University of California
Los Angeles

Decomposition methods for semidefinite optimization

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical Engineering

by

Yifan Sun

2015

Abstract of the Dissertation

Decomposition methods for semidefinite optimization

by

Yifan Sun

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 2015

Professor Lieven Vandenberghe, Chair

Semidefinite optimization problems (SDPs) arise in many applications, including combinatorial optimization, control and signal processing, structural optimization, statistics, and machine learning. Currently, most SDPs are solved using interior-point methods. These methods are typically robust and accurate, and converge in few iterations. However, their per-iteration cost may be high. At each iteration, an interior-point method solves a large and generally dense system of linear equations, and this limits the scalability of these methods. In contrast, first-order methods may require many iterations to converge and often reach a much lower accuracy, but have a very low per-iteration complexity and memory requirement. This allows them to scale to much larger problems. However, both interior-point methods and first-order methods have difficulty exploiting sparsity in SDPs, because the matrix inequality constraint introduces a nonlinear coupling between all the elements of the matrix variable.

In this thesis, we present decomposition methods for sparse semidefinite optimization. The techniques exploit partial separability properties of cones of chordal sparse matrices with a positive semidefinite completion. They can be used for general sparsity patterns by applying them to chordal extensions. Partial separability allows us to break the large semidefinite constraint into a set of smaller constraints

that can be handled by decomposition and splitting algorithms. We first use these methods to solve large sparse matrix nearness problems, in which a large symmetric matrix is projected on the set of sparse matrices with a positive semidefinite or Euclidean distance matrix completion. Second, we present a method that combines a proximal splitting method with an interior-point method to solve large linear SDPs. In both cases, the decomposition techniques are shown to effectively exploit sparsity in very large problems, and offer a significant reduction in memory and runtime over existing methods.

The dissertation of Yifan Sun is approved.

Alan Laub

Kung Yao

Tetsuya Iwasaki

Lieven Vandenberghe, Committee Chair

University of California, Los Angeles

2015

To my parents.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First I want to thank my PhD advisor, Lieven Vandenberghe for his tremendous help through every aspect of my PhD. His insights helped me understand and contextualize very difficult topics, and his suggestions and critiques helped ensure that everything I put out there would be something I could be proud of. I am very grateful to have had such an awesome advisor.

I thank my committee members, Alan Laub, Kung Yao, and Tetsuya Iwasaki, for their helpful comments and playing an important part of making my PhD possible. I also thank the EE departmental staff, who helped me through many sticky situations.

I owe a great amount of thanks as well to my classmates at UCLA for their helpful discussions and jovial camaraderie. I especially thank my labmates Daniel O'Conner, Jinchao Li, Suzi Chao, Cameron Gunn, and Rong Rong, for their general awesomeness. I also would like to thank Martin Andersen for introducing me into the world of chordal SDPs, and for continuing to answer my random email questions on this subject. Additionally, I thank my cubicle mates Jun Wang, Tsung-Yi Chen, Tom Courtade, Adam Williamson, Sina Caliskan, Ayça Balkan, Kasra Vakilinia, Sudarsan Ranganathan, Haobo Wang, Kirti Dhwaj, Ning Wang, Cheng-Yi Lin, and Chris Curwen, and other UCLA friends Ryan Gabrys, Dave Gingrich, Shaunak Mishra, Can Karakuş, Jad Hachem, Navid Vadfaee, Mihir Laghate, and so many others, for making UCLA a fantastic experience. Outside of UCLA, I thank Chris Brinton, Vicente Malave, Madeleine Udell, and many other wonderful people in the greater optimization community, who never hesitated to discuss random topics over Skype. I sincerely apologize to anyone I missed!

I would like to thank the AT&T Labs Fellowship Program and National Science Foundation Graduate Research Fellowship Program for their generous financial assistance throughout my graduate experience. In particular, I thank my mentors

at AT&T Labs, Xiang Zhou and Lynn Nelson, for their support and advice, especially at that first summer internship. I also thank Prashanth Iyengar at Masimo Corporation and Niyant Krishnamurthi, and George, Jacob, and Joseph Yadegar at UtopiaCompression, for awesome summer internships, where I gained an appreciation of many important applications and also learned how to play ultimate frisbee. I also want to thank my friends and professors at Olin College, especially Siddhartan Govindasamy, who was never too cool to chat with a former student about grad school experiences.

Finally, I owe the greatest thanks to my parents, Xiaoduo Sun and Weihe Chen, who inspired me to math and science, and continue to guide me through life. This thesis is dedicated to them and their many sacrifices, as a result of which I was always shielded from life's greatest difficulties.

Vita

2006-2010	B. S. Student Electrical and Computer Engineering Department F. W. Olin College of Engineering, Needham, Massachusetts.
2010-2011	M. S. Student Electrical Engineering Department University of California, Los Angeles (UCLA).
2013-2014	Teaching Assistant Electrical Engineering Department University of California, Los Angeles (UCLA).
2012-2015	Ph. D. Student Electrical Engineering Department University of California, Los Angeles (UCLA).

Publications

Yifan Sun, Martin S. Andersen, and Lieven Vandenberghe. "Decomposition in conic optimization with partially separable structure." *SIAM Journal on Optimization*, 24:873-897, 2014

Yifan Sun and Lieven Vandenberghe. "Decomposition methods for sparse matrix nearness problems." 2015. Under review.

CHAPTER 1

Introduction

1.1 Optimization over sparse matrix cones

In this thesis we consider optimization problems of the general form

$$\begin{array}{ll} \underset{X}{\text{minimize}} & f(X) \\ \text{subject to} & \mathbf{tr}(A_k X) = b_k, \quad k = 1, \ldots, m \\ & X \in \mathcal{C} \end{array} \tag{1.1}$$

where X is a symmetric $p \times p$ matrix variable, $\mathcal{C}$ is a convex cone, and f is a convex function. We focus on two cases of f. In the first case, f is a linear function

$$f(X) = \mathbf{tr}(CX) = \sum_{i,j} C_{ij} X_{ij},$$

and (1.1) is a linear conic optimization problem. The best known example is a semidefinite program (SDP), where $\mathcal{C}$ is the set of positive semidefinite (PSD) matrices (denoted $\mathbb{S}_+^p$). This class of conic problems has been studied intensely since the 90s, and is discussed further in section 1.2. In the second case, f is quadratic of the form

$$f(X) = \|X - C\|_F^2 = \sum_{i,j} (X_{ij} - C_{ij})^2$$

and $m = 0$. Problems of this form are called matrix nearness problems, and arise in many applications, including finance [Hig02], statistics [BX05], and computational biology [AKG13]. Although we focus on linear and quadratic functions, many of our algorithms extend to any separable sum of convex functions; that is,

$$f(X) = \sum_{i,j=1}^{p} f_{ij}(X_{ij})$$

where f_{ij} is a convex function.

We are interested in optimization problems where X is constrained to be a matrix with a specific sparsity pattern. We define a *sparsity pattern* E as a subset of index pairs representing the off-diagonal nonzeros in a symmetric matrix, and we denote $\mathbb{S}^p_E$ as the set of $p \times p$ symmetric matrices with sparsity pattern E:

$$\mathbb{S}^p_E = \{X \in \mathbb{S}^p \mid X_{ij} = 0 \text{ for all } i \neq j \text{ where } \{i, j\} \notin E\}.$$

(Here, $\mathbb{S}^p$ is the set of $p \times p$ symmetric matrices.) We focus on three types of sparse matrix cones $\mathcal{C}$, which capture multiple types of problem sparsity.

In the first case, $\mathcal{C}$ is the set of PSD matrices with sparsity pattern E

$$\mathcal{C} = \mathbb{S}^p_{E,+} = \{X \in \mathbb{S}^p_E \mid X \succeq 0\}.$$

For example, this constraint appears when optimizing over covariance matrices with imposed sparsity patterns, and a zero in the sparsity pattern indicates that two random variables are independent.

In the second case, $\mathcal{C}$ is the set of matrices with sparsity pattern E that have a PSD completion

$$\mathcal{C} = \Pi_E(\mathbb{S}^p_+) = \{\Pi_E(X) \mid X \succeq 0\}.$$

This sparse matrix set can be used in a reformulation of a dense problem with

sparse parameters. For example, consider the simple SDP

$$\begin{array}{ll} \underset{Z}{\text{minimize}} & \mathbf{tr}(CZ) \\ \text{subject to} & \mathbf{diag}(Z) = \mathbf{1} \\ & Z \succeq 0 \end{array} \tag{1.2}$$

where C is a sparse matrix with sparsity pattern E. (This problem appears as a popular relaxation of the MAX-CUT problem; see [GR00] for more details.) The matrix variable Z is in general dense; however, the problem can be reformulated with a change of variables $X = \Pi_E(Z)$, as

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \mathbf{tr}(CX) \\ \text{subject to} & \mathbf{diag}(X) = \mathbf{1} \\ & X \in \Pi_E(\mathbb{S}^p_+). \end{array} \tag{1.3}$$

This is an instance of problem (1.1) with $\mathcal{C} = \Pi_E(\mathbb{S}^p_+)$. For an optimal solution $X^\star$ to problem (1.3), any positive semidefinite completion $Z^\star$ of $X^\star$ (that is, $Z^\star \succeq 0$ and $\Pi_E(Z^\star) = X^\star$) will be an optimal solution to (1.2). In this thesis, we are interested in specific choices of a sparsity pattern E, in which computing such a $Z^\star$ from $X^\star$ can be done efficiently. (Algorithms for such completions are given in section 3.3.) However, in many applications, a full completion may not be needed. A similar reformulation can be done for matrix nearness problems with a sparse parameter C. Applications involving this choice of set $\mathcal{C}$ are discussed further in section 1.2.

In the third case, $\mathcal{C}$ is the of sparse matrices with a Euclidean distance matrix (EDM) completion

$$\mathcal{C} = \Pi_E(\mathbb{D}^p) = \{\Pi_E(X) \mid X \in \mathbb{D}^p\}.$$

Here $\mathbb{D}^p$ is the set of Euclidean distance matrices of order p, *i.e.* matrices with entries that can be expressed as squared Euclidean distances between pairs of points

$$\mathbb{D}^p = \{X \in \mathbb{S}^p \mid X_{ij} = \|u_i - u_j\|_2^2, \text{ for points } u_1, \ldots, u_p\}.$$

For example, consider the noisy EDM completion problem

$$\begin{array}{ll} \underset{Z}{\text{minimize}} & \sum_{\{i,j\} \in E} (Z_{ij} - C_{ij})^2 \\ \text{subject to} & Z \in \mathbb{D}^p \end{array} \tag{1.4}$$

where C contains a partial noisy reading of pairwise distances, and has sparsity pattern E. (A variation of this problem appears in sensor network node localization, molecular conformation recovery, and multidimensional scaling.) Then problem (1.4) can be reformulated as

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|X - C\|_F^2 \\ \text{subject to} & X \in \Pi_E(\mathbb{D}^p). \end{array} \tag{1.5}$$

These two problems (1.4) and (1.5) are equivalent in the sense that, at optimality, $X^\star = \Pi_E(Z^\star)$. As in the previous case, we consider problems where the dense completion $Z^\star$ is easy to compute from $X^\star$, though a full EDM completion may not always be needed. More examples of this type of problem can be found in section 1.3.

For the first two cases of $\mathcal{C}$, problem (1.1) is an instance and a reformulation of an SDP. In the third case of $\mathcal{C}$, problem (1.1) is a reformulation of an optimization problem over the set of EDMs. In the next two sections, we give a more detailed introduction of these two classes of optimization problems, followed by an overview of important algorithms in section 1.4.

1.2 Semidefinite optimization

Semidefinite matrix constraints arise naturally in many applications. (See, for example, the surveys [PL03, VB95, VB99, KW12, WSV00].) In statistics, PSD matrices appear as covariance or correlation matrices, for example in the problem of finding a sparse covariance estimate [BL08] or inverse covariance estimate [Dem72, dBE08] for a Gaussian model, or in adjusting noisy correlation measurements [Hig02, Mal04, HM12]. In machine learning, SDPs also appear in dimensionality reduction [WS06] and in learning nonlinear kernel matrices [LCB04].

