UNIVERSITY OF CALIFORNIA

Los Angeles

Primal–dual proximal optimization algorithms with Bregman divergences

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Xin Jiang

2022

ABSTRACT OF THE DISSERTATION

Primal–dual proximal optimization algorithms with Bregman divergences

by

Xin Jiang
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2022
Professor Lieven Vandenberghe, Chair

Proximal methods are an important class of algorithms for solving nonsmooth, constrained, large-scale or distributed optimization problems. Because of their flexibility and scalability, they are widely used in current applications in engineering, machine learning, and data science. The key idea of proximal algorithms is the decomposition of a large-scale optimization problem into several smaller, simpler problems, in which the basic operation is the evaluation of the proximal operator of a function. The proximal operator minimizes the function regularized by a squared Euclidean distance, and it generalizes the Euclidean projection onto a closed convex set. Since the cost of the evaluation of proximal operators often dominates the per-iteration complexity in a proximal algorithm, efficient evaluation of proximal operators is critical. To this end, generalized Bregman proximal operators based on non-Euclidean distances have been proposed and incorporated in many algorithms and applications. In the first part of this dissertation, we present primal–dual proximal splitting methods for convex optimization, in which generalized Bregman distances are used to define the primal and dual update steps. The proposed algorithms can be viewed as Bregman extensions of many well-known proximal methods. For these algorithms, we analyze the theoretical convergence and

develop techniques to improve practical implementation.

In the second part of the dissertation, we apply the Bregman proximal splitting algorithms to the centering problem in large-scale semidefinite programming with sparse coefficient matrices. The logarithmic barrier function for the cone of positive semidefinite completable sparse matrices is used as the distance-generating kernel. For this distance, the complexity of evaluating the Bregman proximal operator is shown to be roughly proportional to the cost of a sparse Cholesky factorization. This is much cheaper than the standard proximal operator with Euclidean distances, which requires an eigenvalue decomposition. Therefore, the proposed Bregman proximal algorithms can handle sparse matrix constraints with sizes that are orders of magnitude larger than the problems solved by standard interior-point methods and proximal methods.

The dissertation of Xin Jiang is approved.

Vwani Roychowdhury

Arash A. Amini

Wotao Yin

Lieven Vandenberghe, Committee Chair

University of California, Los Angeles

2022

To my family

for their unconditional love and support.

TABLE OF CONTENTS

1 Introduction **1**

1.1 Algorithms for large-scale optimization problems 1

1.2 Proximal methods with Bregman distances 4

1.3 Contributions and outline of the dissertation 5

2 Primal–dual proximal splitting methods **8**

2.1 Problem formulation 8

2.2 Duality and optimality conditions 9

2.3 Merit functions 11

2.4 Proximal operator 14

2.5 First-order proximal algorithms: survey and connections 15

2.5.1 Condat–Vũ three-operator splitting algorithm 16

2.5.2 Primal–dual three-operator (PD3O) splitting algorithm 20

2.5.3 Primal–dual Davis–Yin (PDDY) splitting algorithm 22

3 Bregman proximal splitting algorithms **24**

3.1 Bregman proximal operators 26

3.2 Bregman Condat–Vũ three-operator splitting algorithms 28

3.2.1 Derivation from Bregman proximal point method 30

3.2.2 Convergence analysis 34

3.2.3 Relation to other Bregman proximal splitting algorithms 41

3.3 Bregman dual Condat–Vũ algorithm with line search 44

3.3.1 Algorithm . . . 46
3.3.2 Convergence analysis . . . 47
3.4 Bregman PD3O algorithm . . . 54
3.4.1 Convergence analysis . . . 56
3.4.2 Relation to other Bregman proximal algorithms . . . 59
3.5 Numerical experiment . . . 61

4 Application to sparse semidefinite programming . . . 66
4.1 Sparse semidefinite programming . . . 67
4.2 Primal and dual barriers . . . 71
4.3 The centering problem . . . 72
4.4 Barrier proximal operator for sparse PSD matrix cone . . . 74
4.5 Newton's method for barrier proximal operator . . . 76
4.6 Numerical experiments . . . 79
4.6.1 Maximum cut problem . . . 80
4.6.2 Graph partitioning . . . 84

5 Conclusions . . . 88

References . . . 90

LIST OF FIGURES

2.1 Proximal methods derived from primal Condat–Vũ algorithm. 17

2.2 Proximal methods derived from dual Condat–Vũ algorithm. 19

2.3 Proximal algorithms derived from PD3O. 20

2.4 Proximal algorithms derived from PDDY. 22

3.1 Proximal algorithms derived from Bregman primal Condat–Vũ algorithm (3.6). 42

3.2 Proximal algorithms derived from Bregman dual Condat–Vũ algorithm (3.7). . . 43

3.3 Acceptable stepsizes in Condat–Vũ algorithms and PD3O. We assume the same matrix norm $\|A\|$ and Lipschitz constant L are used in the analysis of the two algorithms. The light gray region under the blue curve is defined by the inequality for the Condat–Vũ algorithms in (3.52). The region under the red curve shows the values allowed by the stepsized conditions for PD3O. 55

3.4 Proximal algorithms derived from Bregman PD3O. 60

3.5 The blue and red curves show the boundaries of the stepsize regions for Bregman Condat–Vũ algorithms and Bregman PD3O, respectively. The blue and red points indicate the chosen parameters in (3.67) (red for for PD3O, blue for Condat–Vũ). In the Bregman dual Condat–Vũ algorithm with line search, the stepsizes are selected on the dashed straight line. The solid line segment shows the range of stepsizes that were selected, with dots indicating the largest, median, and smallest stepsizes. 63

3.6 Comparison of three algorithms (Bregman primal Condat–Vũ, Bregman dual Condat–Vũ with line search, and Bregman PD3O) in terms of objective values. The top two figures plot the relative error of the function value versus CPU time and number of iterations for one problem instance (3.65), respectively. The bottom two figures correspond to another problem instance. 64

4.1 *Left.* The function $\zeta(\nu) = \sum_i 1/(\nu + \lambda_i)$ for $\lambda = (-5, 0, 5, 10)$. We are interested in the solution of $\zeta(\nu) = 1$ larger than $-\lambda_{\min} = 5$. *Right.* The function $1/\zeta(\nu) - 1$. 76

LIST OF TABLES

4.1 Results for four instances of the MAXCUT problem from SDPLIB [Bor99]. Column 3 is the optimal value computed by MOSEK. Column 4 is the difference with the optimal value of the centering problem computed by algorithm (4.20). The last two columns give the primal and dual residuals in the computed solution. 82

4.2 The four MAXCUT problems from SDPLIB plus four larger graphs from the SuiteSparse collection [KAB19]. The last column ("PDHG iterations") gives the number of iterations in the primal–dual algorithm. Columns 3–5 describe the complexity of one iteration of the algorithm. The CPU time is measured in seconds. 83

4.3 Results for four graph partitioning problems from SDPLIB. Column 3 is the optimal value computed by MOSEK. Column 4 is the difference with the optimal value of the centering problem computed by algorithm (4.20). The last two columns give the primal and dual residuals in the computed solution. 86

4.4 The four graph partitioning problems from SDPLIB plus four larger graphs from the SuiteSparse collection. The last column gives the number of iterations in the primal–dual algorithm. Columns 3–5 describe the complexity of one iteration of the algorithm. The CPU time is measured in seconds. 87

ACKNOWLEDGMENTS

This dissertation is the culmination of my five years of study and research at UCLA. This could not have been possible without the help and support of so many people.

First and foremost, I would express my sincere gratitude to my advisor, Professor Lieven Vandenberghe, to whom this dissertation owes its existence. In his course on convex optimization, he aroused my initial interest in optimization via his elegant and rigorous exposition. Moreover, he treats his students with meticulous care, provides them with appropriate guidance and motivation, and addresses their questions seriously and carefully. I am forever thankful for the opportunity to work with and to learn from Professor Vandenberghe.

In addition, I would like to thank my committee members. Professor Wotao Yin is an excellent instructor at UCLA, and also a visionary leader at Alibaba. I would like to thank him for all the knowledge I learnt in his class as well as all the help he offered during my internship at Alibaba. I also would like to express my appreciation to Professor Arash A. Amini, in whose class I built a solid foundation in statistics. I am very grateful to Professor Vwani Roychowdhury, for his help and scholarly examples. I also thank the staff in ECE department for their helpful assistance and student-centered service. Special thanks for Ryo Arreola. My graduate student life at UCLA would not have been the same with Ryo. His kindness made all the difference for international students like me.

I would like to thank my colleagues and friends at UCLA for many helpful support and company. I am especially grateful to Martin S. Andersen, Yifan Sun, Cameron Gunn, Suzi Chao, Tianyi Wang, Qiujing Lu, and Huiyu Wang, for the good time we shared with each other. My old friends have also been unforgettable emotional support in my PhD life. Here I would like to give special thanks to Ziyao Chen and Qianwen Yu, for their unwavering confidence on me and forever friendship.

Finally, my parents deserve my deepest gratitude for all their love and encouragement. I shall thank them for being there with me throughout all stages of life.

VITA

2015 B.Eng., Electronic and Communication Engineering
Department of Electrical and Electronic Engineering
The University of Hong Kong, Hong Kong, China.

2017 M.S., Electrical and Computer Engineering
Department of Electrical and Computer Engineering
University of California, Los Angeles (UCLA).

2016–2022 Teaching Assistant
Department of Electrical and Computer Engineering
University of California, Los Angeles (UCLA).

2017–2022 Graduate Student Researcher
Department of Electrical and Computer Engineering
University of California, Los Angeles (UCLA).

PUBLICATIONS

Xin Jiang and Lieven Vandenberghe. Bregman three-operator splitting methods. *arXiv e-prints*, arXiv:2203:00252, 2022.

Xin Jiang and Lieven Vandenberghe. Bregman primal–dual first-order method and applications to sparse semidefinite programming. *Computational Optimization and Applications*, 81(1):127–159, 2022.

Jiarong Xu, Yizhou Sun, Xin Jiang, Yanhao Wang, Chuping Wang, Jiangang Lu and Yang Yang. Blindfolded attackers still threatening: Strict black-box adversarial attacks on graphs. In *Proceedings of the 36th Conference on Artificial Intelligence*. 2022.

Jiarong Xu, Yang Yang, Junru Chen, Xin Jiang, Chuping Wang, Jiangang Lu and Yizhou Sun. Unsupervised adversarially robust representation learning on graphs. In *Proceedings of the 36th Conference on Artificial Intelligence*, 2022.

Ziyuan Jiao, Zeyu Zhang, Xin Jiang, David Han, Song-Chun Zhu, Yixin Zhu and Hangxin Liu. Consolidating kinematic models to promote coordinated mobile manipulations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2021

CHAPTER 1

Introduction

1.1 Algorithms for large-scale optimization problems

The recent development of optimization algorithms is driven by a broad spectrum of applications from engineering, machine learning, and data science. During the 1990s, the focus of research on optimization methods was extending interior-point methods from linear programming to nonlinear, convex optimization problems [NN94, Wri97, NT98, BN01, Ren01, PRT02, BV04, NW06, Gon12]. To this end, a useful canonical form of optimization problems is the conic LP:

$$\begin{array}{ll} \text{minimize} & \langle c, x \rangle \\ \text{subject to} & Ax = b \\ & x \in \mathcal{K}, \end{array} \tag{1.1}$$

where the optimization variable is the vector x, and $\mathcal{K}$ is a proper (or regular) convex cone. Examples of conic LPs include many important convex optimization problems, *e.g.*, second-order cone programs (SOCPs) [AG03], geometric programs (GPs), and semidefinite programs (SDPs) [Tod01]. Due to their broad applicability to conic LPs, interior-point methods serve as the computational backbone for a number of optimization software packages, including many general-purpose solvers (*e.g.*, SeDuMi [Stu99], SDPT3 [TTT02], Gurobi [Gur22], MOSEK [MOS19], CVXOPT [ADV20]) as well as several modeling tools (*e.g.*, CVX [GB14], CVXPY [DB16], CVXR [FNB20]).

However, for modern applications in image processing, machine learning, and data science, the dimensions of the optimization problems grow rapidly [SNW12, BCN18, GR18] and

off-the-shelf interior-point methods are often impractical for such large-scale applications. More specifically, for conic LPs, the per-iteration complexity of interior-point methods is dominated by the formulation and solution of a large set of linear equations. Alternative approaches, especially for large-scale optimization problems, include first-order methods, coordinate descent methods, and decomposition (or splitting) methods. Typical examples of first-order methods include the gradient (subgradient) descent method, its acceleration and extensions [Nes83, Nes88, NW06, Nes18, dST21]. The coordinate descent method and its variations [LT92, Wri15] successively minimize along the coordinate directions. Decomposition (or splitting) methods iteratively decompose a large-scale optimization problem into smaller, simpler problems [GOY17, RY22] and then solve them separately.

In this dissertation, we focus on a canonical form of optimization problems

$$\text{minimize} \quad f(x) + g(Ax) + h(x), \tag{1.2}$$

where f, g, and h are convex functions, h is differentiable, and f, g are nonsmooth. This general problem is a useful formulation for studying most first-order and decomposition methods, and it is sufficient to represent many important structures arising from a wide variety of large-scale applications in machine learning, signal and image processing, operations research, control, and other fields [PB14, KP15, CP16a, GOY17, CKC22]. For example, when $f = 0$ and $g = 0$, problem (1.2) reduces to minimizing a differentiable function, which can be solved by the gradient descent method. As another example, when the, nonsmooth function f is the indicator function of a closed, convex set $\mathcal{C}$

$$\delta_{\mathcal{C}}(x) = \begin{cases} 0 & x \in \mathcal{C} \\ \infty & x \notin \mathcal{C}, \end{cases}$$

problem (1.2) becomes a set-constrained problem

$$\begin{aligned} \text{minimize} \quad & g(Ax) + h(x) \\ \text{subject to} \quad & x \in \mathcal{C}. \end{aligned}$$

When $g = \delta_{\{b\}}$ is the indicator function of a singleton, the problem (1.2) reduces to an equality-constrained problem

$$\begin{array}{ll} \text{minimize} & f(x) + h(x) \\ \text{subject to} & Ax = b. \end{array} \tag{1.3}$$

Furthermore, when f is the indicator function of a proper, convex cone $\mathcal{K}$ and $h(x) = \langle c, x \rangle$ is a linear function, problem (1.3) reduces to the conic LP (1.1).

In the general problem (1.2) as well as the special cases, the structure of the nonsmooth function f (and g) is often exploited via its proximal operator:

$$\mathbf{prox}_f(y) = \operatorname*{argmin}_x \big(f(x) + \frac{1}{2}\|x - y\|^2 \big),$$

where $\|\cdot\|$ indicates the Euclidean norm. By definition, the proximal operator minimizes the nonsmooth function f regularized by a squared Euclidean distance, and generalizes the Euclidean projection onto a closed, convex set. Evaluation of proximal operators is the basic operation in proximal splitting methods, and thus efficient evaluation is critical in these algorithms. Many convex functions have simple proximal operators; *i.e.*, the proximal operator either has a closed-form formula or can be evaluated efficiently via a simple algorithm. For example, when f is the indicator function of a closed, convex set $\mathcal{C}$, the proximal operator of f is the Euclidean projection onto the set $\mathcal{C}$. Another classical example is the ℓ_1-norm, *i.e.*, $f(x) = \|x\|_1$, the proximal operator of f is well-known as the "soft-threshold" operation (see, *e.g.*, [BT09a]):

$$\mathbf{prox}_f(x)_i = \begin{cases} x_i - 1 & x_i > 1 \\ 0 & |x_i| \leq 1 \\ x_i + 1 & x_i < 1. \end{cases}$$

Moreover, convex duality theory have also been used to design more efficient, scalable optimization algorithms. As a result, the development of primal–dual proximal splitting methods has become an active research area. Primal–dual proximal methods solve the

primal and dual problems simultaneously in an intertwined manner, and recent examples of primal–dual proximal methods for the three-term problem (1.2) include the Condat–Vũ algorithm [Con13, Vu13], the primal–dual three-operator (PD3O) algorithm [Yan18], and the primal–dual Davis–Yin (PDDY) algorithm [SCM20]. Algorithms for some special cases of (1.2) are also of interest. These include the Chambolle–Pock algorithm, also known as the primal–dual hybrid gradient (PDHG) method [PCB09, EZC10, CP11a, CP16b] (when $h = 0$), the alternating direction method of multipliers (ADMM) [GM75, GM76, BPC11] (when $h = 0$), the Loris–Verhoeven algorithm [LV11, CHZ13, DST15] (when $f = 0$), the proximal gradient algorithm (when $g = 0$), the Davis–Yin splitting algorithm [DY17] (when $A = I$), and the Douglas–Rachford splitting (DRS) algorithm [LM79]. These methods are closely related to each other. For example, ADMM is known to be equivalent to DRS applied to the dual problem [Gab83]. More details are discussed in Chapter 2.

1.2 Proximal methods with Bregman distances

In view of the flexibility and scalability of primal–dual proximal methods for large-scale convex optimization problems, many efforts have been made to further improve the efficiency of proximal algorithms, and in particular, the evaluation of proximal operators. To this end, proximal operators based on generalized distances have been proposed [CZ97] and among different definitions of generalized distances, Bregman divergence (or Bregman distance) [Bre67] is often used in many optimization algorithms [CT93, Eck93, Gul94, BL00, BMN01, BT03, AT06, Tse08, BBT17, BST18, LFN18, Teb18]. For example, the renowned mirror descent method [NY83, BMN01, BT03, Bub15, Bec17] is shown to be the projected subgradient descent method with Bregman distances. The Dykstra's algorithm with Bregman distances [BL00] allows non-Euclidean projections onto the constraint set. Proximal algorithms have also been integrated with Bregman distances, and examples include the proximal point method [CT93, Ha90, CZ92, Eck93, Gul94, Kiw97, AT06], the proximal gra-

dient method [Tse08, BBT17, LFN18, BST18, Teb18, HRX21] and recently, primal–dual proximal methods [CP16b, WX17, YA21].

In general, Bregman distances offer two potential benefits. First, the Bregman distance can help build a more accurate local optimization model around the current iterate. This is often interpreted as a form of preconditioning. For example, diagonal or quadratic preconditioning [PC11, JLL19, LXY21] has been shown to improve the convergence rate of PDHG, as well as the accuracy of the computed solution [ADH21]. As a second benefit, a Bregman proximal operator of a function may be easier to compute than the standard Euclidean proximal operator, and thus reduce the complexity per iteration of an optimization algorithm. The idea of incorporating Bregman distances into optimization algorithms has been exploited in many research areas, including signal processing [CV18], optimal transport [CLM21, CC22], matrix optimization problems [DT08], statistical estimation [TLJ06], and machine learning [Rd20].

1.3 Contributions and outline of the dissertation

Despite the advantages offered by the Bregman distances and the numerous applications, extending standard proximal methods and their convergence analysis to Bregman distances is not straightforward because some fundamental properties of the Euclidean proximal operators no longer hold for Bregman proximal operators. An example is the Moreau decomposition [Mor65]; see the definition in Section 2.4. Moreau decomposition relates the (Euclidean) proximal operators of a closed, convex function and its conjugate, and serves as a fundamental pillar for the convergence analysis of most primal–dual proximal splitting methods. Another example is the simple relation between the proximal operators of a function g and the composition with a linear function $g(Ax)$ when AA^T is a multiple of the identity; see also Section 2.4 for more details. This composition rule is used in [OV20] to establish the equivalence between some well-known primal–dual proximal methods for problem (1.2) with

$A = I$ and with general A.

In the first part of the dissertation, we propose several primal–dual proximal methods that incorporate Bregman distances. The presented algorithms include most well-known proximal algorithms as special cases. First, we discuss two variants of the Bregman Condat–Vũ algorithm, in which generalized Bregman proximal operators are used in both primal and dual updates. We give a new derivation for both algorithms, and based on the interpretation, we provide a unified framework for the convergence analysis. To improve practical implementation, we propose a line search technique for the Bregman dual Condat–Vũ algorithm for equality-constrained problems. Finally, we discuss a Bregman extension to PD3O and establish an ergodic convergence result.

The second part of the dissertation is motivated by the difficulty of exploiting sparsity in large-scale semidefinite programming (SDP). A semidefinite program is the conic LP (1.1) where the cone $\mathcal{K}$ is the positive semidefinite (PSD) matrix cone:

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) \\ \text{subject to} & \mathbf{tr}(A_i X) = b_i, \quad i = 1, \ldots, m \\ & X \succeq 0, \end{array}$$

where the optimization variable is the symmetric matrix X, and the generalized inequality constraint $X \succeq 0$ indicates that the matrix X is in the PSD matrix cone. The scalability of interior-point methods is limited by the need to form and solve a set of m linear equations at each iteration. Customization is often difficult, and depends on the structure of the problem. For standard proximal methods, on the other hand, one needs to compute a Euclidean projection onto the PSD matrix cone at each iteration. This involves an eigenvalue decomposition, and thus exploiting sparsity is often difficult for standard proximal methods. Therefore, to improve the efficiency and scalability of proximal algorithms, we apply the proposed Bregman proximal methods to the centering problem in large-scale SDPs with sparse coefficient matrices. We show that if the Bregman distance generated by the barrier function for the cone of PSD completable matrices is used, the generalized projections can

be computed efficiently, with a complexity dominated by the cost of a sparse Cholesky factorization. This is much cheaper than the eigenvalue decomposition needed in every iteration of standard proximal algorithms for SDPs. Hence, while the Bregman proximal method only solves an approximation of the SDP, it can handle problem sizes that are orders of magnitude than the problems solved by standard interior-point methods and proximal methods.

The rest of the dissertation is organized as follows. In Chapter 2 we review some basic results from convex duality theory and present a comprehensive survey for most primal–dual proximal splitting methods. Chapter 3 presents the proposed Bregman primal–dual proximal methods, the convergence analysis, and line search techniques for practical implementation. In Chapter 4 we apply the proposed Bregman proximal methods to the centering problem of sparse SDPs. We describe in detail the Bregman proximal operator designed for sparse SDPs, and also present results of numerical experiments. Chapter 5 concludes this dissertation with some final remarks.

CHAPTER 2

Primal–dual proximal splitting methods

The main focus of this dissertation is on primal–dual proximal splitting methods. Thus in this chapter we start by formulating the optimization problem that we study throughout the dissertation, and review some basic results from convex duality theory. In addition, we present a comprehensive survey of the state-of-the-art primal–dual first-order methods and show in details the connections between them.

This chapter is organized as follows. We introduce the problem formulation in Section 2.1, and in Section 2.2 we summarize the facts from convex duality theory that underlie the primal–dual algorithms for the optimization problem. In Section 2.3 we describe in detail the merit functions and give two illustrative examples. Finally, Section 2.5 discusses several proximal methods and their connections.

2.1 Problem formulation

We discuss primal–dual proximal splitting methods for optimization problems in the form

$$\text{minimize} \quad f(x) + g(Ax) + h(x), \tag{2.1}$$

where the optimization variable is $x \in \mathbf{R}^n$, and A is an $m \times n$ matrix. We assume the functions f, g, and h are proper (with nonempty domain), closed, and convex, and h is differentiable with an open convex domain $\mathbf{dom}\, h$. The notation $\langle x, y\rangle = x^T y$ is used for the standard inner product of vectors x and y, and $\|x\| = \langle x, x\rangle^{1/2}$ for the Euclidean norm of a vector x.

This general problem covers a wide variety of applications in machine learning, signal and image processing, operations research, control and other fields [CP11b, PB14, KP15, CP16a, CKC22]. An important example of (2.1) is $g = \delta_{\mathcal{C}}$, the indicator function of a closed convex set $\mathcal{C}$. With $g = \delta_{\mathcal{C}}$, the problem is equivalent to

$$\begin{array}{ll} \text{minimize} & f(x) + h(x) \\ \text{subject to} & Ax \in \mathcal{C}. \end{array}$$

For $\mathcal{C} = \{b\}$ the constraints are a set of linear equations $Ax = b$. This special case actually covers all applications of the more general problem (2.1), since (2.1) can be reformulated as

$$\begin{array}{ll} \text{minimize} & f(x) + g(y) + h(x) \\ \text{subject to} & Ax = y, \end{array}$$

at the expense of increasing the problem size by introducing a splitting variable y.

2.2 Duality and optimality conditions

In this section we present the dual problem as well as the primal–dual optimality conditions for the problem (2.1). These basic results play a fundamental role in the analysis of most primal–dual proximal algorithms.

To derive the dual of (2.1), we first reformulate the problem as

$$\begin{array}{ll} \text{minimize} & f(x) + g(y) + h(x) \\ \text{subject to} & Ax = y \end{array} \tag{2.2}$$

with variables $x \in \mathbf{R}^n$ and $y \in \mathbf{R}^m$. The *Lagrangian* for the reformulated problem (2.2) is

$$\tilde{\mathcal{L}}(x, y, z) = f(x) + g(y) + h(x) + \langle z, Ax - y \rangle.$$

Taking the infimum with respect to y yields the convex–concave function

$$\mathcal{L}(x, z) = f(x) + h(x) + \langle z, Ax \rangle - g^*(z),$$

which will be referred to as the Lagrangian of (2.1). We follow the convention that $\mathcal{L}(x,z) = +\infty$ if $x \notin \mathbf{dom}(f+h)$ and $\mathcal{L}(x,z) = -\infty$ if $x \in \mathbf{dom}(f+h)$ and $z \notin \mathbf{dom}\, g^*$. The objective function in (2.1) can be expressed as

$$\sup_z \mathcal{L}(x,z) = f(x) + h(x) + g(Ax).$$

The dual function is defined as

$$\inf_x \mathcal{L}(x,z) = -(f+h)^*(-A^Tz) - g^*(z),$$

where $(f+h)^*$ and g^* are the conjugates of $f+h$ and g:

$$(f+h)^*(w) = \sup_x \left(\langle w, x\rangle - f(x) - h(x)\right), \qquad g^*(z) = \sup_y \left(\langle z, y\rangle - g(y)\right).$$

The conjugate $(f+h)^*$ is the infimal convolution of f^* and h^* [Roc70], denoted by $f^* \,\square\, h^*$:

$$f^* \,\square\, h^*(z) = \inf_w \left((f^*(w) + h^*(z-w)\right).$$

Then the problem of maximizing the dual function is called the dual problem:

$$\text{maximize} \quad -(f+h)^*(-A^Tz) - g^*(z) \tag{2.3}$$

The primal–dual optimality conditions for (2.1) and (2.3) are

$$0 \in \partial f(x) + \nabla h(x) + A^Tz, \qquad 0 \in \partial g^*(z) - Ax,$$

where ∂f and ∂g^* are the subdifferentials of f and g^*. We often write the optimality conditions concisely as

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} \partial f(x) + \nabla h(x) \\ \partial g^*(z) \end{bmatrix}. \tag{2.4}$$

Throughout the dissertation, we assume that the optimality conditions (2.4) are solvable. Solutions $x^\star$, $z^\star$ of the optimality conditions (2.4) form a saddle-point of $\mathcal{L}$, *i.e.*, satisfy

$$\inf_x \sup_z \mathcal{L}(x,z) = \sup_z \mathcal{L}(x^\star, z) = \mathcal{L}(x^\star, z^\star) = \inf_x \mathcal{L}(x, z^\star) = \sup_z \inf_x \mathcal{L}(x,z). \tag{2.5}$$

In particular, $\mathcal{L}(x^\star, z^\star)$ is the optimal value of (2.1) and (2.3).

2.3 Merit functions

Most algorithms discussed in this dissertation generate primal and dual iterates and approximate solutions x, z with $x = \mathbf{dom}\, f \cap \mathbf{dom}\, h$ and $z \in \mathbf{dom}\, g^*$. The feasibility conditions $Ax \in \mathbf{dom}\, g$ and $-A^T z \in \mathbf{dom}(f+h)^*$ are not necessarily satisfied. Hence the duality gap

$$\sup_{z'} \mathcal{L}(x, z') - \inf_{x'} \mathcal{L}(x', z) = f(x) + h(x) + g(Ax) + (f+h)^*(-A^T z) + g^*(z) \tag{2.6}$$

may not always be useful as a merit to measure convergence. In this section, we introduce a merit function that will be used later to express convergence results. The merit function can also be used to design stopping conditions for primal–dual proximal splitting algorithms.

If we add constraints $x' \in \mathcal{X}$ and $z' \in \mathcal{Z}$ to the optimization problems on the left-hand side of (2.6), where $\mathcal{X}$ and $\mathcal{Z}$ are compact sets, we obtain a function

$$\eta(x, z) = \sup_{z' \in \mathcal{Z}} \mathcal{L}(x, z') - \inf_{x' \in \mathcal{X}} \mathcal{L}(x', z) \tag{2.7}$$

defined for all $x \in \mathbf{dom}(f+h)$ and $z \in \mathbf{dom}\, g^*$. This follows from the fact that the functions $f + h + \delta_{\mathcal{X}}$ and $g^* + \delta_{\mathcal{Z}}$ are closed and co-finite, so their conjugates have full domain [Roc70, Corollary 13.3.1]. If $\eta(x, z)$ is easily computed, and $\eta(x, z) \geq 0$ for all $x \in \mathbf{dom}(f+h)$ and $z \in \mathbf{dom}\, g^*$ with equality only if x and z are optimal, then the function η can serve as a *merit function* in primal–dual algorithms for problem (2.1). In the special case $\mathcal{X} = \{x^\star\}$ and $\mathcal{Z} = \{z^\star\}$, the merit function reduces to

$$\eta(x, z) = \mathcal{L}(x, z^\star) - \mathcal{L}(x^\star, z), \tag{2.8}$$

and is widely used in the literature [CP11a, Con13, CP16b]. The function (2.8) involves the optimal points $x^\star$, $z^\star$, so it is useful in convergence results but not in practical stopping conditions.

If $\mathbf{dom}(f+h)$ and $\mathbf{dom}\, g^*$ are bounded, then $\mathcal{X}$ and $\mathcal{Z}$ can be chosen to contain $\mathbf{dom}(f+h)$ and $\mathbf{dom}\, g^*$. Then the constraints in (2.7) are redundant and $\eta(x, z)$ is the duality gap (2.6). Boundedness of $\mathbf{dom}(f+h)$ and $\mathbf{dom}\, g^*$ or existence of such compact sets $\mathcal{X}$

and $\mathcal{Z}$ is a common assumption in the literature on primal–dual first-order methods [Nem04, JN12a, JN12b, Bub15].

A weaker assumption is that (2.1) and (2.3) have an optimal primal solution $x^\star \in \mathcal{X}$ and an optimal dual solution $z^\star \in \mathcal{Z}$. If so, we have

$$\eta(x, z) \geq \mathcal{L}(x, z^\star) - \mathcal{L}(x^\star, z) \geq 0$$

for all $x \in \mathbf{dom}(f + h)$ and $z \in \mathbf{dom}\, g^*$. The second inequality follows from (2.5) and is an equality only if x and z are optimal. It follows that $\eta(x, z) \geq 0$ with equality only if x, z are optimal.

