. Iteration-complexity of first-order augmented Lagrangian methods for convex programming. *Mathematical Programming*, 155(1-2):511–547, 2016.

[71] Guanghui Lan and Zhiqiang Zhou. Algorithms for stochastic optimization with functional or expectation constraints. *arXiv preprint arXiv:1604.03887*, 2019. 6, 40

[72] Gert RG Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5(Jan):27–72, 2004. 2, 4

[73] Jean-Bernard Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001. 4, 27

[74] Jean-Bernard Lasserre and Victor Magron. In SDP relaxations, inaccurate solvers do robust optimization. *SIAM Journal on Optimization*, 29(3):2128–2145, 2019. 28

[75] Huan Li and Zhouchen Lin. Accelerated alternating direction method of multipliers: An optimal $O(1/K)$ nonergodic analysis. *Journal of Scientific Computing*, pages 1–29, 2016. 53

[76] Zichong Li and Yangyang Xu. Augmented Lagrangian based first-order methods for convex and nonconvex programs: nonergodic convergence and iteration complexity. *arXiv preprint arXiv:2003.08880*, 2020. 7

[77] Qihang Lin, Runchao Ma, and Tianbao Yang. Level-set methods for finite-sum constrained convex optimization. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 3112–3121, 2018.

[78] Qihang Lin, Selvaprabu Nadarajah, and Negar Soheili. A level-set method for convex optimization with a feasible solution path. *SIAM Journal on Optimization*, 28(4):3290–3311, 2018. 7

[79] Tianyi Lin, Chi Jin, and Michael Jordan. Near-optimal algorithms for minimax optimization. *arXiv preprint arXiv:2002.02417*, 2020. 7, 36, 40

[80] Pierre-Louis Lions and Bertrand Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16(6):964–979, 1979. 6

[81] Minghui Liu and Gábor Pataki. Exact duals and short certificates of infeasibility and weak infeasibility in conic linear programming. *Mathematical Programming*, 167(2):435–480, 2018. 14, 23, 24

[82] László Lovász. Semidefinite programs and combinatorial optimization. In *Recent Advances in Algorithms and Combinatorics*, pages 137–194. Springer, 2003. 4

[83] Juan C Moreno, VB Surya Prasath, and Joao C Neves. Color image processing by vectorial total variation with gradient channels coupling. *Inverse Problems & Imaging*, 10(2):461–497, 2016. 83

[84] ApS MOSEK. The MOSEK optimization toolbox for MATLAB manual, version 8.0, 2017. 4, 11, 76

[85] Ion Necoara, Andrei Patrascu, and Francois Glineur. Complexity of first-order inexact Lagrangian and penalty methods for conic convex programming. *Optimization Methods & Software*, 34(2):305–335, 2019. 6

[86] Ion Necoara and Johan AK Suykens. Application of a smoothing technique to decomposition in convex optimization. *IEEE Transactions on Automatic control*, 53(11):2674–2679, 2008.

[87] Valentin Nedelcu, Ion Necoara, and Quoc Tran-Dinh. Computational complexity of inexact gradient augmented Lagrangian methods: Application to constrained MPC. *SIAM Journal on Control and Optimization*, 52(5):3109–3134, 2014. 6

[88] Arkadi Nemirovski. Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15(1):229–251, 2004. 6, 7, 41, 47, 78

[89] Yurii Nesterov. A method for unconstrained convex minimization problem with the rate of convergence $\mathcal{O}(1/k^2)$. In *Doklady AN USSR*, volume 269, pages 543–547, 1983. 37, 45

[90] Yurii Nesterov. Excessive gap technique in nonsmooth convex minimization. *SIAM Journal on Optimization*, 16(1):235–249, 2005. 6

[91] Yurii Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005. 6, 43, 80, 81

[92] Yurii Nesterov. Dual extrapolation and its applications to solving variational inequalities and related problems. *Mathematical Programming*, 109(2-3):319–344, 2007. 7

[93] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*, volume 87. Springer Science & Business Media, 2013. 5

[94] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Science & Business Media, 2006. 5

[95] Daniel O'Connor and Lieven Vandenberghe. Primal-dual decomposition by operator splitting and applications to image deblurring. *SIAM Journal on Imaging Sciences*, 7(3):1724–1754, 2014. 3

[96] Brendan Odonoghue and Emmanuel Candes. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, 15(3):715–732, 2015. 74

[97] Pablo Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96(2):293–320, 2003. 4, 27

[98] Gábor Pataki. The geometry of semidefinite programming. In *Handbook of Semidefinite Programming*, pages 29–65. Springer, 2000. 14

[99] Gábor Pataki. A simple derivation of a facial reduction algorithm and extended dual systems. Technical report, Columbia University, 2000. 5, 11, 14

[100] Gábor Pataki. Strong duality in conic linear programming: Facial reduction and extended duals. In *Computational and Analytical Mathematics*, pages 613–634. Springer, 2013. 5, 11, 14

[101] Gábor Pataki. Bad semidefinite programs: They all look the same. *SIAM Journal on Optimization*, 27(1):146–172, 2017. 119