The most common occurrence of SDPs is as the backbone of general purpose solvers like CVX [GB12] and Yalmip [Lof04]. These solvers reformulate general convex problems as conic problems, where $\mathcal{C}$ is a product of nonnegative orthants, second order cones, and (most generally) PSD cones; the problem is then solved using a state-of-the-art interior-point method.

Solving an SDP can also be used as a polynomial-time approximation of NP-hard problems, and to compute bounds on the optimal value. For example, SDPs are often used as a relaxation of combinatorial problems [BV96, Par03], such as in relaxations of MAX-CUT [DH73, GW95], approximating the graph chromatic and independence number [Lov79, GLS88, LS91] and finding the maximum weight equipartition [KR98]. Because SDP relaxations are tighter than LP relaxations of combinatorial problems, SDP solvers can be used to construct faster converging branch-and-bound solvers [Hu06, BV08, KMR14]. And, it has also been shown that polynomial optimization problems can be approximated using a hierarchy of SDPs [Las01, Par00]. As an example, we show how SDPs arise from relaxations of quadratically constrained quadratic programs (QCQP), a popular subset of nonconvex optimization problems.

Example: Nonconvex optimization Many nonconvex optimization problems can be reformulated as QCQPs

$$\begin{array}{ll} \underset{x}{\text{minimize}} & x^T A_0 x + b_0^T x + c_0 \\ \text{subject to} & x^T A_i x + b_i^T x + c_i \leq 0, \quad i = 1..., m \end{array} \tag{1.6}$$

where $A_i \in \mathbb{S}^p$ and may be sparse, $b_i \in \mathbb{R}^p$, and $c_i \in \mathbb{R}$ for $i = 0, \ldots, m$. (Here, $\mathbb{S}^p$ is the set of $p \times p$ symmetric matrices.) This problem covers many types of nonconvex objectives and constraints, since the matrices A_i may not be PSD. (For example, the constraint $x \in \{-1, 1\}$ can be written as $x^2 = 1$, which is a combination of two quadratic inequalities.) We can write (1.6) as a matrix optimization problem using a change of variables $X = xx^T$. The convex relaxation comes by replacing this equality with an SDP constraint

$$X \succeq xx^T \iff \begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0,$$

resulting in the following convex relaxation of (1.6)

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \mathbf{tr}(A_0 X) + b_0^T x + c_0 \\ \text{subject to} & \mathbf{tr}(A_i X) + b_i^T x + c_i \leq 0, \quad i = 1, ..., m \\ & \begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0. \end{array} \tag{1.7}$$

Problem (1.7) is an SDP with aggregate sparsity pattern E if the matrices

$$\begin{bmatrix} A_i & b_i \\ b_i^T & c_i \end{bmatrix} \in \mathbb{S}_E^p, \text{ for all } i = 0, \ldots, m.$$

This type of problem and relaxations appear in several applications, such as combinatorial optimization over graphs [GR00]. Recently, this SDP relaxation is

used as a relaxation of the optimal power flow problem [Low14a, Low14b, LL12, Jab12, BHF08, MLD14, AHV14]. Here, solving the relaxation (1.7) provides useful bounds on the solution to the original problem (1.6). In these applications, the sparsity of the relaxed problem corresponds to the underlying network topology of the application, which tends to be very sparse.

1.3 Euclidean distance matrices

The problem of finding an EDM completion [BJ95] or projection [Hig88, GHH90], [GM89, §5.3] appears many seemingly unrelated fields, such as dimensionality reduction for machine learning, sensor network node localization, and molecular conformation recovery. For example, in sensor network node localization, the objective is to recover the configuration of a family of sensors based on noisy distance measurements between nearby pairs [BY04b, KW12]. This can be done by first finding an (approximate) low-rank EDM completion and then directly transforming it into a configuration matrix. A closely related problem is to find the atomic configuration of macromolecules based on noisy bond length measurements [Tro97, CH88, Wut89, AKG13]. In both cases, the desired EDM must be for points in $\mathbb{R}^2$ or $\mathbb{R}^3$, making the problem nonconvex. However, finding an EDM without this dimension restriction (the convex relaxation) can be solved using an SDP, and is often used as an important substep to finding the feasible completion [AKG13].

In general, EDM optimization problems can be interpreted as an SDP, since EDM constraints can be expressed equivalently as semidefinite and affine constraints. They can also be regarded as an instance of the general distance matrix completion problem, and solved using a very different set of techniques [CH88, MLL12]. Surveys for Euclidean distance problems include [KW12, Dat10, AKW99]. The problem will be discussed in more detail in section 3.1.

Example: Sensor network node localization In this application (described by Biswas and Ye [BY04b]) low-cost sensors are arbitrarily distributed across some topology. These sensors communicate noisily with other nearby sensors to estimate their pairwise distances, and the problem is to recover the node configurations from these noisy, partial distance readings. This is done by first finding the best fitting EDM to the partial noisy measurements

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \sum_{\{i,j\}\in E} (X_{ij} - \hat{D}_{ij})^2 \\ \text{subject to} & X \in \mathbb{D}^p \\ & X \text{ has embedding dimension } r, \end{array} \tag{1.8}$$

where E is the sparsity pattern of $\hat{D}$ ($\{i,j\} \in E$ if sensor i and j are close) and r is the dimension of the topology (usually 2 or 3). (We say that an EDM X has embedding dimension r if X is the EDM for p points $u_k \in \mathbb{R}^r$ for each $k = 1, \ldots, p$.) Applications in biology [AKG13] and multidimensional scaling [Kru64] involve similar problems. A difficulty with solving problem (1.8) is the nonconvex constraint that X has embedding dimension r. In section 4.4.3, we discuss solving the convex relaxation, where this constraint is dropped; in this case, (1.8) is a convex sparse matrix nearness problem, with E as its aggregate sparsity pattern.

Example: Maximum variance unfolding A convex optimization method that heuristically finds low-rank EDM completions is maximum variance unfolding, proposed by Weinberger and Saul [WS06]. This is in the context of machine learning, where dimensionality reduction of high-dimensional features is necessary for computational efficiency. The hypothesis is that these features live on a low-dimensional nonlinear manifold, making it difficult to use linear methods like PCA to extract low-rank solutions. Maximum variance unfolding addresses this by first "stretching" the nonlinear manifold so that it fits in a linear subspace.

This is done by solving the following linear SDP

$$\begin{array}{ll} \underset{X}{\text{maximize}} & \mathbf{tr}(X) \\ \text{subject to} & X_{ii} + X_{jj} - 2X_{ij} = \|\hat{u}_i - \hat{u}_j\|_2^2, \quad \text{for all } \|\hat{u}_i - \hat{u}_j\|_2^2 < \rho \\ & X\mathbf{1} = 0 \\ & X \succeq 0. \end{array} \tag{1.9}$$

Here, $\hat{u}_i$ are the given feature vectors, and $X \succeq 0$ is the Gram matrix of the recovered points. ($X_{ij} = u_i^T u_j$ where u_i are the estimated points lying in a low-dimensional subspace.) In the objective, $\mathbf{tr}(X) = \sum_{i=1}^{p} \|u_i\|^2$, is maximizing the variance of points (blowing them apart). The first constraint forces the local configuration to stay constant, and the constraint $X\mathbf{1} = 0$ (where $\mathbf{1}$ is the vector of all 1's) keeps the center of mass of the points at the origin. (Otherwise, the problem would be unbounded.) Often, the solution to (1.9) is low rank, and can be converted to a EDM

$$D = \mathbf{diag}(X)\mathbf{1}^T + \mathbf{1}\mathbf{diag}(X)^T - 2X$$

where the embedding dimension of D is the rank of X. (Here, $\mathbf{diag}(X) \in \mathbb{R}^p$ contains the diagonal of X.) The derivation of this linear map is discussed further in section 3.1.

Both (1.8) and (1.9) are examples of important sparse matrix problems, and demonstrate the close relationship between the PSD cone and the EDM cone. In both cases, the sparsity pattern is given by the underlying network. As a convex heuristic for finding a low-rank solution, maximum variance unfolding is surprisingly successful; however, in practice it is difficult to solve for large matrices, as it requires solving a large SDP.

1.4 Algorithms

In this thesis, we explore two common classes of algorithms for semidefinite programming: interior-point methods and first-order methods. Interior-point methods are similar in per-iteration complexity to Newton's method, which at each iteration requires computing and factoring a Hessian (second order derivative). This can be costly for large matrix variables. Recently, however, there has been an increasing trend in developing first-order methods (which do not use second or higher-order derivatives). These methods are usually capable of handling larger problems, and often exploit sparsity more readily. The tradeoff is that first-order methods generally require many iterations to converge, and may taper off at low-accuracy solutions; in contrast, interior-point methods are known to converge to high-accuracy solutions fairly quickly (in about 20-50 iterations).

1.4.1 Interior-point methods

An interior-point method solves the following conic optimization problem and its dual

$$\begin{array}{lll} \underset{x}{\text{minimize}} & c^T x & \\ \text{subject to} & Ax = b & \\ & x \in \mathcal{C} & \end{array} \qquad \begin{array}{lll} \underset{y,s}{\text{maximize}} & b^T y & \\ \text{subject to} & A^T y + s = c & \\ & s \in \mathcal{C}^* & \end{array} \tag{1.10}$$

where $\mathcal{C}^*$ is the dual cone of $\mathcal{C}$. At each iteration, the interior-point method solves the linear system

$$\begin{bmatrix} H & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_x \\ r_y \end{bmatrix} \tag{1.11}$$

where H depends on the type of scaling used. In most implementations, the first block row is eliminated, and the Schur complement system is solved instead:

$$AH^{-1}A^T \Delta y = AH^{-1}r_x - r_y.$$

The efficiency of the interior-point method then depends on the ease of forming and factoring the $AH^{-1}A^T$ term. In linear programming, the matrix H is diagonal, and $AH^{-1}A^T$ is sparse if AA^T is sparse. Interior-point methods for linear conic optimization are discussed in more detail in section 2.5.

In the case of semidefinite optimization, it is more convenient to write (1.10) as two matrix problems

$$\begin{array}{lll} \min\limits_{X} & \mathbf{tr}(CX) & \\ \text{s.t.} & \mathbf{tr}(A_i X) = b_i,\ k = 1, \ldots, m & \\ & X \succeq 0 & \end{array} \qquad \begin{array}{lll} \max\limits_{y,S} & b^T y & \\ \text{s.t.} & \sum\limits_{i=1}^{m} A_i y_i + S = C & \\ & S \succeq 0 & \end{array} \tag{1.12}$$

where, now, C and A_i, $i = 1, \ldots, m$ are symmetric $p \times p$ matrices. However, unlike for linear programming, the corresponding H matrix in (1.11) for the semidefinite cone is not diagonal, and H^{-1} (and therefore the Schur complement system) is in general dense. This presents a difficulty in semidefinite optimization, in that it is not easy to exploit sparsity when factoring the Schur complement system.

However, certain techniques exist. When C and A_k are sparse with pattern E, then the primal problem has aggregate sparsity E, and a dual feasible S must be sparse with pattern E. Both Choi and Ye [CY00] and Benson, Ye, and Zhang [BYZ00, BY04a] exploit this in proposing dual scaling methods which can solve problems up to $p = 20,000$. In dual scaling, the Hessian is then efficiently constructed from the sparse dual intermediate variables. A similar trick can be applied for primal scaling methods, where the primal variable is substituted by a sparse matrix variable $X \in \Pi_E(\mathbb{S}^p_+)$; see [FKM00, NFF03, YFK03].

The ideas can be combined to construct a primal-dual interior-point method,

solving a nonsymmetric reformulation of (1.12)

$$\begin{array}{lll} \min\limits_{X} & \mathbf{tr}(CX) & \max\limits_{y,S} \quad b^T y \\ \text{s.t.} & \mathbf{tr}(A_k X) = b_k,\ k = 1, \ldots, m & \text{s.t.} \quad \sum\limits_{k=1}^{m} y_k A_k + S = C \\ & X \in \Pi_E(\mathbb{S}^p_+) & \qquad S \in \mathbb{S}^p_{E,+} \end{array}$$

where the sparse cone $\Pi_E(\mathbb{S}^p_+)$ is our second case of $\mathcal{C}$. (This is the method of Srijuntongsiri and Vavasis [SV04].) Additionally, Andersen *et al.* gives efficient implementations for constructing sparse gradients and Hessians for these non-symmetric cones [ADV10c, ADV10b].