Whether $\eta(x, z)$ is easy to evaluate depends on the problem and choice of sets $\mathcal{X}$ and $\mathcal{Z}$. A general expression for the first term in (2.7) is

$$\begin{aligned}\sup_{z' \in \mathcal{Z}} \mathcal{L}(x, z') &= f(x) + h(x) + (g \,\square\, \sigma_{\mathcal{Z}})(Ax) \\ &= f(x) + h(x) + \inf_y \left(g(y) + \sigma_{\mathcal{Z}}(Ax - y)\right),\end{aligned}$$

where $\sigma_{\mathcal{Z}}(w) = \sup_{z \in \mathcal{Z}} \langle z, w \rangle$ is the support function of $\mathcal{Z}$. The corresponding expression for the second term in (2.7) are

$$\begin{aligned}\inf_{x' \in \mathcal{X}} \mathcal{L}(x', z) &= -g^*(z) - ((f + h)^* \,\square\, \sigma_{\mathcal{X}})(-A^T z) \\ &= -g^*(z) - \inf_w \left((f + h)^*(w) + \sigma_{\mathcal{X}}(A^T z + w)\right).\end{aligned}$$

Consider for example the primal and dual pair

$$\begin{array}{lll} \text{minimize} & f(x) + h(x) & \qquad\qquad \text{maximize} \quad -b^T z - (f + h)^*(-A^T z). \\ \text{subject to} & Ax = b & \end{array}$$

Here $g = \delta_{\{b\}}$. If we take $\mathcal{Z} = \{z \mid \|z\| \leq \zeta\}$, then $\sigma_{\mathcal{Z}}(y) = \zeta \|y\|$ and the infimal convolution $(g \,\square\, \sigma_{\mathcal{Z}})(Ax)$ is given by

$$(g \,\square\, \sigma_{\mathcal{Z}})(Ax) = \inf_y \left(g(y) + \sigma_{\mathcal{Z}}(Ax - y)\right)$$

$$
\begin{aligned}
&= \sigma_{\mathcal{Z}}(Ax - b) \\
&= \sup_{\|z\| \leq \zeta} \langle Ax - b, z \rangle \\
&= \zeta \|Ax - b\|.
\end{aligned}
$$

The second equality follows from the definition $g = \delta_{\{b\}}$. If in addition $\mathbf{dom}(f + h)$ is bounded and we take $\mathcal{X} \supseteq \mathbf{dom}(f + h)$, then

$$
\eta(x, z) = f(x) + h(x) + \zeta \|Ax - b\| + b^T z + (f + h)^*(-A^T z)
$$

with domain $\mathbf{dom}(f + h) \times \mathbf{R}^m$. The first three terms are the primal objective augmented with an exact penalty for the constraint $Ax = b$, with a sufficiently large ζ (*i.e.*, $\zeta > \|z^\star\|$).

As another example, consider the pair of primal and dual problems

$$
\begin{array}{lll}
\text{minimize} & \|x\|_1 & \qquad \text{maximize} \quad b^T z \\
\text{subject to} & Ax \leq b & \qquad \text{subject to} \quad \|A^T z\|_\infty \leq 1 \\
& & \qquad\qquad\quad z \geq 0.
\end{array}
$$

This is an example of (2.1) with $f(x) = \|x\|_1$, $h(x) = 0$, and g the indicator function of the set $\{y \mid y \leq b\}$. The domains $\mathbf{dom}\, f$ and $\mathbf{dom}\, g^*$ are unbounded. If we choose $\mathcal{X} = \{x \mid \|x\|_\infty \leq \kappa\}$ and $\mathcal{Z} = \{z \mid 0 \leq z \leq \lambda \mathbf{1}\}$, then

$$
\sigma_{\mathcal{X}}(w) = \kappa \|w\|_1, \qquad (f^* \,\square\, \sigma_{\mathcal{X}})(w) = \kappa \sum_{i=1}^n \max\{0, |w_i| - 1\}
$$

and

$$
\sigma_{\mathcal{Z}}(y) = \lambda \sum_{i=1}^m \max\{0, y_i\}, \qquad (g \,\square\, \sigma_{\mathcal{Z}})(y) = \lambda \sum_{i=1}^m \max\{0, y_i - b_i\}.
$$

Hence, for this example the merit function (2.7) is

$$
\eta(x, z) = \|x\|_1 + \lambda \sum_{i=1}^m \max\{0, (Ax - b)_i\} - b^T z + \kappa \sum_{i=1}^n \max\{0, |(A^T z)_i| - 1\}
$$

with domain $\mathbf{R}^n \times \mathbf{R}^m_+$. The second term is an exact penalty for the primal constraint $Ax \leq b$. The last term is an exact penalty for the dual constraint $\|A^T z\|_\infty \leq 1$.

Proof. The support function of the infinity norm ball $\mathcal{X}$ is the scaled 1-norm function $\sigma_{\mathcal{X}}(w) = \kappa\|w\|_1$, and the conjugate function of the 1-norm function is the indicator function of the unit infinity norm ball. Then the infimal convolution reduces to

$$\begin{aligned}(f^* \square \sigma_{\mathcal{X}})(w) &= \inf_x (f^*(x) + \sigma_{\mathcal{X}}(w)(w - x)) \\ &= \inf_{\|x\|_{\inf} \leq 1} \kappa\|x - w\|_1 \\ &= \kappa \sum_{i=1}^{n} \max\{0, |w_i| - 1\}.\end{aligned}$$

The support function of $\mathcal{Z}$ is

$$\sigma_{\mathcal{Z}}(y) = \sup_{z \in \mathcal{Z}} \langle z, y \rangle = \lambda \sum_{i=1}^{m} \sup_{0 \leq z_i \leq \lambda} y_i z_i = \lambda \sum_{i=1}^{m} \max\{0, y_i\}.$$

Then the infimal convolution is

$$\begin{aligned}(g \square \sigma_{\mathcal{Z}})(y) &= \inf_w (g(w) + \sigma_{\mathcal{Z}}(y - w)) \\ &= \lambda \sum_{i=1}^{m} \inf_{w_i \leq b_i} \max\{0, y_i - w_i\} \\ &= \lambda \sum_{i=1}^{m} \max\{0, y_i - b_i\}.\end{aligned}$$

□

2.4 Proximal operator

The *proximal operator* or *proximal mapping* of a closed convex function f is defined as

$$\mathbf{prox}_f(y) = \operatorname*{argmin}_x \big(f(x) + \frac{1}{2}\|x - y\|^2\big). \tag{2.9}$$

If f is closed and convex, the minimizer in the definition exists and is unique for all y [Mor65]. We will call (2.9) the *standard* or the *Euclidean proximal operator* when we need to distinguish it from Bregman proximal operators defined in Section 3.1. For detailed discussion on the proximal operator as well as the properties, we refer interested readers to

the survey papers [CP11b, PB14], and the books [BC17, Bec17, RY22]. Below we review two important properties of the proximal operator, which will be used later in this dissertation.

The first property is the composition rule with affine mapping. In general, the proximal operator of $f(x) = g(Ax + b)$ does not follow from the proximal operator of g. However, if $AA^T = (1/\alpha)I$, then the proximal operator of f can be derived easily from that of g:

$$\begin{aligned}\mathbf{prox}_f(x) &= (I - \alpha A^T A)x + \alpha A^T(\mathbf{prox}_{\alpha^{-1}g}(Ax + b) - b) \\ &= x - \alpha A^T(Ax + b) - \mathbf{prox}_{\alpha^{-1}g}(Ax + b)).\end{aligned} \tag{2.10}$$

This property is the cornerstone for the "completion" trick, which shows equivalence between several primal–dual proximal splitting algorithms [OV20]; see also Section 2.5.

Another important property of the proximal operator is called *Moreau decomposition* or *Moreau identity*. For any $\lambda > 0$ and any $x \in \mathbf{R}^n$,

$$x = \mathbf{prox}_{\lambda f}(x) + \lambda \mathbf{prox}_{\lambda^{-1} f^*}(x/\lambda). \tag{2.11}$$

This property relates the proximal operator of a closed convex function to that of its conjugate, and is widely used in convergence analysis for primal–dual proximal splitting algorithms.

2.5 First-order proximal algorithms: survey and connections

Over the past few decades, first-order proximal methods have been rapidly developed and widely applied in various applications in machine learning, signal and image processing, operations research, control, and other fields [CP11b, PB14, KP15, CP16b]. In this section, we review several first-order proximal algorithms and their connections. We start with four three-operator splitting algorithms for problem (2.1): the primal and dual variants of the Condat–Vũ algorithm [Con13, Vu13], the primal–dual three-operator (PD3O) algorithm [Yan18], and the primal–dual Davis–Yin (PDDY) algorithm [SCM20]. For each of the

four algorithms, we make connections with other first-order proximal algorithms, using reduction (*i.e.*, setting some parts in (2.1) to zero) and the "completion" reformulation [OV20]. The key idea in the "completion" trick is to reformulate the problem (2.1) (assuming $h = 0$ for simplicity) as

$$\text{minimize} \quad \tilde{f}(x, y) + \tilde{g}(x, y),$$

where

$$\tilde{f}(x, y) = f(x) + \delta_{\{0\}}(y), \qquad \tilde{g}(x, y) = g(Ax + By).$$

The auxiliary matrix B is chosen to satisfy

$$AA^T + BB^T = (1/\alpha)I \quad \text{with } \frac{1}{\alpha} \geq \|A\|_2^2.$$

Here $\|A\|_2$ is the spectral norm of A. After simplifications and use of the properties (2.10) and (2.11), the Douglas–Rachford splitting method applied to the reformulated problem will reduce to the iterations of PDHG. Detailed proofs and extensions can be found in [OV20].

The rest of this section is divided into three subsections, and each subsection discusses one primal–dual proximal splitting method. We focus on the formal connections between algorithms, and the presented connections do not necessarily provide the best approach for convergence analysis or the best known convergence results.

2.5.1 Condat–Vũ three-operator splitting algorithm

We start with the (primal) Condat–Vũ three-operator splitting algorithm, which was proposed independently by Condat [Con13] and Vũ [Vu13],

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau(A^T z^{(k)} + \nabla h(x^{(k)}))\big) \tag{2.12a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}\big(z^{(k)} + \sigma A(2x^{(k+1)} - x^{(k)})\big). \tag{2.12b}$$

The stepsizes σ and τ must satisfy

$$\sigma\tau\|A\|_2^2 + \tau L \leq 1,$$

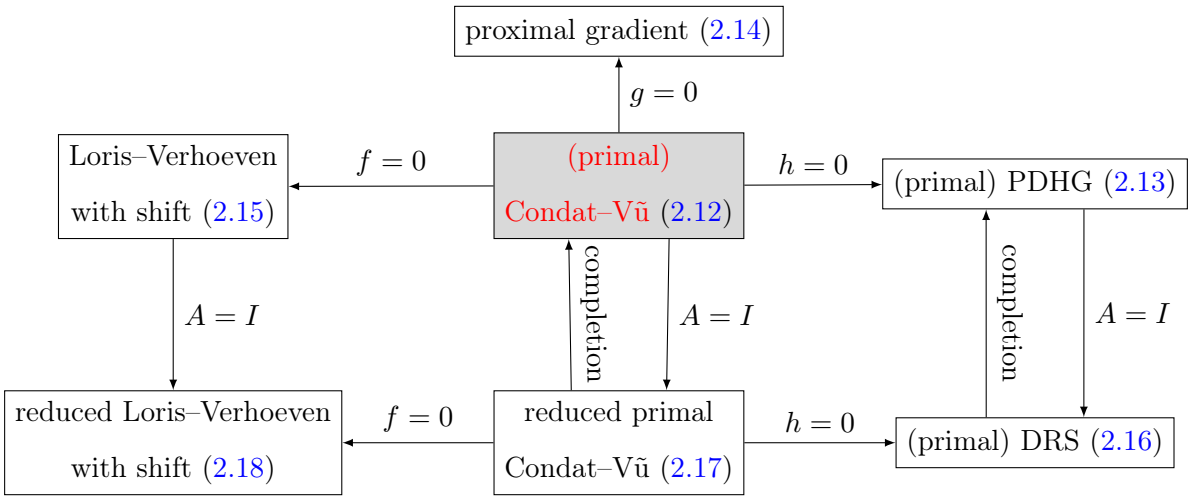

Figure 2.1: Proximal methods derived from primal Condat–Vũ algorithm.

where $\|A\|_2$ is the spectral norm of A, and L is the Lipschitz constant of ∇h with respect to the Euclidean norm. Many other first-order proximal algorithms can be viewed as special cases of (2.12), and their connections are summarized in Figure 2.1. When $h = 0$, algorithm (2.12) reduces to the (primal) primal–dual hybrid gradient (PDHG) method [PCB09, CP11a, CP16b], or PDHGMu in [EZC10]:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau A^T z^{(k)}\big) \tag{2.13a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}\big(z^{(k)} + \sigma A(2x^{(k+1)} - x^{(k)})\big). \tag{2.13b}$$

When $g = 0$ in (2.12) (and assuming $z^{(0)} = 0$), we apply the Moreau identity (2.11) and obtain the proximal gradient algorithm:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}(x^{(k)} - \tau \nabla h(x^{(k)})). \tag{2.14}$$

When $f = 0$, we obtain a variant of the Loris–Verhoeven algorithm [LV11, CHZ13, DST15],

$$x^{(k+1)} = x^{(k)} - \tau(A^T z^{(k)} + \nabla h(x^{(k)})) \tag{2.15a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}\big((I - \sigma\tau AA^T)z^{(k)} + \sigma A(x^{(k+1)} - \tau \nabla h(x^{(k)}))\big). \tag{2.15b}$$

We refer to this as the *Loris–Verhoeven algorithm with shift*, for reasons that will be clarified later. Furthermore, when $A = I$ and $\sigma = 1/\tau$ in PDHG, we obtain the Douglas–Rachford splitting (DRS) algorithm [LM79, EB92, CP07]:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau z^{(k)}\big) \tag{2.16a}$$

$$z^{(k+1)} = \mathbf{prox}_{\tau^{-1} g^*}\big(z^{(k)} + \frac{1}{\tau}(2x^{(k+1)} - x^{(k)})\big). \tag{2.16b}$$

Conversely, the "completion" technique in [OV20] shows that PDHG coincides with DRS applied to a reformulation of the problem. Similarly, when $A = I$ in the primal Condat–Vũ algorithm (2.12), we obtain a new algorithm and refer to it as the *reduced primal Condat–Vũ algorithm*:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau(z^{(k)} + \nabla h(x^{(k)}))\big) \tag{2.17a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}\big(z^{(k)} + \sigma(2x^{(k+1)} - x^{(k)})\big). \tag{2.17b}$$

Conversely, the reduced primal Condat–Vũ algorithm reverts to (2.12) via the "completion" trick. We can also set $f = 0$ in the reduced Condat–Vũ algorithm or $A = I$ in (2.15), and obtain the *reduced Loris–Verhoeven algorithm with shift*:

$$x^{(k+1)} = x^{(k)} - \tau(z^{(k)} + \nabla h(x^{(k)})) \tag{2.18a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}\big((1 - \sigma\tau)z^{(k)} + \sigma(x^{(k+1)} - \tau\nabla h(x^{(k)})\big). \tag{2.18b}$$

Finally, due to the absence of f in (2.18), it is not clear how to apply the "completion" trick to (2.18) to obtain (2.15).

Condat [Con13] also discusses a variant of (2.12), which we will call the dual Condat–Vũ algorithm:

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma A x^{(k)}) \tag{2.19a}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau(A^T(2z^{(k+1)} - z^{(k)}) + \nabla h(x^{(k)}))\big). \tag{2.19b}$$

Figure 2.2 summarizes the proximal algorithms derived from (2.19). When $h = 0$, algorithm (2.19) reduces to PDHG applied to the dual of (2.1) (with $h = 0$), which is shown to

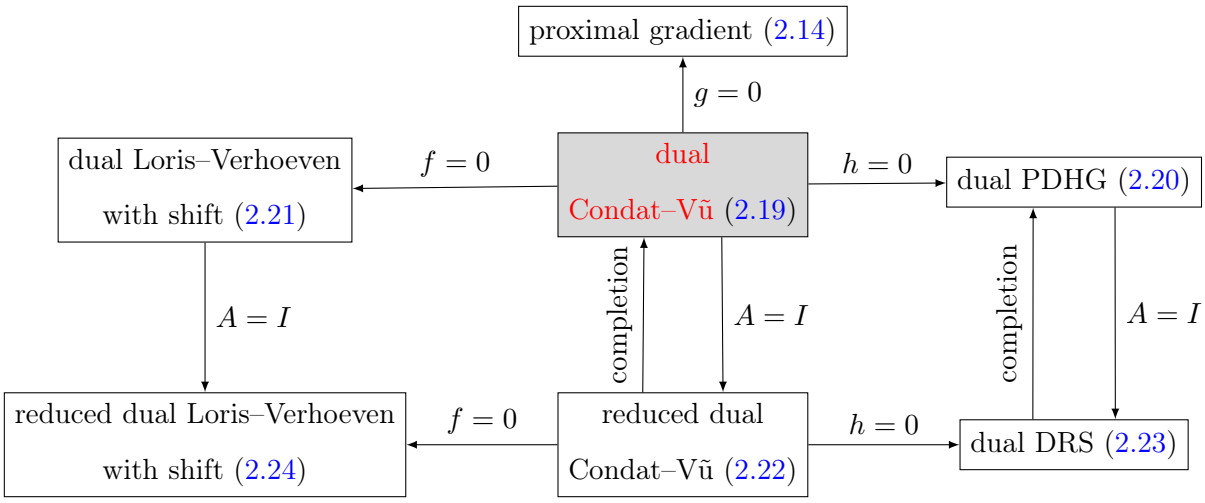

Figure 2.2: Proximal methods derived from dual Condat–Vũ algorithm.

be equivalent to linearized ADMM [PB14] (also called Split Inexact Uzawa in [EZC10]):

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma A x^{(k)}) \tag{2.20a}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau(A^T(2z^{(k+1)} - z^{(k)}))\big). \tag{2.20b}$$

Setting $g = 0$ in (2.19) yields the proximal gradient algorithm (2.14). When $f = 0$, we obtain a new algorithm:

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma A x^{(k)}) \tag{2.21a}$$

$$x^{(k+1)} = x^{(k)} - \tau(A^T(2z^{(k+1)} - z^{(k)}) + \nabla h(x^{(k)})). \tag{2.21b}$$

Following the previous naming convention, we call it *dual Loris–Verhoeven algorithm with shift*. Furthermore, setting $A = I$ in (2.19) gives the *reduced dual Condat–Vũ algorithm*:

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma x^{(k)}) \tag{2.22a}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau(2z^{(k+1)} - z^{(k)} + \nabla h(x^{(k)}))\big). \tag{2.22b}$$

Conversely, applying the "completion" trick to this reduced algorithm recovers (2.19). Similarly, setting $A = I$ in dual PDHG gives dual DRS:

$$z^{(k+1)} = \mathbf{prox}_{\tau^{-1} g^*}(z^{(k)} + \frac{1}{\tau} x^{(k)}) \tag{2.23a}$$

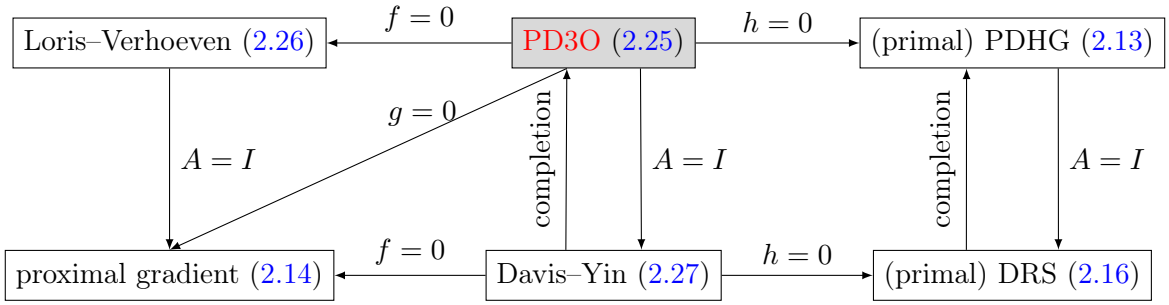

Figure 2.3: Proximal algorithms derived from PD3O.

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau(2z^{(k+1)} - z^{(k)})\big). \tag{2.23b}$$

It is just DRS (2.16) with f and g switched. Conversely, the "completion" trick recovers dual PDHG (2.20) from dual DRS (2.23). We can also set $A = I$ in (2.21) or $f = 0$ in the reduced dual Condat–Vũ algorithm, and obtain the *reduced dual Loris–Verhoeven algorithm with shift*:

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma x^{(k)}) \tag{2.24a}$$

$$x^{(k+1)} = x^{(k)} - \tau(2z^{(k+1)} - z^{(k)} + \nabla h(x^{(k)})). \tag{2.24b}$$

Again, owing to the lack of f in (2.24), it is unclear how to apply the "completion" trick to (2.24) to obtain (2.21).

2.5.2 Primal–dual three-operator (PD3O) splitting algorithm

The third diagram, Figure 2.3, starts with the primal–dual three-operator (PD3O) splitting algorithm [Yan18]

$$x^{(k+1)} = \mathbf{prox}_{\tau f}(x^{(k)} - \tau(A^T z^{(k)} + \nabla h(x^{(k)}))) \tag{2.25a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma A(2x^{(k+1)} - x^{(k)} + \tau\nabla h(x^{(k)}) - \tau\nabla h(x^{(k+1)}))). \tag{2.25b}$$

Compared with the Condat–Vũ algorithm (2.12), PD3O seems to have slightly more complicated updates and larger complexity per iteration, but the requirement for the stepsizes is

looser: $\sigma\tau\|A\|_2^2 \leq 1$ and $\tau \leq 1/L$. When $h = 0$, (2.25) reduces to the (primal) PDHG (2.13). The classical proximal gradient algorithm (2.14) can be obtained by setting $g = 0$. When $f = 0$, it reduces to the iterations

$$x^{(k+1)} = x^{(k)} - \tau(A^T z^{(k)} + \nabla h(x^{(k)})) \tag{2.26a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}\big((I - \sigma\tau AA^T)z^{(k)} + \sigma A(x^{(k+1)} - \tau\nabla h(x^{(k+1)}))\big). \tag{2.26b}$$

This algorithm was discovered independently as the Loris–Verhoeven algorithm [LV11], the primal–dual fixed point algorithm based on proximity operator (PDFP^{2}O) [CHZ13], and the proximal alternating predictor corrector (PAPC) [DST15]. Comparison with (2.15) reveals a minor difference between these two algorithms: the gradient term in the z-update is taken at the newest primal iterate $x^{(k+1)}$ in Loris–Verhoeven (2.26) and at the previous point $x^{(k)}$ in the shifted version. This difference is inherited in the proximal gradient algorithm (2.14) and its shifted version (2.18).

Furthermore, when $A = I$ and $\sigma = 1/\tau$ in PD3O, we recover the well-known Davis–Yin splitting (DYS) algorithm [DY17]:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}(x^{(k)} - \tau(z^{(k)} + \nabla h(x^{(k)}))) \tag{2.27a}$$

$$z^{(k+1)} = \mathbf{prox}_{\tau^{-1} g^*}(z^{(k)} + \frac{1}{\tau}(2x^{(k+1)} - x^{(k)}) + \nabla h(x^{(k)}) - \nabla h(x^{(k+1)})). \tag{2.27b}$$

We can also set $A = I$ in (2.26) and obtain the iterations

$$x^{(k+1)} = x^{(k)} - \tau(z^{(k)} + \nabla h(x^{(k)})) \tag{2.28a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}\big((1 - \sigma\tau)z^{(k)} + \sigma(x^{(k+1)} - \tau\nabla h(x^{(k+1)})\big). \tag{2.28b}$$

The stepsize conditions require $\sigma\tau \leq 1$ and $\tau \leq 1/L$. Thus we can set $\sigma = 1/\tau$ and apply Moreau decomposition. The resulting algorithm is exactly the proximal gradient algorithm (2.14). The only difference in the z-update between (2.18) and (2.28) is the point at which the gradient of h is taken. The second algorithm uses the most up-to-date iterate $x^{(k+1)}$ when evaluating the gradient of h, and this choice allows a larger stepsize τ.

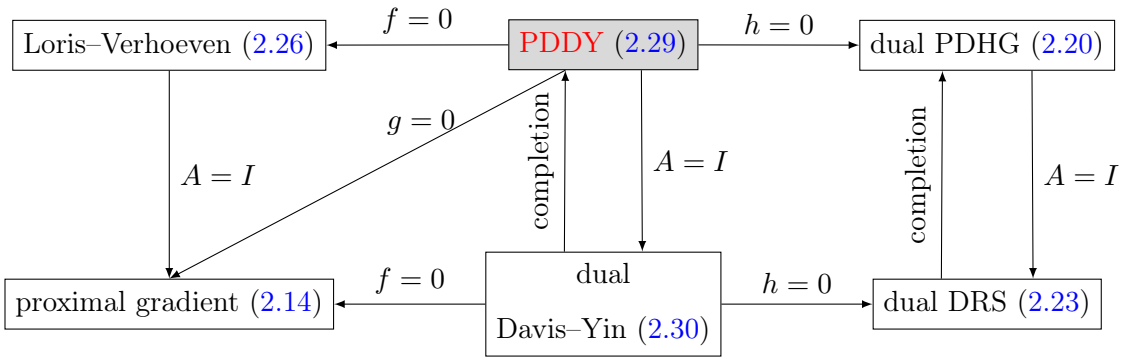

Figure 2.4: Proximal algorithms derived from PDDY.

2.5.3 Primal–dual Davis–Yin (PDDY) splitting algorithm

The core algorithm in Figure 2.4 is the primal–dual Davis–Yin (PDDY) splitting algorithm [SCM20]

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma A x^{(k)}) \tag{2.29a}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau A^T(2z^{(k+1)} - z^{(k)}) - \tau\nabla h(x^{(k)} + \tau A^T(z^{(k)} - z^{(k+1)}))\big). \tag{2.29b}$$

The requirement for stepsizes is the same as that in PD3O: $\sigma\tau\|A\|_2^2 \leq 1$ and $\tau \leq 1/L$. Figure 2.4 is almost identical to Figure 2.3 with the roles of f and g exchanged. When $h = 0$, PDDY reduces to the dual PDHG (2.20). In addition, when $A = I$ and $\sigma = 1/\tau$, PDDY reduces to the Davis–Yin algorithm, but with f and g exchanged:

$$z^{(k+1)} = \mathbf{prox}_{\tau^{-1}g^*}(z^{(k)} + \frac{1}{\tau}x^{(k)}) \tag{2.30a}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}\big(x^{(k)} - \tau(2z^{(k+1)} - z^{(k)} - \nabla h(x^{(k)} + \tau(z^{(k)} - z^{(k+1)}))\big). \tag{2.30b}$$

Similarly, when $h = 0$, $A = I$ and $\sigma = 1/\tau$, PDDY reverts to the Douglas–Rachford algorithm with f and g switched (2.23).

We have seen that the middle and right parts of Figure 2.4 are those of Figure 2.3 with f and g switched. However, when one of the functions f or g is absent, the algorithms reduced from PD3O and PDDY are exactly the same. In particular, when $f = 0$, PDDY reduces to the Loris–Verhoeven algorithm.

Proof. With a change of variables $u^{(k+1)} = x^{(k)} + \tau A^T(z^{(k)} - z^{(k+1)})$, (2.29) with $f = 0$ becomes

$$\begin{aligned} z^{(k+1)} &= \mathbf{prox}_{\sigma g^*}(z^{(k)} + \sigma A x^{(k)}) \\ u^{(k+1)} &= x^{(k)} + \tau A^T(z^{(k)} - z^{(k+1)}) \\ x^{(k+1)} &= u^{(k+1)} - \tau A^T z^{(k+1)} - \tau \nabla h(u^{(k+1)}). \end{aligned}$$

Eliminating x yields

$$\begin{aligned} z^{(k+1)} &= \mathbf{prox}_{\sigma g^*}\big((1 - \sigma\tau A A^T) z^{(k)} + \sigma A(u^{(k)} - \tau \nabla h(u^{(k)}))\big) \\ u^{(k+1)} &= u^{(k)} - \tau(A^T z^{(k+1)} + \nabla h(u^{(k)})), \end{aligned}$$

which is Loris–Verhoeven algorithm (2.26) with the order of updates switched. □

CHAPTER 3

Bregman proximal splitting algorithms

As seen in Chapter 2, primal–dual proximal splitting methods are prevalent in solving large-scale convex optimization problems. In these algorithms, the proximal operator is used to capture the structure of the problem, and its evaluation is in general the computational bottleneck in the implementation of proximal methods. To further improve the efficiency of proximal algorithms, proximal operators based on generalized Bregman distances have been proposed and incorporated in many methods [CT93, Eck93, Gul94, AT06, Tse08, BBT17, BST18, LFN18, Teb18]. A Bregman proximal operator of a function may be easier to compute than the standard Euclidean proximal operator, and hence reduce the complexity per iteration of an optimization algorithm. This idea has been exploited in the optimal transport problem [CLM21, CC22], where the relative entropy distance and its variant are chosen as the distance-generating kernel, and then the computation of the corresponding Bregman proximal operator is accelerated. Another example is the optimization problems over nonnegative trigonometric polynomials. In these problems, Itakura–Saito distance is used to formulate the Bregman proximal operator, and then advanced techniques in numerical linear algebra can be utilized in the computation of proximal operators [CV18]. As a second benefit of Bregman proximal operators, the generalized Bregman distance can help build a more accurate local optimization model around the current iterate. This is often interpreted as a form of preconditioning. For example, diagonal or quadratic preconditioning [PC11, JLL19, LXY21] has been shown to improve the convergence rate of PDHG, as well as the accuracy of the computed solution [ADH21].

In view of the two potential benefits of Bregman proximal operators, in this chapter we present new Bregman extensions and convergence results for the Condat–Vũ and PD3O algorithms. Specifically, the Condat–Vũ algorithm exists in a primal and dual variant. We discuss extensions of the two algorithms that use Bregman proximal operators in the primal and dual updates. The Bregman primal Condat–Vũ algorithm first appeared in [CP16b, Algorithm 1], and is also a special case of the algorithm proposed in [YA21] for a more general convex–concave saddle point problem. We give a new derivation of this method and its dual variant, by applying the Bregman proximal point method to the primal–dual optimality conditions. Based on the interpretation, we provide a unified framework for the convergence analysis of the two variants, and show an $O(1/k)$ ergodic convergence rate, which is consistent with previous results for Euclidean proximal operators in [Con13, Vu13] and Bregman proximal operators in [CP16b]. We also give a convergence result for the primal and dual iterates. Moreover, we propose an easily implemented backtracking line search technique for selecting stepsizes in the Bregman dual Condat–Vũ algorithm for problems with equality constraints. We give a detailed analysis of the algorithm with line search and obtain an $O(1/k)$ ergodic rate of convergence. Last, we propose a Bregman extension for PD3O and establish an ergodic convergence result.

This chapter is organized as follows. Section 3.1 provides some necessary background on Bregman distances. In Section 3.2 we discuss the Bregman primal and dual Condat–Vũ algorithms and analyze their convergence. The line search technique and its technique and its convergence are discussed in Section 3.3. In Section 3.4 we extend PD3O to a Bregman proximal method and analyze its convergence. Section 3.5 contains results of a numerical experiment. Most content of this chapter is adapted from [JV22b].

3.1 Bregman proximal operators

In this section we give the definition of Bregman divergence, Bregman proximal operators, and the basic properties that will be used in the dissertation. Bregman divergence was first introduced in [Bre67], and named by Censor and Lent [CL81]. Later, Bregman divergence has been widely used in first-order proximal methods [CZ92, CT93, Eck93, Gul94, Kiw97, Teb97, BBC03], and some other applications of Bregman divergence arise in statistics and information theory (see, for example, [BGW05] and references therein). We refer interested readers to [CZ97, Bub15] for an in-depth discussion of Bregman divergence, their history, and applications.

Let ϕ be a convex function with a domain that has nonempty interior, and assume ϕ is continuous on $\mathbf{dom}\,\phi$ and continuously differentiable on $\mathbf{int}(\mathbf{dom}\,\phi)$. The *generalized distance* (or *Bregman divergence*) generated by the *kernel function* ϕ is defined as the function

$$d(x,y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x-y\rangle,$$

with domain $\mathbf{dom}\,d = \mathbf{dom}\,\phi \times \mathbf{int}(\mathbf{dom}\,\phi)$. The distance generated by the kernel $\phi(x) = (1/2)\|x\|^2$ is the squared Euclidean distance $d(x,y) = (1/2)\|x-y\|^2$. The best known non-quadratic example is the relative entropy

$$d(x,y) = \sum_{i=1}^{n}(x_i\log(x_i/y_i) - x_i + y_i), \qquad \mathbf{dom}\,d = \mathbf{R}^n_+ \times \mathbf{R}^n_{++}. \tag{3.1}$$

This generalized distance is generated by the kernel $\phi(x) = \sum_i x_i \log x_i$.

Generalized distances are not necessarily symmetric ($d(x,y) \neq d(y,x)$), and thus some literature call it Bregman divergence instead of Bregman distance. But they still share some other important properties with the Euclidean distance. An important example is the *triangle identity* [CT93, Lemma 3.1]

$$\langle \nabla\phi(y) - \nabla\phi(z), x-y\rangle = d(x,z) - d(x,y) - d(y,z),$$

which holds for all $x \in \mathbf{dom}\,\phi$ and $y, z \in \mathbf{int}(\mathbf{dom}\,\phi)$. This generalizes the identity

$$\langle y - z, x - y \rangle = \frac{1}{2}\big(\|x - z\|^2 - \|x - y\|^2 - \|y - z\|^2\big).$$

Additional assumptions may have to be imposed on the kernel function ϕ, depending on the application and the algorithm in which the generalized distance is used. For now we only assume convexity, continuity, and continuously differentiability on the interior of the domain. Other properties will be mentioned when needed.