[102] Gábor Pataki and Stefan H Schmieta. The DIMACS library of mixed semidefinite-quadratic-linear programs. `http://archive.dimacs.rutgers.edu/Challenges/Seventh/Instances/`, 1999. 17, 20

[103] Frank Permenter, Henrik A Friberg, and Erling D Andersen. Solving conic optimization problems via self-dual embedding and facial reduction: A unified approach. *SIAM Journal on Optimization*, 27(3):1257–1282, 2017. 5, 14

[104] Frank Permenter and Pablo Parrilo. Partial facial reduction: Simplified, equivalent SDPs via approximations of the PSD cone. *Mathematical Programming*, pages 1–54, 2017. xiii, 5, 11, 14, 15, 16, 17, 19, 119, 120

[105] Gabriel Peyré and Marco Cuturi. Computational optimal transport with applications to data sciences. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019. 2

[106] Boris Teodorovich Polyak. A general method for solving extremal problems. *Doklady Akademii Nauk*, 174(1):33–36, 1967. 6

[107] Roman A Polyak, James Costa, and Saba Neyshabouri. Dual fast projected gradient method for quadratic programming. *Optimization Letters*, 7(4):631–645, 2013. 6

[108] Michael Posa, Mark Tobenkin, and Russ Tedrake. Lyapunov analysis of rigid body systems with impacts and friction via sums-of-squares. In *Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control*, pages 63–72. ACM, 2013. 17

[109] Songdi Qian, Ying'ai Gan, Feng Tian, Weizheng Li, Meisheng Li, Bingzheng Chen, Yuquan Hu, Jifa Gu, and Yaohuang Guo. *Operations Research*. Tsinghua University Press, Beijing, 2012. 5

[110] Arie J Quist, Etienne de Klerk, Cornelis Roos, and Tamas Terlaky. Copositive realxation for general quadratic programming. *Optimization Methods & Software*, 9(1-3):185–208, 1998. 17

[111] Hamed Rahimian and Sanjay Mehrotra. Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659*, 2019. 2

[112] James Renegar. *A Mathematical View of Interior-Point Methods in Convex Optimization*, volume 3. SIAM, 2001. 22

[113] R Tyrell Rockafellar. The multiplier method of Hestenes and Powell applied to convex programming. *Journal of Optimization Theory and applications*, 12(6):555–562, 1973. 42

[114] R Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970. 5

[115] R Tyrrell Rockafellar. Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research*, 1(2):97–116, 1976. 7, 37

[116] R Tyrrell Rockafellar and Roger J-B Wets. *Variational Analysis*, volume 317. Springer Science & Business Media, 2009. 5

[117] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1-4):259–268, 1992. 84

[118] Sohil Shah, Abhay Kumar Yadav, Carlos D Castillo, David W Jacobs, Christoph Studer, and Tom Goldstein. Biconvex relaxation for semidefinite programming in computer vision. In *European Conference on Computer Vision*, pages 717–735. Springer, 2016. 4

[119] Vikas Singh, Lopamudra Mukherjee, Jiming Peng, and Jinhui Xu. Ensemble clustering using semidefinite programming with applications. *Machine Learning*, 79(1-2):177–200, 2010. 4

[120] Jos F Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods & Software*, 11(1-4):625–653, 1999. 4, 11, 19

[121] Weijie Su, Stephen Boyd, and Emmanuel J Candes. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. *The Journal of Machine Learning Research*, 17(1):5312–5354, 2016. 78

[122] Defeng Sun, Kim-Chuan Toh, and Liuqin Yang. A convergent 3-block semiproximal alternating direction method of multipliers for conic programming with 4-type constraints. *SIAM Journal on Optimization*, 25(2):882–915, 2015. viii, 17, 30

[123] Defeng Sun, Kim-Chuan Toh, Yancheng Yuan, and Xin-Yuan Zhao. SDPNAL+: A Matlab software for semidefinite programming with bound constraints (version 1.0). *Optimization Methods & Software*, pages 1–29, 2019. 4, 11

[124] Shin-Ichi Tanigawa. Singularity degree of the positive semidefinite matrix completion problem. *SIAM Journal on Optimization*, 27(2):986–1009, 2017. 5, 14

[125] Quoc Tran-Dinh, Ahmet Alacaoglu, Olivier Fercoq, and Volkan Cevher. An adaptive primal-dual framework for nonsmooth convex minimization. *Mathematical Programming Computation*, pages 1–41, 2019. 6, 91

[126] Quoc Tran-Dinh, Olivier Fercoq, and Volkan Cevher. A smooth primal-dual optimization framework for nonsmooth composite convex minimization. *SIAM Journal on Optimization*, 28(1):96–134, 2018. 6, 37

[127] Quoc Tran-Dinh, Suat Gumussoy, Wim Michiels, and Moritz Diehl. Combining convex–concave decompositions and linearization approaches for solving BMIs, with application to static output feedback. *IEEE Transactions on Automatic Control*, 57(6):1377–1390, 2011. 2, 4