The methods closest to our proposed method are the conversion methods of Fukuda *et al.* [FKM00, NFF03, FKK09], which break the sparse conic constraint into overlapping conic constraints, allowing H^{-1} to be block diagonal rather than dense. We discuss this method in more detail in chapter 5.

1.4.2 First-order methods

The term "first-order methods" encompasses a variety of popular algorithms, including proximal algorithms (*e.g.* the proximal gradient method and the alternating direction method of multipliers (ADMM)) and block coordinate descent methods (*e.g.* Dykstra's alternating projection method). It is impossible to give a comprehensive overview, since the idea of only using first-order information is so general; however, more details on these two classes of algorithms are given in sections 2.3 and 2.4.

Unlike interior-point methods, first-order methods may require many iterations to converge. However, their per-iteration costs are usually significantly lower. This can be understood intuitively in the context of problem (1.1), where applying a first-order method typically entails two main steps: the projection on the affine constraint, and the projection on the conic constraint. The projection on the set

$Ax = b$ can be done by computing a factorization of A in preprocessing, and if A is sparse, this can be done very efficiently. The factors can then be reused for different intermediate values of b. In contrast, an interior-point method combines both projections in one linear system (1.11), where the matrix H provides a linearization of the conic constraint at each step, and therefore changes each time. An interior-point method has to recompute the factorization of $AH^{-1}A^T$ at each iteration (which for SDPs is dense even if A is sparse) and without special structure, cannot reuse much of this expensive computation.

However, in the applications presented in this thesis, we often find that a simple application of a first-order method still does not exploit sparsity effectively. In semidefinite optimization, when $\mathcal{C} = \mathbb{S}^p_{E,+}$, each step requires computing a $p \times p$ eigenvalue decomposition to project on $\mathcal{C}$. This is often much cheaper than forming and solving $AH^{-1}A^T$ at each step; however, for very large matrices, even this may not be enough, as computing eigenvalue decompositions generally does not scale well with matrix size.

Example: Projection on the sparse PSD cone Consider the problem of projecting a $p \times p$ matrix C on $\mathbb{S}^p_{E,+}$:

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|X - C\|_F^2 \\ \text{subject to} & X \in \mathbb{S}^p_{E,+} \end{array}$$

where the conic constraint $X \in \mathbb{S}^p_{E,+}$ simultaneously enforces a sparsity constraint and a PSD constraint. Without the sparsity constraint, the problem requires a single eigenvalue decomposition of order p:

$$X = \sum_{i=1}^{p} \max\{\lambda_i, 0\} u_i u_i^T$$

where λ_i and u_i are the eigenvalues and corresponding eigenvectors of C. With the sparsity constraint, an intuitive first-order method is to alternately project on the sparsity and PSD constraint (as described in section 2.3). This requires not one, but a sequence of expensive dense eigenvalue decompositions, each with significant complexity if C is large. To get an idea, on a desktop computer with 32 GB RAM and an Intel Xeon CPU E31225 processor, interior-point methods could not solve problems with p more than a few thousand, where p is the order of the matrix variable. First-order methods scale slightly better, but a $p \times p$ eigenvalue decomposition will be very slow for $p > 10,000$, and will exceed memory for $p > 50,000$. In contrast, using our decomposition methods, we solve problems with $p = 100,000$ without significant memory usage.

1.5 Contributions

The main contribution of this thesis is the formulation of new decomposition methods for sparse matrix optimization problems. This involves first transforming problems with aggregate sparsity as optimizations over a sparse matrix variable, and showing that efficient completion methods exist (chapter 3). The sparse matrix cones are expressed as either partially separable cones or their dual (chapter 3). The decomposition algorithms are formed by applying first-order methods to optimization problems over partially separable cones, and fall into three categories:

1. dual decomposition methods (such as dual proximal gradient and dual coordinate descent) for minimizing strongly convex functions over partially separable cones (chapter 4),

2. a variation of a Douglas-Rachford method on a consensus reformulation of (1.1) when the projection on the affine constraints is simple (chapter 4),

3. and a hybrid Spingarn and interior-point method approach when the affine constraints are nontrivial (chapter 5).

To demonstrate the scalability of our algorithms, we apply these algorithms to large sparse SDPs. We give numerical examples with matrix variable sizes ranging from 1000 to 100,000.

Outline Chapter 2 discusses optimization theory, including convex analysis, interior-point methods, and first-order methods. Chapter 3 gives the necessary background for matrices, including EDM properties, sparse matrix decomposition theorems, and matrix completion methods. Chapter 4 introduces the dual decomposition and Douglas-Rachford consensus methods for matrix nearness problems. Finally, chapter 5 introduces a hybrid Spingarn and interior-point method that tackles SDPs with aggregate sparsity but have nontrivial affine constraints.

CHAPTER 2

Convex optimization

In this chapter, we begin by summarizing important concepts related to the problem formulation in section 2.1. We discuss the fundamental algorithms in section 2.2. We then present the existing methods in sections 2.3, 2.4, and 2.5 that are used to design the decomposition methods in chapters 4 and 5.

2.1 Convex analysis

The following concepts will be used throughout the thesis.

Convex set A set $\mathcal{S}$ is *convex* if, for all $x, y \in \mathcal{S}$, the point $\theta x + (1-\theta)y \in \mathcal{S}$ whenever $0 \leq \theta \leq 1$. In general, we use $\Pi_{\mathcal{S}}(x)$ to denote the Euclidean projection of x on $\mathcal{S}$ and $\delta_{\mathcal{S}}$ to denote the indicator function of $\mathcal{C}$:

$$\delta_{\mathcal{S}}(x) = \begin{cases} 0, & \text{if } x \in \mathcal{S}, \\ +\infty, & \text{else.} \end{cases}$$

Convex cones A *cone* or *conic set* $\mathcal{C}$ is a set that is invariant to nonnegative scaling:

$$x \in \mathcal{C} \iff \alpha x \in \mathcal{C}, \quad \forall \alpha \geq 0.$$

In this thesis we deal with only convex cones, *i.e.* cones that are also convex sets. We say $\mathcal{C}$ is pointed if $x \in \mathcal{C}$ and $-x \in \mathcal{C}$ implies $x = 0$. A cone is *proper* if it is

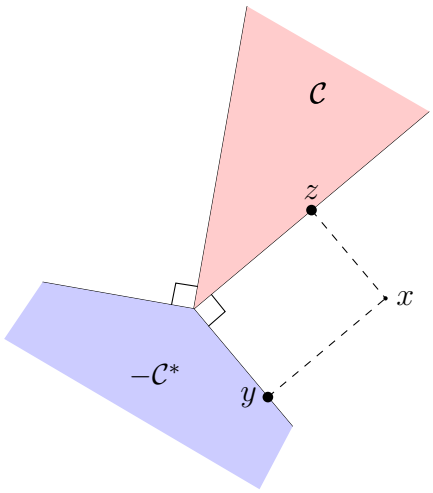

Figure 2.1: *Projection on dual cones.* Any vector in $\mathbb{R}^n$ can be decomposed in terms of its projection on a convex cone $\mathcal{C}$ and its negative dual cone $-\mathcal{C}^*$. Here, $z = \Pi_{\mathcal{C}}(x)$ and $y = \Pi_{-\mathcal{C}^*}(x) = -\Pi_{\mathcal{C}^*}(-x)$, and the vector $x = y + z$.

convex, closed, pointed, and has nonempty interior.

Dual cones The *dual cone* of $\mathcal{C}$ is

$$\mathcal{C}^* = \{y \mid x^T y \geq 0, \quad \forall x \in \mathcal{C}\},$$

and is always a closed convex cone, even if $\mathcal{C}$ is not. The dual cone to a proper cone is also always proper. The Euclidean projection on a closed convex cone can be expressed in terms of the projection on its dual cone

$$x = \Pi_{\mathcal{C}}(x) + \Pi_{-\mathcal{C}^*}(x) = \Pi_{\mathcal{C}}(x) - \Pi_{\mathcal{C}^*}(-x). \tag{2.1}$$

(See figure 2.1 for an illustration.) When $\mathcal{C} = \mathcal{C}^*$ then $\mathcal{C}$ is a self-dual cone. This is true, for example, if $\mathcal{C}$ is the nonnegative orthant, or the set of PSD matrices.

Convex functions A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if **dom** f (the domain of f) is convex and Jensen's inequality is satisfied

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y), \quad \forall x, y \in \textbf{dom } f, \quad 0 \leq \theta \leq 1.$$

A function is *closed* when its epigraph

$$\textbf{epi } f = \{(x, y) \mid x \in \textbf{dom } f, y \geq f(x)\}$$

is closed.

First order condition If the function is differentiable at every point in the domain, then it satisfies the following first-order condition: for all x in the domain of f,

$$f(x) - f(y) \leq \nabla f(x)^T (x - y), \quad \forall y \in \textbf{dom } f.$$

If $\nabla f(x) = 0$ this implies x is a minimizer of f. Points x satisfying this condition are *stability points*, and correspond to the global minima of a convex function f.

Conjugate function The function

$$f^*(z) = \sup_x (x^T z - f(x))$$

is the *convex conjugate* (also called the *Fenchel* or *Legendre-Fenchel conjugate*) of f. Regardless of the convexity of f, the conjugate function f^* is always closed and convex. From the definition, it follows that the conjugate of a separable function is also separable:

$$f(x) = \sum_{k=1}^{n} f_k(x_k) \iff f^*(z) = \sum_{k=1}^{n} f_k^*(z_k),$$

where f_k^* are the conjugate functions of f_k.

Subgradients For convex nondifferentiable $f(x)$, we define a *subgradient* of $f(x)$ as a vector $g \in \mathbb{R}^n$ satisfying

$$f(y) - f(x) \geq g^T(y - x), \quad \forall y \in \mathbf{dom}\ f. \tag{2.2}$$

The set of all g satisfying (2.2) is the *subdifferential* of $f(x)$. The subdifferential $\partial f : \mathbb{R}^n \to 2^{\mathbb{R}^n}$ is an example of an *operator*, or a point-to-set mapping. If f is smooth at x, it can be shown that $\partial f(x) = \{\nabla f(x)\}$ [Pol87, Ch. 5 L. 5]. As a small example, consider the absolute value function $f : \mathbb{R} \to \mathbb{R}$, $f(x) = |x|$. Then

$$\partial f(x) = \begin{cases} \{1\}, & \text{if } x > 0, \\ \{-1\}, & \text{if } x < 0, \\ [-1, 1], & \text{if } x = 0. \end{cases}$$

When the function is smooth, the subdifferential is single-valued. When the function is nonsmooth, the subdifferential contains the slopes of the supporting hyperplanes of the function's epigraph at that point. Additionally, from the definition (2.2), note that if $g = 0$, then

$$f(y) - f(x) \geq 0,\ \forall y.$$

In this case, we say that x is a stability point of f if $0 \in \partial f(x)$, and since f is convex,

$$x \text{ minimizes } f(x) \iff 0 \in \partial f(x).$$

Subdifferentials of conjugates For a closed convex function f and its convex conjugate f^*, the subdifferentials are related through

$$z \in \partial f(x) \iff x \in \partial f^*(z) \iff z^T x = f(x) + f^*(z) \tag{2.3}$$

which is discussed and proven in [Roc70, 23.5]. We can intuitively prove this by noting that for some vector z,

$$x = \arg\max_y \; y^T z - f(y) \iff z \in \partial f(x).$$

From the definition of the conjugate function, it follows that

$$f^*(z) = \max_y y^T z - f(y) = x^T z - f(x). \tag{2.4}$$

And, since f is a closed, convex function, then $(f^*)^* = f$, and

$$f(x) = \max_y \; y^T x - f^*(y) = x^T z - f^*(z).$$

where the third term is from rearranging (2.4). This implies the following two statements

$$z = \arg\max_y \; y^T x - f^*(y) \iff x \in \partial f^*(z).$$

Subgradients are positive homogeneous An important theorem regarding subgradients is the Moreau-Rockafellar theorem [Roc70, Th. 23.8], [Pol87, Ch.5, Lemma 10], which says that subgradients are positive homogeneous

$$\partial(f_1 + f_2)(x) = \partial f_1(x) + \partial f_2(x) \quad \text{and} \quad \partial(\alpha f) = \alpha \partial f, \; \alpha > 0, \tag{2.5}$$

if there exists some point in both the relative interiors of the domains of f_1 and f_2:

$$(\textbf{ri dom } f_1) \cap (\textbf{ri dom } f_2) \neq \emptyset.$$

(Here, $\textbf{ri } \mathcal{S}$ is the relative interior of a set $\mathcal{S}$.)