The *Bregman proximal operator* of a function f is

$$\mathbf{prox}_f^\phi(y, a) = \operatorname*{argmin}_x(f(x) + \langle a, x\rangle + d(x, y)) \tag{3.2}$$

$$= \operatorname*{argmin}_x(f(x) + \langle a, x\rangle + \phi(x) - \langle \nabla\phi(y), x\rangle). \tag{3.3}$$

It is assumed that for every a and every $y \in \mathbf{int}(\mathbf{dom}\,\phi)$ the minimizer $\hat{x} = \mathbf{prox}_f^\phi(y, a)$ is unique and in $\mathbf{int}(\mathbf{dom}\,\phi)$. From the expression (3.3) we see that $\hat{x} = \mathbf{prox}_f^\phi(y, a)$ satisfies

$$\nabla\phi(y) - \nabla\phi(\hat{x}) - a \in \partial f(\hat{x}).$$

Equivalently, by definition of subgradient,

$$\begin{aligned} f(x) + \langle a, x\rangle &\geq f(\hat{x}) + \langle a, \hat{x}\rangle + \langle \nabla\phi(y) - \nabla\phi(\hat{x}), x - \hat{x}\rangle \\ &= f(\hat{x}) + \langle a, \hat{x}\rangle + d(\hat{x}, y) + d(x, \hat{x}) - d(x, y) \end{aligned} \tag{3.4}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\,\phi$.

When $d(x, y) = \frac{1}{2}\|x - y\|^2$, the corresponding Bregman proximal operator is the standard proximal operator applied to $y - a$:

$$\mathbf{prox}_f^\phi(y, a) = \mathbf{prox}_f(y - a).$$

For this distance, closedness and convexity of f guarantee that the proximal operator is well defined [Mor65]. The questions of existence and uniqueness are more complicated for general Bregman distances. There are no simple general conditions that guarantee that for every a

and every $y \in \mathbf{int}(\mathbf{dom}\,\phi)$ the generalized proximal operator (3.2) is uniquely defined and in $\mathbf{int}(\mathbf{dom}\,\phi)$. Some sufficient conditions are provided (see, for example, [Bub15, Section 4.1], [BBT17, Assumption A]), but they may be quite restrictive or difficult to verify in practice. In applications, however, the Bregman proximal operator is used with specific combinations of f and ϕ, for which the minimization problem in (3.2) is particularly easy to solve. In those applications, existence and uniqueness of the solution follow directly from the closed-form solution or availability of a fast algorithm to compute it. A classical example is the relative entropy distance (3.1) with f given by the indicator function of the hyperplane $\{x \mid \mathbf{1}^T x = 1\}$. Problem (3.2) can be written as

$$\begin{array}{ll} \text{minimize} & a^T x + \sum_{i=1}^n (x_i \log(x_i/y_i) - x_i) \\ \text{subject to} & \mathbf{1}^T x = 1. \end{array}$$

For any a and any positive y, the solution of (3.2) is unique and equal to the positive vector

$$\mathbf{prox}_f^d(y, a) = \frac{1}{\sum_{i=1}^n y_i e^{-a_i}} \begin{bmatrix} y_1 e^{-a_1} \\ \vdots \\ y_n e^{-a_n} \end{bmatrix}. \tag{3.5}$$

3.2 Bregman Condat–Vũ three-operator splitting algorithms

We now discuss two Bregman three-operator splitting algorithms for the problem (2.1). The algorithms use a generalized distance d_{p} in the primal space, generated by a kernel ϕ_{p}, and a generalized distance d_{d} in the dual space, generated by a kernel ϕ_{d}. The first algorithm is

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\mathrm{p}}}\big(x^{(k)}, \tau A^T z^{(k)} + \tau \nabla h(x^{(k)})\big) \tag{3.6a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^{\phi_{\mathrm{d}}}\big(z^{(k)}, -\sigma A(2x^{(k+1)} - x^{(k)})\big) \tag{3.6b}$$

and will be referred to as the *Bregman primal Condat–Vũ algorithm*. The second algorithm will be called the *Bregman dual Condat–Vũ algorithm*:

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^{\phi_{\mathrm{d}}}(z^{(k)}, -\sigma A x^{(k)}) \tag{3.7a}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\mathrm{p}}}(x^{(k)}, \tau A^T(2z^{(k+1)} - z^{(k)}) + \tau \nabla h(x^{(k)})). \tag{3.7b}$$

The two algorithms need starting points $x^{(0)} \in \mathbf{int}(\mathbf{dom}\,\phi_{\mathrm{p}}) \cap \mathbf{dom}\,h$, and $z^{(0)} \in \mathbf{int}(\mathbf{dom}\,\phi_{\mathrm{d}})$. Conditions on stepsizes σ, τ will be specified later. When Euclidean distances are used for the primal and dual proximal operators, the two algorithms reduce to the primal and dual variants of the Condat–Vũ algorithm (2.12) and (2.19), respectively. Algorithm (3.6) has been proposed in [CP16a]. Here we discuss it together with (3.6) in a unified framework.

In Section 3.2.1 we show that the proposed algorithms can be interpreted as the Bregman proximal point method applied to a monotone inclusion problem. In Section 3.2.2 we analyze their convergence. In Section 3.2.3 we discuss the connections between the two algorithms and other Bregman proximal splitting methods.

Assumptions Throughout Section 3.2 we make the following assumptions. The kernels ϕ_{p} and ϕ_{d} are 1-strongly convex with respect to norms $\|\cdot\|_{\mathrm{p}}$ and $\|\cdot\|_{\mathrm{d}}$, respectively:

$$d_{\mathrm{p}}(x, x') \geq \frac{1}{2}\|x - x'\|_{\mathrm{p}}^2, \qquad d_{\mathrm{d}}(z, z') \geq \frac{1}{2}\|z - z'\|_{\mathrm{d}}^2 \tag{3.8}$$

for all $(x, x') \in \mathbf{dom}\,d_{\mathrm{p}}$ and $(z, z') \in \mathbf{dom}\,d_{\mathrm{d}}$. The assumption that the strong convexity constants are equal to one can be made without loss of generality, by scaling the norms (or distances) if needed. We also assume that the function $L\phi_{\mathrm{p}} - h$ is convex for some $L > 0$. More precisely, $\mathbf{dom}\,\phi_{\mathrm{p}} \subseteq \mathbf{dom}\,h$ and

$$h(x) - h(x') - \langle \nabla h(x'), x - x' \rangle \leq L d_{\mathrm{p}}(x, x') \quad \text{for all } (x, x') \in \mathbf{dom}\,d_{\mathrm{p}}. \tag{3.9}$$

Note that this assumption is looser than the one in [CP16a, Equation (4)]. We denote by $\|A\|$ the matrix norm

$$\|A\| = \sup_{u \neq 0, v \neq 0} \frac{\langle v, Au \rangle}{\|v\|_{\mathrm{d}} \|u\|_{\mathrm{p}}} = \sup_{u \neq 0} \frac{\|Au\|_{\mathrm{d},*}}{\|u\|_{\mathrm{p}}} = \sup_{v \neq 0} \frac{\|A^T v\|_{\mathrm{p},*}}{\|v\|_{\mathrm{d}}}, \tag{3.10}$$

where $\|\cdot\|_{\mathrm{p},*}$ and $\|\cdot\|_{\mathrm{d},*}$ are the dual norms of $\|\cdot\|_{\mathrm{p}}$ and $\|\cdot\|_{\mathrm{d}}$.

It is also assumed that the primal–dual optimality conditions (2.4) have a solution $(x^\star, z^\star)$ with $x^\star \in \mathbf{dom}\,\phi_{\mathrm{p}}$ and $z^\star \in \mathbf{dom}\,\phi_{\mathrm{d}}$.

3.2.1 Derivation from Bregman proximal point method

The Bregman Condat–Vũ algorithms (3.6) and (3.7) can be viewed as applications of the Bregman proximal point algorithm to the optimality conditions (2.4). This interpretation extends the derivation of the Bregman PDHG algorithm from the Bregman proximal point algorithm given in [JV22a]. The idea originates with He and Yuan's interpretation of PDHG as a "preconditioned" proximal point algorithm [HY12].

The Bregman proximal point algorithm [Eck93, CZ97, Gul94] is an algorithm for monotone inclusion problems $0 \in F(u)$. The update $u^{(k+1)}$ in one iteration of the algorithm is defined as the solution of the inclusion

$$\nabla\phi(u^{(k)}) - \nabla\phi(u^{(k+1)}) \in F(u^{(k+1)}),$$

where ϕ is a Bregman kernel function. Applied to (2.4), with a kernel function ϕ_{pd}, the algorithm generates a sequence $(x^{(k)}, z^{(k)})$ defined by

$$\nabla\phi_{\mathrm{pd}}(x^{(k)}, z^{(k)}) - \nabla\phi_{\mathrm{pd}}(x^{(k+1)}, z^{(k+1)}) \in \begin{bmatrix} A^T z^{(k+1)} + \partial f(x^{(k+1)}) + \nabla h(x^{(k+1)}) \\ -Ax^{(k+1)} + \partial g^*(z^{(k+1)}) \end{bmatrix}. \tag{3.11}$$

3.2.1.1 Primal–dual Bregman distances

We introduce four possible primal–dual kernel functions: the functions

$$\phi_+(x,z) = \frac{1}{\tau}\phi_{\mathrm{p}}(x) + \frac{1}{\sigma}\phi_{\mathrm{d}}(z) + \langle z, Ax\rangle, \qquad \phi_-(x,z) = \frac{1}{\tau}\phi_{\mathrm{p}}(x) + \frac{1}{\sigma}\phi_{\mathrm{d}}(z) - \langle z, Ax\rangle,$$

where $\sigma, \tau > 0$, and the functions

$$\phi_{\mathrm{dcv}}(x,z) = \phi_+(x,z) - h(x), \qquad \phi_{\mathrm{pcv}}(x,z) = \phi_-(x,z) - h(x).$$

The subscripts in ϕ_+ and ϕ_- refer to the sign of the inner product term $\langle z, Ax\rangle$. The subscripts in ϕ_{pcv} and ϕ_{dcv} indicate the algorithm (Bregman primal or dual Condat-Vũ) for which these distances will be relevant. If these kernel functions are convex, they generate

the following Bregman distances. The distances generated by ϕ_+ and ϕ_- are

$$d_+(x,z;x',z') = \frac{1}{\tau}d_{\mathrm{p}}(x,x') + \frac{1}{\sigma}d_{\mathrm{d}}(z,z') + \langle z-z', A(x-x')\rangle$$
$$d_-(x,z;x',z') = \frac{1}{\tau}d_{\mathrm{p}}(x,x') + \frac{1}{\sigma}d_{\mathrm{d}}(z,z') - \langle z-z', A(x-x')\rangle,$$

respectively, and the distances generated by ϕ_{dcv} and ϕ_{pcv} are

$$d_{\mathrm{dcv}}(x,z;x',z') = d_+(x,z;x',z') - h(x) + h(x') + \langle \nabla h(x'), x-x'\rangle$$
$$d_{\mathrm{pcv}}(x,z;x',z') = d_-(x,z;x',z') - h(x) + h(x') + \langle \nabla h(x'), x-x'\rangle.$$

We now show that ϕ_+ and ϕ_- are convex if

$$\sigma\tau\|A\|^2 \le 1$$

and strongly convex if $\sigma\tau\|A\|^2 < 1$, and that the functions ϕ_{dcv} and ϕ_{pcv} are convex if

$$\sigma\tau\|A\|^2 + \tau L \le 1 \tag{3.12}$$

and strongly convex if $\sigma\tau\|A\|^2 + \tau L < 1$.

Proof. To show that the kernel functions ϕ_+ and ϕ_- are convex, we show that d_+ and d_- are nonnegative. Suppose $\sigma\tau\|A\|^2 \le \delta^2$ with $0 < \delta \le 1$. Then (3.8) and the arithmetic–geometric mean inequality imply that

$$\begin{aligned}
|\langle z-z', A(x-x')\rangle| &\le \|A\|\|z-z'\|_{\mathrm{d}}\|x-x'\|_{\mathrm{p}} \\
&\le \frac{\delta}{\sqrt{\sigma\tau}}\|z-z'\|_{\mathrm{d}}\|x-x'\|_{\mathrm{p}} \\
&\le \frac{\delta}{2\tau}\|x-x'\|_{\mathrm{p}}^2 + \frac{\delta}{2\sigma}\|z-z'\|_{\mathrm{p}}^2 \\
&\le \frac{\delta}{\tau}d_{\mathrm{p}}(x,x') + \frac{\delta}{\sigma}d_{\mathrm{d}}(z,z').
\end{aligned} \tag{3.13}$$

Therefore

$$d_+(x,z;x',z') = \frac{1}{\tau}d_{\mathrm{p}}(x,x') + \frac{1}{\sigma}d_{\mathrm{d}}(z,z') + \langle z-z', A(x-x')\rangle$$

$$
\begin{aligned}
&\geq \frac{1-\delta}{\tau} d_{\mathrm{p}}(x, x') + \frac{1-\delta}{\sigma} d_{\mathrm{d}}(z, z') \\
&\geq \frac{1-\delta}{2\tau} \|x - x'\|_{\mathrm{p}}^2 + \frac{1-\delta}{2\sigma} \|z - z'\|_{\mathrm{d}}^2, \\
d_-(x, z; x', z') &= \frac{1}{\tau} d_{\mathrm{p}}(x, x') + \frac{1}{\sigma} d_{\mathrm{d}}(z, z') - \langle z - z', A(x - x') \rangle \\
&\geq \frac{1-\delta}{\tau} d_{\mathrm{p}}(x, x') + \frac{1-\delta}{\sigma} d_{\mathrm{d}}(z, z') \\
&\geq \frac{1-\delta}{2\tau} \|x - x'\|_{\mathrm{p}}^2 + \frac{1-\delta}{2\sigma} \|z - z'\|_{\mathrm{d}}^2.
\end{aligned}
$$

With $\delta = 1$, this shows convexity of ϕ_+ and ϕ_-; with $\delta < 1$, strong convexity.

Similarly, if $\sigma\tau\|A\|^2 \leq \delta(\delta - \tau L)$, with $0 < \delta \leq 1$, then

$$
\begin{aligned}
|\langle z - z', A(x - x') \rangle| &\leq \frac{\sqrt{\delta(\delta - \tau L)}}{\sqrt{\sigma\tau}} \|z - z'\|_{\mathrm{d}} \|x - x'\|_{\mathrm{p}} \\
&\leq \frac{\delta - \tau L}{2\tau} \|x - x'\|_{\mathrm{p}}^2 + \frac{\delta}{2\sigma} \|z - z'\|_{\mathrm{d}}^2 \\
&\leq \frac{\delta - \tau L}{\tau} d_{\mathrm{p}}(x, x') + \frac{\delta}{\sigma} d_{\mathrm{d}}(z, z')
\end{aligned}
$$

and

$$
\begin{aligned}
d_{\mathrm{dcv}}(x, z; x', z') &= \frac{1}{\tau} d_{\mathrm{p}}(x, x') + \frac{1}{\sigma} d_{\mathrm{d}}(z, z') + \langle z - z', A(x - x') \rangle \\
&\quad - h(x) + h(x') + \langle \nabla h(x'), x - x' \rangle \\
&\geq (\frac{1-\delta}{\tau} + L) d_{\mathrm{p}}(x, x') + \frac{1-\delta}{\sigma} d_{\mathrm{d}}(z, z') - h(x) + h(x') + \langle \nabla h(x'), x - x' \rangle \\
&\geq \frac{1-\delta}{\tau} d_{\mathrm{p}}(x, x') + \frac{1-\delta}{\sigma} d_{\mathrm{d}}(z, z'), \\
d_{\mathrm{pcv}}(x, z; x', z') &= \frac{1}{\tau} d_{\mathrm{p}}(x, x') + \frac{1}{\sigma} d_{\mathrm{d}}(z, z') - \langle z - z', A(x - x') \rangle \\
&\quad - h(x) + h(x') + \langle \nabla h(x'), x - x' \rangle \\
&\geq (\frac{1-\delta}{\tau} + L) d_{\mathrm{p}}(x, x') + \frac{1-\delta}{\sigma} d_{\mathrm{d}}(z, z') - h(x) + h(x') + \langle \nabla h(x'), x - x' \rangle \\
&\geq \frac{1-\delta}{\tau} d_{\mathrm{p}}(x, x') + \frac{1-\delta}{\sigma} d_{\mathrm{d}}(z, z').
\end{aligned}
$$

□

3.2.1.2 Bregman Condat-Vũ algorithms from proximal point method

The Bregman primal Condat–Vũ algorithm (3.6) is the Bregman proximal point method with the kernel function $\phi_{\mathrm{pd}} = \phi_{\mathrm{pcv}}$. If we take $\phi_{\mathrm{pd}} = \phi_{\mathrm{pcv}}$ in (3.11), we obtain two coupled inclusions that determine $x^{(k+1)}$, $z^{(k+1)}$. The first one is

$$\begin{aligned} 0 \;\in\; & \frac{1}{\tau}(\nabla\phi_{\mathrm{p}}(x^{(k+1)}) - \nabla\phi_{\mathrm{p}}(x^{(k)})) - A^T(z^{(k+1)} - z^{(k)}) - \nabla h(x^{(k+1)}) + \nabla h(x^{(k)}) \\ & + A^T z^{(k+1)} + \partial f(x^{(k+1)}) + \nabla h(x^{(k+1)}) \\ = \; & \frac{1}{\tau}(\nabla\phi_{\mathrm{p}}(x^{(k+1)}) - \nabla\phi_{\mathrm{p}}(x^{(k)})) + A^T z^{(k)} + \nabla h(x^{(k)}) + \partial f(x^{(k+1)}). \end{aligned}$$

This shows that $x^{(k+1)}$ solves the optimization problem

$$\text{minimize} \quad f(x) + \langle A^T z^{(k)} + \nabla h(x^{(k)}), x\rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)}).$$

The solution is the x-update (3.6a) in the Bregman primal Condat–Vũ method. The second inclusion is

$$\begin{aligned} 0 \;\in\; & \frac{1}{\sigma}(\nabla\phi_{\mathrm{d}}(z^{(k+1)}) - \nabla\phi_{\mathrm{d}}(z^{(k)})) - A(x^{(k+1)} - x^{(k)}) - Ax^{(k+1)} + \partial g^*(z^{(k+1)}) \\ = \; & \frac{1}{\sigma}(\nabla\phi_{\mathrm{d}}(z^{(k+1)}) - \nabla\phi_{\mathrm{d}}(z^{(k)})) - A(2x^{(k+1)} - x^{(k)}) + \partial g^*(z^{(k+1)}). \end{aligned}$$

This shows that $z^{(k+1)}$ solves the optimization problem

$$\text{minimize} \quad g^*(z) - \langle z, A(2x^{(k+1)} - x^{(k)})\rangle + \frac{1}{\sigma} d_{\mathrm{d}}(z, z^{(k)}).$$

The solution is the z-update (3.6b).

Choosing $\phi_{\mathrm{pd}} = \phi_{\mathrm{dcv}}$ in (3.11) yields the Bregman dual Condat–Vũ algorithm (3.7). Substituting $\phi_{\mathrm{pd}} = \phi_{\mathrm{dcv}}$ in (3.11) gives the inclusions

$$\begin{aligned} 0 \;\in\; & \frac{1}{\tau}(\nabla\phi_{\mathrm{p}}(x^{(k+1)}) - \nabla\phi_{\mathrm{p}}(x^{(k)})) + A^T(z^{(k+1)} - z^{(k)}) - \nabla h(x^{(k+1)}) + \nabla h(x^{(k)}) \\ & + A^T z^{(k+1)} + \partial f(x^{(k+1)}) + \nabla h(x^{(k+1)}) \\ = \; & \frac{1}{\tau}(\nabla\phi_{\mathrm{p}}(x^{(k+1)}) - \nabla\phi_{\mathrm{p}}(x^{(k)})) + A^T(2z^{(k+1)} - z^{(k)}) + \nabla h(x^{(k)}) + \partial f(x^{(k+1)}) \end{aligned}$$

and

$$\begin{aligned} 0 &\in \frac{1}{\sigma}(\nabla\phi_{\mathrm{d}}(z^{(k+1)}) - \nabla\phi_{\mathrm{d}}(z^{(k)})) + A(x^{(k+1)} - x^{(k)}) - Ax^{(k+1)} + \partial g^*(z^{(k+1)}) \\ &= \frac{1}{\sigma}(\nabla\phi_{\mathrm{d}}(z^{(k+1)}) - \nabla\phi_{\mathrm{d}}(z^{(k)})) - Ax^{(k)} + \partial g^*(z^{(k+1)}). \end{aligned}$$

The second inclusion shows that $z^{(k+1)}$ solves the optimization problem

$$\text{minimize} \quad g^*(z) + \langle z, Ax^{(k)} \rangle + \frac{1}{\sigma} d_{\mathrm{d}}(z, z^{(k)}).$$

The solution is given by the z-update (3.7a). Given $z^{(k+1)}$, one can solve the first inclusion for $x^{(k+1)}$, which involves the optimization problem

$$\text{minimize} \quad f(x) + \langle A^T(2z^{(k+1)} - z^{(k)}) + \nabla h(x^{(k)}), x \rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)}).$$

The solution is the x-update (3.7b).

3.2.2 Convergence analysis

The derivation in Section 3.2.1 allows us to apply existing convergence theory for the Bregman proximal point method for monotone inclusions to the proposed algorithms (3.6) and (3.7). The literature on the Bregman proximal point method for monotone inclusions [Eck93, Gul94, CZ97] focuses on the convergence of iterates, and this generally requires additional assumptions on ϕ_{p} and ϕ_{d} (beyond the assumptions of convexity made in Section 3.2.1). In this section we present a self-contained convergence analysis and give a direct proof of an $O(1/k)$ rate of ergodic convergence. We also give a self-contained proof of convergence of the iterates $x^{(k)}$ and $z^{(k)}$.

We make the assumptions listed in Section 3.2.1: the strong convexity assumption (3.8) for the primal and dual kernels ϕ_{p} and ϕ_{d}, and the *relative smoothness* property (3.9) of the function h. We assume that the stepsizes σ, τ satisfy (3.12), and that the primal–dual optimality condition (2.4) has a solution $(x^\star, z^\star) \in \mathbf{dom}\, \phi_{\mathrm{p}} \times \mathbf{dom}\, \phi_{\mathrm{d}}$.

For the sake of brevity we combine the analysis of the Bregman primal and the Bregman dual Condat-Vũ algorithms. In the following, d, $\tilde{d}$, $\tilde{\phi}$ are defined as

$$d = d_-, \qquad \tilde{d} = d_{\mathrm{pcv}}, \qquad \tilde{\phi} = \phi_{\mathrm{pcv}}$$

for the Bregman primal Condat–Vũ algorithm (3.6) and as

$$d = d_+, \qquad \tilde{d} = d_{\mathrm{dcv}}, \qquad \tilde{\phi} = \phi_{\mathrm{dcv}}$$

for the Bregman dual Condat–Vũ algorithm (3.7).

3.2.2.1 One-iteration analysis

We first show that the iterates $x^{(k+1)}$, $z^{(k+1)}$ generated by the Bregman Condat–Vũ algorithms (3.6) and (3.7) satisfy

$$\begin{aligned} &\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\ &\quad \leq \; d(x, z; x^{(k)}, z^{(k)}) - d(x, z; x^{(k+1)}, z^{(k+1)}) - \tilde{d}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}) \end{aligned} \tag{3.14}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$ and $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\mathrm{d}}$. More specifically, for Bregman primal Condat–Vũ algorithm, we have

$$\begin{aligned} &\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\ &\quad \leq \; d_-(x, z; x^{(k)}, z^{(k)}) - d_-(x, z; x^{(k+1)}, z^{(k+1)}) - d_{\mathrm{pcv}}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}), \end{aligned}$$

and for Bregman dual Condat–Vũ algorithm,

$$\begin{aligned} &\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\ &\quad \leq \; d_+(x, z; x^{(k)}, z^{(k)}) - d_+(x, z; x^{(k+1)}, z^{(k+1)}) - d_{\mathrm{dcv}}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}), \end{aligned}$$

Proof. We write (3.6) and (3.7) in a unified notation as

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\mathrm{p}}}(x^{(k)}, \tau(A^T \tilde{z} + \nabla h(x^{(k)}))) \tag{3.15a}$$

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}(z^{(k)}, -\sigma A\tilde{x}) \tag{3.15b}$$

where $\tilde{x}$ and $\tilde{z}$ are defined in the following table:

Bregman primal Condat–Vũ algorithm	$\tilde{x} = 2x^{(k+1)} - x^{(k)}$	$\tilde{z} = z^{(k)}$
Bregman dual Condat–Vũ algorithm	$\tilde{x} = x^{(k)}$	$\tilde{z} = 2z^{(k+1)} - z^{(k)}$.

The optimality condition (3.4) for the proximal operator evaluation (3.15a) is that

$$\tau(f(x^{(k+1)}) - f(x)) \leq d_{\mathrm{p}}(x, x^{(k)}) - d_{\mathrm{p}}(x^{(k+1)}, x^{(k)}) - d_{\mathrm{p}}(x, x^{(k+1)}) + \tau\langle A^T\tilde{z} + \nabla h(x^{(k)}), x - x^{(k+1)}\rangle$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$. The optimality condition for (3.15b) is that

$$\sigma(g^*(z^{(k+1)}) - g^*(z)) \leq d_{\mathrm{d}}(z, z^{(k)}) - d_{\mathrm{d}}(z^{(k+1)}, z^{(k)}) - d_{\mathrm{d}}(z, z^{(k+1)}) - \sigma\langle z - z^{(k+1)}, A\tilde{x}\rangle$$

for all $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\mathrm{d}}$. Combining the two inequalities gives

$$\begin{aligned}
&\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\
&\quad = f(x^{(k+1)}) - f(x) + h(x^{(k+1)}) - h(x) + g^*(z^{(k+1)}) - g^*(z) + \langle A^T z, x^{(k+1)}\rangle - \langle z^{(k+1)}, Ax\rangle \\
&\quad \leq \frac{1}{\tau}\Big(d_{\mathrm{p}}(x, x^{(k)}) - d_{\mathrm{p}}(x, x^{(k+1)}) - d_{\mathrm{p}}(x^{(k+1)}, x^{(k)})\Big) \\
&\qquad + \frac{1}{\sigma}\Big(d_{\mathrm{d}}(z, z^{(k)}) - d_{\mathrm{d}}(z, z^{(k+1)}) - d_{\mathrm{d}}(z^{(k+1)}, z^{(k)})\Big) \\
&\qquad + h(x^{(k+1)}) - h(x) + \langle \nabla h(x^{(k)}), x - x^{(k+1)}\rangle \\
&\qquad + \langle A^T\tilde{z}, x - x^{(k+1)}\rangle - \langle z - z^{(k+1)}, A\tilde{x}\rangle + \langle A^T z, x^{(k+1)}\rangle - \langle z^{(k+1)}, Ax\rangle \\
&\quad \leq \frac{1}{\tau}\Big(d_{\mathrm{p}}(x, x^{(k)}) - d_{\mathrm{p}}(x, x^{(k+1)}) - d_{\mathrm{p}}(x^{(k+1)}, x^{(k)})\Big) \\
&\qquad + \frac{1}{\sigma}\Big(d_{\mathrm{d}}(z, z^{(k)}) - d_{\mathrm{d}}(z, z^{(k+1)}) - d_{\mathrm{d}}(z^{(k+1)}, z^{(k)})\Big) \\
&\qquad + h(x^{(k+1)}) - h(x^{(k)}) + \langle \nabla h(x^{(k)}), x^{(k+1)} - x^{(k)}\rangle \\
&\qquad + \langle A^T\tilde{z}, x - x^{(k+1)}\rangle - \langle z - z^{(k+1)}, A\tilde{x}\rangle + \langle A^T z, x^{(k+1)}\rangle - \langle z^{(k+1)}, Ax\rangle
\end{aligned} \tag{3.16}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$ and all $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\mathrm{d}}$. The second inequality follows from convexity of h. Substituting the expressions for $\tilde{x}$ and $\tilde{z}$ in the Bregman primal Condat–Vũ algorithm (3.6), we obtain on the last line of (3.16)

$$\langle A^T\tilde{z}, x - x^{(k+1)}\rangle - \langle z - z^{(k+1)}, A\tilde{x}\rangle + \langle A^T z, x^{(k+1)}\rangle - \langle z^{(k+1)}, Ax\rangle$$

$$
\begin{aligned}
&= \langle z^{(k)}, A(x - x^{(k+1)})\rangle - \langle z - z^{(k+1)}, A(2x^{(k+1)} - x^{(k)})\rangle + \langle A^T z, x^{(k+1)}\rangle - \langle z^{(k+1)}, Ax\rangle \\
&= \langle z^{(k)} - z^{(k+1)}, A(x - x^{(k+1)})\rangle + \langle z - z^{(k+1)}, A(x^{(k)} - x^{(k+1)})\rangle \\
&= -\langle z - z^{(k)}, A(x - x^{(k)})\rangle + \langle z - z^{(k+1)}, A(x - x^{(k+1)})\rangle + \langle z^{(k+1)} - z^{(k)}, A(x^{(k+1)} - x^{(k)})\rangle.
\end{aligned}
$$

Then for Bregman primal Condat–Vũ algorithm, (3.16) implies

$$
\begin{aligned}
&\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\
&\quad\leq \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)}) + \frac{1}{\sigma} d_{\mathrm{d}}(z, z^{(k)}) - \langle z - z^{(k)}, A(x - x^{(k)})\rangle \\
&\qquad - \Big(\frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k+1)}) + \frac{1}{\sigma} d_{\mathrm{d}}(z, z^{(k+1)}) - \langle z - z^{(k+1)}, A(x - x^{(k+1)})\rangle\Big) \\
&\qquad - \Big(\frac{1}{\tau} d_{\mathrm{p}}(x^{(k+1)}, x^{(k)}) + \frac{1}{\sigma} d_{\mathrm{d}}(z^{(k+1)}, z^{(k)}) - \langle z^{(k+1)} - z^{(k)}, A(x^{(k+1)} - x^{(k)})\rangle\Big) \\
&\qquad + h(x^{(k+1)}) - h(x^{(k)}) - \langle \nabla h(x^{(k)}), x^{(k+1)} - x^{(k)}\rangle \\
&\quad= d_-(x, z; x^{(k)}, z^{(k)}) - d_-(x, z; x^{(k+1)}, z^{(k+1)}) - d_{\mathrm{pcv}}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}).
\end{aligned}
$$

Similarly, if we substitute the expressions for $\tilde{x}$ and $\tilde{z}$ in the Bregman dual Condat–Vũ algorithm, the last line of (3.16) becomes

$$
\begin{aligned}
&\langle A^T \tilde{z}, x - x^{(k+1)}\rangle - \langle z - z^{(k+1)}, A\tilde{x}\rangle + \langle A^T z, x^{(k+1)}\rangle - \langle z^{(k+1)}, Ax\rangle \\
&\quad= \langle A^T(z - z^{(k)}), x - x^{(k)}\rangle - \langle A^T(z - z^{(k+1)}), x - x^{(k+1)}\rangle - \langle A^T(z^{(k+1)} - z^{(k)}), x^{(k+1)} - x^{(k)}\rangle,
\end{aligned}
$$

and then (3.16) implies that

$$
\begin{aligned}
&\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\
&\quad\leq \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)}) + \frac{1}{\sigma} d_{\mathrm{d}}(z, z^{(k)}) + \langle z - z^{(k)}, A(x - x^{(k)})\rangle \\
&\qquad - \Big(\frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k+1)}) + \frac{1}{\sigma} d_{\mathrm{d}}(z, z^{(k+1)}) + \langle z - z^{(k+1)}, A(x - x^{(k+1)})\rangle\Big) \\
&\qquad - \Big(\frac{1}{\tau} d_{\mathrm{p}}(x^{(k+1)}, x^{(k)}) + \frac{1}{\sigma} d_{\mathrm{d}}(z^{(k+1)}, z^{(k)}) + \langle z^{(k+1)} - z^{(k)}, A(x^{(k+1)} - x^{(k)})\rangle\Big) \\
&\qquad + h(x^{(k+1)}) - h(x^{(k)}) - \langle \nabla h(x^{(k)}), x^{(k+1)} - x^{(k)}\rangle \\
&\quad= d_+(x, z; x^{(k)}, z^{(k)}) - d_+(x, z; x^{(k+1)}, z^{(k+1)}) - d_{\mathrm{dcv}}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}).
\end{aligned}
$$

□

3.2.2.2 Ergodic convergence

We define averaged iterates

$$x_{\text{avg}}^{(k)} = \frac{1}{k}\sum_{i=1}^{k} x^{(i)}, \qquad z_{\text{avg}}^{(k)} = \frac{1}{k}\sum_{i=1}^{k} z^{(i)} \tag{3.17}$$

for $k \geq 1$. We show that for both algorithms (3.6) and (3.7),

$$\mathcal{L}(x_{\text{avg}}^{(k)}, z) - \mathcal{L}(x, z_{\text{avg}}^{(k)}) \leq \frac{2}{k}\Big(\frac{1}{\tau}d_{\text{p}}(x, x^{(0)}) + \frac{1}{\sigma}d_{\text{d}}(z, z^{(0)})\Big) \tag{3.18}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\text{p}}$ and $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\text{d}}$.