[128] Quoc Tran-Dinh, Ion Necoara, and Moritz Diehl. Fast inexact decomposition algorithms for large-scale separable convex optimization. *Optimization*, 65(2):325–356, 2016. 6

[129] Quoc Tran-Dinh and Yuzixuan Zhu. Augmented Lagrangian-based decomposition methods with non-ergodic optimal rates. *arXiv preprint arXiv:1806.05280*, 2018. 34, 35

[130] Quoc Tran-Dinh and Yuzixuan Zhu. Non-stationary first-order primal-dual algorithms with fast non-ergodic convergence rates. *arXiv preprint arXiv:1903.05282*, 2020. 9, 34, 35, 37

[131] Lloyd N Trefethen and David Bau III. *Numerical Linear Algebra*, volume 50. SIAM, 1997. 15

[132] Paul Tseng. On accelerated proximal gradient methods for convex-concave optimization. Technical report, Department of Mathematics, University of Washington, Seattle, 2008. 6, 59

[133] Levent Tunçel. *Polyhedral and Semidefinite Programming Methods in Combinatorial Optimization*, volume 27. American Mathematical Society, 2016. 5, 11, 14

[134] Reha H Tütüncü, Kim-Chuan Toh, and Michael J Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming*, 95(2):189–217, 2003. 4, 11

[135] Bng Công Vũ. A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Advances in Computational Mathematics*, 38(3):667–681, 2013. 6

[136] David G Wagner and Yehua Wei. A criterion for the half-plane property. *Discrete Mathematics*, 309(6):1385–1390, 2009. 17

[137] Hayato Waki. How to generate weakly infeasible semidefinite programs via Lasserre's relaxations for polynomial optimization. *Optimization Letters*, 6(8):1883–1896, 2012. vii, 17, 23

[138] Hayato Waki and Masakazu Muramatsu. Facial reduction algorithms for conic optimization problems. *Journal of Optimization Theory and Applications*, 158(1):188–215, 2013. 5, 11, 14

[139] Hayato Waki, Maho Nakata, and Masakazu Muramatsu. Strange behaviors of interior-point methods for solving semidefinite programming problems in polynomial optimization. *Computational Optimization and Applications*, 53(3):823–844, 2012. vii, 17, 24, 122, 126

[140] Weiran Wang and Miguel A Carreira-Perpinán. Projection onto the probability simplex: An efficient algorithm with a simple proof, and an application. *arXiv preprint arXiv:1309.1541*, 2013. 78

[141] Zaiwen Wen, Donald Goldfarb, and Wotao Yin. Alternating direction augmented Lagrangian methods for semidefinite programming. *Mathematical Programming Computation*, 2(3-4):203–230, 2010. 7

[142] Blake E Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, pages 3639–3647, 2016. 53, 65

[143] Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems 17 (NIPS 2004)*, pages 1537–1544, 2004. 2

[144] Yangyang Xu. First-order methods for constrained convex programming based on linearized augmented Lagrangian function. *arXiv preprint arXiv:1711.08020*, 2017. 7, 37, 42, 53

[145] Yangyang Xu. Global convergence rates of augmented Lagrangian methods for constrained convex programming. *arXiv preprint arXiv:1711.05812*, 2017.

[146] Yangyang Xu. Primal-dual stochastic gradient method for convex programs with many functional constraints. *arXiv preprint arXiv:1802.02724*, 2018.

[147] Yangyang Xu. Iteration complexity of inexact augmented Lagrangian methods for constrained convex programming. *Mathematical Programming*, pages 1–46, 2019. 7, 37, 40, 42

[148] Yan Yan, Yi Xu, Qihang Lin, Wei Liu, and Tianbao Yang. Optimal epoch stochastic gradient descent ascent methods for min-max optimization. *arXiv preprint arXiv:2002.05309*, 2020. 7

[149] Liuqin Yang, Defeng Sun, and Kim-Chuan Toh. SDPNAL+: a majorized semismooth Newton-CG augmented Lagrangian method for semidefinite programming with nonnegative constraints. *Mathematical Programming Computation*, 7(3):331–366, 2015. viii, 4, 11, 17, 30

[150] Xiaoqun Zhang, Martin Burger, and Stanley Osher. A unified primal-dual algorithm framework based on Bregman iteration. *Journal of Scientific Computing*, 46(1):20–46, 2011. 6

[151] Xin-Yuan Zhao, Defeng Sun, and Kim-Chuan Toh. A Newton-CG augmented Lagrangian method for semidefinite programming. *SIAM Journal on Optimization*, 20(4):1737–1765, 2010. 4, 11

[152] Yuzixuan Zhu, Deyi Liu, and Quoc Tran-Dinh. Accelerated primal-dual algorithms for a class of convex-concave saddle-point problems with non-bilinear coupling term. *arXiv preprint arXiv:2006.09263*, 2020. 9, 34, 35

[153] Yuzixuan Zhu, Gábor Pataki, and Quoc Tran-Dinh. Sieve-SDP: A simple facial reduction algorithm to preprocess semidefinite programs. *Mathematical Programming Computation*, 11(3):503–586, 2019. 9, 10, 90