Proximal operator For a closed, convex function f, its *proximal operator* is:

$$\textbf{prox}_f(z) = \operatorname*{argmin}_x \ f(x) + \frac{1}{2}\|x - z\|_2^2.$$

It can be shown that $\textbf{prox}_f(z)$ is unique and exists for all z [Mor65, BC11]. We often also parametrize the proximal operator by a step size t:

$$\textbf{prox}_{tf}(z) = \operatorname*{argmin}_x \ tf(x) + \frac{1}{2}\|x - z\|_2^2 = \operatorname*{argmin}_x \ f(x) + \frac{1}{2t}\|x - z\|_2^2.$$

We can also interpret $\textbf{prox}_{tf}$ as the gradient of the Moreau-Yosida regularized form

$$\tilde{f}_t(z) = \min_x f(x) + \frac{1}{2t}\|x - z\|_2^2.$$

which can be viewed as a locally smoothed version of f.

The evaluation $u = \textbf{prox}_{tf}(z)$ is itself an optimization problem, with the optimality condition

$$0 \in \partial(f(u) + \frac{1}{2t}\|u - z\|_2^2) \iff \frac{z - u}{t} \in \partial f(u). \tag{2.6}$$

Equivalently, we can write this as $(I + t\partial f)(u) = z$, where $I(x) = \{x\}$ is the identity operator. It can be shown that if $f(x)$ is closed, then the operator $I + t\partial f$ is invertible, and the proximal operator can be written equivalently as

$$\textbf{prox}_{tf}(z) = (I + t\partial f)^{-1} z.$$

Properties of the proximal operator The following are some properties of proximal operators which make them useful for decomposition methods.

1. (Separability) If $f(x)$ is separable, then $\mathbf{prox}_f$ can be computed in independent segments:

$$f(x) = f_1(x_1) + f_2(x_2) \;\Rightarrow\; \mathbf{prox}_f(z) = (\mathbf{prox}_{f_1}(z_1),\; \mathbf{prox}_{f_2}(z_2)).$$

   This makes the operator particularly amenable to parallelism.

2. (Moreau decomposition) [Mor65] If f^* is the convex conjugate of a closed, convex function f, then

$$\mathbf{prox}_f(x) + \mathbf{prox}_{f^*}(x) = x. \tag{2.7}$$

   The property follows from (2.3) and (2.6):

$$\begin{aligned} u = \mathbf{prox}_f(x) &\iff x - u \in \partial f(u) \\ &\iff u \in \partial f^*(x-u) \\ &\iff x - u = \mathbf{prox}_{f^*}(x). \end{aligned}$$

   Note that for $f(x) = \delta_{\mathcal{C}}(-x)$ where $\mathcal{C}$ is a closed convex cone, (2.7) reduces to (2.1).

3. (Firmly nonexpansive) If f is a closed, convex function, then

$$(u-v)^T(x-y) \geq \|u-v\|_2^2 \text{ where } u = \mathbf{prox}_{tf}(x) \text{ and } v = \mathbf{prox}_{tf}(y),$$

   a property we call *firm nonexpansiveness*. This is useful because it is known that, for firmly nonexpansive operators $T(x)$, the iterates

$$x^+ = (1-\rho)x + \rho T(x), \quad 0 < \rho < 2$$

converge weakly to a fixed point [EB92, Roc76].

Evaluating a proximal operator The evaluation of a proximal operator involves solving an optimization problem, and in general, it may be as difficult as optimizing f itself. However, there are several key cases where the evaluation is cheap. For our methods, we will only need three of these simple cases.

- (Indicator functions.) If $f(x) = \delta_{\mathcal{S}}(x)$, then $\mathbf{prox}_{tf}(z) = \Pi_{\mathcal{S}}(z)$.
- (Linear functions.) If $f(x) = c^T x$ then $\mathbf{prox}_{tf}(z) = z - tc$.
- (Euclidean distance.) If $f(x) = \|x - c\|_2^2$, then $\mathbf{prox}_{tf}(z) = \dfrac{tc + 2z}{2 + t}$.

There also exist many other examples where evaluating $\mathbf{prox}_{tf}$ is cheap (*e.g.* the shrinkage operator). References for convex analysis include [Pol87, Roc70, BV04, BC11].

2.2 Gradient methods

In the simplest case, consider the minimization of an unconstrained, smooth convex function $f(x)$, with full domain. Since f is convex, then $x^\star$ minimizes f if $\nabla f(x^\star) = 0$

Lipschitz gradient and strong convexity We say f has an L-Lipschitz continuous gradient if

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L\|x - y\|_2,$$

and is μ-strongly convex if

$$(\nabla f(x) - \nabla f(y))^T (x - y) \geq \mu \|x - y\|_2^2, \tag{2.8}$$

or simply strongly convex if there exist some $\mu > 0$ which satisfies (2.8). While a convex function in general may have many optimal variables, strongly convex functions always have a unique optimal variable. (Assume that $x \neq y$ but $\nabla f(x) = \nabla f(y) = 0$. Then (2.8) is clearly violated.) Strong convexity and Lipschitz continuous gradients are related through duality; if $f(x)$ is closed and μ-strongly convex, then $f^*(z)$ is differentiable and defined for all z, where $\nabla f^*(z)$ is $1/\mu$-Lipschitz continuous. For twice-differentiable convex functions, the constants L and μ bound the Hessian:

$$LI \succeq \nabla^2 f(x) \succeq \mu I$$

and the ratio L/μ upper bounds the condition number of $\nabla^2 f(x)$.

Gradient descent A basic method for minimizing a convex differentiable function $f(x)$ is the gradient descent method, with references dating back to Cauchy [Cau47, Gol62]. This method uses the iterative scheme

$$x^+ = x - t\nabla f(x)$$

where $t > 0$ is a chosen step size. It is known that, if the optimum exists, then using $0 < t < 2/L$, the iterates converges to the set satisfying the first-order optimality condition $0 = \nabla f(x^\star)$ [Pol87, Ch 1, Th 1] where $|f(x^i) - f(x^\star)|$ converges to 0 at a rate of $O(1/i)$ [BT09, §3]. Polyak also shows that, with t in this range, the method provably monotonically decreases in function value; for each iteration i, [Pol87, Ch. 1, Th 2]

$$f(x^i) < f(x^{i-1}).$$

If f is both μ-strongly convex and with L-Lipschitz continuous gradient, then the convergence rate is improved to linear [Pol87, Ch. 1, Th. 2, 3],[Nes04, Th. 1.2.4]

$$\|x^{i+1} - x^\star\|_2 \leq c^i \|x^i - x^\star\|_2, \quad c = 1 - t\frac{2\mu L}{\mu + L}.$$

Of course, most optimization problems are more involved than just minimizing a differentiable convex function with full domain. When $f(x)$ is nondifferentiable and its domain is restricted, more interesting gradient methods are needed.

Proximal point method The differentiability condition on $f(x)$ can be dropped in the *proximal point algorithm*, which minimizes any closed convex f by repeating the iteration

$$x^+ = \mathbf{prox}_{tf}(x)$$

for some $t > 0$ [Roc76, EB92, Mar70]. The proof of convergence follows from two observations. First, $\mathbf{prox}_{tf}$ is a firmly nonexpansive operator, so it converges to a fixed point. Second, by (2.6),

$$x - x^+ \in \partial f(x^+) \text{ and } x = x^+ \iff 0 \in \partial f(x),$$

which means that any fixed point solution x minimizes f. However, the proximal point method is not very common in practice, mostly because there are not many applications where minimizing $f(x) + \|x - z\|_2^2$ is much easier than minimizing f.

Further references for these fundamental algorithms include [Pol87, BT09, Nes04]. In the following sections we present practically efficient first-order methods that can solve specific classes of convex problems.

2.3 Dykstra's algorithm and dual block coordinate ascent

We first consider the optimization problem of projecting on the intersection of convex sets

$$\begin{array}{ll} \underset{x}{\text{minimize}} & (1/2)\|x - c\|_2^2 \\ \text{subject to} & x \in \mathcal{C}_1 \cap \cdots \cap \mathcal{C}_l. \end{array} \tag{2.9}$$

A popular method for solving (2.9) is Dykstra's alternating projection method, which uses a sequence of projections on each set $\mathcal{C}_k$. For the algorithm to be efficient, we assume that $\mathcal{C}_k$, for $k = 1, \ldots, l$, are simple sets; that is, the projections $\Pi_{\mathcal{C}_k}$ are easy to compute.

If $\mathcal{C}_k$ are all convex cones, then the dual problem of (2.9) is

$$\begin{array}{ll} \underset{s_1,\ldots,s_l}{\text{maximize}} & -(1/2)\| \sum_{k=1}^{l} s_k + c\|_2^2 + (1/2)\|c\|_2^2 \\ \text{subject to} & s_k \in \mathcal{C}_k^*, \quad k = 1, \ldots, l \end{array} \tag{2.10}$$

and the optimality conditions include primal and dual feasibility, and

$$x - c = \sum_{k=1}^{l} s_k. \tag{2.11}$$

From (2.11), the primal variable can be written in terms of the dual variables, so solving the dual also solves the primal. However, extracting the dual variable from the primal solution cannot be done trivially.

Dykstra's method has been applied to several types of dense matrix nearness problems in the literature. For example, the problem of finding the nearest EDM matrix to a given matrix is discussed in [GHH90] and [GM89, section 5.3], and the nearest correlation matrix problem is discussed in [Hig02, Mal04, HM12]. Additionally, it can be shown that Dykstra's method is equivalent to the block coordinate ascent method applied to the dual problem.

2.3.1 Dykstra's projection algorithm

Von Neumann's theorem and proof in [Neu50, Th. 13.7] showed that by alternately projecting a vector x on affine sets $\mathcal{C}_1, \ldots, \mathcal{C}_l$ for some finite l, the vector x will eventually converge on their intersection $\mathcal{C}_1 \cap \cdots \cap \mathcal{C}_l$. Dykstra extended the method to projections on general convex sets with a simple correction term [Dyk83, BD86,

BB96]. It can be summarized as follows: with variables initialized as

$$x_l^0 = c, \quad s_k^0 = 0, \quad k = 1, \ldots, l,$$

compute for cycles $i = 1, 2, \ldots$

$$x_k^i = \Pi_{\mathcal{C}_k}(x_{k-1}^i - s_k^{i-1}), \qquad s_k^i = x_k^i - (x_{k-1}^i - s_k^{i-1}), \quad k = 1, \ldots, l, \tag{2.12}$$

with $x_0^{i+1} = x_l^i$. The method is a special instance of the successive projection method [Han88, Tse93], which minimizes the convex function $f(x) + \sum_{k=1}^{m} g_k(x)$ where $f(x)$ is strongly convex, by cyclically evaluating the proximal operator of $g_k(x)$. The convergence of Dykstra's algorithm can also be derived by the convergence of this larger class of algorithms.

2.3.2 Block coordinate ascent

Block coordinate ascent algorithms maximize a function by optimizing over variable blocks one at a time. (Equivalently, block coordinate descent minimizes a convex function the same way.) There are several schemes in deciding the block order. The most basic scheme to solve

$$\underset{x_1,\ldots,x_n}{\text{minimize}} \quad f(x_1, \ldots, x_n)$$

is the Gauss-Seidel ordering, where the iterates are updated in order, and always uses the newest information:

$$x_i^+ = \underset{y}{\text{argmin}}\ f(x_1^+, \ldots, x_{i-1}^+, y, x_{i+1}, \ldots, x_m).$$

A parallelizable version is the Gauss-Jacobi ordering, where no new information is used in the same cycle:

$$x_i^+ = \underset{y}{\operatorname{argmin}}\ f(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_m).$$

A third popular scheme that requires more optimization is the Gauss-Southwell scheme, where the newest information is always used, and at each iteration, the block x_i to be updated corresponds to the block of $\nabla f(x)$ with smallest norm.