Proof. From (3.14), since $\mathcal{L}(u, v)$ is convex in u and concave in v,

$$\begin{aligned}
\mathcal{L}(x_{\text{avg}}^{(k)}, z) - \mathcal{L}(x, z_{\text{avg}}^{(k)}) &\leq \frac{1}{k}\sum_{i=1}^{k}\big(\mathcal{L}(x^{(i)}, z) - \mathcal{L}(x, z^{(i)})\big) \\
&\leq \frac{1}{k}\big(d(x, z; x^{(0)}, z^{(0)}) - d(x, z; x^{(k)}, z^{(k)})\big) \\
&\leq \frac{1}{k}d(x, z; x^{(0)}, z^{(0)}) \\
&\leq \frac{2}{k}\Big(\frac{1}{\tau}d_{\text{p}}(x, x^{(0)}) + \frac{1}{\sigma}d_{\text{d}}(z, z^{(0)})\Big)
\end{aligned}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\text{p}}$ and $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\text{d}}$. The last step follows from (3.13). □

Substituting $x = x^\star$, $z = z^\star$ in (3.18) gives

$$\mathcal{L}(x_{\text{avg}}^{(k)}, z^\star) - \mathcal{L}(x^\star, z_{\text{avg}}^{(k)}) \leq \frac{2}{k}\Big(\frac{1}{\tau}d_{\text{p}}(x^\star, x^{(0)}) + \frac{1}{\sigma}d_{\text{d}}(z^\star, z^{(0)})\Big).$$

More generally, if $\mathcal{X} \subseteq \mathbf{dom}\, \phi_{\text{p}}$ and $\mathcal{Z} \subseteq \mathbf{dom}\, \phi_{\text{d}}$ are compact sets that contain optimal solutions $x^\star$, $z^\star$, respectively, then the merit function η (2.7) is upper bounded by

$$\eta(x_{\text{avg}}^{(k)}, z_{\text{avg}}^{(k)}) \leq \frac{2}{k}\Big(\frac{1}{\tau}\sup_{x\in\mathcal{X}} d_{\text{p}}(x, x^{(0)}) + \frac{1}{\sigma}\sup_{z\in\mathcal{Z}} d_{\text{d}}(z, z^{(0)})\Big).$$

3.2.2.3 Monotonicity properties

For $x = x^\star$, $z = z^\star$, the left-hand side of (3.14) is nonnegative and therefore

$$d(x^\star, z^\star; x^{(k+1)}, z^{(k+1)}) \leq d(x^\star, z^\star; x^{(k)}, z^{(k)}) - \tilde{d}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}) \tag{3.19}$$

for $k \geq 0$. Hence $d(x^\star, z^\star; x^{(k+1)}, z^{(k+1)}) \leq d(x^\star, z^\star; x^{(k)}, z^{(k)})$ and

$$d(x^\star, z^\star; x^{(k)}, z^{(k)}) \leq d(x^\star, z^\star; x^{(0)}, z^{(0)}). \tag{3.20}$$

The inequality (3.19) also implies that

$$\sum_{i=0}^{k} \tilde{d}(x^{(i+1)}, z^{(i+1)}; x^{(i)}, z^{(i)}) \leq d(x^\star, z^\star; x^{(0)}, z^{(0)}).$$

Hence $\tilde{d}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}) \to 0$.

3.2.2.4 Convergence of iterates

Convergence of iterates can be shown under additional assumptions about the primal and dual distance functions. The following two assumptions are common in the literature on Bregman distances [CT93, Eck93, Gul94, CZ97].

1. For fixed x and z, the sublevel sets

$$\{x' \mid d_{\mathrm{p}}(x, x') \leq \gamma\}, \qquad \text{and} \qquad \{z' \mid d_{\mathrm{d}}(z, z') \leq \gamma\}$$

   are closed. In other words, the distances $d_{\mathrm{p}}(x, x')$ and $d_{\mathrm{d}}(z, z')$ are closed functions of x' and z', respectively. Since a sum of closed functions is closed, the distance $d(x, z; x', z')$ is a closed function of (x', z'), for fixed (x, z).

2. If $\tilde{x}^{(k)} \in \mathbf{int}(\mathbf{dom}\, \phi_{\mathrm{p}})$ converges to $x \in \mathbf{dom}\, \phi_{\mathrm{p}}$, then $d_{\mathrm{p}}(x, \tilde{x}^{(k)}) \to 0$. Similarly, if $\tilde{z}^{(k)} \in \mathbf{int}(\mathbf{dom}\, \phi_{\mathrm{d}})$ converges to $z \in \mathbf{dom}\, \phi_{\mathrm{d}}$, then $d_{\mathrm{d}}(z, \tilde{z}^{(k)}) \to 0$.

We also assume that $\sigma\tau\|A\|^2 + \tau L < 1$. As shown in Section 3.2.1.1 this implies that the kernel functions ϕ_{pcv} and ϕ_{dcv} are strongly convex and that

$$\tilde{d}(x, z; x', z') \geq \frac{\alpha}{2\tau}\|x - x'\|_{\text{p}}^2 + \frac{\alpha}{2\sigma}\|z - z'\|_{\text{d}}^2 \tag{3.21}$$

for some $\alpha > 0$. Similarly, $\sigma\tau\|A\|^2 < 1$ implies that

$$d(x, z; x', z') \geq \frac{\beta}{2\tau}\|x - x'\|_{\text{p}}^2 + \frac{\beta}{2\sigma}\|z - z'\|_{\text{d}}^2 \tag{3.22}$$

for some $\beta > 0$. Recall that $d = d_-$, $\tilde{d} = d_{\text{pcv}}$ for the Bregman primal Condat–Vũ algorithm (3.6), and $d = d_+$, $\tilde{d} = d_{\text{dcv}}$ for the Bregman dual Condat–Vũ algorithm.

Proof. We first note that $\tilde{d}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}) \to 0$ and (3.21) imply that

$$x^{(k+1)} - x^{(k)} \to 0, \qquad \text{and} \qquad z^{(k+1)} - z^{(k)} \to 0.$$

The inequality (3.20), together with (3.22), implies that the sequence $(x^{(k)}, z^{(k)})$ is bounded. Let $(x^{(k_i)}, z^{(k_i)})$ be a convergent subsequence of $(x^{(k)}, z^{(k)})$ with limit $(\hat{x}, \hat{z})$. Since $x^{(k_i+1)} - x^{(k_i)} \to 0$ and $z^{(k_i+1)} - z^{(k_i)} \to 0$, the sequence $(x^{(k_i+1)}, z^{(k_i+1)})$ also converges to $(\hat{x}, \hat{z})$. We show that $(\hat{x}, \hat{z})$ satisfies the optimality condition (2.4).

From (3.20), $d(x^\star, z^\star; x^{(k_i)}, z^{(k_i)})$ is bounded. Since the sublevel sets

$$\{(x', z') \mid d(x^\star, z^\star; x', z') \leq \gamma\}$$

are closed subsets of $\mathbf{int}(\mathbf{dom}\,\phi_{\text{p}}) \cap \mathbf{int}(\mathbf{dom}\,\phi_{\text{d}})$, so is the limit:

$$(\hat{x}, \hat{z}) \in \mathbf{int}(\mathbf{dom}\,\phi_{\text{p}}) \cap \mathbf{int}(\mathbf{dom}\,\phi_{\text{d}}).$$

The iterates in the subsequence satisfy

$$\nabla\phi_{\text{pd}}(x^{(k_i)}, z^{(k_i)}) - \nabla\phi_{\text{pd}}(x^{(k_i+1)}, z^{(k_i+1)}) + \begin{bmatrix} -A^T z^{(k_i+1)} \\ Ax^{(k_i+1)} \end{bmatrix} \in \begin{bmatrix} \partial f(x^{(k_i+1)}) + \nabla h(x^{(k_i+1)}) \\ \partial g^*(z^{(k_i+1)}) \end{bmatrix}, \tag{3.23}$$

where $\phi_{\rm pd} = \phi_{\rm pcv}$ in the Bregman primal Condat–Vũ algorithm and $\phi_{\rm pd} = \phi_{\rm dcv}$ in the Bregman dual Condat–Vũ algorithm. The left-hand side of (3.23) converges to $(-A^T\hat{z}, A\hat{x})$ because $\nabla\phi_{\rm pd}$ is continuous on $\mathbf{int}(\mathbf{dom}\,\phi_{\rm pd})$. Since the operator on right-hand side of (3.23) is maximal monotone the limit point $(\hat{x}, \hat{z})$ satisfies the optimality condition

$$\begin{bmatrix} -A^T\hat{z} \\ A\hat{x} \end{bmatrix} \in \begin{bmatrix} \partial f(\hat{x}) + \nabla h(\hat{x}) \\ \partial g^*(\hat{z}) \end{bmatrix}$$

(see [Bre73, page 27], [Tse00, Lemma 3.2]).

To show convergence of the entire sequence $(x^{(k)}, z^{(k)})$, we substitute $(\hat{x}, \hat{z})$ in (3.14):

$$\mathcal{L}(x^{(k+1)}, \hat{z}) - \mathcal{L}(\hat{x}, z^{(k+1)}) \leq d(\hat{x}, \hat{z}; x^{(k)}, z^{(k)}) - d(\hat{x}, \hat{z}; x^{(k+1)}, z^{(k+1)}).$$

Since the left-hand side is nonnegative, we have $d(\hat{x}, \hat{z}; x^{(k)}, z^{(k)}) \leq d(\hat{x}, \hat{z}; x^{(k-1)}, z^{(k-1)})$ for all $k \geq 1$. This further implies that

$$d(\hat{x}, \hat{z}; x^{(k)}, z^{(k)}) \leq d(\hat{x}, \hat{z}; x^{(k_i)}, z^{(k_i)})$$

for all $k \geq k_i$. By the second additional assumption mentioned above, the right-hand side converges to zero. Then the left-hand side also converges to zero and, from (3.22) $x^{(k)} \to \hat{x}$ and $z^{(k)} \to \hat{z}$. □

3.2.3 Relation to other Bregman proximal splitting algorithms

Following similar steps as in Section 2.5, we obtain several Bregman proximal splitting methods as special cases of (3.6) and (3.7). The connections are summarized in Figure 3.1 and Figure 3.2. A comparison of Figures 2.1 and 3.1 shows that all the reduction relations ($A = I$) are still valid. However, it is unclear how to apply the "completion" operation to algorithms based on non-Euclidean Bregman distances.

When $h = 0$, (3.6) reduces to Bregman PDHG [CP16b]:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\rm p}}(x^{(k)}, \tau A^T z^{(k)}) \tag{3.24a}$$

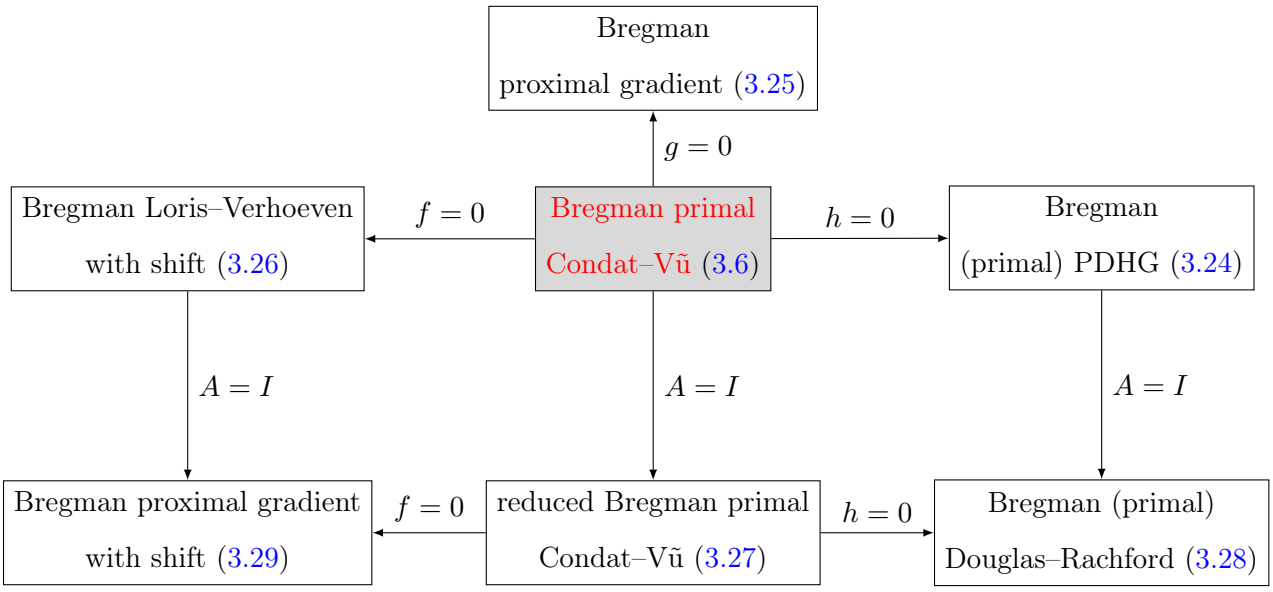

Figure 3.1: Proximal algorithms derived from Bregman primal Condat–Vũ algorithm (3.6).

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}(z^{(k)}, -\sigma A(2x^{(k+1)} - x^{(k)})). \tag{3.24b}$$

When $g = 0$, $g^* = \delta_{\{0\}}$ (and assuming $z^{(0)} = 0$), we obtain the Bregman proximal gradient algorithm [BBT17]:

$$x^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{p}}}_{\tau f}(x^{(k)}, \tau \nabla h(x^{(k)})). \tag{3.25}$$

When $f = 0$ in (3.6), we obtain the *Bregman Loris–Verhoeven algorithm with shift:*

$$x^{(k+1)} = \operatorname*{argmin}_x \big(\langle \nabla h(x^{(k)}) - A^T z^{(k)}, x\rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)})\big) \tag{3.26a}$$

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}\big(z^{(k)}, -\sigma A(2x^{(k+1)} - x^{(k)})\big). \tag{3.26b}$$

Furthermore, when $A = I$ in (3.6), we recover the reduced Bregman primal Condat–Vũ algorithm:

$$x^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{p}}}_{\tau f}\big(x^{(k)}, \tau(z^{(k)} + \nabla h(x^{(k)})\big) \tag{3.27a}$$

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}\big(z^{(k)}, -\sigma(2x^{(k+1)} - x^{(k)})\big). \tag{3.27b}$$

Similarly, setting $A = I$ in Bregman PDHG (3.24) yields the Bregman Douglas–Rachford

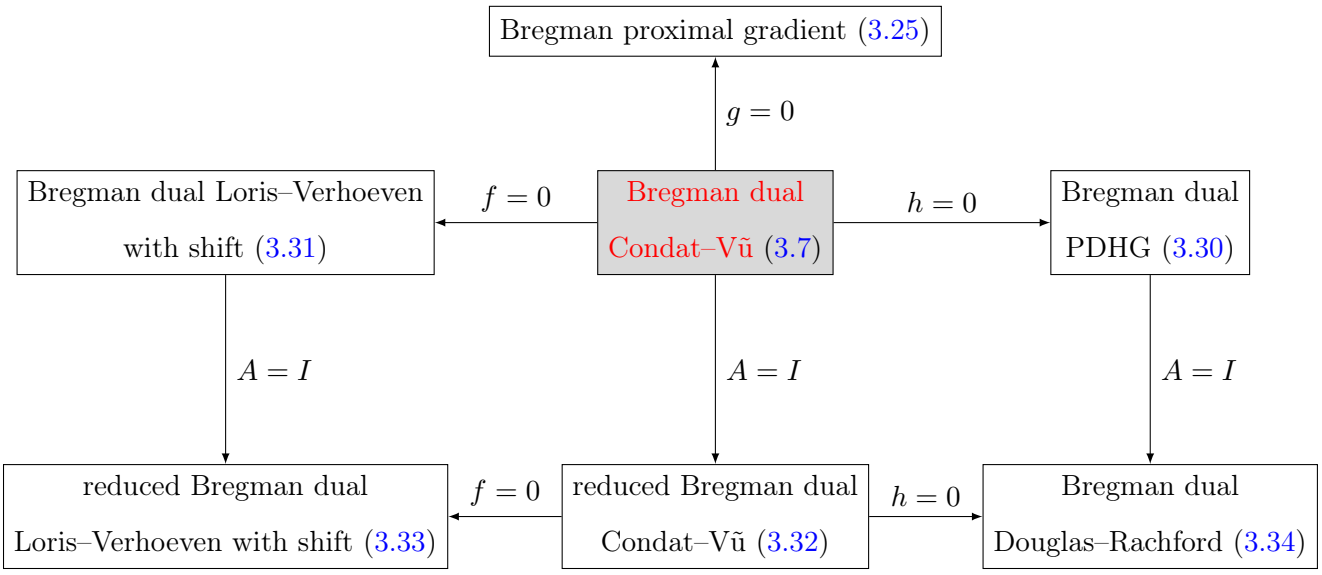

Figure 3.2: Proximal algorithms derived from Bregman dual Condat–Vũ algorithm (3.7).

algorithm:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\mathrm{p}}}(x^{(k)}, \tau z^{(k)}) \tag{3.28a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^{\phi_{\mathrm{d}}}(z^{(k)}, -\sigma(2x^{(k+1)} - x^{(k)})). \tag{3.28b}$$

Last, when we set $A = I$ in (3.26), we have the *Bregman reduced Loris–Verhoeven algorithm with shift:*

$$x^{(k+1)} = \operatorname*{argmin}_x \big(\langle \nabla h(x^{(k)}) - z^{(k)}, x\rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)})\big) \tag{3.29a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^{\phi_{\mathrm{d}}}\big(z^{(k)}, -\sigma(2x^{(k+1)} - x^{(k)})\big). \tag{3.29b}$$

Similarly, the Bregman dual Condat–Vũ algorithm (3.7) can be reduced to some other Bregman proximal splitting methods, as summarized in Figure 3.2. When $h = 0$, (3.7) reduces to the Bregman dual PDHG:

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^{\phi_{\mathrm{d}}}(z^{(k)}, -\sigma A x^{(k)}) \tag{3.30a}$$

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\mathrm{p}}}(x^{(k)}, \tau A^T(2z^{(k+1)} - x^{(k)})). \tag{3.30b}$$

When $g = 0$, $g^* = \delta_{\{0\}}$ (and assuming $z^{(0)} = 0$), we obtain the Bregman proximal gradient algorithm (3.25). When $f = 0$ in (3.7), we obtain the *Bregman dual Loris–Verhoeven algorithm with shift*:

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}(z^{(k)}, -\sigma A x^{(k)}) \tag{3.31a}$$

$$x^{(k+1)} = \operatorname*{argmin}_x \big(\langle A^T(2z^{(k+1)} - z^{(k)}) + \nabla h(x^{(k)}), x \rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)}) \big). \tag{3.31b}$$

Moreover, if we set $A = I$ in the Bregman dual Condat–Vũ algorithm (3.7), we obtain its reduced variant:

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}(z^{(k)}, -\sigma x^{(k)}) \tag{3.32a}$$

$$x^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{p}}}_{\tau f}\big(x^{(k)}, \tau(2z^{(k+1)} - x^{(k)} + \nabla h(x^{(k)}))\big). \tag{3.32b}$$

Similarly setting $A = I$ in (3.31) yields the *reduced Bregman Loris–Verhoeven algorithm with shift:*

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}(z^{(k)}, -\sigma x^{(k)}) \tag{3.33a}$$

$$x^{(k+1)} = \operatorname*{argmin}_x \big(\langle 2z^{(k+1)} - z^{(k)} + \nabla h(x^{(k)}), x \rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)}) \big), \tag{3.33b}$$

and setting $A = I$ in (3.30) gives the Bregman dual Douglas–Rachford algorithm:

$$z^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{d}}}_{\sigma g^*}(z^{(k)}, -\sigma x^{(k)}) \tag{3.34a}$$

$$x^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{p}}}_{\tau f}(x^{(k)}, \tau(2z^{(k+1)} - x^{(k)})). \tag{3.34b}$$

3.3 Bregman dual Condat–Vũ algorithm with line search

The algorithms (3.6) and (3.7) use constant parameters σ and τ. The stepsize condition (3.12) involves the matrix norm $\|A\|$ and the Lipschitz constant L in (3.9). Estimating or bounding $\|A\|$ for a large matrix can be difficult. As an added complication, the norms $\|\cdot\|_{\mathrm{p}}$ and $\|\cdot\|_{\mathrm{d}}$ in the definition of the matrix norm (3.10) are assumed to be scaled so that the strong convexity parameters of the primal and dual kernels are equal to one. Close bounds on

the strong convexity parameters may also be difficult to obtain. Using conservative bounds for $\|A\|$ and L results in unnecessarily small values of σ and τ, and can dramatically slow down the convergence. Even when the estimates of $\|A\|$ and L are accurate, the requirements for the stepsizes (3.12) are still too strict in most iterations, as observed in [ADH21]. In view of the above arguments, line search techniques for primal–dual proximal methods have recently become an active area of research. Malitsky and Pock [MP18] proposed a line search technique for PDHG and the Condat–Vũ algorithm in the Euclidean case. The algorithm with adaptive parameters in [VMC21] focuses on a special case of (2.1) (*i.e.*, $f = 0$) and extends the Loris–Verhoeven algorithm (2.26). A Bregman proximal splitting method with line search is discussed in [JV22a] and considers the problem (2.1) with $h = 0$ and $g = \delta_{\{b\}}$. In this section, we extend the Bregman dual Condat–Vũ algorithm (3.7) with a varying parameter option, in which the stepsizes are chosen adaptively without requiring any estimates or bounds for $\|A\|$ or the strong convexity parameter of the kernels. The algorithm is restricted to problems in the equality constrained form

$$\begin{aligned} &\text{minimize} && f(x) + h(x) \\ &\text{subject to} && Ax = b. \end{aligned} \tag{3.35}$$

This is a special case of (2.1) with $g = \delta_{\{b\}}$, the indicator function of the singleton $\{b\}$.

The details of the algorithm are discussed in Section 3.3.1 and a convergence analysis is presented in Section 3.3.2. The main conclusion is an $O(1/k)$ rate of ergodic convergence, consistent with previous results for related algorithms [MP18, JV22a].

Assumptions We make the same assumptions as in Section 3.2.1, but define

$$\phi_{\mathrm{d}}(z) = \frac{1}{2}\|z\|^2, \qquad d_{\mathrm{d}}(z, z') = \frac{1}{2}\|z - z'\|^2, \qquad \|z\|_{\mathrm{d}} = \|z\|,$$

where $\|\cdot\|$ is the Euclidean norm, and define $\|A\|$ accordingly as

$$\|A\| = \sup_{u\neq 0, v\neq 0} \frac{\langle v, Au\rangle}{\|v\|\|u\|_{\mathrm{p}}} = \sup_{u\neq 0} \frac{\|Au\|}{\|u\|_{\mathrm{p}}} = \sup_{v\neq 0} \frac{\|A^T v\|_{\mathrm{p},*}}{\|v\|}.$$

3.3.1 Algorithm

The algorithm uses the following iteration, with starting points $x^{(0)} \in \mathbf{int}(\mathbf{dom}\,\phi_{\mathrm{p}}) \cap \mathbf{dom}\,h$ and $z^{(-1)} = z^{(0)}$:

$$\bar{z}^{(k+1)} = z^{(k)} + \theta_k(z^{(k)} - z^{(k-1)}) \tag{3.36a}$$

$$x^{(k+1)} = \mathbf{prox}^{\phi_{\mathrm{p}}}_{\tau_k f}\big(x^{(k)}, \tau_k(A^T\bar{z}^{(k+1)} + \nabla h(x^{(k)}))\big) \tag{3.36b}$$

$$z^{(k+1)} = z^{(k)} + \sigma_k(Ax^{(k+1)} - b). \tag{3.36c}$$

With constant parameters $\theta_k = 1$, $\sigma_k = \sigma$, $\tau_k = \tau$, the algorithm can be simplified as

$$\begin{aligned} x^{(k+1)} &= \mathbf{prox}^{\phi_{\mathrm{p}}}_{\tau f}\big(x^{(k)}, \tau A^T(2z^{(k)} - z^{(k-1)}) + \tau\nabla h(x^{(k)}))\big) \\ z^{(k+1)} &= z^{(k)} + \sigma(Ax^{(k+1)} - b). \end{aligned}$$

Except for the numbering of the dual iterates, this is the Bregman dual Condat–Vũ algorithm (3.7) applied to (3.35).

In the line search algorithm, the parameters θ_k, τ_k, σ_k are determined by a backtracking search. At the start of the algorithm, we set τ_{-1} and σ_{-1} to some positive values. To start the search in iteration k we choose $\bar{\theta}_k \geq 1$. For $i = 0, 1, 2, \ldots$, we set $\theta_k = 2^{-i}\bar{\theta}_k$, $\tau_k = \theta_k\tau_{k-1}$, $\sigma_k = \theta_k\sigma_{k-1}$, and compute $\bar{z}_{k+1}$, x_{k+1}, z_{k+1} using (3.36). For some $\delta \in (0, 1]$, if

$$\begin{aligned} &\langle z^{(k+1)} - \bar{z}^{(k+1)}, A(x^{(k+1)} - x^{(k)})\rangle + h(x^{(k+1)}) - h(x^{(k)}) - \langle\nabla h(x^{(k)}), x^{(k+1)} - x^{(k)}\rangle \\ \leq\ & \frac{\delta^2}{\tau_k} d_{\mathrm{p}}(x^{(k+1)}, x^{(k)}) + \frac{1}{2\sigma_k}\|\bar{z}^{(k+1)} - z^{(k+1)}\|^2, \end{aligned} \tag{3.37}$$

we accept the computed iterates $\bar{z}^{(k+1)}$, $x^{(k+1)}$, $z^{(k+1)}$ and parameters θ_k, σ_k, τ_k, and terminate the backtracking search. If (3.37) does not hold, we increment i and continue the backtracking search.

The backtracking condition (3.37) is similar to the condition in the line search algorithm for PDHG with Euclidean proximal operators [MP18, Algorithm 4], but it is not identical, even in the Euclidean case. The proposed condition is weaker and allows larger stepsizes than the condition in [MP18, Algorithm 4].

3.3.2 Convergence analysis

In this section we present the convergence analysis for the line search algorithm (3.36). The main conclusion is an $O(1/k)$ rate of ergodic convergence, shown in equation (3.45).

3.3.2.1 Lower bound on algorithm parameters

We first show that the stepsizes are bounded below by

$$\tau_k \geq \tau_{\min} \triangleq \min\left\{\tau_{-1}, \frac{-L + \sqrt{L^2 + 4\delta^2\beta^2\|A\|^2}}{2\beta^2\|A\|^2}\right\}, \qquad \sigma_k \geq \sigma_{\min} \triangleq \beta\tau_{\min}, \tag{3.38}$$

where $\beta = \sigma_{-1}/\tau_{-1}$. The lower bounds imply that the backtracking eventually terminates with positive stepsizes σ_k and τ_k.

Proof. Applying the result in Section 3.2.1.1, with $\tau = \tau_k/\delta^2$, $\sigma = \sigma_k$, we see that the backtracking condition (3.37) holds at iteration k if

$$\tau_k\sigma_k\|A\|^2 + \tau_k L \leq \delta^2.$$

Then mathematical induction can be used to prove (3.38). The two lower bounds (3.38) hold at $k = 0$ by the definition of $\tau_{\min}$ and $\sigma_{\min}$. Now assume $\tau_{k-1} \geq \tau_{\min}$, $\sigma_{k-1} \geq \sigma_{\min}$, and consider the kth iteration. The first attempt of θ_k is $\theta_k = \bar{\theta}_k \geq 1$. If this value is accepted, then

$$\tau_k = \bar{\theta}_k\tau_{k-1} \geq \tau_{k-1} \geq \tau_{\min}, \qquad \sigma_k = \bar{\theta}_k\sigma_{k-1} \geq \sigma_{k-1} \geq \sigma_{\min}.$$

Otherwise, one or more backtracking steps are needed. Denote by $\tilde{\theta}_k$ the last rejected value. Then $\tilde{\theta}_k^2\tau_{k-1}^2\beta^2\|A\|^2 + \tilde{\theta}_k\tau_{k-1}L > \delta^2$ and the accepted θ_k satisfies

$$\theta_k = \frac{\tilde{\theta}_k}{2} \geq \frac{-L + \sqrt{L^2 + 4\delta^2\beta^2\|A\|^2}}{2\tau_{k-1}\beta^2\|A\|^2}.$$

Therefore,

$$\tau_k = \theta_k\tau_{k-1} > \frac{-L + \sqrt{L^2 + 4\delta^2\beta^2\|A\|^2}}{2\beta^2\|A\|^2}, \qquad \sigma_k = \beta\tau_k \geq \beta\tau_{\min}.$$

□

3.3.2.2 One-iteration analysis

The iterates $x^{(k+1)}$, $z^{(k+1)}$, $\bar{z}^{(k+1)}$ generated by the algorithm (3.36) satisfy

$$\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, \bar{z}^{(k+1)}) \leq \frac{1}{\tau_k}\left(d_{\mathrm{p}}(x, x^{(k)}) - d_{\mathrm{p}}(x, x^{(k+1)}) - (1-\delta^2) d_{\mathrm{p}}(x^{(k+1)}, x^{(k)})\right) + \frac{1}{2\sigma_k}\left(\|z - z^{(k)}\|^2 - \|z - z^{(k+1)}\|^2 - \|\bar{z}^{(k+1)} - z^{(k)}\|^2\right) \quad (3.39)$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$ and all z. Here the Lagrangian is given by

$$\mathcal{L}(x, z) = f(x) + h(x) + \langle z, Ax - b\rangle.$$

Proof. The optimality condition for the primal prox-operator (3.36b) gives

$$\begin{aligned} & f(x^{(k+1)}) - f(x) \\ & \leq \frac{1}{\tau_k}\left(d_{\mathrm{p}}(x, x^{(k)}) - d_{\mathrm{p}}(x, x^{(k+1)}) - d_{\mathrm{p}}(x^{(k+1)}, x^{(k)})\right) + \langle A^T \bar{z}^{(k+1)} + \nabla h(x^{(k)}), x - x^{(k+1)}\rangle \end{aligned}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$. Hence

$$\begin{aligned} & f(x^{(k+1)}) + h(x^{(k+1)}) - f(x) - h(x) \\ & \leq \frac{1}{\tau_k}(d_{\mathrm{p}}(x, x^{(k)}) - d_{\mathrm{p}}(x, x^{(k+1)}) - d_{\mathrm{p}}(x^{(k+1)}, x^{(k)})) + \langle A^T \bar{z}_{k+1}, x - x^{(k+1)}\rangle \\ & \quad + h(x^{(k+1)}) - h(x) + \langle \nabla h(x^{(k)}), x - x^{(k+1)}\rangle \\ & \leq \frac{1}{\tau_k}(d_{\mathrm{p}}(x, x^{(k)}) - d_{\mathrm{p}}(x, x^{(k+1)}) - d_{\mathrm{p}}(x^{(k+1)}, x^{(k)})) + \langle A^T \bar{z}^{(k+1)}, x - x^{(k+1)}\rangle \\ & \quad + h(x^{(k+1)}) - h(x^{(k)}) - \langle \nabla h(x^{(k)}), x^{(k+1)} - x^{(k)}\rangle. \end{aligned} \quad (3.40)$$

The second inequality follows from the convexity of h:

$$h(x) \geq h(x^{(k)}) + \langle \nabla h(x^{(k)}), x - x^{(k)}\rangle.$$

The dual update (3.36c) implies that

$$\langle z - z^{(k+1)}, Ax^{(k+1)} - b\rangle = \frac{1}{\sigma_k}\langle z - z^{(k+1)}, z^{(k+1)} - z^{(k)}\rangle \quad \text{for all } z. \quad (3.41)$$

This equality at $k = i - 1$ is

$$\langle z - z^{(i)}, Ax^{(i)} - b\rangle = \frac{1}{\sigma_{i-1}}\langle z - z^{(i)}, z^{(i)} - z^{(i-1)}\rangle$$

$$= \frac{1}{2\sigma_{i-1}} \left(\|z - z^{(i-1)}\|^2 - \|z - z^{(i)}\|^2 - \|z^{(i)} - z^{(i-1)}\|^2 \right). \tag{3.42}$$

The equality (3.41) at $k = i - 2$ is

$$\begin{aligned} \langle z - z^{(i-1)}, Ax^{(i-1)} - b \rangle &= \frac{1}{\sigma_{i-2}} \langle z - z^{(i-1)}, z^{(i-1)} - z^{(i-2)} \rangle \\ &= \frac{\theta_{i-1}}{\sigma_{i-1}} \langle z - z^{(i-1)}, z^{(i-1)} - z^{(i-2)} \rangle \\ &= \frac{1}{\sigma_{i-1}} \langle z - z^{(i-1)}, \bar{z}^{(i)} - z^{(i-1)} \rangle. \end{aligned}$$

We evaluate this at $z = z^{(i)}$ and add it to the equality at $z = z^{(i-2)}$ multiplied by θ_{i-1}:

$$\begin{aligned} \langle z^{(i)} - \bar{z}^{(i)}, Ax^{(i-1)} - b \rangle &= \frac{1}{\sigma_{i-1}} \langle z^{(i)} - \bar{z}^{(i)}, \bar{z}^{(i)} - z^{(i-1)} \rangle \\ &= \frac{1}{2\sigma_{i-1}} \left(\|z^{(i)} - z^{(i-1)}\|^2 - \|z^{(i)} - \bar{z}^{(i)}\|^2 - \|\bar{z}^{(i)} - z^{(i-1)}\|^2 \right). \end{aligned} \tag{3.43}$$

Now we combine (3.40) for $k = i - 1$, with (3.42) and (3.43). For $i \geq 1$,

$$\begin{aligned} & \mathcal{L}(x^{(i)}, z) - \mathcal{L}(x, \bar{z}^{(i)}) \\ &= f(x^{(i)}) + h(x^{(i)}) + \langle z, Ax^{(i)} - b \rangle - f(x) - h(x) - \langle \bar{z}^{(i)}, Ax - b \rangle \\ &\leq \frac{1}{\tau_{i-1}} \Big(d_{\mathrm{p}}(x, x^{(i-1)}) - d_{\mathrm{p}}(x, x^{(i)}) - d_{\mathrm{p}}(x^{(i)}, x^{(i-1)}) \Big) + \langle A^T \bar{z}^{(i)}, x - x^{(i)} \rangle + \langle z, Ax^{(i)} - b \rangle \\ &\quad - \langle \bar{z}^{(i)}, Ax - b \rangle + h(x^{(i)}) - h(x^{(i-1)}) - \langle \nabla h(x^{(i-1)}), x^{(i)} - x^{(i-1)} \rangle \\ &= \frac{1}{\tau_{i-1}} \Big(d_{\mathrm{p}}(x, x^{(i-1)}) - d_{\mathrm{p}}(x, x^{(i)}) - d_{\mathrm{p}}(x^{(i)}, x^{(i-1)}) \Big) + \langle z - \bar{z}^{(i)}, Ax^{(i)} - b \rangle \\ &\quad + h(x^{(i)}) - h(x^{(i-1)}) - \langle \nabla h(x^{(i-1)}), x^{(i)} - x^{(i-1)} \rangle \\ &= \frac{1}{\tau_{i-1}} \Big(d_{\mathrm{p}}(x, x^{(i-1)}) - d_{\mathrm{p}}(x, x^{(i)}) - d_{\mathrm{p}}(x^{(i)}, x^{(i-1)}) \Big) \\ &\quad + \langle z^{(i)} - \bar{z}^{(i)}, A(x^{(i)} - x^{(i-1)}) \rangle + \langle z - z^{(i)}, Ax^{(i)} - b \rangle + \langle z^{(i)} - \bar{z}^{(i)}, Ax^{(i-1)} - b \rangle \\ &\quad + h(x^{(i)}) - h(x^{(i-1)}) - \langle \nabla h(x^{(i-1)}), x^{(i)} - x^{(i-1)} \rangle \\ &= \frac{1}{\tau_{i-1}} \Big(d_{\mathrm{p}}(x, x^{(i-1)}) - d_{\mathrm{p}}(x, x^{(i)}) - d_{\mathrm{p}}(x^{(i)}, x^{(i-1)})) \Big) \\ &\quad + \frac{1}{2\sigma_{i-1}} \Big(\|z - z^{(i-1)}\|^2 - \|z - z^{(i)}\|^2 - \|\bar{z}^{(i)} - z^{(i-1)}\|^2 - \|\bar{z}^{(i)} - z^{(i)}\|^2 \Big) \\ &\quad + \langle A^T (z^{(i)} - \bar{z}^{(i)}), x^{(i)} - x^{(i-1)} \rangle + h(x^{(i)}) - h(x^{(i-1)}) - \langle \nabla h(x^{(i-1)}), x^{(i)} - x^{(i-1)} \rangle \\ &\leq \frac{1}{\tau_{i-1}} \left(d_{\mathrm{p}}(x, x^{(i-1)}) - d_{\mathrm{p}}(x, x^{(i)}) - (1 - \delta^2) d_{\mathrm{p}}(x^{(i)}, x^{(i-1)}) \right) \end{aligned}$$

$$+\frac{1}{2\sigma_{i-1}}\left(\|z-z^{(i-1)}\|^2-\|z-z^{(i)}\|^2-\|\bar{z}^{(i)}-z^{(i-1)}\|^2\right),$$

which is the desired result (3.39). The first inequality follows from (3.40). In the second last step we substitute (3.42) and (3.43). The last step uses the line search exit condition (3.37) at $k=i-1$. □

3.3.2.3 Ergodic convergence

We define the averaged primal and dual sequences

$$x_{\text{avg}}^{(k)}=\frac{1}{\sum_{i=1}^k\tau_{i-1}}\sum_{i=1}^k\tau_{i-1}x^{(i)},\qquad \bar{z}_{\text{avg}}^{(k)}=\frac{1}{\sum_{i=1}^k\tau_{i-1}}\sum_{i=1}^k\tau_{i-1}\bar{z}^{(i)}$$

for $k\geq 1$. We show that

$$\mathcal{L}(x_{\text{avg}}^{(k)},z)-\mathcal{L}(x,\bar{z}_{\text{avg}}^{(k)})\leq\frac{1}{\sum_{i=1}^k\tau_{i-1}}\left(d(x,x^{(0)})+\frac{1}{2\beta}\|z-z^{(0)}\|^2\right) \tag{3.44}$$

$$\leq\frac{1}{k\tau_{\min}}\left(d(x,x^{(0)})+\frac{1}{2\beta}\|z-z^{(0)}\|^2\right) \tag{3.45}$$

for all $x\in\mathbf{dom}\,f\cap\mathbf{dom}\,\phi_{\text{p}}$ and all z. This holds for any choice of $\delta\in(0,1]$ in (3.37). If we compare (3.44) and (3.18), we note that the two left-hand sides involve different dual iterates ($\bar{z}_{\text{avg}}^{(k)}$ as opposed to $z_{\text{avg}}^{(k)}$).

Proof. From (3.39),

$$\mathcal{L}(x^{(i)},z)-\mathcal{L}(x,\bar{z}^{(i)})\leq\frac{1}{\tau_{i-1}}\left(d_{\text{p}}(x,x^{(i-1)})-d_{\text{p}}(x,x^{(i)})+\frac{1}{2\beta}\|z-z^{(i-1)}\|^2-\frac{1}{2\beta}\|z-z^{(i)}\|^2\right).$$

Since $\mathcal{L}$ is convex in x and affine in z,

$$\begin{aligned}&(\sum_{i=1}^k\tau_{i-1})\left(\mathcal{L}(x_{\text{avg}}^{(k)},z)-\mathcal{L}(x,\bar{z}_{\text{avg}}^{(k)})\right)\\&\leq\sum_{i=1}^k\tau_{i-1}(\mathcal{L}(x^{(i)},z)-\mathcal{L}(x,\bar{z}^{(i)}))\\&\leq d_{\text{p}}(x,x^{(0)})-d_{\text{p}}(x,x^{(k)})+\frac{1}{2\beta}(\|z-z^{(0)}\|^2-\|z-z^{(k)}\|^2)\end{aligned}$$

$$\leq \quad d_{\mathrm{p}}(x, x^{(0)}) + \frac{1}{2\beta}\|z - z^{(0)}\|^2. \tag{3.46}$$

Dividing by $\sum_{i=1}^{k} \tau_{i-1}$ and plugging in $x = x^\star$, $z = z^\star$ gives (3.44). □

If we substitute an optimal $x = x^\star$ in (3.46), we obtain that

$$f(x_{\mathrm{avg}}^{(k)}) + h(x_{\mathrm{avg}}^{(k)}) + z^T(Ax_{\mathrm{avg}}^{(k)} - b) - f(x^\star) \leq \frac{1}{\sum_{i=1}^{k} \tau_{i-1}}\big(d_{\mathrm{p}}(x^\star, x^{(0)}) + \frac{1}{2\beta}\|z - z^{(0)}\|^2\big)$$

for all z, since $Ax^\star = b$. More generally, suppose $\mathcal{X} \subseteq \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$ is a compact set containing an optimal solution $x^\star$, and $\mathcal{Z} = \{z \mid \|z\| \leq \zeta\}$ contains a dual optimal $z^\star$, then the merit function η defined in (2.7) satisfies

$$\begin{aligned}
\eta(x_{\mathrm{avg}}^{(k)}, \bar{z}_{\mathrm{avg}}^{(k)}) &= f(x_{\mathrm{avg}}^{(k)}) + h(x_{\mathrm{avg}}^{(k)}) + \zeta\|Ax_{\mathrm{avg}}^{(k)} - b\| - \inf_{x\in\mathcal{X}}(f(x) + h(x) + z^T(Ax - b)) \\
&\leq \frac{1}{\sum_{i=1}^{k} \tau_{i-1}}\Big(\sup_{x\in\mathcal{X}} d_{\mathrm{p}}(x, x^{(0)}) + \frac{1}{2\beta}(\zeta + \|z^{(0)}\|)^2\Big) \\
&\leq \frac{1}{k\tau_{\min}}\Big(\sup_{x\in\mathcal{X}} d_{\mathrm{p}}(x, x^{(0)}) + \frac{1}{2\beta}(\zeta + \|z^{(0)}\|)^2\Big).
\end{aligned}$$

Especially, we have

$$\begin{aligned}
& f(x_{\mathrm{avg}}^{(k)}) + h(x_{\mathrm{avg}}^{(k)}) + \zeta\|Ax_{\mathrm{avg}}^{(k)} - b\| - f(x^\star) - h(x^\star) \\
&\leq \eta(x_{\mathrm{avg}}^{(k)}, \bar{z}_{\mathrm{avg}}^{(k)}) \\
&\leq \frac{1}{k\tau_{\min}}\Big(\sup_{x\in\mathcal{X}} d_{\mathrm{p}}(x, x^{(0)}) + \frac{1}{2\beta}(\zeta + \|z^{(0)}\|)^2\Big).
\end{aligned}$$

For $\zeta > \|z^\star\|$, the penalty function in the merit function is exact, so

$$f(x) + h(x) + \zeta\|Ax - b\| - f(x^\star) - h(x^\star) \geq 0$$

with equality only if x is optimal. Therefore, the above inequality the inequality shows that the merit function decreases as $O(1/k)$.

3.3.2.4 Monotonicity properties

For $x = x^\star$, $z = z^\star$, the left-hand side of (3.39) is nonnegative and we obtain

$$d_{\mathrm{p}}(x^\star, x^{(k+1)}) + \frac{1}{2\beta}\|z^\star - z^{(k+1)}\|^2$$

$$
\begin{aligned}
&\leq d_{\mathrm{p}}(x^\star, x^{(k)}) + \frac{1}{2\beta}\|z^\star - z^{(k)}\|^2 - \big((1-\delta^2)d_{\mathrm{p}}(x^\star, x^{(k)}) + \frac{1}{2\beta}\|z^\star - z^{(k)}\|^2\big) \\
&\leq d_{\mathrm{p}}(x^\star, x^{(k)}) + \frac{1}{2\beta}\|z^\star - z^{(k)}\|^2 \qquad (3.47)
\end{aligned}
$$

for $k \geq 0$. Moreover,

$$
\sum_{i=0}^{k} \Big((1-\delta^2)(d_{\mathrm{p}}(x^{(i+1)}, x^{(i)}) + \frac{1}{2\beta}\|\bar{z}^{(i+1)} - z^{(i)}\|^2\Big) \leq d_{\mathrm{p}}(x^\star, x^{(0)}) + \frac{1}{2\beta}\|z^\star - \bar{z}^{(0)}\|^2. \quad (3.48)
$$

These inequalities hold for any value $\delta \in (0, 1]$. In particular, the last inequality implies that $\bar{z}^{(i+1)} - z^{(i)} \to 0$. When $\delta < 1$ it also implies that $d_{\mathrm{p}}(x^{(i+1)}, x^{(i)}) \to 0$ and, by the strong convexity assumption on ϕ_{p}, that $x^{(i+1)} - x^{(i)} \to 0$.

3.3.2.5 Convergence of iterates

With additional assumptions similar to those in Section 3.2.2.3, one can show the convergence of iterates. More specifically, we make two additional assumptions on the Bregman kernel ϕ_{p}.

1. For fixed x, the sublevel sets $\{x' \mid d(x, x') \leq \gamma\}$ are closed. In other words, the distance $d(x, x')$ is a closed function of x'.

2. If $\tilde{x}^{(k)} \in \mathbf{int}(\mathbf{dom}\, \phi_{\mathrm{p}})$ converges to $x \in \mathbf{dom}\, \phi_{\mathrm{p}}$, then $d_{\mathrm{p}}(x, \tilde{x}^{(k)}) \to 0$.

Again, these two assumptions are not restrictive and are very common in the literature on Bregman distances [CT93, Eck93, Gul94, CZ97].

We also make the (minor) assumptions that $\delta < 1$ in (3.37) and that θ_k is bounded above (which is easily satisfied, since the user chooses $\bar{\theta}_k$). With these additional assumptions it can be shown that the sequences $x^{(k)}$, $z^{(k)}$ converge to optimal solutions.

Proof. The inequality (3.47) and strong convexity of ϕ_{p} show that the sequences $x^{(k)}$, $z^{(k)}$ are bounded. Let $(x^{(k_i)}, z^{(k_i)})$ be a convergent subsequence with limit $(\hat{x}, \hat{z})$. With $\delta < 1$, (3.48) shows that $d_{\mathrm{p}}(x^{(k_i+1)}, x^{(k_i)})$ converges to zero. By strong convexity of the kernel, $x^{(k_i+1)} -$

$x^{(k_i)} \to 0$ and therefore the subsequence $x^{(k_i+1)}$ also converges to $\hat{x}$. Since $z^{(k_i+1)} - z^{(k_i)} \to 0$, the subsequence $z^{(k_i+1)}$ converges to $\hat{z}$. Since θ_k is bounded above,

$$\bar{z}^{(k_i+1)} = z^{(k_i)} + \theta_k(z^{(k_i)} - z^{(k_i-1)})$$

also converges to $\hat{z}$.

The dual update (3.36c) can be written as

$$Ax^{(k_i+1)} - b = \frac{1}{\sigma_{k_i}}(z^{(k_i+1)} - z^{(k_i)}). \tag{3.49}$$

Since $z^{(k_i+1)} - z^{(k_i)} \to 0$ and $\sigma_{k_i} \geq \sigma_{\min}$, the left-hand side converges to zero, so $A\hat{x} = b$.

From (3.47), $d_{\mathrm{p}}(x^\star, x^{(k_i)})$ is bounded above. Since the sublevel set

$$\{x' \mid d_{\mathrm{p}}(x^\star, x') \leq \gamma\}$$

are closed subsets of $\mathbf{int}(\mathbf{dom}\,\phi_{\mathrm{p}})$, the limit $\hat{x}$ is in $\mathbf{int}(\mathbf{dom}\,\phi_{\mathrm{p}})$. The left-hand side of the optimality condition

$$\frac{1}{\tau_{k_i}}(\nabla\phi_{\mathrm{p}}(x^{(k_i)}) - \nabla\phi_{\mathrm{p}}(x^{(k_i+1)})) - (A^T\bar{z}^{(k_i+1)} + \nabla h(x^{(k)})) \in \partial f(x^{(k_i+1)}) \tag{3.50}$$

converges to $-A^T\hat{z}$, because $\tau_k \geq \tau_{\min}$ and $\nabla\phi_{\mathrm{p}}$ is continuous on $\mathbf{int}(\mathbf{dom}\,\phi_{\mathrm{p}})$. By maximal monotonicity of ∂f, this implies that $-A^T\hat{z} - \nabla h(\hat{x}) \in \partial f(\hat{x})$ (see [Bre73, page 27], [Tse00, Lemma 3.2]). We conclude that $\hat{x}$, $\hat{z}$ satisfy the optimality conditions $A\hat{x} = b$ and $-A^T\hat{z} \in \partial f(\hat{x}) + \nabla h(\hat{x})$.

To show that the entire sequence converges, we substitute $x = \hat{x}$, $z = \hat{z}$ in (3.39):

$$\mathcal{L}(x^{(k)}, \hat{z}) - \mathcal{L}(\hat{x}, \bar{z}^{(k)}) \leq \frac{1}{\tau_k}\big(d_{\mathrm{p}}(\hat{x}, x^{(k)}) - d_{\mathrm{p}}(\hat{x}, x^{(k+1)})\big) + \frac{1}{2\sigma_k}\big(\|\hat{z} - z^{(k)}\|^2 - \|\hat{z} - z^{(k+1)}\|^2\big).$$

The left-hand side is nonnegative, and thus

$$d_{\mathrm{p}}(\hat{x}, x^{(k+1)}) + \frac{1}{2\beta}\|\hat{z} - z^{(k+1)}\|^2 \leq d_{\mathrm{p}}(\hat{x}, x^{(k)}) + \frac{1}{2\beta}\|\hat{z} - z^{(k)}\|^2$$

for all k. This shows that

$$d_{\mathrm{p}}(\hat{x}, x^{(k)}) + \frac{1}{2\beta}\|\hat{z} - z^{(k)}\|^2 \leq d_{\mathrm{p}}(\hat{x}, x^{(k_i)}) + \frac{1}{2\beta}\|\hat{z} - z^{(k_i)}\|^2$$

for all $k \geq k_i$. By the second additional assumption mentioned above, the right-hand side converges to zero. Therefore $d_{\mathrm{p}}(\hat{x}, x^{(k)}) \to 0$ and $z^{(k)} \to \hat{z}$. If $d_{\mathrm{p}}(\hat{x}, x^{(k)}) \to 0$, then the strong convexity property of the kernel implies that $x^{(k)} \to \hat{x}$. □

3.4 Bregman PD3O algorithm

In this section we propose the *Bregman PD3O algorithm*, another Bregman proximal splitting method for the problem (2.1). Bregman PD3O also involves two generalized distances, d_{p} and d_{d}, generated by ϕ_{p} and ϕ_{d}, respectively, and it consists of the iterations

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\mathrm{p}}}(x^{(k)}, \tau A^T z^{(k)} + \tau \nabla h(x^{(k)})) \tag{3.51a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^{\phi_{\mathrm{d}}}(z^{(k)}, -\sigma A(2x^{(k+1)} - x^{(k)} + \tau(\nabla h(x^{(k)}) - \nabla h(x^{(k+1)})))). \tag{3.51b}$$

The only difference between Bregman PD3O and Bregman primal Condat–Vũ algorithm (3.6) is the additional term $\tau(\nabla h(x^{(k)}) - \nabla h(x^{(k+1)}))$. Thus the two algorithms (3.6) and (3.51) reduce to the same method when h is absent from problem (2.1). The additional term allows PD3O to use larger stepsizes than the Condat–Vũ algorithm. If we use the same matrix norm $\|A\|$ and Lipschitz constant L in the analysis for the two methods, then the conditions are

$$\begin{aligned} &\text{Condat–Vũ:} && \sigma\tau\|A\|^2 + \tau L \leq 1 \\ &\text{PD3O:} && \sigma\tau\|A\|^2 \leq 1, \quad \tau \leq 1/L. \end{aligned} \tag{3.52}$$

The range of possible parameters is illustrated in Figure 3.3.

In Section 3.4.1 we provide the detailed convergence analysis of the Bregman PD3O method. The connections between Bregman PD3O and several other Bregman proximal methods are discussed in Section 3.4.2.

Assumptions Throughout Section 3.4 we make the following assumptions. The kernel functions ϕ_{p} and ϕ_{d} are 1-strongly convex with respect to the Euclidean norm and an arbi-

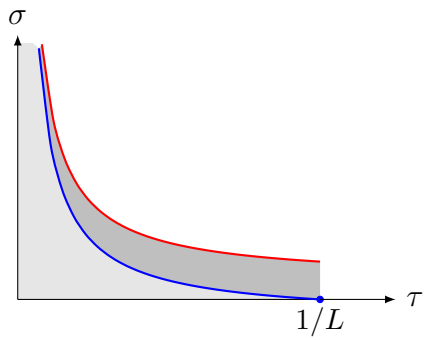

Figure 3.3: Acceptable stepsizes in Condat–Vũ algorithms and PD3O. We assume the same matrix norm $\|A\|$ and Lipschitz constant L are used in the analysis of the two algorithms. The light gray region under the blue curve is defined by the inequality for the Condat–Vũ algorithms in (3.52). The region under the red curve shows the values allowed by the stepsized conditions for PD3O.

trary norm $\|\cdot\|_{\mathrm{d}}$, respectively:

$$d_{\mathrm{p}}(x,x') \geq \frac{1}{2}\|x-x'\|^2, \qquad d_{\mathrm{d}}(z,z') \geq \frac{1}{2}\|z-z'\|_{\mathrm{d}}^2. \tag{3.53}$$

The assumptions that the strong convexity constants are one can be made without loss of generality, by scaling the distances. The definition of $\|A\|$ follows (3.10) and reduces to

$$\|A\| = \sup_{v\neq 0} \frac{\|Av\|_{\mathrm{d},*}}{\|v\|}.$$

We also assume that the gradient of h is L-Lipschitz continuous with respect to the Euclidean norm: $\mathbf{dom}\, h = \mathbf{R}^n$ and

$$h(y) - h(x) - \langle \nabla h(x), y-x\rangle \leq \frac{L}{2}\|y-x\|^2, \quad \text{for any } x,y \in \mathbf{dom}\, h. \tag{3.54}$$

The parameters τ and σ must satisfy

$$\sigma\tau\|A\|^2 \leq 1, \qquad \tau \leq 1/L. \tag{3.55}$$

Finally, we assume that the optimality condition (2.4) has a solution

$$(x^\star, z^\star) \in \mathbf{dom}\, \phi_{\mathrm{p}} \times \mathbf{dom}\, \phi_{\mathrm{d}}.$$

Note that (3.54) is a stronger assumption than (3.9). (Combined with the first inequality in (3.53), it implies (3.9).) We will use the following consequence of (3.54):

$$h(y) - h(x) - \langle \nabla h(x), y - x \rangle \geq \frac{1}{2L} \|\nabla h(y) - \nabla h(x)\|^2 \tag{3.56}$$

for all x, y [Nes18, Theorem 2.1.5].

3.4.1 Convergence analysis

In this section we present the convergence analysis for Bregman PD3O. The main result is an $O(1/k)$ rate of ergodic convergence, given in (3.61).

3.4.1.1 A primal–dual Bregman distance

We introduce a primal–dual kernel

$$\phi_{\text{pd3o}}(x, y, z) = \frac{1}{\tau}\phi_{\text{p}}(x) + \frac{1}{\sigma}\phi_{\text{d}}(z) + \frac{\tau}{2}\|y\|^2 - \langle y, x \rangle - \langle z, A(x - \tau y) \rangle,$$

where $\sigma, \tau > 0$. If ϕ_{pd3o} is convex, the generated Bregman distance is given by

$$\begin{aligned} & d_{\text{pd3o}}(x, y, z; x', y', z') && (3.57) \\ &= \frac{1}{\tau} d_{\text{p}}(x, x') + \frac{1}{\sigma} d_{\text{d}}(z, z') + \frac{\tau}{2}\|y - y'\|^2 \\ &\quad - \langle y - y', x - x' \rangle - \langle z - z', A(x - x') \rangle + \tau \langle z - z', A(y - y') \rangle. && (3.58) \end{aligned}$$

We now show that ϕ_{pd3o} is convex if $\sigma\tau\|A\|^2 \leq 1$.

Proof. It is sufficient to show that d_{pd3o} is nonnegative:

$$\begin{aligned} d_{\text{pd3o}}(x, y, z; x', y', z') &\geq \frac{1}{2\tau}\|x - x'\|^2 + \frac{\tau}{2}\|A^T(z - z')\|^2 + \frac{\tau}{2}\|y - y'\|^2 \\ &\quad - \langle y - y', x - x' \rangle - \langle z - z', A(x - x') \rangle + \tau \langle z - z', A(y - y') \rangle \\ &= \frac{1}{2}\left\| \frac{1}{\sqrt{\tau}}(x - x') - \sqrt{\tau}(y - y') - \sqrt{\tau} A^T(z - z') \right\|^2 \\ &\geq 0. \end{aligned} \tag{3.59}$$

In step 1 we use the strong convexity assumption (3.53), the definition of $\|A\|$ (3.10) with $\|\cdot\|_{\mathrm{p}} = \|\cdot\|$, and the assumption $\sigma\tau\|A\|^2 \leq 1$. The bound on $d_{\mathrm{d}}(z, z')$ follows from

$$\frac{1}{\sigma} d_{\mathrm{d}}(z, z') \geq \frac{1}{2\sigma}\|z - z'\|_{\mathrm{d}}^2 \geq \frac{\|A^T(z - z')\|^2}{2\sigma\|A\|^2} \geq \frac{\tau}{2}\|A^T(z - z')\|^2.$$

□

Note that the convexity of ϕ_{pd3o} only requires the first inequality in the stepsize condition (3.55). Although the Bregman PD3O algorithm (3.51) is not the Bregman proximal point method for the Bregman kernel ϕ_{pd3o}, the distance d_{pd3o} will appear in the key inequality (3.60) of the convergence analysis.

3.4.1.2 One-iteration analysis

We first show that the iterates $x^{(k+1)}$, $z^{(k+1)}$ generated by Bregman PD3O (3.51) satisfy

$$\begin{aligned}
&\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\
&\quad \leq \; d_{\mathrm{pd3o}}\big(x, \nabla h(x), z; x^{(k)}, \nabla h(x^{(k)}), z^{(k)}\big) - d_{\mathrm{pd3o}}\big(x, \nabla h(x), z; x^{(k+1)}, \nabla h(x^{(k+1)}), z^{(k+1)}\big) \\
&\qquad - d_{\mathrm{pd3o}}\big(x^{(k+1)}, \nabla h(x), z^{(k+1)}; x^{(k)}, \nabla h(x^{(k)}), z^{(k)}\big)
\end{aligned} \tag{3.60}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$ and $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\mathrm{d}}$.

Proof. Recall that Bregman PD3O differs from the Bregman primal Condat–Vũ algorithm (3.6) only in an additional term in the dual update. The proof in Section 3.3.2.2 therefore applies up to (3.16), with

$$\tilde{x} = 2x^{(k+1)} - x^{(k)} + \tau(\nabla h(x^{(k)}) - \nabla h(x^{(k+1)})), \qquad \tilde{z} = z^{(k)}.$$

Hence,

$$\begin{aligned}
&\mathcal{L}(x^{(k+1)}, z) - \mathcal{L}(x, z^{(k+1)}) \\
&\quad \leq \; d_{-}(x, z; x^{(k)}, z^{(k)}) - d_{-}(x, z; x^{(k+1)}, z^{(k+1)}) - d_{-}(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)})
\end{aligned}$$

$$
\begin{aligned}
&\quad - \tau\langle A^T(z - z^{(k+1)}), \nabla h(x^{(k)}) - \nabla h(x^{(k+1)})\rangle \\
&\quad - h(x^{(k+1)}) + h(x^{(k)}) + \langle \nabla h(x^{(k)}), x^{(k+1)} - x^{(k)}\rangle \\
&= d_-(x, z; x^{(k)}, z^{(k)}) + \frac{\tau}{2}\|\nabla h(x) - \nabla h(x^{(k)})\|^2 \\
&\quad - \langle (x - \tau A^T z) - (x^{(k)} - \tau A^T z^{(k)}), \nabla h(x) - \nabla h(x^{(k)})\rangle \\
&\quad - \Big(d_-(x, z; x^{(k+1)}, z^{(k+1)}) + \frac{\tau}{2}\|\nabla h(x) - \nabla h(x^{(k+1)})\|^2 \\
&\quad - \big\langle x - \tau A^T z - (x^{(k+1)} - \tau A^T z^{(k+1)}), \nabla h(x) - \nabla h(x^{(k+1)})\big\rangle\Big) \\
&\quad - \Big(d_-(x^{(k+1)}, z^{(k+1)}; x^{(k)}, z^{(k)}) + \frac{\tau}{2}\|\nabla h(x) - \nabla h(x^{(k)})\|^2 \\
&\quad - \big\langle (x^{(k+1)} - \tau A^T z^{(k+1)}) - (x^{(k)} - \tau A^T z^{(k)}), \nabla h(x) - \nabla h(x^{(k)})\big\rangle\Big) \\
&\quad - (h(x) - h(x^{(k+1)}) - \langle \nabla h(x^{(k+1)}), x - x^{(k+1)}\rangle - \frac{\tau}{2}\|\nabla h(x) - \nabla h(x^{(k+1)})\|^2) \\
&= d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(k)}, \nabla h(x^{(k)}), z^{(k)}) - d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(k+1)}, \nabla h(x^{(k+1)}), z^{(k+1)}) \\
&\quad - d_{\mathrm{pd3o}}(x^{(k+1)}, \nabla h(x), z^{(k+1)}; x^{(k)}, \nabla h(x^{(k)}), z^{(k)}) \\
&\quad - (h(x) - h(x^{(k+1)}) - \langle \nabla h(x^{(k+1)}), x - x^{(k+1)}\rangle - \frac{\tau}{2}\|\nabla h(x) - \nabla h(x^{(k+1)})\|^2) \\
&\le d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(k)}, \nabla h(x^{(k)}), z^{(k)}) - d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(k+1)}, \nabla h(x^{(k+1)}), z^{(k+1)}) \\
&\quad - d_{\mathrm{pd3o}}(x^{(k+1)}, \nabla h(x), z^{(k+1)}; x^{(k)}, \nabla h(x^{(k)}), z^{(k)}) \\
&\le d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(k)}, \nabla h(x^{(k)}), z^{(k)}) - d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(k+1)}, \nabla h(x^{(k+1)}), z^{(k+1)}).
\end{aligned}
$$

Step 3 follows from definition of d_{pd3o} (3.58). In step 4 we use the Lipschitz condition (3.56) and the second inequality in the stepsize condition (3.55). The last step follows from the fact that d_{pd3o} is nonnegative (3.59). □

3.4.1.3 Ergodic convergence

The iterates generated by Bregman PD3O (3.51) satisfy

$$
\mathcal{L}(x_{\mathrm{avg}}^{(k)}, z) - \mathcal{L}(x, z_{\mathrm{avg}}^{(k)}) \le \frac{2}{k}\Big(\frac{1}{\tau}d_{\mathrm{p}}(x, x^{(0)}) + \frac{1}{\sigma}d_{\mathrm{d}}(z, z^{(0)}) + \frac{\tau}{2}\|\nabla h(x) - \nabla h(x^{(0)})\|^2\Big), \quad (3.61)
$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$ and all $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\mathrm{d}}$, where the averaged iterates are defined in (3.17).