Block coordinate descent algorithms have been around for a long time, but have drawn much recent attention due to their ability to handle large-scale methods. Unlike gradient descent or steepest descent methods, in general block coordinate methods are not guaranteed to converge [Pow73]; for special cases, however, convergence has been proven [Pow73, LT93, Tse88, dE59, SS73, Tse93, Tse01, BT13, GS00]. Recently, there has been interest in randomized block coordinate descent, where the choice of block update order is randomized; see, for example, [Nes12, RT14] and [Wri15] for a survey.

2.3.3 Equivalence of the two methods

When block coordinate ascent with Gauss-Seidel ordering is applied to the dual projection problem (2.10), it is equivalent to Dykstra's algorithm [Han88, GM89, Tse93] for (2.9). One way to show this equivalence when $\mathcal{C}_k$ are closed convex cones is to rewrite (2.12) compactly as

$$s_k^i = \Pi_{\mathcal{C}_k^*}\left(-c - \sum_{j=1}^{k-1} s_j^i - \sum_{j=k+1}^{l} s_j^{i-1}\right) \tag{2.13}$$

which is exactly the coordinate ascent step for the dual problem (2.10), using Gauss-Seidel ordering. To verify, telescoping the s_k update in (2.12) by k, and

again by iteration gives

$$\sum_{j=1}^{k}(s_j^i - s_j^{i-1}) = x_k^i - x_0^i, \quad \text{and} \quad \sum_{j=1}^{l} s_j^i = x_0^{i+1} - c, \tag{2.14}$$

respectively. It then follows that

$$s_k^{i-1} - x_{k-1}^i = -c - \sum_{j=1}^{k-1} s_j^i - \sum_{j=k+1}^{l} s_j^{i-1}.$$

Finally, using the simple relation for dual cones from (2.1),

$$s_k^i = \Pi_{\mathcal{C}_k}(x_{k-1}^i - s_k^{i-1}) - (x_{k-1}^i - s_k^{i-1}) = \Pi_{\mathcal{C}^*}(s_k^{i-1} - x_{k-1}^i), \tag{2.15}$$

and (2.13) follows. This shows that the helper variables s_k in Dykstra's algorithm are exactly the dual variables in (2.9). Additionally, the equivalence allows us to use convergence results and stopping criteria interchangeably between dual block coordinate ascent and Dykstra's algorithm.

Though this proof is specific for $\mathcal{C}_k$ conic sets, the properties extend to $\mathcal{C}_k$ any closed convex sets. In the more general case, the dual update is

$$s_k^i = \mathbf{prox}_{\delta^*_{\mathcal{C}_k}}\left(-c - \sum_{j=1}^{k-1} s_j^i - \sum_{j=k+1}^{l} s_j^{i-1}\right)$$

where $\delta^*_{\mathcal{C}}(x) = \sup_{x \in \mathcal{C}} x^T y$ is the support function of $\mathcal{C}$. This can be written in terms of projections on $\mathcal{C}_k$ using the Moreau decomposition (2.7)

$$s_k^i = -c - \sum_{j=1}^{k-1} s_j^i - \sum_{j=k+1}^{l} s_j^{i-1} - \Pi_{\mathcal{C}_k}\left(-c - \sum_{j=1}^{k-1} s_j^i - \sum_{j=k+1}^{l} s_j^{i-1}\right).$$

2.3.4 Convergence and stopping condition

We can give a convergence rate by applying a recent result by Beck and Tetruashvili on block coordinate gradient projection algorithms [BT13, Th. 6.3]: since l is finite, we have for any k the relation $f(s_k^i) - f(s_k^\star) \leq \tau/i$, where τ is a constant, and is proportional to $L_{\max}$, the maximum Lipschitz constant of the gradient of f corresponding to each variable block. From (2.14) it then follows that

$$\|x_l^i - x_l^\star\|_2 = \|\sum_k s_k^i - \sum_k s_k^\star\|_2 = O(1/\sqrt{i}).$$

In practice, in cases where $L_{\max}$ is much smaller than L, block coordinate ascent methods may converge much more quickly than vanilla gradient methods.

In general, deciding when to terminate the block coordinate ascent method can be difficult, since the iterates may remain constant for several successive iterations [Pow73]. However, since the block coordinate ascent applied to this problem is equivalent to Dykstra's method, which is guaranteed to converge, an effective stopping condition is available. Note that by (2.15), s_k^i is always dual feasible. Additionally, by the second part of (2.14), x_l^i and s_k^i, $k = 1, \ldots, l$, always satisfy stationarity conditions. The sequence has converged to an optimal point when primal feasibility and complementary slackness is attained.

The stopping condition proposed in [Ray05, eq. (13)] is based on measuring the residual $r^i = \sum_k \|s_k^i - s_k^{i-1}\|_2$, *i.e.* the difference between the values of s_k at the end of two successive cycles. It can be shown that if $r^i = 0$ then the iterates x_l^i have remained constant during cycle i. This can be seen from the expression

$$\|x_k^i - x_0^i\|_2 = \|\sum_{j=1}^k s_j^i - \sum_{j=1}^k s_j^{i-1}\|_2 \leq \sum_{j=1}^l \|s_j^i - s_j^{i-1}\|_2 = \|r^i\|_2.$$

In this case,

$$x_0^i = x_1^i = \ldots = x_l^i \in \mathcal{C}_1 \cap \cdots \cap \mathcal{C}_l$$

is primal feasible. Additionally, at each iteration, for $w = x_{k-1}^i - s_k^{i-1}$, the iterates

$$x_k^i = \Pi_{\mathcal{C}_k}(w) \text{ and } s_k^i = \Pi_{\mathcal{C}_k}(w) - w = -\Pi_{-\mathcal{C}_k^*}(w),$$

and are orthogonal ($(x_k^i)^T s_k^i = 0$). In the case that x_k^i is constant for all k, then

$$\sum_{k=1}^{l} (s_k^i)^T (x_l^i) = 0$$

and optimality is achieved. This observation implies that x_l^i and $s_k^i,\ k = 1, \ldots, l$, are optimal if $r^i = 0$.

2.4 Proximal splitting methods

In this section we investigate a class of first-order methods that use the proximal operator to handle nondifferentiable functions. Although the history of these methods extends to the 1950s, their popularity in large-scale convex optimization is more recent than interior-point methods. The ideas follow from the success of splitting methods in linear algebra, which reduce the complexity of solving large linear systems into iteratively solving simpler systems. Similarly, proximal splitting methods solve problems of the form

$$\underset{x}{\text{minimize}} \quad f(x) + g(x)$$

where f and g are closed convex functions, by solving over the $f(x)$ and $g(x)$ terms separately. Examples of splitting methods include the forward-backward and double-backward methods [Pas79], and Douglas-Rachford and Peaceman-Rachford [LM79] methods. In this work, we will only deal with the forward-backward (proximal gradient) algorithm and Douglas-Rachford methods (discussed in the next two sections), and refer to the surveys [BC11, EB92, BPC11,

PB13, DY14] for more details on the other splitting algorithms.

An instance of the forward-backward algorithm is the gradient projection method, first presented by Goldstein [Gol64] and Levitin and Polyak [LP66], which is known to converge at a rate of $O(1/i)$. (See [BT09].) Nesterov's first method [Nes83] adds an acceleration scheme to this method, giving an overall $O(1/i^2)$ rate of convergence, where i is the number of iterations needed to reach a desired tolerance. The Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) [BT09] generalizes the projected gradient method to a proximal gradient method, which handles problems of the form $f(x) + g(x)$ where one of the two terms may be nondifferentiable. Interestingly, though FISTA is a popular algorithm in many fields, the convergence of iterates to a fixed point (for functions that are not strongly convex) has only recently been proved [CD14].

More generally, the Douglas-Rachford method solves problems of the type $f(x)+g(x)$ where both f and g may be nondifferentiable. Both Spingarn's method of partial inverses [Spi83, Spi85], and the alternating direction method of multipliers (ADMM) [BPC11] have been shown to be variations of the Douglas-Rachford splitting. Several important works [CP11b] and surveys [EB92, PB13] have been written about these methods; see also [BC11] on more general operator theory. Douglas-Rachford splitting methods are preferred for their generality; unlike proximal gradient methods, they make no assumptions on differentiability, and, unlike the proximal gradient method, have no restrictions on step size. Additionally, they are easy to implement in a decomposed manner, where important steps may be computed concurrently and in parallel.

2.4.1 Proximal gradient method

Consider the problem

$$\underset{x}{\text{minimize}} \quad f(x) + g(x) \tag{2.16}$$

where $f(x)$ is differentiable and the proximal operator of $g(x)$ is easy to evaluate. Then (2.16) can be solved using the proximal gradient method, with the iteration scheme

$$x^i = \mathbf{prox}_{tg}\left(x^{i-1} - t\nabla f(x^{i-1})\right). \tag{2.17}$$

where $t > 0$ is a step size. If $g(x) = \delta_{\mathcal{C}}(x)$, then (2.17) is the projected gradient method: [Pol87, §7.2]

$$x^i = \Pi_{\mathcal{C}}\left(x^{i-1} - t\nabla f(x^{i-1})\right).$$

When $\mathcal{C}$ is a conic set, the projections on $\mathcal{C}$ can also be expressed in terms of projections on $\mathcal{C}^*$ via the identity $u = \Pi_{\mathcal{C}}(u) - \Pi_{\mathcal{C}^*}(-u)$. (See (2.1).)

Convergence A standard convergence result states that for a fixed step size $t = 1/L$ (where ∇f is L-Lipschitz continuous) the sequence z^i converges to a minimizer of f over $\mathcal{C}$, even when the minimizers are not unique [Pol87, Ch 7, Th 1]. (This is the same step size bound as required in the gradient method.) Moreover the dual optimality gap decreases as

$$f(x^i) - f(x^\star) \leq \frac{L}{2i}\|x^0 - x^\star\|^2, \tag{2.18}$$

where $x^\star$ is any optimal solution [BT09, Th. 3.1]. If $f(x)$ is also μ-strongly convex, then the convergence rate is linear

$$\|x^{k+1} - x^\star\|_2 \leq c\|x^k - x^\star\|_2, \quad c = \max\{|1 - tL|, |1 - t\mu|\},$$

which is also shown in [Pol87, Ch 7, Th 1]

Interpretation as splitting method We can consider the minimization of $f(x) + g(x)$ as finding x satisfying the optimality condition

$$0 \in \partial f(x) + \partial g(x) = (I + \partial g(x)) - (I - \partial f(x)).$$

Then the proximal gradient algorithm can be interpreted as a splitting method that deals with these two terms separately

$$x^+ = (I + t\partial g)^{-1}(I - t\partial f)(x)$$

and the operators $(I - t\partial f)$ and $(I + t\partial g)^{-1}$ are a forward and backward step, respectively (hence the name forward-backward algorithm). It can be shown that if $0 < t < 1/L$ then $I - t\partial f$ is firmly nonexpansive. Since $\mathbf{prox}_{tg} = (I + t\partial g)^{-1}$ is also firmly nonexpansive, then the compound operator is also firmly nonexpansive, and (2.16) converges weakly to a fixed point. To see that all fixed points are also optimal points, note that

$$x^+ = x \Rightarrow (I + t\partial g)(x) = (I - t\partial f)(x) \iff 0 \in \partial f(x) + \partial g(x).$$

This is a quick alternative proof for the convergence of the forward-backward algorithm.

Acceleration An important advantage of the proximal gradient method is the availability of Nesterov-type accelerations [Nes83, Nes04, Tse08, BT09]. We will use Beck and Teboulle's accelerated proximal gradient method, widely known under the acronym FISTA [BT09]; see also [Tse08]. This is a generalization of Nesterov's first accelerated proximal gradient method [Nes83], used when $g(x) = \delta_{\mathcal{C}}(x)$.