Proof. From (3.60), since $\mathcal{L}(u, v)$ is convex in u and concave in v,

$$\begin{aligned}
&\mathcal{L}(x_{\mathrm{avg}}^{(k)}, z) - \mathcal{L}(x, z_{\mathrm{avg}}^{(k)}) \\
&\quad\leq \frac{1}{k}\sum_{i=1}^{k}\big(\mathcal{L}(x^{(i)}, z) - \mathcal{L}(x, z^{(i)})\big) \\
&\quad\leq \frac{1}{k}\Big(d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(0)}, \nabla h(x^{(0)}), z^{(0)}) - d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(k)}, \nabla h(x^{(k)}), z^{(k)})\Big) \\
&\quad\leq \frac{1}{k} d_{\mathrm{pd3o}}(x, \nabla h(x), z; x^{(0)}, \nabla h(x^{(0)}), z^{(0)}) \\
&\quad\leq \frac{2}{k}\Big(\frac{1}{\tau} d_{\mathrm{p}}(x, x^{(0)}) + \frac{1}{\sigma} d_{\mathrm{d}}(z, z^{(0)}) + \frac{\tau}{2}\|\nabla h(x) - \nabla h(x^{(0)})\|^2\Big)
\end{aligned}$$

for all $x \in \mathbf{dom}\, f \cap \mathbf{dom}\, \phi_{\mathrm{p}}$ and $z \in \mathbf{dom}\, g^* \cap \mathbf{dom}\, \phi_{\mathrm{d}}$. □

3.4.2 Relation to other Bregman proximal algorithms

The proposed algorithm (3.51) can be viewed as an extension to PD3O (2.25) using generalized distances, and reduces to several Bregman proximal methods by reduction. These algorithms can also be organized into a diagram similar to Figure 2.3. Figure 3.4 starts from Bregman PD3O (3.51), and summarizes its connection to several Bregman proximal methods. When $h = 0$, (3.51) reduces to Bregman PDHG (3.51), and when $g = 0$, (3.51) reduces to the Bregman proximal gradient algorithm (3.25). More interestingly, when $f = 0$, Bregman PD3O reduces to the Bregman Loris–Verhoeven algorithm

$$x^{(k+1)} = \operatorname*{argmin}_x\big(\langle \nabla h(x^{(k)}) - A^T z^{(k)}, x\rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)})\big) \tag{3.62a}$$

$$z^{(k+1)} = \mathbf{prox}_{\sigma g^*}^{\phi_{\mathrm{d}}}\big(z^{(k)}, -\sigma A\big(2x^{(k+1)} - x^{(k)} + \tau\big(\nabla h(x^{(k)}) - \nabla h(x^{(k+1)})\big)\big)\big). \tag{3.62b}$$

Setting $A = I$ (with $\sigma = 1/\tau$), we obtain a new variant of Bregman proximal gradient algorithm:

$$x^{(k+1)} = \operatorname*{argmin}_x\big(\langle \nabla h(x^{(k)}) - z^{(k)}, x\rangle + \frac{1}{\tau} d_{\mathrm{p}}(x, x^{(k)})\big) \tag{3.63a}$$

$$z^{(k+1)} = \mathbf{prox}_{\tau^{-1} g^*}^{\phi_{\mathrm{d}}}\big(z^{(k)}, -\frac{1}{\tau} A\big(2x^{(k+1)} - x^{(k)}\big) - A\big(\nabla h(x^{(k)}) - \nabla h(x^{(k+1)})\big)\big). \tag{3.63b}$$

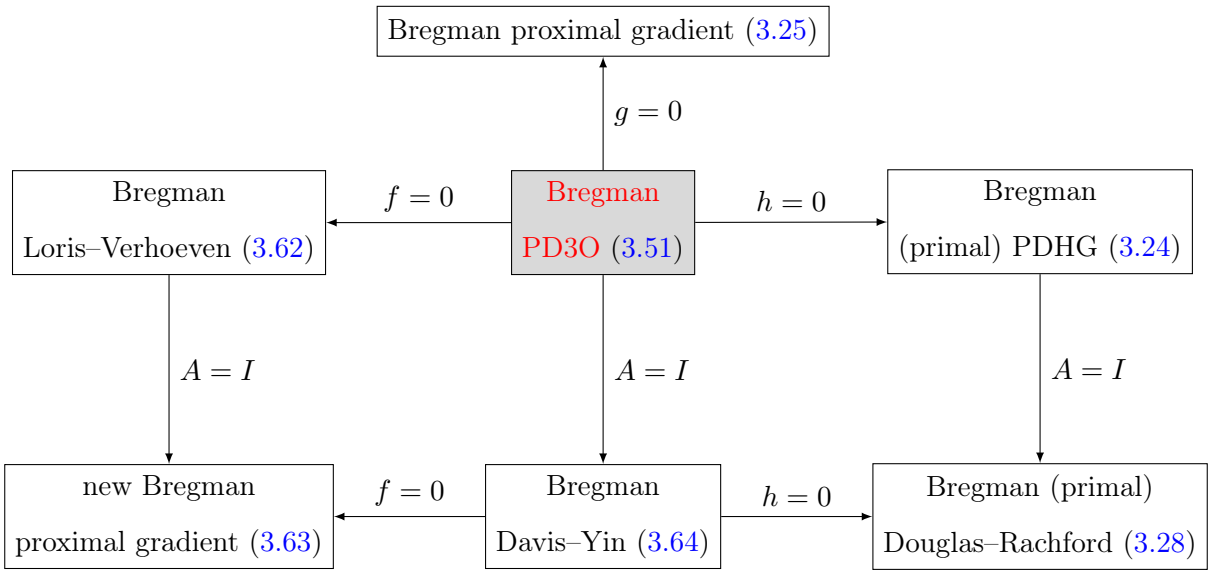

Figure 3.4: Proximal algorithms derived from Bregman PD3O.

The difference between (3.63) and (3.29) is the additional term $\tau(\nabla h(x^{(k)}) - \nabla h(x^{(k+1)}))$, the same as the difference between (3.6) and (3.51). When the Euclidean proximal operator is used, (3.63) reduces to the proximal gradient method. However, the new algorithm (3.63) does not seem to be equivalent to the Bregman proximal gradient algorithm due to the lack of Moreau decomposition in the generalized case. Nevertheless, the new algorithm (3.63) may still be interesting on its own, especially when the generalized proximal operator of g^* is easy to compute while the (Euclidean or generalized) proximal operator of g is computationally expensive. Finally, setting $A = I$ (and $\sigma = 1/\tau$) in Bregman PD3O (3.51) gives a Bregman Davis–Yin algorithm:

$$x^{(k+1)} = \mathbf{prox}_{\tau f}^{\phi_{\mathrm{p}}}\big(x^{(k)}, \tau(z^{(k)} + \nabla h(x^{(k)}))\big) \tag{3.64a}$$

$$z^{(k+1)} = \mathbf{prox}_{\tau^{-1} g^*}^{\phi_{\mathrm{d}}}\big(z^{(k)}, -\frac{1}{\tau}(2x^{(k+1)} - x^{(k)}) + \nabla h(x^{(k)}) - \nabla h(x^{(k+1)})\big). \tag{3.64b}$$

3.5 Numerical experiment

In this section we evaluate the performance of the Bregman primal Condat–Vũ algorithm (3.6), Bregman dual Condat–Vũ algorithm with line search (3.36), and Bregman PD3O (3.51). The main goal of the example is to validate and illustrate the difference in the stepsize conditions (3.52), and the usefulness of the line search procedure. We consider the convex optimization problem

$$\begin{array}{ll} \text{minimize} & \psi(x) = \lambda\|Ax\|_1 + \frac{1}{2}\|Cx - b\|^2 \\ \text{subject to} & \mathbf{1}^T x = 1, \quad x \succeq 0, \end{array} \tag{3.65}$$

where $x \in \mathbf{R}^n$ is the optimization variable, $C \in \mathbf{R}^{m\times n}$, and $A \in \mathbf{R}^{(n-1)\times n}$ is the difference matrix

$$A = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 \end{bmatrix}. \tag{3.66}$$

This problem is of the form of (2.1) with

$$f(x) = \delta_H(x), \qquad g(y) = \lambda\|y\|_1, \qquad g^*(z) = \begin{cases} 0 & \|z\|_\infty \le \lambda \\ +\infty & \text{otherwise,} \end{cases} \qquad h(x) = \frac{1}{2}\|Cx - b\|^2,$$

and δ_H is the indicator function of the hyperplane $H = \{x \in \mathbf{R}^n \mid \mathbf{1}^T x = 1\}$. We use the relative entropy distance

$$d_{\mathrm{p}}(x, y) = \sum_{i=1}^n (x_i \log(x_i/y_i) - x_i + y_i), \qquad \mathbf{dom}\, d_{\mathrm{p}} = \mathbf{R}^n_+ \times \mathbf{R}^n_{++}.$$

in the primal space. This distance is 1-strongly convex with respect to ℓ_1-norm [BT09b] (and also ℓ_2-norm). With the relative entropy distance, all the primal iterates $x^{(k)}$ remain feasible. In the dual space we use the Euclidean distance. Thus, the matrix norm (3.10) in the stepsize condition (3.12) for the Bregman Condat–Vũ algorithms is the (1,2)-operator

norm

$$\|A\|_{1,2} = \sup_{v \neq 0} \frac{\|Av\|}{\|v\|_1} = \max_{i=1,\ldots,n} \|a_i\| = \sqrt{2},$$

where a_i is the ith column of A. In the Bregman PD3O algorithm, we use the squared Euclidean distance $d_{\mathrm{p}}(x, y) = \frac{1}{2}\|x-y\|^2$, and the matrix norm in the stepsize condition (3.55) is the spectral norm $\|A\|_2$. For the difference matrix (3.66), $\|A\|_2$ is bounded above by 2, and very close to this upper bound for large n.

The Lipschitz constant for h with respect to the ℓ_1-norm is the largest absolute value of the elements in C^TC, *i.e.*, $L_1 = \max_{i,j} |(C^TC)_{ij}|$. This value is used in the stepsize condition (3.12) for the Bregman Condat–Vũ algorithms. The Lipschitz constant with respect to the ℓ_2-norm is $L_2 = \|C\|_2$, which is used in the stepsize condition (3.55) for Bregman PD3O.

The matrix norms and Lipschitz constants are summarized as follows:

	matrix norm	Lipschitz constant
Bregman Condat–Vũ	$\|A\|_{1,2} = \sqrt{2}$	$L_1 = \max_{i,j} \|(C^TC)_{ij}\|$
Bregman PD3O	$\|A\|_2 \leq 2$	$L_2 = \|C\|_2$.

In the example we use the exact values of L_1 and L_2,

The Bregman proximal operator of f has a closed-form solution, given in (3.5), and the (Euclidean) proximal operator of g^* is the projection onto the infinity norm ball:

$$\mathbf{prox}_{g^*}(z)_i = \begin{cases} \lambda & z_i > \lambda \\ z_i & |z_i| \leq \lambda \\ -\lambda & z_i < -\lambda. \end{cases}$$

The experiment is carried out in Python 3.6 on a desktop with an Intel Core i5 2.4GHz CPU and 8GB RAM. We set $m = 500$ and $n = 10,000$. The elements in the matrix $C \in \mathbf{R}^{m \times n}$ and $b \in \mathbf{R}^m$ are randomly generated from independent standard Gaussian distributions. For the constant stepsize option, we choose

$$\begin{array}{lll} \text{Condat-Vũ} & \sigma = L_1/2 & \tau = 1/(2L_1) \\ \text{PD3O} & \sigma = L_2/4 & \tau = 1/L_2. \end{array} \tag{3.67}$$

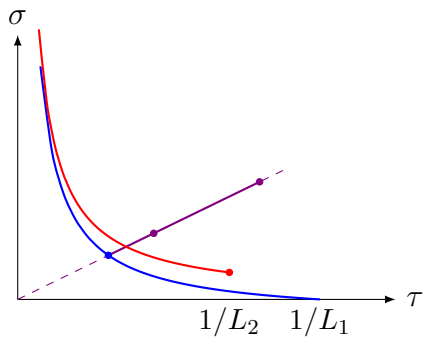

Figure 3.5: The blue and red curves show the boundaries of the stepsize regions for Bregman Condat–Vũ algorithms and Bregman PD3O, respectively. The blue and red points indicate the chosen parameters in (3.67) (red for for PD3O, blue for Condat–Vũ). In the Bregman dual Condat–Vũ algorithm with line search, the stepsizes are selected on the dashed straight line. The solid line segment shows the range of stepsizes that were selected, with dots indicating the largest, median, and smallest stepsizes.

These two choices, as well as the range of possible parameters, are illustrated in Figure 3.5. The two choices are on the blue and red curve, respectively, and satisfy the requirement (3.52) with equality. For the line search algorithm, we set $\bar{\theta}_k = 1.2$ to encourage more aggressive updates, and $\beta = \sigma_{-1}/\tau_{-1} = L_1^2$, which is consistent with the choice in (3.67).

Numerical results We solve the problem (3.65) using the Bregman primal Condat–Vũ algorithm (3.6), the Bregman dual Condat–Vũ algorithm with line search (3.36), and Bregman PD3O (3.51). Figure 3.6 reports the relative distance between the function values to the optimal value $\psi^\star$, which is computed via CVXPY [DB16]. Comparison between the Bregman primal Condat–Vũ algorithm and Bregman PD3O shows that Bregman PD3O converges faster. Figure 3.6 also compares the performance between the Bregman primal Condat–Vũ algorithm with constant stepsizes and Bregman dual algorithm with line search. One can see clearly that the line search significantly improves the convergence. On the other hand, the line search does not add much computation overhead, as the plots of the CPU time and

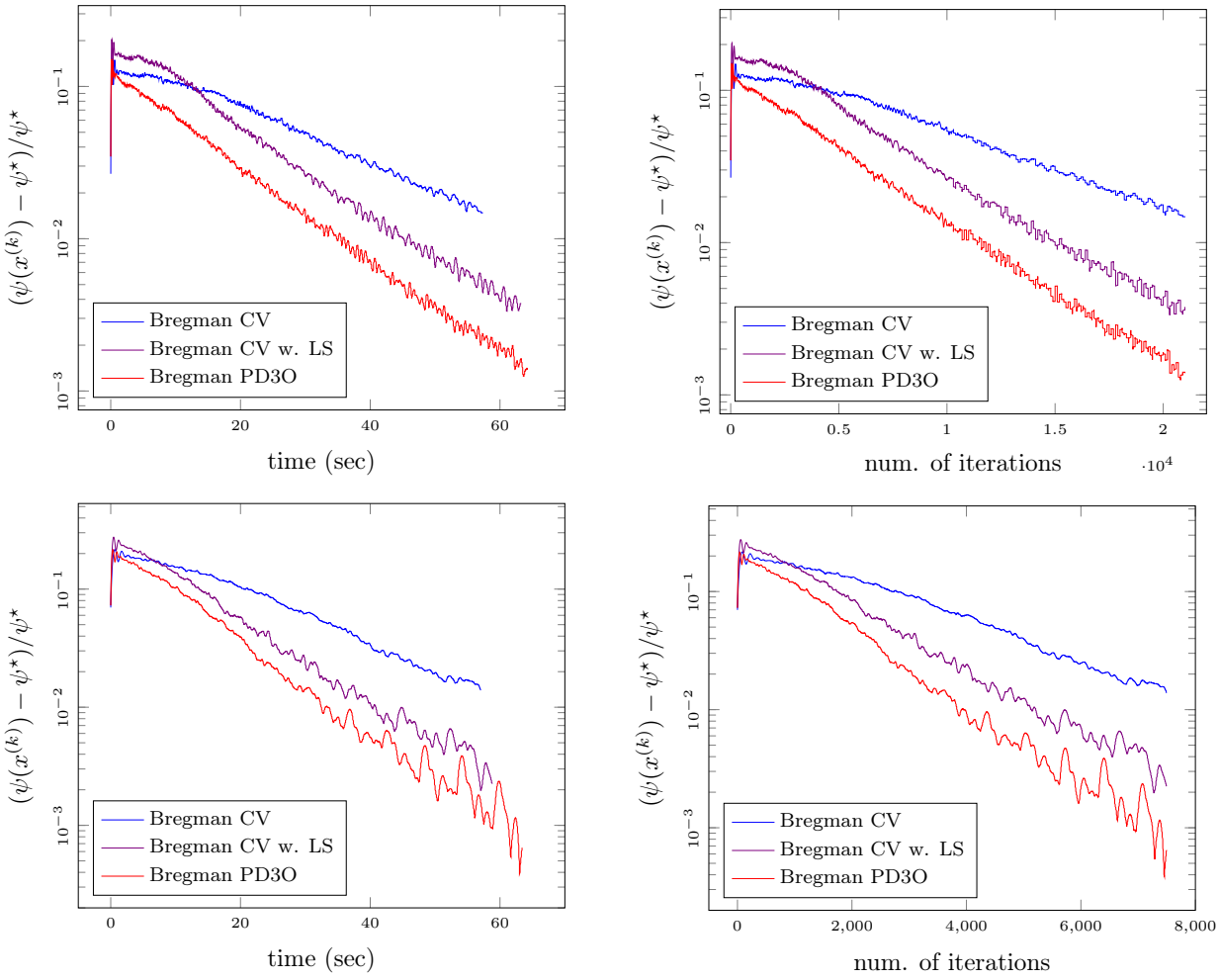

Figure 3.6: Comparison of three algorithms (Bregman primal Condat–Vũ, Bregman dual Condat–Vũ with line search, and Bregman PD3O) in terms of objective values. The top two figures plot the relative error of the function value versus CPU time and number of iterations for one problem instance (3.65), respectively. The bottom two figures correspond to another problem instance.

the number of iterations are roughly identical. In these experiments Bregman PD3O and the Bregman dual Condat–Vũ algorithm with line search have a similar performance, without one algorithm being conclusively better than the other.

CHAPTER 4

Application to sparse semidefinite programming

Bregman proximal splitting methods presented in Chapter 3 offer the flexibility of choosing specific Bregman distances for different problems, with the goal of accelerating the evaluation of proximal operators. In this chapter, we apply this idea to the centering problem in sparse semidefinite programming (SDP). It is motivated by the difficulty of exploiting sparsity in large-scale semidefinite programming in general and, for proximal methods, the need for eigenvalue decompositions to compute Euclidean projections on the positive semidefinite matrix cone. In this chapter, the logarithmic barrier function for the cone of positive semidefinite completable sparse matrices is used as the distance-generating kernel. For this distance, the complexity of evaluating the Bregman proximal operator is shown to be roughly proportional to the cost of a sparse Cholesky factorization, and thus Bregman proximal methods in Chapter 3 can be applied to solve large-scale SDP centering problems efficiently.

This chapter is organized as follows. We first introduce sparse semidefinite programming in Section 4.1. Section 4.2 reviews the logarithmic barriers for two sparse matrix cones and in Section 4.3 we describe the centering problem for sparse SDPs. Based on the discussion in Sections 4.2 and 4.3, we present in Section 4.4 the Bregman proximal operator generated by the logarithmic barrier function for the sparse PSD matrix cone. Then in Section 4.5 we describe in detail the Newton's method used to compute the barrier proximal operator. Section 4.6 contains results of numerical experiments. In the experiments, we apply the Bregman dual PDHG with line search presented in Chapter 3 to solve two sets of sparse SDPs. Most content in this chapter is adapted from [JV22a].

4.1 Sparse semidefinite programming

We consider semidefinite programs (SDPs) in the standard form

$$\begin{array}{lllll} \text{primal:} & \text{minimize} & \mathbf{tr}(CX) & \text{dual:} \quad \text{maximize} & b^T y \\ & \text{subject to} & \mathcal{A}(X) = b & \text{subject to} & \mathcal{A}^*(y) + S = C \\ & & X \in \mathbf{S}^n_+ & & S \in \mathbf{S}^n_+, \end{array} \tag{4.1}$$

with primal variable $X \in \mathbf{S}^n$ and dual variables $S \in \mathbf{S}^n$, $y \in \mathbf{R}^m$, where $\mathbf{S}^n$ is the set of symmetric $n \times n$ matrices. The linear operator $\mathcal{A}\colon \mathbf{S}^n \to \mathbf{R}^m$ is defined as

$$\mathcal{A}(X) = \big(\mathbf{tr}(A_1X), \mathbf{tr}(A_2X), \ldots, \mathbf{tr}(A_mX)\big)$$

and $\mathcal{A}^*(y) = \sum_{i=1}^m y_i A_i$ is its adjoint operator. The coefficients $C, A_1, \ldots, A_m$ are symmetric $n \times n$ matrices. The notation $\mathbf{S}^n_+$ is used for the cone of positive semidefinite (PSD) matrices in $\mathbf{S}^n$.

In many large-scale applications of semidefinite programming, the coefficient matrices are sparse. The sparsity pattern of a symmetric $n \times n$ matrix can be represented by an undirected graph $G = (V, E)$ with vertex set $V = \{1, 2, \ldots, n\}$ and edge set E. The set of matrices with sparsity pattern E is then defined as

$$\mathbf{S}^n_E = \{Y \in \mathbf{S}^n \mid Y_{ij} = Y_{ji} = 0 \ \text{ if } i \neq j \text{ and } \{i, j\} \notin E\}.$$

In this chapter, E will denote the common (or *aggregate*) sparsity pattern of the coefficient matrices in the SDP, *i.e.*, we assume that $C, A_1, \ldots, A_m \in \mathbf{S}^n_E$. Note that the sparsity pattern E is not uniquely defined (unless it is dense, *i.e.*, the sparsity graph G is complete): if the coefficients are in $\mathbf{S}^n_E$ then they are also in $\mathbf{S}^n_{E'}$ where $E \subset E'$. In particular, E can always be extended to make the graph $G = (V, E)$ *chordal* or *triangulated* [BP93, VA15]. Without loss of generality, we will assume that this is the case.

Although aggregate sparsity is not always present in large SDPs with sparse coefficients, it appears naturally in applications with an underlying graph structure. The graph structure in the application is often inherited by all the coefficients of the SDP. Examples include

relaxations of combinatorial graph optimization problems [Lov79, Ali95, GW95, GR00] and symmetric eigenvalue optimization problems associated with graphs [BDX04, XB04]. Aggregate sparsity also arises in SDP formulations of Euclidean distance geometry problems with network structure, with applications to network node localization [BY04, BLT06, BLW06, KKW09, DKQ10] and machine learning [WS06, WSZ07]. Another important source of SDPs with aggregate sparsity is given by the semidefinite relaxations of the optimal power flow problem in electricity networks [Jab12, Low14a, Low14b, MKL15, MHL13, Tay15]. When the aggregate sparsity pattern is dense or almost dense, it is sometimes possible to introduce or enhance sparsity via variable transformation [FKM01, Section 6]. Applications of this technique arise in the graph partition problem [FKM01] and the SDP formulation of the maximum variance unfolding problem [WS04].

Sparse semidefinite programming Even when the coefficient matrices in the SDP (4.1) share an aggregate sparsity pattern, the primal variable X generally needs to be dense to be feasible. However, the cost function and the linear equality constraints only depend on the diagonal entries X_{ii} and the off-diagonal entries $X_{ij} = X_{ji}$ for $\{i, j\} \in E$. For the other entries the only requirement is to make the matrix positive semidefinite. In the dual problem, $S \in \mathbf{S}^n_E$ holds at all dual feasible points. These observations imply that the SDPs (4.1) can be equivalently rewritten as a pair of primal and dual conic linear programs

$$\begin{array}{lllll} \text{primal:} & \text{minimize} & \mathbf{tr}(CX) & \text{dual:} & \text{maximize} \quad b^T y \\ & \text{subject to} & \mathcal{A}(X) = b & & \text{subject to} \quad \mathcal{A}^*(y) + S = C \\ & & X \in K & & \qquad\qquad S \in K^*, \end{array} \tag{4.2}$$

with *sparse* matrix variables $X, S \in \mathbf{S}^n_E$, and a vector variable $y \in \mathbf{R}^m$. The primal cone K in this problem is the set of matrices in $\mathbf{S}^n_E$ which have a positive semidefinite completion, *i.e.*, $K = \Pi_E(\mathbf{S}^n_+)$ where Π_E stands for projection on $\mathbf{S}^n_E$. The dual cone K^* of K is the set of positive semidefinite matrices with sparsity pattern E, *i.e.*, $K^* = \mathbf{S}^n_+ \cap \mathbf{S}^n_E$. The formulation (4.2) is attractive when the aggregate sparsity pattern E is very sparse, in which

case $\mathbf{S}^n_E$ is a much lower-dimensional space than $\mathbf{S}^n$. The nonsymmetric formulation of sparse semidefinite programming was applied in [ADV10, Bur03, SV04, SAV14]; see also [VA15, Section 14.4].

Algorithms for sparse SDPs Sparse structure in semidefinite programming has been extensively explored by many authors and leveraged in many algorithms. The scalability of interior-point methods is limited by the need to form and solve a set of m linear equations in m variables, known as the *Schur complement system*, at each iteration. This system is usually dense. Sparsity in the coefficients A_i can be exploited to reduce the cost of assembling the Schur complement equations. This process is efficient especially in extremely sparse problems, where the coefficients A_i may also have low rank. In dual barrier methods, one can also take advantage of sparsity of dual feasible variables S. These properties are leveraged in the dual interior-point methods described in [BYZ00, BY08, BGM13, BGP19, BGP21].

In another line of research, techniques based on properties and algorithms for chordal sparsity patterns have been applied to semidefinite programming since the late 1990s [FKN97, FKM01, NFF03, BYZ00, Bur03, KKK08, KKM11, ADV13, ZL21, SAV14, PHA18, SV04]; see [VA15, ZFP21] for recent surveys. An important tool from this literature is the *conversion* or *clique decomposition method* proposed by Fukuda *et al.* [FKM01, NFF03]. It is based on a fundamental result from linear algebra, stating that for a chordal pattern E, a matrix $X \in \mathbf{S}^n_E$ has a positive semidefinite completion if and only if $X_{\gamma_k\gamma_k} \succeq 0$ for $k = 1, \ldots, r$, where $\gamma_1, \ldots, \gamma_r$ are the maximal cliques in the graph [GJS84]. In the conversion method, the large sparse variable matrix X in (4.2) is replaced with smaller dense matrix variables $X_k = X_{\gamma_k\gamma_k}$. Each of these new variables is constrained to be positive semidefinite. Linear equality constraints need to be added to couple the variables X_k, as they represent overlapping subblocks of a single matrix X. Thus, a large sparse SDP is converted in an equivalent problem with several smaller, dense variables X_k, and additional sparse equality constraints. This equivalent problem may be considerably easier to solve by interior-point methods than

the original SDP (4.1). Recent examples where the clique decomposition is applied to solve large sparse SDPs can be found in [EDA20, ZL21].

Proximal splitting methods, which are surveyed in Chapter 2, are perhaps the most popular alternatives to interior-point methods in machine learning, image processing, and other applications, involving large-scale convex programming, and typical examples include (accelerated) proximal gradient methods [BT09a, BT09b, Nes18], ADMM [BPC11], and the primal–dual hybrid gradient (PDHG) or Chambolle–Pock method [PCB09, EZC10, CP11a] (see more examples in Chapter 2). When applied to the SDPs (4.1), they require at each iteration a Euclidean projection on the positive semidefinite cone $\mathbf{S}_+^n$ and hence, a symmetric eigenvalue decomposition of order n. This contributes an order n^3 term to the per-iteration complexity. In the nonsymmetric formulation (4.2) of the sparse SDP, the projections on K^* or (equivalently) K cannot be computed directly, and must be handled by introducing splitting variables and alternating projections on $\mathbf{S}_E^n$, which is trivial, and on $\mathbf{S}_+^n$, which requires an eigenvalue decomposition. The clique decomposition used in the conversion method described above, which was originally developed for interior-point methods, lends itself naturally to splitting algorithms as well. It allows us to replace the matrix constraint $X \in K$ with several smaller dense inequalities $X_k \succeq 0$, one for each maximal clique in the sparsity graph. In a proximal method, this means that projection on the $n \times n$ positive semidefinite cone can be replaced by less expensive projections on lower-dimensional positive semidefinite cones [MKL15, SV15, ZFP17, ZFP20]. This advantage of the conversion method is tempered by the large number of consistency constraints that must be introduced to link the splitting variables X_k. First-order methods typically do not compute very accurate solutions and if the residual error in the consistency constraints is not small, it may be difficult to convert the computed solution of the decomposed problem back to an accurate solution of the original SDP [EDA20].

4.2 Primal and dual barriers

In this section we introduce the logarithmic barrier functions for the pair of primal and dual cones K and K^*, and later in Section 4.4 the primal barrier will be used as the kernel function to generate the Bregman distance. The logarithmic barrier functions for the cones $K^* = \mathbf{S}^n_+ \cap \mathbf{S}^n_E$ and $K = \Pi_E(\mathbf{S}^n_+)$ are defined as

$$\phi_*(S) = -\log\det S, \qquad \phi(X) = \sup_S \big(-\mathbf{tr}(XS) - \phi_*(S)\big), \tag{4.3}$$

with domains $\mathbf{dom}\,\phi_* = \mathbf{int}\,K^*$ and $\mathbf{dom}\,\phi = \mathbf{int}\,K$, respectively. Note that $\phi(X)$ is the conjugate of ϕ_* evaluated at $-X$.

In [ADV13, VA15] efficient algorithms are presented for evaluating the two barrier functions, their gradients, and their directional second derivatives, when the sparsity pattern E is chordal. The value of the dual barrier $\phi_*(S) = -\log\det S$ is easily computed from the diagonal entries in a sparse Cholesky factor of S. More specifically, if a factorization $PSP^T = LDL^T$ is available, with P a permutation matrix, D positive diagonal and L unit lower-triangular, then $\phi_*(S) = -\sum_i \log D_{ii}$. The gradient and Hessian are given by

$$\nabla\phi_*(S) = -\Pi_E(S^{-1}), \qquad \nabla^2\phi_*(S)[V] = \frac{d}{dt}\nabla\phi_*(S+tV) = \Pi_E(S^{-1}VS^{-1}). \tag{4.4}$$

Given a Cholesky factorization of S, these expressions can be evaluated via one or two recursions on the elimination tree [ADV13, VA15], without explicitly computing the entire inverse S^{-1} or the matrix product $S^{-1}VS^{-1}$. The cost of these recursions is roughly the same as the cost of a sparse Cholesky factorization with the sparsity pattern E [ADV13, VA15].

The primal barrier function ϕ and its gradient can be evaluated by solving the optimization problem in the definition of $\phi(X)$. The optimal solution $\hat{S}_X$ is the matrix in $\mathbf{S}^n_{++} \cap \mathbf{S}^n_E$ that satisfies

$$\Pi_E(\hat{S}_X^{-1}) = X. \tag{4.5}$$

Its inverse $\hat{S}_X^{-1}$ is also the maximum determinant positive definite completion of X, *i.e.*,

$Z = \hat{S}_X^{-1}$ is the solution of

$$\begin{array}{ll} \text{maximize} & \log\det Z \\ \text{subject to} & \Pi_E(Z) = X \end{array} \tag{4.6}$$

(where we take $\mathbf{S}^n_{++}$ as the domain of the cost function). The solution Z is also called the *maximum entropy* completion of X, since it maximizes the entropy of the normal distribution $\mathcal{N}(0, Z)$, which is given by

$$\frac{1}{2}(\log\det Z + n\log(2\pi) + n),$$

subject to the constraint $\Pi_E(Z) = X$; see [Dem72]. From $\hat{S}_X$, one obtains

$$\phi(X) = \log\det \hat{S}_X - n, \qquad \nabla\phi(X) = -\hat{S}_X, \qquad \nabla^2\phi(X) = \nabla^2\phi_*(\hat{S}_X)^{-1}. \tag{4.7}$$

Comparing the expressions for the gradients of ϕ and ϕ_* in (4.7) and (4.4), and using (4.5), we see that $\nabla\phi$ and $\nabla\phi_*$ are inverse mappings, up to a change in sign:

$$\nabla\phi(X) = -\hat{S}_X = -(\nabla\phi_*)^{-1}(-X), \qquad \nabla\phi_*(S) = -(\nabla\phi)^{-1}(-S).$$

For general sparsity patterns, the determinant maximization problem (4.6) or the convex optimization problem in the definition of ϕ must be solved by an iterative optimization algorithm. If the pattern is chordal, these optimization problems can be solved by finite recursive algorithms, again at a cost that is comparable with the cost of a sparse Cholesky factorization for the same pattern [ADV13, VA15].