FISTA applies a proximal gradient update after an extrapolation step:

$$x^i = \mathbf{prox}_{tg}(v^i - t\nabla f(v^i)) \quad \text{where } v^i = x^{i-1} + \frac{i-2}{i+1}(x^{i-1} - x^{i-2})$$

(with the assumption $x^{-1} = x^0$, so the first iteration is the standard proximal gradient update). The same step size $1/L$ is used as in the proximal gradient method. The accelerated proximal gradient algorithm has the same complexity per iteration as the basic algorithm. The iterates x^i can be shown to converge, even when the optimal solutions are not unique. This is discussed in the recent paper [CD14]. The dual optimality gap decreases as $1/i^2$:

$$f(x^i) - f(x^\star) \leq \frac{2L}{(i+1)^2}\|x^0 - x^\star\|_2^2,$$

where $x^\star$ is any optimal solution [BT09, Th. 4.4]. The $O(1/i^2)$ convergence rate in optimal value is an improvement over the $O(1/i)$ rate of the proximal gradient method.

Stopping conditions Various stopping criteria can be used for these algorithms. For example, one can bound the error with which the iterates satisfy the optimality condition

$$x = \mathbf{prox}_{tg}(x - t\nabla f(x)) \iff 0 \in \nabla f(x) + \partial g(x) \tag{2.19}$$

for problem (2.16), where t is any positive number. In the proximal gradient algorithm, at each iteration we have

$$x^{i-1} = \mathbf{prox}_{tg}(x^{i-1} - t\nabla f(x^{i-1})) + r^i$$

with $r^i = x^{i-1} - x^i$. In the fast proximal gradient algorithm we have

$$v^i = \mathbf{prox}_{tg}(v^i - t\nabla f(v^i)) + r^i$$

with $r^i = v^i - x^i$. In both cases, $r^i = 0$ implies optimality (2.19). This suggests using stopping conditions

$$\frac{\|x^i - x^{i-1}\|_2}{\max\{\|x^i\|_2, 1\}} \leq \epsilon, \qquad \frac{\|v^i - x^i\|_2}{\max\{\|x^i\|_2, 1\}} \leq \epsilon \tag{2.20}$$

for the proximal gradient method and the fast proximal gradient method, respectively.

2.4.2 Douglas-Rachford method

The Douglas-Rachford method [LM79, EB92, BC11] is a popular method for solving

$$\underset{x}{\text{minimize}} \quad f(x) + g(x)$$

where $f(x)$ and $g(x)$ are both closed, convex, and possibly nonsmooth functions. The method is efficient when the proximal operators of both f and g are easy to compute. It uses the iteration scheme

$$\begin{aligned} x^{i+1} &= \mathbf{prox}_{tf}(z^i) \\ y^{i+1} &= \mathbf{prox}_{tg}(2x^{i+1} - z^i) \\ z^{i+1} &= z^i + \rho(y^{i+1} - x^{i+1}). \end{aligned} \tag{2.21}$$

The algorithm depends on two algorithm parameters: a positive *steplength* t and a *relaxation parameter* ρ_k, which can change at each iteration but must remain in an interval $(\rho_{\min}, \rho_{\max})$ with $0 < \rho_{\min} < \rho_{\max} < 2$. More details on the Douglas-Rachford method and its applications can be found in [Eck94, CP07, BC11, PB13].

Convergence It can be shown that if f and g are closed convex functions, with **ri dom** $f \cap$ **ri dom** $g \neq \emptyset$, and the problem (2.23) has a solution, then the sequences x^i and y^i converge to a solution [BC11, corollary 27.2]. Recent results by Davis and Yin [DY14, Th. 7] also give a convergence rate for the objective value: it is shown that

$$|(f(x^i) + g(x^i)) - (f(x^\star) + g(x^\star))| = o(1/\sqrt{i}).$$

The bound is improved to $O(1/i)$ in [PSB14, Thm. 3] when $f(x)$ is a convex quadratic, *i.e.* $f(x) = x^T Q x + q^T x$ where $Q \succeq 0$.

Interpretation as fixed-point iteration The three steps in the Douglas-Rachford iteration can be combined into a single update

$$z^i = z^{i-1} - \rho G(z^{i-1})$$

with the operator G defined as

$$G(z) = \mathbf{prox}_{tf}(z) - \mathbf{prox}_{tg}(2\mathbf{prox}_{tf}(z) - z).$$

For $\rho = 1$ this is a fixed-point iteration $z^i = z^{i-1} - G(z^{i-1})$ for solving $G(z) = 0$; for other values of ρ it is a fixed-point iteration with relaxation (underrelaxation for $\rho < 1$, overrelaxation for $\rho > 1$).

The vector z is a zero of G if and only if

$$\text{for } x = \mathbf{prox}_{tf}(z), \text{ then } x = \mathbf{prox}_{tg}(2x - z).$$

Written as optimality conditions, they give the result

$$\frac{z - x}{t} \in \partial f(x), \quad \text{and} \quad \frac{x - z}{t} \in \partial g(x) \tag{2.22}$$

respectively. By positive homogeneity of the subdifferential (2.5), this implies $0 \in \partial(f+g)(x)$, and thus x is a minimizer of $f(x)+g(x)$. Conversely, suppose that x minimizes $f(x)+g(x)$, and therefore $0 \in \partial f(x) + \partial g(x)$. Then there exists a vector z satisfying (2.22). Assume there is a method for finding such a z. Then condition (2.22) is equivalent to $x = \mathbf{prox}_{tf}(z) = \mathbf{prox}_{tg}(2x-z)$, and z is a fixed point of $G(z)$.

Stopping condition The Douglas-Rachford iteration reaches optimality for x when $G(z)=0$, so a reasonable stopping condition is when $\|z^+ - z\|_2$ is small, or equivalently when $\|y-x\|_2$ is smaller than some tolerance. This suggests using the stopping condition

$$\frac{\|y^i - x^i\|_2}{\max\{\|x^i\|_2, \|y^i\|_2, 1\}} \leq \epsilon.$$

2.4.3 Spingarn's method

A useful instance of the Douglas-Rachford method is when $g(x)$ is the indicator function of a subspace $\mathcal{V}$; *i.e.*, we solve

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & x \in \mathcal{V}. \end{array} \tag{2.23}$$

The Douglas-Rachford method specialized to this problem is also known as Spingarn's method, or the method of partial inverses [Spi83, Spi85, EB92]. The algorithm starts at an arbitrary z^0 and repeats the following iteration:

$$\begin{aligned} x^{i+1} &= \mathbf{prox}_{tf}(z^i) \\ y^{i+1} &= \Pi_{\mathcal{V}}(2x^{i+1} - z^i) \\ z^{i+1} &= z^i + \rho(y^{i+1} - x^{i+1}). \end{aligned} \tag{2.24}$$

Interpretation as fixed-point iteration We can also interpret (2.21) as a fixed point iteration for $\rho = 1$ where

$$G(z) = \mathbf{prox}_{tf}(z) - \Pi_{\mathcal{V}}(2\mathbf{prox}_{tf}(z) - z). \tag{2.25}$$

Defining $x = \mathbf{prox}_{tf}(z)$, we see that $G(z) = 0$ if and only if $x = \Pi_{\mathcal{V}}(2x - z)$. This is satisfied if and only if $x \in \mathcal{V}$ and $x - z \in \mathcal{V}^{\perp}$. From the optimality conditions of $\mathbf{prox}_{tf}$, if we define $v = t^{-1}(z - x)$ then it follows that $v \in \partial f(x)$. This gives the optimality conditions for (2.23) as

$$x \in \mathcal{V}, \qquad v \in \mathcal{V}^{\perp}, \qquad v \in \partial f(x). \tag{2.26}$$

Stopping condition From step 1 in the algorithm (2.24) and the definition of the proximal operator we see that the vector $v^i = t^{-1}(z^{i-1} - x^i)$ satisfies $v^i \in \partial f(x^i)$. If we define

$$r_{\mathrm{p}}^i = \Pi_{\mathcal{V}}(x^i) - x^i, \qquad r_{\mathrm{d}}^i = -\Pi_{\mathcal{V}}(v^i)$$

then

$$x^i + r_{\mathrm{p}}^i \in \mathcal{V}, \qquad v^i + r_{\mathrm{d}}^i \in \mathcal{V}^{\perp}, \qquad v^i \in \partial f(x^i).$$

The vectors r_{p}^i and r_{d}^i can be interpreted as primal and dual residuals in the optimality conditions (2.26), evaluated at the approximate primal and dual solutions x^i, v^i. One can also note from (2.25) that

$$\begin{aligned} G(z^{i-1}) &= x^i - \Pi_{\mathcal{V}}(x^i - tv^i) \\ &= (x^i - \Pi_{\mathcal{V}} x^i) - t\Pi_{\mathcal{V}}(v^i) \\ &= -r_{\mathrm{p}}^i - tr_{\mathrm{d}}^i \end{aligned}$$

and since the two terms on the right-hand side are orthogonal, then

$$\|G(z^{i-1})\|_2^2 = \|r_{\mathrm{p}}^i\|_2^2 + t^2\|r_{\mathrm{d}}^i\|_2^2. \tag{2.27}$$

A simple stopping criterion is to terminate when

$$\frac{\|r_{\mathrm{p}}^i\|_2}{\max\{1.0, \|x^i\|_2\}} \leq \epsilon_{\mathrm{p}} \qquad \text{and} \qquad \frac{\|r_{\mathrm{d}}^i\|_2}{\max\{1.0, \|v^i\|_2\}} \leq \epsilon_{\mathrm{d}} \tag{2.28}$$

for some relative tolerances ϵ_{p} and ϵ_{d}.

Choice of steplength In the standard convergence analysis of the Douglas-Rachford algorithm the parameter t is assumed to be an arbitrary positive constant [EB92]. However the efficiency in practice is greatly influenced by the steplength choice and several strategies have been proposed for varying t during the algorithm [HYW00, WL01, HLW03]. As a guideline, it is often observed that the convergence is slow if one of the two components of $\|G(z^{i-1})\|_2^2$ in (2.27) decreases much more rapidly than the other, and that adjusting t can help control the balance between the primal and dual residuals. A simple strategy is to take

$$t^{i+1} = \begin{cases} t^i\tau^i & \sigma^i > \mu \\ t^i/\tau^i & \sigma^i < 1/\mu \\ t^i & \text{otherwise,} \end{cases} \tag{2.29}$$

where σ^i is the ratio of relative primal and dual residuals,

$$\sigma^i = \frac{\|r_{\mathrm{p}}^i\|_2}{\|x^i\|_2} \cdot \frac{\|v^i\|_2}{\|r_{\mathrm{d}}^i\|_2},$$

and τ^i and μ are parameters greater than one. A numerical evaluation of this strategy is given in section 5.5.1.

2.5 Linear conic optimization

First-order methods are important historically, and are readily used to solve large, sparse problems. For smaller problems, often convex optimization problems are solved by first reformulating it as a linear conic optimization problem and solving it with an interior-point method. This is the approach used in software packages like CVX and Yalmip, for example. Linear conic optimization problems are written in form

$$\begin{array}{ll} \underset{x}{\text{minimize}} & c^T x \\ \text{subject to} & Ax = b \\ & x \in \mathcal{C} \end{array} \tag{2.30}$$

where $x \in \mathbb{R}^n$ is a variable and $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$ are problem parameters. The set $\mathcal{C}$ is a proper convex cone. If $\mathcal{C} = \{x \mid x_i \geq 0\}$ the nonnegative orthant, then (2.30) is a *linear program (LP)*. If $\mathcal{C}$ is the vectorized PSD cone $\mathbb{S}^p_+$, then (2.30) is a *linear SDP*.

The Lagrangian dual of problem (2.30) is

$$\begin{array}{ll} \underset{s,y}{\text{maximize}} & b^T y \\ \text{subject to} & s + A^T y = c \\ & s \in \mathcal{C}^*, \end{array}$$

where $\mathcal{C}^*$ is the dual cone of $\mathcal{C}$, and the dual variables $y \in \mathbb{R}^m$ and $s \in \mathbb{R}^n$ correspond to the affine and conic constraints in (2.30). If both the primal and dual optima are attained, and if there exists a feasible $x \in \mathbf{int}\ \mathcal{C}$ (the interior of $\mathcal{C}$) where also $Ax = b$, then by Slater's condition, strong duality holds, and at optimality the primal and dual objectives agree ($c^T x^\star = b^T y^\star$) [Roc70, Th 28.2, 28.4]. In this case, we have

$$Ax^\star = b, \quad x^\star \in \mathcal{C}, \quad s^\star + A^T y^\star = c, \quad s^\star \in \mathcal{C}^*, \quad (s^\star)^T x^\star = 0$$

2.5.1 History of linear and conic optimization

Linear conic optimization grew as a generalization of linear programming, whose history begins around the late 1930s with Kantorovich [Kan39] in economics and von Neumann and Morgenstern [NM53] in game theory. An early practical algorithm for solving linear programs is the simplex method, given by Dantzig in 1947 (see [Dan63]), which did well in practice but did not have a polynomial worst-case runtime. Karmarkar's interior-point method in 1984 is credited as the first practically efficient and provably polynomial time method for linear programming [Kar84]. A variation of his method is the Mehrotra's predictor-corrector method [Meh92], and is the basis of most interior-point solvers today.

The extension of interior-point methods from linear programs to general conic programs is done by extending the barrier functions, which penalize the objective to enforce the conic constraint. In the 1990s, Nesterov and Nemirovski [NN94] provided a series of self-concordant barrier functions for general convex sets, effectively extending Karmakar's method to conic programming. A similar extension and proof was offered by Alizadeh [Ali95] for SDPs. Since then, interior-point methods have become a commonplace tool for conic optimization because of their practical success, converging to high-accuracy solutions in about 25-50 iterations. For this reason, general purpose solvers like CVX [GB12] or Yalmip [Lof04] often solve general nonlinear convex optimization problems by first converting them to linear conic programs (2.30), and solving them by invoking popular interior-point solvers. These include the commercial products Mosek [MOS02], and the open-source products Sedumi [Stu99] and SDPT3 [TTT02]. A detailed discussion of interior-point methods can be found in [Wri97, Ali95, BV04].

2.5.2 Solving a linear conic problem

An interior-point method converges to the optimal solution of (2.30) by iteratively solving the following linear system

$$\begin{bmatrix} H & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} r_x \\ r_y \end{bmatrix}. \tag{2.31}$$

Here, H is a positive definite scaling matrix that depends on the algorithm used, the cone $\mathcal{C}$, and the current primal and dual iterates in the algorithm. The system (2.31) is often called the *Karush-Kuhn-Tucker (KKT)* equation. We will assume that $H = \nabla^2\phi(w)$ where $\phi(w)$ is a logarithmic barrier function for $\mathcal{C}$ and w is some point in **int** $\mathcal{C}$. This assumption is sufficiently general to cover path-following methods based on primal scaling, dual scaling, and the Nesterov-Todd primal-dual scaling.

In most implementations, the KKT equation is solved by eliminating Δx and solving a smaller system, called the *Schur complement system*

$$AH^{-1}A^T\Delta y = AH^{-1}r_x - r_y. \tag{2.32}$$

Generally, an interior-point method solves a problem efficiently if forming and factoring the term $AH^{-1}A^T$ can be done efficiently. This is true, for example, in linear programming if the matrix A is sparse; since H is diagonal, then $AH^{-1}A$ is sparse if AA^T is sparse. Unfortunately, for nonlinear optimization (and specifically for semidefinite programming) H^{-1} is dense in general, so $AH^{-1}A^T$ is dense regardless of the sparsity of A, and solving (2.32) has a high computational complexity. In chapter 5, We combat this issue using special reformulations of (2.30).

CHAPTER 3

Matrix cones

In this section we review some key theoretical results from matrix algebra. First, we discuss the important properties and theorems pertaining to the set of Euclidean distance matrices, with no assumptions on sparsity (section 3.1). We then delve into sparse matrix theory, focusing on the graph theoretic results used in the clique decomposition theorems (section 3.2). Finally, we give a minimum rank matrix completion algorithm (section 3.3) to demonstrate that matrix solutions with chordal sparsity patterns can be completed to the dense form systematically and efficiently, if needed. Most of the graph theorems are well known and proven in several texts (*i.e.* [BP93, VA15]).

3.1 Euclidean distance matrices

A *Euclidean distance matrix (EDM)* is a matrix X where each element X_{ij} is the squared Euclidean distance between points u_i and u_j, for some set of points u_i, $i = 1, \ldots, p$. We denote the set of $p \times p$ EDMs as $\mathbb{D}^p$. For any $X \in \mathbb{D}^p$, all elements of X must be nonnegative, and all diagonal elements zero. Distance matrices are also invariant under translations and rotations of the points they represent. Specifically, EDMs can be characterized by matrix algebra using Schoenberg's condition [Sch35, Sch38], which says that EDMs are negative semidefinite in the nullspace of $\mathbf{1}^T$:

$$X \in \mathbb{D}^p \iff \mathbf{diag}(X) = 0 \text{ and } z^T X z \leq 0, \ \forall z \text{ where } z^T \mathbf{1} = 0.$$

Schoenberg's condition can be written in matrix form as $V^T X V \preceq 0$, where V is an orthonormal $p \times p-1$ matrix whose columns span the nullspace of $\mathbf{1}^T$, for example,

$$V = \frac{1}{p+\sqrt{p}} \begin{bmatrix} (1+\sqrt{p})\mathbf{1}^T \\ \mathbf{1}\mathbf{1}^T - (p+\sqrt{p})I \end{bmatrix}. \tag{3.1}$$

We use the notation

$$\begin{aligned} \mathbb{D}_0^p &= \{X \in \mathbb{S}^p \mid z^T X z \leq 0, \quad \forall z \mid \mathbf{1}^T z = 0\} \\ &= \{X \in \mathbb{S}^p \mid V^T X V \preceq 0\} \end{aligned}$$

and the EDMs are the matrices in $\mathbb{D}_0^p$ with zero diagonal. From this it is clear that the sets $\mathbb{D}^p$ and $\mathbb{D}_0^p$ are both intersections of closed convex cones, and therefore are both closed convex cones. There are two noticeable differences. The cone $\mathbb{D}^p$ has empty interior in $\mathbb{S}^p$, while the interior of $\mathbb{D}_0^p$ is nonempty (it contains the matrix $X = -I$). Additionally, $\mathbb{D}^p$ is pointed (since it contains only nonnegative matrices), while $\mathbb{D}_0^p$ is not, as it contains the subspace $\{a\mathbf{1}^T + \mathbf{1}a^T \mid a \in \mathbb{R}^p\}$.

Embedding dimension An EDM has an *embedding dimension* r if it is an EDM for some set of points u_i, $i = 1, \ldots, p$, where $u_i \in \mathbb{R}^r$. Intuitively, it makes sense that a matrix in $\mathbb{D}^p$ can have at most an embedding dimension of $p-1$, since two points can be embedded in a line, three points in a plane, and so forth. Young and Householder [YH38] showed that the embedding dimension of $X \in \mathbb{D}^p$ is exactly the rank of $W = X - Xe_1\mathbf{1}^T - \mathbf{1}e_1^T X$, where $e_1 = (1, 0, \ldots, 0)$ is a unit vector of length p. Since for any $X \in \mathbb{D}^p$,

$$We_1 = Xe_1 - Xe_1\mathbf{1}^T e_1 - \mathbf{1}e_1^T Xe_1 = X_{11}\mathbf{1} = 0$$

then the rank of W (and therefore the embedding dimension of X) is upper bounded by $p-1$.

Linear mappings between $\mathbb{D}^p$ and $\mathbb{S}_+^p$ A PSD matrix $G \succeq 0$ is a *Gram matrix* for a set of points $u_i, i = 1, \ldots, p$, where

$$G = UU^T \text{ where } U = [u_1, \ldots, u_p]^T .$$

Specifically, for any $G \succeq 0$ with rank $r < p$, G is the Gram matrix for some set of points $u_i \in \mathbb{R}^r$. Under a linear transformation, a Gram matrix becomes an EDM; specifically, if G is the Gram matrix for $u_i, i = 1, \ldots, p$, then by definition,

$$D_{ij} = \|u_i - u_j\|_2^2 = G_{ii} - 2G_{ij} + G_{jj},$$

or, written in matrix form, $T(G) = D$ where

$$T(X) = \mathbf{diag}(X)\mathbf{1}^T + \mathbf{1}\mathbf{diag}(X)^T - 2X. \tag{3.2}$$

Note that $T : \mathbb{S}_+^p \to \mathbb{D}^p$ is onto but not one-to-one, since for any $u \in \mathbb{R}^p$,

$$T(G + u\mathbf{1}^T + \mathbf{1}u^T) = T(G).$$

(Physically, this means that distance matrices are invariant to translations in the points u_i, whereas Gram matrices are not.) Therefore T is not invertible unless its domain is restricted. In this case, Johnson and Tarazaga [JT95] give the family of inverses as

$$G_u(D) = -\frac{1}{2}\left(I - \mathbf{1}u^T\right) D (I - u\mathbf{1}^T) \tag{3.3}$$

where the range of $G_u(D)$ is the set of Gram matrices with u in its nullspace. It can be shown that for $D \in \mathbb{D}^p$, $T(G_u(D)) = D$ and for $X \succeq 0$, if $Xu = 0$, then $G_u(T(X)) = X$. Additionally, the embedding dimension of D is exactly the rank of $G_u(D)$, which is at most $p - 1$.

Pseudoinverse of $T(X)$ It can be shown (in [KW12], for example) that taking $c = (1/p)\mathbf{1}$, G_c is the unique Moore-Penrose pseudoinverse of T. Taking this choice of u also has a visual interpretation: the mapping

$$G_c(D) = -\frac{1}{2}\left(I - \frac{1}{p}\mathbf{1}\mathbf{1}^T\right) D \left(I - \frac{1}{p}\mathbf{1}\mathbf{1}^T\right)$$

gives the Gram matrix for points centered at the origin. (That is, $G_c(D) = UU^T$ where $U^T\mathbf{1} = 0$.)

Projection on EDMs Formulas for projecting on $\mathbb{D}_0^p$ can be found in [HW88, GM89, GHH90]. Define $Q = [\ V \quad (1/\sqrt{p})\mathbf{1}\ ]$ where V is as defined in (3.1). In [HW88, GHH90] the projection of a matrix $D \in \mathbb{S}^p$ on $\mathbb{D}_0^p$ is computed directly, as the solution of

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|X - D\|_F^2 \\ \text{subject to} & V^T X V \preceq 0. \end{array} \tag{3.4}$$

Since Q is orthogonal, the problem is equivalent to

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|Q^T X Q - Q^T D Q\|_F^2 \\ \text{subject to} & V^T X V \preceq 0 \end{array}.$$

The objective can be expanded to

$$Q^T X Q - Q^T D Q = \frac{1}{\sqrt{p}} \begin{bmatrix} p(V^T X V - V^T D V) & \sqrt{p}(V^T X \mathbf{1} - V^T D \mathbf{1}) \\ \sqrt{p}(\mathbf{1}^T X V - \mathbf{1}^T D V) & \mathbf{1}^T X \mathbf{1} - \mathbf{1}^T D \mathbf{1}. \end{bmatrix}$$

From this it is clear that at the optimum, the $1,1$ block of $Q^T X Q$ is the projection of $V^T D V$ on the negative semidefinite cone and the other blocks are equal to the

corresponding blocks of $Q^T DQ$. The solution of (3.4) is therefore

$$\Pi_{\mathbb{D}_0^p}(D) = \frac{1}{\sqrt{p}} Q \begin{bmatrix} -\sqrt{p}\left(\Pi_+(-V^T DV)\right) & V^T D\mathbf{1} \\ \mathbf{1}^T Dv & \frac{1}{\sqrt{p}}\mathbf{1}^T D\mathbf{1} \end{bmatrix} Q^T.$$

An alternative method in [GM89] computes the projection indirectly, via the projection on the dual cone of $\mathbb{D}_0^p$ and the formula

$$\Pi_{\mathbb{D}_0^p}(D) = D - \Pi_{-(\mathbb{D}_0^p)^*}(D). \tag{3.5}$$

(See figure 2.1.) The dual cone is $(\mathbb{D}_0^p)^* = \{VZV^T \mid Z \preceq 0\}$, so the projection of D on $-(\mathbb{D}_0^p)^*$ is the solution of

$$\begin{array}{ll} \underset{Z}{\text{minimize}} & \|VZV^T - D\|_F^2 \\ \text{subject to} & Z \succeq 0. \end{array}$$

Equivalently, since Q is orthogonal, we solve

$$\begin{array}{ll} \underset{Z}{\text{minimize}} & \|Q^T(VZV^T - D)Q\|_F^2 = \left\| \begin{bmatrix} Z - V^T DV & -V^T De \\ -e^T DV & -e^T De \end{bmatrix} \right\|_F^2 \\ \text{subject to} & Z \succeq 0 \end{array}$$

where $e = (1/\sqrt{p})\mathbf{1} \in \mathbb{R}^p$. The solution is $Z = \Pi_+(V^T DV)$ and substituting in (3.5) gives

$$\Pi_{\mathbb{D}_0^p}(D) = D - V\Pi_+(V^T DV)V^T.$$

3.2 Decomposition of sparse matrix cones

We now review concepts pertaining to sparse matrix cones. Specifically, a $p \times p$ symmetric matrix X has *sparsity pattern* E if $X_{ij} = 0$ for all $i \neq j, \{i, j\} \notin E$,

Figure 3.1: *Small example.* Left: a sparsity pattern for a set of 6×6 matrices. Note that by default the diagonal is included in the pattern. Right: the corresponding undirected graph, with 6 vertices, and edges between vertices corresponding to nonzeros in the sparsity pattern.

and we denote this set as $\mathbb{S}^p_E$. There are several ways to store a sparse matrix in memory. For example, in triplet notation, the matrix is stored as three arrays: the row indices, the column indices, and the values of the nonzeros. In this case, the cardinality of E directly corresponds to the amount of memory needed to store the matrix (proportional to $|E| + p$).