4.3 The centering problem

To apply Bregman proximal methods discussed in Chapter 3, we consider the centering problem for the sparse SDP (4.2)

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) + \mu\phi(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{4.8}$$

where ϕ is the logarithmic barrier function for the cone K, given in (4.3). The centering parameter $\mu > 0$ controls the duality gap at the solution. Since the barrier function ϕ is n-logarithmically homogeneous (see [VA15]), the optimal solution of the centering problem is a (μn)-suboptimal solution for the original SDP (4.2). The centering problem (4.8) is useful as an approximation to the original problem, because it yields more easily computed suboptimal solutions, with an accuracy that can be controlled by the choice of the barrier parameter μ. The centering problem is also a key component of barrier methods, in which a sequence of centering problems with decreasing values of the barrier parameter are solved. Traditionally, the centering problem in interior-point methods is solved by the Newton's method, possibly accelerated via the preconditioned conjugate gradient method [BGP19, VB95], but recent work has started to examine the use of proximal methods such as the alternating direction method of multipliers (ADMM) or the proximal method of multipliers for this purpose [LMY21, PG21, PG22].

We will assume that the equality constraints in (4.2) include a constraint $\mathbf{tr}(NX) = 1$, where $N \in \mathbf{S}^n_{++} \cap \mathbf{S}^n_E$. To make this explicit we write the centering problem (4.2) as

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) + \mu\phi(X) \\ \text{subject to} & \mathcal{A}(X) = b, \\ & \mathbf{tr}(NX) = 1. \end{array} \tag{4.9}$$

For $N = I$, the normalized cone $\{X \in K \mid \mathbf{tr}(NX) = 1\}$ is a matrix extension of the probability simplex $\{x \succeq 0 \mid \mathbf{1}^T x = 1\}$, sometimes referred to as the *spectraplex*. With minor changes, the techniques we discuss extend to a normalization in the inequality form $\mathbf{tr}(NX) \leq 1$, with $N \in \mathbf{S}^n_{++} \cap \mathbf{S}^n_E$. Here we will discuss (4.9) to retain the standard form of the centering problem.

The constraints $\mathbf{tr}(NX) = 1$ and $\mathbf{tr}(NX) \leq 1$ guarantee the boundedness of the primal feasible set, a common assumption in first-order methods. The added constraint does not diminish the generality of our approach. In many applications an equality $\mathbf{tr}(NX) = 1$ is implied by the contraints $\mathcal{A}(X) = b$ and easily derived from the problem data (see Section 4.6

for two typical examples). When an equality constraint of this form is not readily available, one can add a bounding inequality $\mathbf{tr}(NX) \leq 1$ with N sufficiently small to ensure that the optimal solution is not modified.

To apply first-order proximal methods, we view the problem (4.9) as a linearly constrained optimization problem

$$\begin{array}{ll} \text{minimize} & f(X) \\ \text{subject to} & \mathcal{A}(X) = b, \end{array} \tag{4.10}$$

where f is defined as

$$f(X) = \mathbf{tr}(CX) + \mu\phi(X) + \delta_{\mathcal{H}}(X), \qquad \mathcal{H} = \{X \in \mathbf{S}_E^n \mid \mathbf{tr}(NX) = 1\}, \tag{4.11}$$

and $\delta_{\mathcal{H}}$ is the indicator function of the hyperplane $\mathcal{H}$. Therefore problem (4.10) is in the canonical form (2.1) with $g = \delta_{\{b\}}$ and $h = 0$.

4.4 Barrier proximal operator for sparse PSD matrix cone

The reformulated problem (4.10) can be solved by Bregman proximal splitting methods presented in Chapter 3. Here the primal kernel is chosen as the barrier function ϕ (4.3) for the cone K, and the dual kernel is the squared Euclidean kernel. The primal kernel ϕ satisfies the assumptions listed in Section 3.1, *i.e.*, it is convex, continuous, continuously differentiable on the interior of the cone, and strongly convex on $\mathbf{int}\, K \cap \{X \mid \mathbf{tr}(NX) = 1\}$. It generates the Bregman divergence

$$\begin{aligned} d(X, Y) &= \phi(X) - \phi(Y) - \mathbf{tr}\left(\nabla\phi(Y)(X - Y)\right) \\ &= \phi(X) - \log\det \hat{S}_Y + n + \mathbf{tr}\left(\hat{S}_Y(X - Y)\right) \\ &= \phi(X) - \log\det \hat{S}_Y + \mathbf{tr}\left(\hat{S}_Y X\right). \end{aligned}$$

On line 2 we used the properties (4.7) to express $\phi(Y)$ and $\nabla\phi(Y)$. The Bregman proximal operator (3.2) for the function f defined in (4.11) then becomes

$$\hat{X} = \mathbf{prox}_f^{\phi}(Y, A)$$

$$
\begin{aligned}
&= \operatorname*{argmin}_{\mathbf{tr}(NX)=1} \left(\mathbf{tr}(CX) + \mu\phi(X) + \mathbf{tr}(AX) + d(X,Y)\right) \\
&= \operatorname*{argmin}_{\mathbf{tr}(NX)=1} \left(\mathbf{tr}\left((C + A - \nabla\phi(Y))X\right) + (\mu+1)\phi(X)\right) \\
&= \operatorname*{argmin}_{\mathbf{tr}(NX)=1} \left(\mathbf{tr}(BX) + \phi(X)\right)
\end{aligned}
$$

where

$$
B = \frac{1}{1+\mu}(C + A + \hat{S}_Y).
$$

To compute $\hat{X}$ we therefore need to solve an optimization problem

$$
\begin{array}{ll}
\text{minimize} & \mathbf{tr}(BX) + \phi(X) \\
\text{subject to} & \mathbf{tr}(NX) = 1,
\end{array}
\tag{4.12}
$$

where $B \in \mathbf{S}_E^n$ and $N \in \mathbf{S}_{++}^n \cap \mathbf{S}_E^n$. If we introduce a Lagrange multiplier ν for the equality constraint in (4.12), the optimality condition can be written as

$$
\nabla\phi(X) + B + \nu N = 0, \qquad \mathbf{tr}(NX) = 1.
$$

Equivalently, since $\nabla\phi_*(S) = -(\nabla\phi)^{-1}(-S)$,

$$
X = -\nabla\phi_*(B + \nu N) = \Pi_E((B + \nu N)^{-1}), \qquad \mathbf{tr}(NX) = 1.
$$

Eliminating X we obtain a nonlinear equation in ν:

$$
\mathbf{tr}(N(B + \nu N)^{-1}) = 1. \tag{4.13}
$$

(The projection in $\mathbf{tr}(N\Pi_E((B + \nu N)^{-1}))$ can be omitted because the matrix N has the sparsity pattern E.) The unique solution ν that satisfies $B + \nu N \succ 0$ defines the solution $X = \Pi_E((B + \nu N)^{-1})$ of (4.12).

The equation (4.13) is also the optimality condition for the Lagrange dual of (4.12), which is a smooth unconstrained convex optimization problem in the scalar variable ν:

$$
\text{maximize} \quad -\phi_*(B + \nu N) - \nu. \tag{4.14}
$$

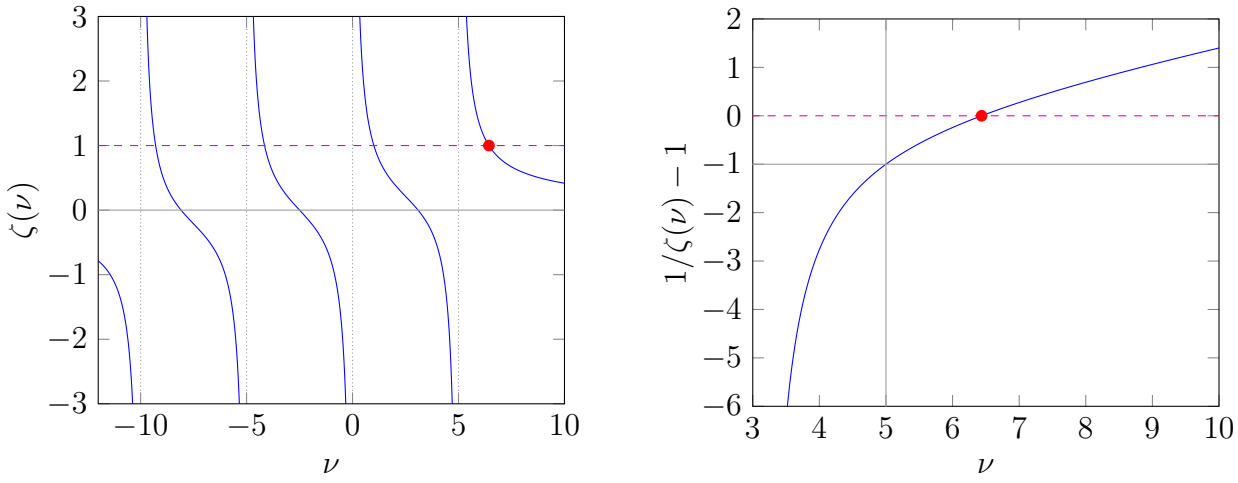

Figure 4.1: *Left.* The function $\zeta(\nu) = \sum_i 1/(\nu + \lambda_i)$ for $\lambda = (-5, 0, 5, 10)$. We are interested in the solution of $\zeta(\nu) = 1$ larger than $-\lambda_{\min} = 5$. *Right.* The function $1/\zeta(\nu) - 1$.

4.5 Newton's method for barrier proximal operator

In this section we discuss in detail Newton's method applied to the dual problem (4.14) and the equivalent nonlinear equation (4.13). We write the equation as $\zeta(\nu) = 1$ where

$$\zeta(\nu) = \mathbf{tr}(N(B + \nu N)^{-1}), \qquad \zeta'(\nu) = -\mathbf{tr}(N(B + \nu N)^{-1}N(B + \nu N)^{-1}). \tag{4.15}$$

The function ζ and its derivative can be expressed in terms of the generalized eigenvalues λ_i of (B, N) as

$$\zeta(\nu) = \sum_{i=1}^{n} \frac{1}{\nu + \lambda_i}, \qquad \zeta'(\nu) = -\sum_{i=1}^{n} \frac{1}{(\nu + \lambda_i)^2}. \tag{4.16}$$

Figure 4.1 shows an example with $n = 4$, $N = I$, and eigenvalues $10, 5, 0, -5$.

We are interested in computing the solution of $\zeta(\nu) = 1$ that satisfies $B + \nu N \succ 0$, *i.e.*, $\nu > -\lambda_{\min}$, where $\lambda_{\min} = \min_i \lambda_i$ is the smallest generalized eigenvalue of (B, N). We denote this interval by $J = (-\lambda_{\min}, \infty)$. The equation $\zeta(\nu) = 1$ is guaranteed to have a unique solution in J because ζ is monotonic and continuous on this interval, with

$$\lim_{\nu \to -\lambda_{\min}} \zeta(\nu) = \infty, \qquad \lim_{\nu \to \infty} \zeta(\nu) = 0.$$

Furthermore, on the interval J, the function ζ and its derivative can be expressed as

$$\zeta(\nu) = -\mathbf{tr}(N\nabla\phi_*(B+\nu N)), \qquad \zeta'(\nu) = -\mathbf{tr}(N(\nabla^2\phi_*(B+\nu N)[N])).$$

Therefore $\zeta(\nu)$ and $\zeta'(\nu)$ can be evaluated by taking the inner product of N with

$$\begin{aligned}\nabla\phi_*(B+\nu N) &= -\Pi_E\left((B+\nu N)^{-1}\right)\\ \nabla^2\phi_*(B+\nu N)[N] &= -\Pi_E\left((B+\nu N)^{-1}N(B+\nu N)^{-1}\right).\end{aligned}$$

Since $B, N \in \mathbf{S}^n_E$, these quantities can be computed by the efficient algorithms for computing the gradient and directional second derivative of ϕ_* described in [ADV13, VA15].

We note a few other properties of ζ. First, the expressions in (4.16) show that ζ is convex, decreasing, and positive on J. Second, if $\nu \in J$, then $\tilde{\nu} \in J$ for all $\tilde{\nu}$ that satisfy

$$\tilde{\nu} > \nu - \frac{1}{\sqrt{|\zeta'(\nu)|}}. \tag{4.17}$$

This follows from

$$|\zeta'(\nu)| = \sum_{i=1}^n \frac{1}{(\nu+\lambda_i)^2} \geq \frac{1}{(\nu+\lambda_{\min})^2},$$

and is also a simple consequence of the Dikin ellipsoid theorem for self-concordant functions [NN94, Theorem 2.1.1.b].

The Newton iteration for the equation $\zeta(\nu) - 1 = 0$ is

$$\nu^+ = \nu + \alpha\frac{1-\zeta(\nu)}{\zeta'(\nu)}, \tag{4.18}$$

where α is a step size. The same iteration can be interpreted as a damped Newton method for the unconstrained problem (4.14). If $\nu^+ \in J$ for a unit step $\alpha = 1$, then

$$\zeta(\nu^+) > \zeta(\nu) + \zeta'(\nu)(\nu^+ - \nu) = 1,$$

from strict convexity of ζ. Hence after one full Newton step, the Newton iteration with unit steps approaches the solution monotonically from the left. If $\zeta(\nu) < 1$ then in general

a non-unit step size must be taken to keep the iterates in J. From the Dikin ellipsoid inequality (4.17), we see that $\nu^+ \in J$ for all positive α that satisfy

$$\alpha < \frac{\sqrt{|\zeta'(\nu)|}}{1-\zeta(\nu)}.$$

The theory of self-concordant functions provides a step size rule that satisfies this condition and guarantees convergence:

$$\alpha = \frac{\sqrt{|\zeta'(\nu)|}}{\sqrt{|\zeta'(\nu)|}+1-\zeta(\nu)} \quad \text{if} \quad \frac{1-\zeta(\nu)}{\sqrt{|\zeta'(\nu)|}} < \eta, \qquad \alpha = 1 \quad \text{otherwise,}$$

where η is a constant in $(0,1)$. As an alternative to this fixed step size rule, a standard backtracking line search can be used to determine a suitable step size α in (4.18). Checking whether $\nu^+ \in J$ can be done by attempting a sparse Cholesky factorization of $B+\nu^+ N$.

Figure 4.1 shows that the function ζ can be quite nonlinear around the solution of the equation if the solution is near $-\lambda_{\min}$. Instead of applying Newton's method directly to (4.15), it is useful to rewrite the nonlinear equation as $\psi(\nu)=0$ where

$$\psi(\nu) = \frac{1}{\zeta(\nu)} - 1. \tag{4.19}$$

The negative smallest eigenvalue $-\lambda_{\min}$ is a pole of $\zeta(\nu)$, but a zero of $1/\zeta(\nu)$. Also the derivative of ψ changes slowly near this zero point; in Figure 4.1, the function ψ is almost linear in the region of interest. This implies that Newton's method applied to (4.19), *i.e.*,

$$\nu^+ = \nu + \beta\frac{\psi(\nu)}{\psi'(\nu)} = \nu + \beta\frac{\zeta(\nu)(1-\zeta(\nu))}{\zeta'(\nu)},$$

should be extremely efficient in this case. Starting the line search at $\beta=1$ is equivalent to starting at $\alpha=\zeta(\nu)$ in (4.18). This often requires fewer backtracking steps than starting at $\alpha=1$.

Newton's method requires a feasible initial point $\nu_0 \in J$. Suppose we know a positive lower bound γ on the smallest eigenvalue of N. Then $\hat{\nu}_0 \in J$ where

$$\hat{\nu}_0 > \max\{0, \frac{-\lambda_{\min}(B)}{\gamma}\}.$$

A lower bound on $\lambda_{\min}(B)$ can be obtained from the Gershgorin circle theorem, which states that the eigenvalues of B are contained in the disks

$$\Big\{ s \;\Big|\; |s - B_{ii}| \leq \sum_{j \neq i} |B_{ij}| \Big\}, \quad i = 1, \ldots, n.$$

Thus, $\lambda_{\min}(B) \geq \min_i \left(B_{ii} - \sum_{j \neq i} |B_{ij}|\right)$. Apart from the above initialization, we find another practically useful initial point $\tilde{\nu}_0 = n - \mathbf{tr}\, B / \mathbf{tr}\, N$, which is the solution for $\mathbf{tr}(N(B + \nu N)^{-1}) = 1$ when B happens to be a multiple of N. This choice is efficient in many practical examples but, unfortunately, not guaranteed to be feasible. Thus, in the implementation, we use $\tilde{\nu}_0$ if it is feasible and $\hat{\nu}_0$ otherwise.

4.6 Numerical experiments

In this section we evaluate the performance of Bregman PDHG with line search, *i.e.* (3.36) with $h = 0$, applied to the centering problem (4.10). For clarity we rewrite the algorithm here in matrix notation:

$$\bar{z}^{(k+1)} = z^{(k)} + \theta_k (z^{(k)} - z^{(k-1)}) \tag{4.20a}$$

$$X^{(k+1)} = \operatorname*{argmin}_x \big(f(X) + \langle \bar{z}^{(k+1)}, \mathcal{A}(X) \rangle + \frac{1}{\tau_k} d(X, X^{(k)}) \big) \tag{4.20b}$$

$$z^{(k+1)} = z^{(k)} + \sigma_k (\mathcal{A}(X^{(k+1)}) - b), \tag{4.20c}$$

where d is the Bregman distance generated by the barrier function ϕ (4.3). The numerical results illustrate that the cost for evaluating the Bregman proximal operator (4.12) is comparable to the cost of a sparse Cholesky factorization with sparsity pattern E. This prox-evaluation dominates the computational cost in each iteration of (4.20), since $\mathcal{A}$ and $\mathcal{A}^*$ are usually easy to evaluate for large-scale problems with sparse or other types of structure. In particular, the proposed method does not need to solve linear equations involving $\mathcal{A}$ or $\mathcal{A}^*$, an important advantage over ADMM and interior-point methods.

In this section we consider the centering problem for two sets of sparse SDPs, the maximum cut problem and the graph partitioning problem. The experiments are carried out

in Python 3.6 on a laptop with an Intel Core i5 2.4GHz CPU and 8GB RAM. The Python library for chordal matrix computations CHOMPACK [AV15] is used to compute chordal extensions (with the AMD reordering [ADD96]), sparse Cholesky factorizations, the primal barrier ϕ, and the gradient and directional second derivative of the dual barrier ϕ_*. Other sparse matrix computations are implemented using CVXOPT [ADV20].

In the experiments, we terminate the iteration (4.20) when the relative primal and dual residuals are less than 10^{-6}. These two stopping conditions are sufficient for our algorithm, as suggested by the convergence proof, in particular, equations (3.49) and (3.50). The two residuals are defined as

$$\text{primal residual} = \frac{\|z_k - z_{k-1}\|_2}{\sigma_k \max\{1, \|z_k\|_\infty\}}, \qquad \text{dual residual} = \frac{\|\nabla\phi(X_k) - \nabla\phi(X_{k-1})\|_2}{\tau_k \max\{1, \|X_k\|_{\max}\}},$$

where $\|Y\|_{\max} = \max_{i,j} |Y_{ij}|$.

4.6.1 Maximum cut problem

Given an undirected graph $G = (V, E)$, the maximum cut problem is to partition the set of vertices into two sets in order to maximize the total number of edges between the two sets. (If every edge $\{i, j\} \in E$ is associated with a nonnegative weight w_{ij}, then the maximum cut problem is to maximize the total weight of the edges between the two sets.) One can show that the maximum cut problem can be represented as a binary quadratic optimization problem

$$\begin{array}{ll} \text{maximize} & (1/4)x^T L x \\ \text{subject to} & x \in \{\pm 1\}^n, \end{array}$$

where $L \in \mathbf{S}^n$ is the Laplacian of an undirected graph $G = (V, E)$ with vertices $V = \{1, 2, \ldots, n\}$. The SDP relaxation of the maximum cut problem is

$$\begin{array}{ll} \text{maximize} & (1/4)\,\mathbf{tr}(LX) \\ \text{subject to} & \mathbf{diag}(X) = \mathbf{1} \\ & X \succeq 0, \end{array} \tag{4.21}$$

with variable $X \in \mathbf{S}^n$. The operator $\mathbf{diag}\colon \mathbf{S}^n \to \mathbf{R}^n$ returns the diagonal elements of the input matrix as a vector: $\mathbf{diag}(X) = (X_{11}, X_{22}, \ldots, X_{nn})$. If moderate accuracy is allowed, we can solve the centering problem of the SDP relaxation

$$\begin{array}{ll} \text{minimize} & -(1/4)\,\mathbf{tr}(LX) + \mu\phi(X) \\ \text{subject to} & \mathbf{diag}(X) = \mathbf{1} \\ & X \in \Pi_{E'}(\mathbf{S}^n_+) \end{array} \tag{4.22}$$

with optimization variable $X \in \mathbf{S}^n_{E'}$ where E' is a chordal extension of E. Note that $\mathbf{tr}(X) = n$ for all feasible X. The centering problem has the form of (4.10) with

$$C = -\frac{1}{4}L, \qquad N = \frac{1}{n}I, \qquad \mathcal{A}(X) = \mathbf{diag}(X).$$

The Lagrangian of (4.22) is given by

$$\mathcal{L}(X, z) = f(X) + \langle z, \mathcal{A}(X) - b\rangle,$$

where f is defined in (4.11), and z is the Lagrange multiplier associated with the equality constraint $\mathbf{diag}(X) = \mathbf{1}$. Thus we have

$$\frac{1}{4}\,\mathbf{tr}(LX^\star) \le p^\star_{\text{sdp}} \le \mathbf{1}^T z^\star, \qquad -\frac{1}{4}\,\mathbf{tr}(LX^\star) + \mathbf{1}^T z^\star = \mu n, \tag{4.23}$$

where $X^\star$ and $z^\star$ are the primal and dual optimal solutions of the centering problem (4.22), and $p^\star_{\text{sdp}}$ is the optimal value of the SDP (4.21).

Numerical results We first collect four MAXCUT problems of moderate size from SDPLIB [Bor99]. The SDP relaxation (4.21) is solved using MOSEK [MOS19] and the optimal value computed by MOSEK is denoted by $p^\star_{\text{sdp}}$. (Note that the source file for the graph maxcutG55 was incorrectly converted into SDPA sparse format. Thus the objective value for the maxG55 problem obtained from the original data file is 1.1039×10^4 instead of 9.9992×10^3 as reported in SDPLIB.)

In (4.22), we set $\mu = 0.001/n$, and report in column 4 of Table 4.1 the difference between $p^\star_{\text{sdp}}$ and the cost function $(1/4)\,\mathbf{tr}(LX)$ at the suboptimal solution returned by the

	n	$p^\star_{\text{sdp}}$	$p^\star_{\text{sdp}} - \frac{1}{4}\,\mathbf{tr}(LX)$	primal residual	dual residual
maxG51	1000	4.0039×10^3	3.12×10^{-4}	2.24×10^{-7}	6.43×10^{-8}
maxG32	2000	1.5676×10^3	6.95×10^{-4}	6.48×10^{-7}	2.23×10^{-7}
maxG55	5000	1.1039×10^4	1.02×10^{-4}	5.32×10^{-7}	7.13×10^{-7}
maxG60	7000	1.5222×10^4	9.91×10^{-5}	1.21×10^{-7}	2.33×10^{-7}

Table 4.1: Results for four instances of the MAXCUT problem from SDPLIB [Bor99]. Column 3 is the optimal value computed by MOSEK. Column 4 is the difference with the optimal value of the centering problem computed by algorithm (4.20). The last two columns give the primal and dual residuals in the computed solution.

algorithm. The last two columns of Table 4.1 give the relative primal and dual residuals. These results show that the proposed algorithm is able to solve the centering SDP (4.22) with the desired accuracy. A comparison of the third and fourth columns of Table 4.1 confirms (4.23), *i.e.*, the objective value of the SDP at X is within $\mu n = 10^{-3}$ of the optimal value. Considering the values of $p^\star_{\text{sdp}}$, we see that the computed points on the central path are close to the optimal solutions of the SDPs.

To test the scalability of algorithm (4.20), we add four larger graphs from the SuiteSparse collection [KAB19]. In Table 4.2 we report the time per Cholesky factorization, the number of Newton steps per iteration, the time per PDHG iteration, and the number of iterations in Bregman PDHG for the eight test problems. As can be seen from the table, the number of Newton iterations per prox-evaluation remains small even when the size of the problem increases. Also, we observe that the time per PDHG iteration is roughly the cost of a sparse Cholesky factorization times the number of Newton steps. This means that the backtracking in Newton's method does not cause a significant overhead. Since the evaluations of $\mathcal{A}$ and $\mathcal{A}^*$ in this problem are very cheap, the cost per prox-evaluation is the dominant term in the per-iteration complexity.

	n	time per Cholesky factorization	Newton steps per iteration	time per PDHG iteration	PDHG iterations
maxG51	1000	0.05	2.45	0.12	267
maxG32	2000	0.12	1.56	0.18	240
maxG55	5000	0.29	2.10	0.58	249
maxG60	7000	0.60	2.55	1.22	279
barth4	6019	0.42	3.57	1.55	346
tuma2	12992	0.48	4.36	1.89	375
biplane-9	21701	0.95	2.58	2.12	287
c-67	57975	0.76	3.58	3.56	378

Table 4.2: The four MAXCUT problems from SDPLIB plus four larger graphs from the SuiteSparse collection [KAB19]. The last column ("PDHG iterations") gives the number of iterations in the primal–dual algorithm. Columns 3–5 describe the complexity of one iteration of the algorithm. The CPU time is measured in seconds.

4.6.2 Graph partitioning

The problem of partitioning the vertices of a graph $G = (V, E)$ in two subsets of equal size (here we assume an even number of vertices), while minimizing the number of edges between the two subsets, can be expressed as

$$\begin{array}{ll} \text{minimize} & (1/4)x^T L x \\ \text{subject to} & \mathbf{1}^T x = 0 \\ & x \in \{-1, 1\}^n, \end{array}$$

where L is the graph Laplacian. The ith entry of the n-vector x indicates the set that vertex i is assigned to. To obtain an SDP relaxation we introduce a matrix variable $Y = xx^T$ and write the problem in the equivalent form

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(LY) \\ \text{subject to} & \mathbf{1}^T Y \mathbf{1} = 0 \\ & \mathbf{diag}(Y) = \mathbf{1} \\ & Y = xx^T, \end{array}$$

and then relax the constraint $Y = xx^T$ as $Y \succeq 0$. This gives the SDP

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(LY) \\ \text{subject to} & \mathbf{1}^T Y \mathbf{1} = 0 \\ & \mathbf{diag}(Y) = \mathbf{1} \\ & Y \succeq 0. \end{array} \tag{4.24}$$

The dual SDP is

$$\begin{array}{ll} \text{maximize} & \mathbf{1}^T z \\ \text{subject to} & \mathbf{diag}(z) + \xi \mathbf{1}\mathbf{1}^T \preceq (1/4)L, \end{array}$$

with variables $\xi \in \mathbf{R}$ and $z \in \mathbf{R}^n$.

The aggregate sparsity pattern of the SDP (4.24) is completely dense, because the equality constraint $\mathbf{1}^T Y \mathbf{1} = 0$ has a coefficient matrix of all ones. We therefore eliminate the dense

constraint using the technique described in [FKM01, page 668]. Let P be the $n \times (n-1)$ matrix

$$P = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -1 & 1 & \cdots & 0 & 0 \\ 0 & -1 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & -1 & 1 \\ 0 & 0 & \cdots & 0 & -1 \end{bmatrix}.$$

The columns of P form a sparse basis for the orthogonal complement of the multiples of the vector $\mathbf{1}$. Suppose Y is feasible in (4.24) and define

$$\begin{bmatrix} X & u \\ u^T & v \end{bmatrix} = \begin{bmatrix} P & \mathbf{1} \end{bmatrix}^{-1} Y \begin{bmatrix} P & \mathbf{1} \end{bmatrix}^{-T}. \tag{4.25}$$

From $\mathbf{1}^T Y \mathbf{1} = 0$, we see that

$$0 = \mathbf{1}^T Y \mathbf{1} = \mathbf{1}^T \begin{bmatrix} P & \mathbf{1} \end{bmatrix} \begin{bmatrix} X & u \\ u^T & v \end{bmatrix} \begin{bmatrix} P & \mathbf{1} \end{bmatrix}^T \mathbf{1} = n^2 v,$$

and therefore $v = 0$. Since the matrix (4.25) is positive semidefinite, we also have $u = 0$. Hence every feasible Y can be expressed as $Y = PXP^T$, with $X \succeq 0$. If we make this substitution in (4.24) we obtain

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(P^T L P X) \\ \text{subject to} & \mathbf{diag}(PXP^T) = \mathbf{1} \\ & X \succeq 0. \end{array}$$

The $(n-1) \times (n-1)$ matrix $P^T L P$ has elements

$$(P^T L P)_{ij} = \begin{cases} L_{ii} - 2L_{i,i+1} + L_{i+1,i+1} & i = j \\ L_{ij} - L_{i+1,j} - L_{i,j+1} + L_{i+1,j+1} & i \neq j. \end{cases}$$

	n	$p^\star_{\rm sdp}$	$p^\star_{\rm sdp} - \frac{1}{4}\,\mathbf{tr}(P^TLPX)$	primal residual	dual residual
gpp100	100	-44.943551	3.78×10^{-4}	3.24×10^{-7}	8.34×10^{-7}
gpp124-1	124	-7.3430761	4.02×10^{-4}	3.86×10^{-8}	7.45×10^{-8}
gpp250-1	250	-45.444917	8.23×10^{-4}	1.28×10^{-7}	8.39×10^{-7}
gpp500-1	500	-25.320544	5.17×10^{-4}	7.42×10^{-8}	7.12×10^{-7}

Table 4.3: Results for four graph partitioning problems from SDPLIB. Column 3 is the optimal value computed by MOSEK. Column 4 is the difference with the optimal value of the centering problem computed by algorithm (4.20). The last two columns give the primal and dual residuals in the computed solution.

Thus the sparsity pattern E' of the matrix P^TLP is denser than E, *i.e.*, $E \subseteq E'$. The n constraints $\mathbf{diag}(PXP^T) = \mathbf{1}$ reduce to

$$X_{11} = 1, \qquad X_{i-1,i-1} + X_{ii} - 2X_{i,i-1} = 1, \quad i = 2, \ldots, n-1, \qquad X_{n-1,n-1} = 1.$$

To apply algorithm (4.20), we first rewrite the graph partitioning problem as

$$\begin{array}{ll} \text{minimize} & (1/4)\,\mathbf{tr}(P^TLPX) \\ \text{subject to} & \mathbf{diag}(PXP^T) = \mathbf{1} \\ & X \in \Pi_{E''}(\mathbf{S}_+^{n-1}) \end{array} \tag{4.26}$$

where E'' is a chordal extension of the aggregate sparsity pattern E'. Note that $\mathbf{tr}(P^TPX) = n-1$ for all feasible X. The centering problem for this sparse SDP is of the form (4.10) with

$$C = \frac{1}{4}P^TLP, \qquad N = \frac{1}{n-1}P^TP, \qquad \mathcal{A}(X) = \mathbf{diag}(PXP^T), \qquad \mathcal{A}^*(y) = P^T\,\mathbf{diag}(y)P.$$

Numerical results Table 4.3 shows the numerical results for four problems from SDPLIB[Bor99]. The SDP relaxation (4.24) is solved by MOSEK and its optimal value is denoted by $p^\star_{\rm sdp}$. In solving (4.26), we set $\mu = 0.001/n$, and report in Table 4.3 the value $(1/4)\,\mathbf{tr}(P^TLPX)$,

	n	time per Cholesky factorization	Newton steps per iteration	time per PDHG iteration	PDHG iterations
gpp100	100	0.01	2.43	0.02	305
gpp124-1	124	0.01	2.00	0.02	392
gpp250-1	250	0.01	2.65	0.03	365
gpp500-1	500	0.02	3.01	0.07	394
delaunay_n10	1024	0.37	4.36	1.76	403
delaunay_n11	2048	0.48	4.70	2.54	420
delaunay_n12	4096	0.60	4.43	3.05	367
delaunay_n13	8192	1.02	4.42	4.98	375

Table 4.4: The four graph partitioning problems from SDPLIB plus four larger graphs from the SuiteSparse collection. The last column gives the number of iterations in the primal–dual algorithm. Columns 3–5 describe the complexity of one iteration of the algorithm. The CPU time is measured in seconds.

where X is the solution returned by the algorithm (4.20). As in the first experiment, the numerical results show that the algorithm is able to solve the centering SDP (4.26) with desired accuracy.

In addition, we test the algorithm for four additional graphs from the SuiteSparse collection [KAB19]. Table 4.4 reports the time per Cholesky factorization, the number of Newton steps per iteration, the time per PDHG iteration, and the number of iterations in the primal–dual algorithm. The same observations as in Section 4.6.1 apply: the number of Newton steps remains moderate as the size of the problem increases, and the cost per iteration is roughly linear in the cost of a Cholesky factorization.

CHAPTER 5

Conclusions

In the first part of the dissertation we presented two variants of Bregman Condat–Vũ algorithms, in which generalized Bregman proximal operators are used in both primal and dual updates. The proposed algorithms extend the Condat–Vũ three-operator splitting algorithm [Con13, Vu13] for convex optimization, and include most well-known proximal splitting methods as special cases. We gave a new derivation for both methods, by applying the Bregman proximal point method to the primal–dual optimality conditions. Based on the interpretation, we provided a unified framework for the convergence analysis. Moreover, we introduced a line search technique for the Bregman dual Condat–Vũ algorithm for equality-constrained problems, and proposed a Bregman extension to PD3O [Yan18].

In the second part of the dissertation, we applied the proposed Bregman proximal splitting algorithms to the centering problem in large-scale semidefinite programming with sparse coefficient matrices. The Bregman distance used in the proximal operator is generated by the logarithmic barrier function for the cone of sparse matrices with a positive semidefinite completion. With this choice of Bregman distance, the per-iteration complexity of the Bregman proximal algorithm is dominated by the cost of a Cholesky factorization with the aggregate (or common) sparsity pattern of the semidefinite program, plus the cost of evaluating the linear mapping in the constraints and its adjoint. Therefore, when applied to the centering problem of SDP, Bregman proximal methods obviate the expensive eigenvalue decomposition needed for standard proximal methods, and are able to handle sparse matrix constraints with sizes that are orders of magnitude larger than the problems solved by general-purpose

solvers.

Many open research directions remain. First, there are still some theoretical questions on how to incorporate Bregman distances into primal–dual proximal methods. For example, it is unclear how to use Bregman distances in PDDY [SCM20]. The derivation of PD3O and other Bregman proximal methods from the Bregman proximal point algorithm is unknown. The Bregman version of the accelerated proximal methods [AT06, CP16b, Teb18] is largely unexplored. Developing Bregman proximal methods for nonconvex optimization problems is also interesting. To further improve the efficiency of practical implementation, it would be useful to conduct a plane search for the two parameters τ and σ,

In addition to the above theoretical questions, we believe that the results in the dissertation will lead to new efficient algorithms for convex optimization applications in control, signal processing, machine learning, and data science, in which domain knowledge is used to formulate specialized Bregman proximal operators that reduce the cost per iteration. Further developing the sparse SDP methods from the results in Chapter 4 is also important.

[ADD96] P. Amestoy, T. Davis, and I. Duff. "An approximate minimum degree ordering algorithm." *SIAM Journal on Matrix Analysis and Applications*, **17**(4):886–905, 1996.

[ADH21] D. Applegate, M. Dóaz, O. Hinder, H. Lu, M. Lubin, B. O'Donoghue, and W. Schudy. "Practical large-scale linear programming using primal–dual hybrid gradient." *arXiv eprints*, **arXiv:2106.04756**, 2021.

[ADV10] M. S. Andersen, J. Dahl, and L. Vandenberghe. "Implementation of nonsymmetric interior-point methods for linear optimization over sparse matrix cones." *Mathematical Programming Computation*, **2**:167–201, 2010.

[ADV13] M. S. Andersen, J. Dahl, and L. Vandenberghe. "Logarithmic barriers for sparse matrix cones." *Optimization Methods and Software*, **28**(3):396–423, 2013.

[ADV20] M. S. Andersen, J. Dahl, and L. Vandenberghe. *CVXOPT: A Python Package for Convex Optimization*, 2020. Available at `www.cvxopt.org`.

[AG03] F. Alizadeh and D. Goldfarb. "Second-order cone programming." *Mathematical Programming*, **95**(1):3–51, 2003.

[Ali95] F. Alizadeh. "Interior point methods in semidefinite programming with applications to combinatorial optimization." *SIAM Journal on Optimization*, **5**(1):13–51, 1995.

[AT06] A. Auslender and M. Teboulle. "Interior gradient and proximal methods for convex and conic optimization." *SIAM Journal on Optimization*, **16**(3):697–725, 2006.

[AV15] M. S. Andersen and L. Vandenberghe. *CHOMPACK: A Python Package for Chordal Matrix Computations*, 2015. `cvxopt.github.io/chompack`.

[BBC03] H. H. Bauschke, J. M. Borwein, and P. L. Combettes. "Bregman monotone optimization algorithms." *SIAM Journal on Control and Optimization*, **42**(2):596–636, 2003.

[BBT17] H. H. Bauschke, J. Bolte, and M. Teboulle. "A descent lemma beyond Lipschitz gradient continuity: first-order methods revisited and applications." *Mathematics of Operations Research*, **42**(2):330–348, 2017.

[BC17] H. H. Bauschke and P. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer Publishing Company, Incorporated, 2nd edition, 2017.

[BCN18] L. Bottou, F. Curtis, and J. Nocedal. "Optimization methods for large-scale machine learning." *SIAM Review*, **60**(2):223–311, 2018.

[BDX04] S. Boyd, P. Diaconis, and L. Xiao. "Fastest mixing Markov chain on a graph." *SIAM Review*, **46**(4):667–689, 2004.

[Bec17] A. Beck. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, 2017.

[BGM13] S. Bellavia, J. Gondzio, and B. Morini. "A matrix-free preconditioner for sparse symmetric positive definite systems and least-squares problems." *SIAM Journal on Scientific Computing*, **35**(1):A192–A211, 2013.

[BGP19] S. Bellavia, J. Gondzio, and M. Porcelli. "An inexact dual logarithmic barrier method for solving sparse semidefinite programs." *Mathematical Programming*, **178**(1-2):109–143, 2019.

[BGP21] S. Bellavia, J. Gondzio, and M. Porcelli. "A relaxed interior point method for low-rank semidefinite programming problems with applications to matrix completion." *Journal of Scientific Computing*, **89**(2):1–36, 2021.

[BGW05] A. Banerjee, X. Guo, and H. Wang. "On the optimality of conditional expectation as a Bregman predictor." *IEEE Transactions on Information Theory*, **51**(7):2664–2669, 2005.

[BL00] H. H. Bauschke and A. Lewis. "Dykstra's algorithm with Bregman projections: A convergence proof." *Optimization*, **48**(4):409–427, 2000.

[BLT06] P. Biswas, T.C. Liang, K. C. Toh, Y. Ye, and T. C. Wang. "Semidefinite programming approaches for sensor network localization with noisy distance measurements." *IEEE Transactions on Automation Science and Engineering*, **3**(4):360–371, 2006.

[BLW06] P. Biswas, T.-C. Lian, T.-C. Wang, and Y. Ye. "Semidefinite programming based algorithms for sensor network localization." *ACM Transactions on Sensor Networks*, **2**(2):188–220, 2006.

[BMN01] A. Ben-Tal, T. Margalit, and A. Nemirovski. "The ordered subsets mirror descent optimization method with applications to tomography." *SIAM Journal on Optimization*, **12**(1):79–108, 2001.

[BN01] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization. Analysis, Algorithms, and Engineering Applications*. SIAM, 2001.

[Bor99] B. Borchers. "SDPLIB 1.2, a library of semidefinite programming test problems." *Optimization Methods and Software*, **11**(1–4):683–690, 1999.

[BP93] J. R. S. Blair and B. Peyton. "An introduction to chordal graphs and clique trees." In A. George, J. R. Gilbert, and J. W. H. Liu, editors, *Graph Theory and Sparse Matrix Computation*. Springer-Verlag, 1993.

[BPC11] S. P. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. "Distributed optimization and statistical learning via the alternating direction method of multipliers." *Foundations and Trends in Machine Learning*, **3**(1):1–122, 2011.

[Bre67] L. M. Bregman. "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming." *USSR Computational Mathematics and Mathematical Physics*, **7**(3):200–217, 1967.

[Bre73] H. Brézis. *Opérateurs maximaux monotones et semi-groupes de contractions dans les espaces de Hilbert*, volume 5 of *North-Holland Mathematical Studies*. North-Holland, 1973.

[BST18] J. Bolte, S. Sabach, S. Teboulle, and Y. Vaisbourd. "First order methods beyond convexity and Lipschitz gradient continuity with applications to quadratic inverse problems." *SIAM Journal on Optimization*, **28**(3):2131–2151, 2018.

[BT03] A. Beck and M. Teboulle. "Mirror descent and nonlinear projected subgradient methods for convex optimization." *Operations Research Letters*, **31**(3):167–175, 2003.

[BT09a] A. Beck and M. Teboulle. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems." *SIAM Journal on Imaging Sciences*, **2**(1):183–202, 2009.

[BT09b] A. Beck and M. Teboulle. "Gradient-based algorithms with applications to signal recovery problems." In Y. Eldar and D. Palomar, editors, *Convex Optimization in Signal Processing and Communications*, pp. 42–88. Cambridge University Press, 2009.

[Bub15] S. Bubeck. "Convex optimization: algorithms and complexity." *Foundations and Trends in Machine Learning*, **8**(3–4):231–357, 2015.

[Bur03] S. Burer. "Semidefinite programming in the space of partial positive semidefinite matrices." *SIAM Journal on Optimization*, **14**(1):139–172, 2003.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[BY04] P. Biswas and Y. Ye. "Semidefinite programming for Ad-Hoc wireless sensor network localization." In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, pp. 46–54, 2004.

[BY08] S. J. Benson and Y. Ye. "Algorithm 875: DSDP5—software for semidefinite programming." *ACM Transactions on Mathematical Software (TOMS)*, **34**(3):16, 2008.

[BYZ00] S. J. Benson, Y. Ye, and X. Zhang. "Solving large-scale sparse semidefinite programs for combinatorial optimization." *SIAM Journal on Optimization*, **10**(2):443–461, 2000.

[CC22] A. Chambolle and J. P. Contreras. "Accelerated Bregman primal–dual methods applied to optimal transport and Wasserstein barycenter problems." *arXiv e-prints*, **arXiv:2203.00802**, 2022.

[CHZ13] P. Chen, J. Huang, and X. Zhang. "A primal–dual fixed point algorithm for convex separable minimization with applications to image restoration." *Inverse Problems*, **29**(2), 2013.

[CKC22] L. Condat, D. Kitahara, A. Contreras, and A. Hirabayashi. "Proximal splitting algorithms for convex optimization: a tour of recent advances, with new twists." *SIAM Review*, 2022. To appear. Preprint available at https://arxiv.org/abs/1912.00137.

[CL81] Y. Censor and A. Lent. "An iterative row-action method for interval convex programming." *Journal of Optimization Theory and Applications*, **34**(3):321–353, 1981.

[CLM21] C. Clason, D. A. Lorenz, H. Mahler, and B. Wirth. "Entropic regularization of continuous optimal transport problems." *Journal of Mathematical Analysis and Applications*, **494**(1):124432, 2021.

[Con13] L. 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, August 2013.

[CP07] P. L. Combettes and J.-C. Pesquet. "A Douglas–Rachford splitting approach to nonsmooth convex variational signal recovery." *IEEE Journal of Selected Topics in Signal Processing*, **1**(4):564–574, 2007.

[CP11a] A. Chambolle and T. 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.

[CP11b] P. L. Combettes and J.-C. Pesquet. "Proximal splitting methods in signal processing." In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, Springer Optimization and Its Applications, pp. 185–212. Springer New York, 2011.

[CP16a] A. Chambolle and T. Pock. "An introduction to continuous optimization for imaging." *Acta Numerica*, **25**:161–319, 2016.

[CP16b] A. Chambolle and T. Pock. "On the ergodic convergence rates of a first-order primal–dual algorithm." *Mathematical Programming*, **159**(1):253–287, 2016.

[CT93] G. Chen and M. Teboulle. "Convergence analysis of a proximal-like minimization algorithm using Bregman functions." *SIAM Journal on Optimization*, **3**(3):538–543, 1993.

[CV18] H. Chao and L. Vandenberghe. "Entropic proximal operators for nonnegative trigonometric polynomials." *IEEE Transactions on Signal Processing*, **66**(18):4826–4838, 2018.

[CZ92] Y. Censor and S. A. Zenios. "Proximal minimization algorithm with D-functions." *Journal of Optimization Theory and Applications*, **73**(3):451–464, 1992.

[CZ97] Y. Censor and S. Zenios. *Parallel Optimization: Theory, Algorithms and Applications*. Oxford University Press, 1997.

[DB16] S. Diamond and S. Boyd. "CVXPY: A Python-embedded modeling language for convex optimization." *Journal of Machine Learning Research*, **17**(83):1–5, 2016.

[Dem72] A. P. Dempster. "Covariance selection." *Biometrics*, **28**(1):157–175, 1972.

[DKQ10] Y. Ding, N. Krislock, J. Qian, and H. Wolkowicz. "Sensor network localization, Euclidean distance matrix completion, and graph realization." *Optimization and Engineering*, **11**(1):45–66, 2010.

[DST15] Y. Drori, S. Sabach, and M. Teboulle. "A simple algorithm for a class of nonsmooth convex-concave saddle-point problems." *Operations Research Letters*, **43**(2):209–214, 2015.

[dST21] A. d'Aspremont, D. Scieur, and A. Taylor. "Acceleration methods." *Foundations and Trends in Optimization*, **5**(1–2):1–245, 2021.

[DT08] I. S. Dhillon and J. A. Tropp. "Matrix nearness problems with Bregman divergences." *SIAM Journal on Matrix Analysis and Applications*, **29**(4):1120–1146, 2008.

[DY17] D. Davis and W. Yin. "A three-operator splitting scheme and its optimization applications." *Set-Valued and Variational Analysis*, **25**:829–858, 2017.

[EB92] J. Eckstein and D. Bertsekas. "On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators." *Mathematical Programming*, **55**(1):293–318, 1992.

[Eck93] J. Eckstein. "Nonlinear proximal point algorithms using Bregman functions, with applications to convex programming." *Mathematics of Operations Research*, **18**(1):202–226, 1993.

[EDA20] A. Eltved, J. Dahl, and M. S. Andersen. "On the robustness and scalability of semidefinite relaxation for optimal power flow problems." *Optimization and Engineering*, **21**(2):375–392, 2020.

[EZC10] E. Esser, X. Zhang, and T. 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.

[FKM01] M. Fukuda, M. Kojima, K. Murota, and K. Nakata. "Exploiting sparsity in semidefinite programming via matrix completion I: general framework." *SIAM Journal on Optimization*, **11**(3):647–674, 2001.

[FKN97] K. Fujisawa, M. Kojima, and K. Nakata. "Exploiting sparsity in primal–dual interior-point methods for semidefinite programming." *Mathematical Programming*, **79**(1–3):235–253, 1997.

[FNB20] A. Fu, B. Narasimhan, and S. Boyd. "CVXR: An R package for disciplined convex optimization." *Journal of Statistical Software*, **94**(14):1–34, 2020.

[Gab83] D. Gabay. "Applications of the method of multipliers to variational inequalities." In M. Fortin and R. Glowinski, editors, *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, volume 15 of *Studies in Mathematics and Its Applications*, pp. 299–331. Elsevier, 1983.

[GB14] M. Grant and S. Boyd. "CVX: Matlab Software for Disciplined Convex Programming, version 2.1." http://cvxr.com/cvx, 2014.

[GJS84] R. Grone, C. R. Johnson, E. M. Sá, and H. Wolkowicz. "Positive definite completions of partial Hermitian matrices." *Linear Algebra and its Applications*, **58**:109–124, 1984.

[GM75] R. Glowinski and A. Marrocco. "Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires." *Revue française d'automatique, informatique, recherche opérationnelle*, **9**(2):41–76, 1975.

[GM76] D. Gabay and B. Mercier. "A dual algorithm for the solution of nonlinear variational problems via finite element approximation." *Computers and Mathematics with Applications*, **2**(1):17–40, 1976.

[Gon12] J. Gondzio. "Interior point methods 25 years later." *European Journal of Operational Research*, **218**(3):587–601, 2012.

[GOY17] R. Glowinski, S. J. Osher, and W. Yin. *Splitting Methods in Communication, Imaging, Science, and Engineering*. Scientific Computation. Springer, 2017.

[GR00] M. Goemans and F. Rendl. "Combinatorial optimization." In H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors, *Handbook of Semidefinite Programming*, chapter 12, pp. 343–360. Kluwer Academic Publishers, 2000.

[GR18] P. Giselsson and A. Rantzer. *Large-Scale and Distributed Optimization*. Springer, 2018.

[Gul94] O. Güler. "Ergodic convergence in proximal point algorithms with Bregman functions." In D.-Z. Du and J. Sun, editors, *Advances in Optimization and Approximation*, pp. 155–165. Springer, 1994.

[Gur22] Gurobi Optimization, LLC. "Gurobi Optimizer Reference Manual.", 2022.

[GW95] M. X. Goemans and D. P. Williamson. "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming." *Journal of the ACM*, **42**(6):1115–1145, 1995.

[Ha90] C. Ha. "A generalization of the proximal point algorithm." *SIAM Journal on Control and Optimization*, **28**(3):503–512, 1990.

[HRX21] F. Hanzely, P. Richtarik, and L. Xiao. "Accelerated Bregman proximal gradient methods for relatively smooth convex optimization." *Computational Optimization and Applications*, **79**(2):405–440, 2021.

[HY12] B. He and X. 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.

[Jab12] R. A. Jabr. "Exploiting sparsity in SDP relaxations of the OPF problem." *IEEE Transactions on Power Systems*, **27**(2):1138–1139, 2012.

[JLL19] M. Jacobs, F. Leger, W. Li, and S. Osher. "Solving large-scale optimization problems with a convergence rate independent of grid size." *SIAM Journal on Numerical Analysis*, **57**(3):1100–1123, 2019.

[JN12a] A. Juditsky and A. Nemirovski. "First-order methods for nonsmooth convex large-scale optimization, I: general purpose methods." In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*, pp. 149–183. MIT Press, 2012.

[JN12b] A. Juditsky and A. Nemirovski. "First-order methods for nonsmooth convex large-scale optimization, II: utilizing problem's structure." In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*, pp. 149–183. MIT Press, 2012.

[JV22a] X. Jiang and L. Vandenberghe. "Bregman primal–dual first-order method and applications to sparse semidefinite programming." *Computational Optimization and Applications*, **81**(1):127–159, 2022.

[JV22b] X. Jiang and L. Vandenberghe. "Bregman three-operator splitting methods." *arXiv e-prints*, **arXiv:2203.00252**, 2022.

[KAB19] S. Kolodziej, M. Aznaveh, M. Bullock, J. David, T. A. Davis, M. Henderson, Y. Hu, and R. Sandstrom. "The SuiteSparse matrix collection website interface." *Journal of Open Source Software*, **35**(4), 2019.

[Kiw97] K. C. Kiwiel. "Proximal minimization methods with generalized Bregman functions." *SIAM Journal on Control and Optimization*, **35**(4):1142–1168, 1997.

[KKK08] K. Kobayashi, S. Kim, and M. Kojima. "Correlative sparsity in primal–dual interior-point methods for LP, SDP, and SOCP." *Applied Mathematics and Optimization*, **58**(1):69–88, 2008.

[KKM11] S. Kim, M. Kojima, M. Mevissen, and M. Yamashita. "Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion." *Mathematical Programming*, **129**(1):33–68, 2011.

[KKW09] S. Kim, M. Kojima, and H. Waki. "Exploiting sparsity in SDP relaxations for sensor network localization." *SIAM Journal on Optimization*, **20**(1):192–215, 2009.

[KP15] N. Komodakis and J. Pesquet. "Playing with duality: an overview of recent primal–dual approaches for solving large-scale optimization problems." *IEEE Signal Processing Magazine*, **32**(6):31–54, 2015.

[LFN18] H. Lu, R. M. Freund, and Y. Nesterov. "Relatively smooth convex optimization by first-order methods, and applications." *SIAM Journal on Optimization*, **28**(1):333–354, 2018.

[LM79] P. Lions and B. Mercier. "Splitting algorithms for the sum of two nonlinear operators." *SIAM Journal on Numerical Analysis*, **16**(6):964–979, 1979.

[LMY21] T. Lin, S. Ma, Y. Ye, and S. Zhang. "An ADMM-based interior-point method for large-scale linear programming." *Optimization Methods and Software*, **36**(2–3):389–424, 2021.

[Lov79] L Lovász. "On the Shannon capacity of a graph." *IEEE Transactions on Information Theory*, **25**:1–7, 1979.

[Low14a] S. H. Low. "Convex relaxation of optimal power flow part I: formulations and equivalence." *IEEE Transactions on Control of Network Systems*, **1**(1):15–27, 2014.

[Low14b] S. H. Low. "Convex relaxation of optimal power flow part II: exactness." *IEEE Transactions on Control of Network Systems*, **1**(2):177–189, 2014.

[LT92] Z. Q. Luo and P. Tseng. "On the convergence of the coordinate descent method for convex differentiable minimization." *Journal of Optimization Theory and Applications*, **72**(1):7–35, 1992.

[LV11] I. Loris and C. Verhoeven. "On a generalization of the iterative soft-thresholding algorithm for the case of non-separable penalty." *Inverse Problems*, **27**(12), 2011.

[LXY21] Y. Liu, Y. Xu, and W. Yin. "Acceleration of primal–dual methods by preconditioning and simple subproblem procedures." *Journal of Scientific Computing*, **86**(2):21, 2021.

[MHL13] D. K. Mohlzahn, J. T. Holzer, B. C. Lesieutre, and C. L. DeMarco. "Implementation of a large-scale optimal power flow solver based on semidefinite programming." *IEEE Transactions on Power Systems*, **28**(4):3987–3998, 2013.

[MKL15] R. Madani, A. Kalbat, and J. Lavaei. "ADMM for sparse semidefinite programming with applications to optimal power flow problem." In *Proceedings of the 54th IEEE Converence on Decision and Control*, pp. 5932–5939, 2015.

[Mor65] J. Moreau. "Proximité et dualité dans un espace Hilbertien." *Bulletin de la Société Mathématique de France*, **93**:273–299, 1965.

[MOS19] MOSEK ApS. *The MOSEK Optimization Tools Manual. Version 8.1.*, 2019. Available at www.mosek.com.

[MP18] Y. Malitsky and T. Pock. "A first-order primal–dual algorithm with linesearch." *SIAM Journal on Optimization*, **28**(1):411–432, 2018.

[Nem04] A. 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.

[Nes83] Y. Nesterov. "A method of solving a convex programming problem with convergence rate $O(1/k^2)$." *Soviet Mathematics Doklady*, **27**(2):372–376, 1983.

[Nes88] Y. Nesterov. "On an approach to the construction of optimal methods of minimization of smooth convex functions." *Ekonomika i Mateaticheskie Metody*, **24**(3):509–517, 1988.

[Nes18] Y. Nesterov. *Lectures on Convex Optimization.* Springer Publishing Company, Incorporated, 2018.

[NFF03] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota. "Exploiting sparsity in semidefinite programming via matrix completion II: implementation and numerical results." *Mathematical Programming*, **95**(2):303–327, 2003.

[NN94] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.

[NT98] Y. Nesterov and M. J. Todd. "Primal–dual interior-point methods for self-scaled cones." *SIAM Journal on Optimization*, **8**(2):324–364, May 1998.

[NW06] J. Nocedal and S. Wright. *Numerical Optimization*. Springer-Verlag, 2nd edition, 2006.

[NY83] A. Nemirovsky and D. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley–Interscience, 1983.

[OV20] D. O'Connor and L. Vandenberghe. "On the equivalence of the primal–dual hybrid gradient method and Douglas–Rachford splitting." *Mathematical Programming*, **179**(1–2):85–108, 2020.

[PB14] N. Parikh and S. Boyd. "Proximal algorithms." *Foundations and Trends in Optimization*, **1**(3):127–239, 2014.

[PC11] T. Pock and A. Chambolle. "Diagonal preconditioning for first order primal–dual algorithms in convex optimization." In *International Conference on Computer Vision*, pp. 1762–1769, 2011.

[PCB09] T. Pock, D. Cremers, H. Bischof, and A. Chambolle. "An algorithm for minimizing the Mumford–Shah functional." In *International Conference on Computer Vision*, pp. 1133–1140, 2009.

[PG21] S. Pougkakiotis and J. Gondzio. "An interior point-proximal method of multipliers for convex quadratic programming." *Computational Optimization and Applications*, **78**:307–351, 2021.

[PG22] S. Pougkakiotis and J. Gondzio. "An interior point-proximal method of multipliers for linear positive semi-definite programming." *Journal of Optimization Theory and Applications*, **192**:97–129, 2022.

[PHA18] S. K. Pakazad, A. Hansson, M. S. Andersen, and A. Rantzer. "Distributed semidefinite programming with application to large-scale system analysis." *IEEE Transactions on Automatic Control*, **63**(4):1045–1058, April 2018.

[PRT02] J. Peng, C. Roos, and T. Terlaky. *Self-Regularity. A New Paradigm for Primal–Dual Interior-Point Algorithms*. Princeton University Press, 2002.

[Rd20] M. Romain and A. d'Aspremont. "A Bregman method for structure learning on sparse directed acyclic graphs." *arXiv e-prints*, **arXiv:2011.02764**, 2020.

[Ren01] J. Renegar. *A Mathematical View of Interior-Point Methods in Convex Optimization*. Society for Industrial and Applied Mathematics, 2001.

[Roc70] R. T. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, 1970.

[RY22] E. K. Ryu and W. Yin. *Large-Scale Convex Optimization via Monotone Operators*. Cambridge University Press, 2022. To be published.

[SAV14] Y. Sun, M. S. Andersen, and L. Vandenberghe. "Decomposition in conic optimization with partially separable structure." *SIAM Journal on Optimization*, **24**:873–897, 2014.

[SCM20] A. Salim, L. Condat, K. Mishchenko, and P. Richtárik. "Dualize, split, randomize: fast nonsmooth optimization algorithms." *arXiv preprint*, **arXiv:2004.0263**, 2020.

[SNW12] S. Sra, S. Nowozin, and S. J. Wright. *Optimization for Machine Learning*. The MIT Press, 2012.

[Stu99] J. F. Sturm. "Using SeDuMi 1.02, A MATLAB toolbox for optimization over symmetric cones." *Optimization Methods and Software*, **11**(1–4):625–653, 1999.

[SV04] G. Srijuntongsiri and S. A. Vavasis. "A fully sparse implementation of a primal–dual interior-point potential reduction method for semidefinite programming." *arXiv preprint*, **arXiv:cs/0412009**, 2004.

[SV15] Y. Sun and L. Vandenberghe. "Decomposition methods for sparse matrix nearness problems." *SIAM Journal on Matrix Analysis and Applications*, **36**(4):1691–1717, 2015.

[Tay15] J. A. Taylor. *Convex Optimization of Power Systems*. Cambridge University Press, 2015.

[Teb97] M. Teboulle. "Convergence of proximal-like algorithms." *SIAM Journal on Optimization*, **7**(4):1069–1083, 1997.

[Teb18] M. Teboulle. "A simplified view of first order methods for optimization." *Mathematical Programming*, **170**(1):67–96, 2018.

[TLJ06] B. Taskar, S. Lacoste-Julien, and M. I. Jordan. "Structured prediction, dual extragradient and Bregman projections." *Journal of Machine Learning Research*, **7**(60):1627–1653, 2006.

[Tod01] M. J. Todd. "Semidefinite optimization." *Acta Numerica*, **10**:515–560, 2001.

[Tse00] P. Tseng. "A modified forward-backward splitting method for maximal monotone mappings." *SIAM Journal on Control and Optimization*, **38**(2):431–446, 2000.

[Tse08] P. Tseng. "On accelerated proximal gradient methods for convex-concave optimization." Unpublished preprint available at https://www.mit.edu/~dimitrib/PTseng/papers/apgm.pdf, 2008.

[TTT02] K. C. Toh, R. H. Tütüncü, and M. J. Todd. *SDPT3 version 3.02. A MATLAB software for semidefinite-quadratic-linear programming*, 2002. Available at www.math.nus.edu.sg/~mattohkc/sdpt3.html.

[VA15] L. Vandenberghe and M. S. Andersen. "Chordal graphs and semidefinite optimization." *Foundations and Trends in Optimization*, **1**(4):241–433, 2015.

[VB95] L. Vandenberghe and S. Boyd. "A primal–dual potential reduction method for problems involving matrix inequalities." *Mathematical Programming*, **69**(1):205–236, 1995.

[VMC21] M.-L. Vladarean, Y. Malitsky, and V. Cevher. "A first-order primal–dual method with adaptivity to local smoothness." In *Advances in Neural Information Processing Systems*, 2021.

[Vu13] B. C. Vũ. "A splitting algorithm for dual monotone inclusions involving cocoercive operators." *Advances in Computational Mathematics*, **38**(3):667–681, 2013.

[Wri97] S. J. Wright. *Primal–dual interior-point methods*. SIAM, Philadelphia, 1997.

[Wri15] S. J. Wright. "Coordinate descent algorithms." *Mathematical Programming*, **151**(1):3–34, 2015.

[WS04] K. Q. Weinberger and L. K. Saul. "Unsupervised learning of image manifolds by semidefinite programming." In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, 2004.

[WS06] K. Q. Weinberger and L. K. Saul. "An introduction to nonlinear dimensionality reduction by maximum variance unfolding." In *National Conference on Artificial Intelligence*, pp. 1683–1686, 2006.

[WSZ07] K. Q. Weinberger, F. Sha, Q. Zhu, and L. Saul. "Graph Laplacian regularization for large-scale semidefinite programming." In *Advances in Neural Information Processing Systems*. MIT Press, 2007.

[WX17] J. Wang and L. Xiao. "Exploiting strong convexity from data with primal–dual first-order algorithms." In *International Conference on Machine Learning*, pp. 3694–3702, 2017.

[XB04] L. Xiao and S. Boyd. "Fast linear iterations for distributed averaging." *Systems & Control Letters*, **53**(1):65–78, 2004.

[YA21] E. Yazdandoost Hamedani and N. S. Aybat. "A primal–dual algorithm with line search for general convex-concave saddle point problems." *SIAM Journal on Optimization*, **31**(2):1299–1329, 2021.

[Yan18] M. Yan. "A new primal–dual algorithm for minimizing the sum of three functions with a linear operator." *Journal of Scientific Computing*, **76**(3):1698–1717, September 2018.

[ZFP17] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn. "Fast ADMM for semidefinite programs with chordal sparsity." In *American Control Conference*, pp. 3335–3340, 2017.

[ZFP20] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn. "Chordal decomposition in operator-splitting methods for sparse semidefinite programs." *Mathematical Programming*, **180**:489–532, 2020.

[ZFP21] Y. Zheng, G. Fantuzzi, and A. Papachristodoulou. "Chordal and factor-width decompositions for scalable semidefinite and polynomial optimization." *Annual Reviews in Control*, **52**, 2021.

[ZL21] R. Y. Zhang and J. Lavaei. "Sparse semidefinite programs with guaranteed near-linear time complexity via dualized clique tree conversion." *Mathematical Programming*, **188**(1):351–393, 2021.