The sparsity pattern of a sparse symmetric matrix X can be interpreted as the edge set for an undirected graph $G = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, \ldots p\}$ are the graph vertices and $\mathcal{E} = E$ are the undirected edges between vertices. (For example, see figure 3.1.) When it is unambiguous, we can use the shorthand $G_E = G(\{1, \ldots, p\}, E)$, or just refer to the graph as E.

3.2.1 Cliques and chordal graphs

Adjacent and simplicial vertices For a general undirected graph $G = (\mathcal{V}, \mathcal{E})$, two vertices $u, v \in \mathcal{V}$ are *adjacent* if $\{u, v\} \in \mathcal{E}$. The set of vertices adjacent to v is denoted

$$\mathbf{adj}_G(v) = \{u \in \mathcal{V} \mid \{u, v\} \in \mathcal{E}\}.$$

Note that $\mathbf{adj}(v)$ does not include v; we denote $\mathbf{cadj}_G(v) = \{v\} \cup \mathbf{adj}_G(v)$. In cases where it is unambiguous, we interchange the notations $\mathbf{adj}_G(v) = \mathbf{adj}_E(v) = \mathbf{adj}(v)$, and $\mathbf{cadj}_G(v) = \mathbf{cadj}_E(v) = \mathbf{cadj}(v)$. We say a graph is *complete* if there

exists an edge between every pair of vertices. A vertex is *simplicial* if the subgraph of G induced by the vertices in $\mathbf{cadj}(v)$ is complete. A set of vertices $\mathcal{V}_c \subseteq \mathcal{V}$ forms a *clique* if the subgraph of G induced by the vertices in $\mathcal{V}_c$ is complete. A clique that is not a strict subset of another clique is a *maximal clique*.

Paths and cycles In a graph G, a set of vertices $\{v_1, v_2, \ldots, v_k\}$ form a *path* if $\{v_i, v_{i+1}\} \in \mathcal{E}$ for $i = 1, \ldots, k-1$. If, in addition, $\{v_1, v_k\} \in \mathcal{E}$ then these vertices also form a *cycle*. Two vertices v_1 and v_2 are *connected* if there exists a path from v_1 to v_2. A graph is connected if every pair of vertices are connected.

Chordal graphs A cycle $\{v_1, v_2, \ldots, v_k\}$ has a *chord* if, in addition to the edges (v_1, v_k) and $(v_i, v_{i+1}), i = 1, \ldots, k-1$, there is an additional edge connecting two vertices in the cycle. If a cycle has no chord, then the shortest path between any two vertices in the cycle is along the cycle. A graph (and corresponding sparsity pattern) is *chordal* if every cycle with four or more vertices has a chord. A *chordal extension* can be computed for graphs that are not chordal; G' is a chordal extension of $G = (\mathcal{V}, \mathcal{E})$ if $G' = (\mathcal{V}, \mathcal{E}')$ where $\mathcal{E} \subseteq \mathcal{E}'$ and G' is chordal. (Equivalently, we write $\mathcal{E}'$ is the chordal extension of $\mathcal{E}$.)

3.2.2 Clique decomposition theorems

We now show that when E is an edge set to a chordal graph, then certain sparse matrix cones with pattern E can be written in terms of smaller, overlapping conic constraints.

We use the following notation for representing subvectors and submatrices. If β is an index set (ordered subset) of $\{1, \ldots, p\}$, we define P_β as the $|\beta| \times p$-matrix

$$(P_\beta)_{ij} = \begin{cases} 1, & j = \beta(i), \\ 0, & \text{otherwise.} \end{cases} \tag{3.6}$$

Multiplying a vector with P_β selects the subvector indexed by β:

$$P_\beta x = (x_{\beta(1)}, x_{\beta(2)}, \ldots, x_{\beta(r)})$$

if β has r elements, denoted $\beta(1)$, $\beta(2)$, ..., $\beta(r)$. Similarly, the matrix P_β can be used to select a principal submatrix of a $p \times p$ matrix:

$$(P_\beta X P_\beta^T)_{ij} = X_{\beta(i)\beta(j)}.$$

The multiplication $x = P_\beta^T y$ of a $|\beta|$-vector y with the transpose of P_β gives a p-vector x with $P_\beta x = y$ and $x_j = 0$ for $j \notin \beta$. The operation $X = P_\beta^T Y P_\beta$ creates a $p \times p$ matrix X from an $r \times r$ matrix, with $P_\beta X P_\beta^T = Y$ and $X_{ij} = 0$ for $i \notin \beta$ or $j \notin \beta$. For example, if $p = 5$ and $\beta = \{1, 3, 4\}$ then

$$P_\beta = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \qquad P_\beta^T y = \begin{bmatrix} y_1 \\ 0 \\ y_2 \\ y_3 \\ 0 \end{bmatrix}, \qquad P_\beta^T Y P_\beta = \begin{bmatrix} Y_{11} & 0 & Y_{12} & Y_{13} & 0 \\ 0 & 0 & 0 & 0 & 0 \\ Y_{21} & 0 & Y_{22} & Y_{33} & 0 \\ Y_{31} & 0 & Y_{32} & Y_{33} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Sparse PSD completable matrices We first consider the set of sparse matrices with a PSD completion:

$$\Pi_E(\mathbb{S}_+^p) = \{\Pi_E(X) \mid X \in \mathbb{S}_+^p\}.$$

The set $\Pi_E(\mathbb{S}_+^p)$ is a closed convex cone, since if $\Pi_E(X) = 0$ then the diagonal of X is zero. Therefore $\Pi_E(X) = 0$ and $X \in \mathbb{S}_+^p$ imply $X = 0$. This is a sufficient condition for the projection $\Pi_E(\mathbb{S}_+^p)$ to be closed [Roc70, Th. 9.1]. The following theorem gives a characterization of $\Pi_E(\mathbb{S}_+^p)$ when the sparsity pattern E is chordal [GJS84].

Theorem 1 *Let E be a chordal sparsity pattern of order p, with cliques β_k, $k = 1, \ldots, m$. A matrix $X \in \mathbb{S}^p_E$ is in $\Pi_E(\mathbb{S}^p_+)$ if and only if*

$$P_{\beta_k} X P_{\beta_k}^T \succeq 0, \quad k = 1, \ldots, m \tag{3.7}$$

where the matrices P_{β_k} are as defined in (3.6).

Proof Suppose that for some $c \in \mathbb{R}^{|\beta|}$, $c^T(X_{\beta,\beta})c < 0$ for any index set β. Then define $d = P_\beta^T c$ and, if Z is any completion of X, then

$$d^T Z d = c^T(X_{\beta,\beta})c < 0.$$

Therefore the condition (3.7) is necessary. To show that (3.7) is sufficient, in section 3.3 we give a direct method for completing $X \in \Pi_E(\mathbb{S}^p_+)$ when E is chordal. □

If E is nonchordal, condition (3.7) is necessary but not sufficient. The following simple counterexample is given in Grone *et al.* [GJS84]:

$$B = \begin{bmatrix} 1 & 1 & ? & -1 \\ 1 & 1 & 1 & ? \\ ? & 1 & 1 & 1 \\ -1 & ? & 1 & 1 \end{bmatrix}.$$

Here E contains the edges for a cycle of length 4 with no chord, and it is clear that each 2×2 principal dense submatrix is PSD. To see that it has no PSD completion, note that the determinant of $B_{\beta,\beta}$ where $\beta = \{2, 3, 4\}$ is

$$\det\left(\begin{bmatrix} 1 & 1 & -1 \\ 1 & 1 & \alpha \\ -1 & \alpha & 1 \end{bmatrix}\right) = -(\alpha + 1)^2 < 0, \quad \forall \alpha.$$

Sparse positive semidefinite matrices Next, we consider the cone of sparse PSD matrices:

$$\mathbb{S}^p_{E,+} = \mathbb{S}^p_E \cap \mathbb{S}^p_+.$$

This is a closed convex cone for any E, since both $\mathbb{S}^p_E$ and $\mathbb{S}^p_+$ are closed convex cones. The following theorem from [GT84, AHM88] provides a characterization when the sparsity pattern is chordal.

Theorem 2 *Let E be a chordal sparsity pattern of order p, with cliques β_k, $k = 1, \ldots, m$. A matrix $S \in \mathbb{S}^p_E$ is PSD if and only if it can be expressed as*

$$S = \sum_{k=1}^{m} P_{\beta_k}^T Z_k P_{\beta_k} \tag{3.8}$$

with $Z_k \succeq 0$ for $k = 1, \ldots, m$.

Proof The theorem is equivalent to theorem 1 by duality. By theorem 3.7, the set of PSD completable matrices can be written as an intersection of cones

$$\Pi_E(\mathbb{S}^p_+) = \mathcal{K}_1 \cap \ldots \cap \mathcal{K}_m$$

where $\mathcal{K}_k = \{X \mid X_{\beta_k,\beta_k} \succeq 0\}$. Since $\mathcal{K}_1, \ldots, \mathcal{K}_m$ are closed, convex cones, then the dual cone is

$$\mathbb{S}^p_{E,+} = \sum_{k=1}^{m} \mathcal{K}_k^*$$

where $\mathcal{K}_k^* = \{P_{\beta_k}^T Z P_{\beta_k} \mid Z \succeq 0\}$, which is exactly condition (3.8). □

When E is not chordal, the condition in the theorem is sufficient for the positive semidefiniteness of S, but not necessary.

Sparse EDM completable matrices Finally, we consider the set of sparse matrices with an EDM completion. The cone of EDM completable matrices

$$\Pi_E(\mathbb{D}^p) = \{\Pi_E(X) \mid X \in \mathbb{D}^p\}$$

is a closed convex cone. To see this, first suppose the sparsity graph G_E is connected. If $\Pi_E(X) = 0$ for some $X \in \mathbb{D}^p$, with $X_{ij} = \|u_i - u_j\|_2^2$ for $i, j = 1, \ldots, p$, then $u_i = u_j$ for all $\{i, j\} \in E$. If the graph is connected, this implies that the position vectors u_i are all equal, *i.e.*, $X = 0$. Hence $\Pi_E(X) = 0$, $X \in \mathbb{D}^p$ only holds if $X = 0$. It then follows from [Roc70, Th. 9.1] that $\Pi_E(\mathbb{D}^p)$ is closed. Next, assume G_E has d connected components, with vertex sets $\alpha_1, \ldots, \alpha_d \subseteq \{1, 2, \ldots, p\}$. For $k = 1, \ldots, d$, let $E_k = \{\{i, j\} \mid \{\alpha_k(i), \alpha_k(j)\} \in E\}$ be the edge sets of the connected components of G_E. Since for each k, the graph with vertex set $\{1, 2, \ldots, |\alpha_k|\}$ and edge set E_k is connected, the sets $\Pi_{E_k}(\mathbb{D}^{|\alpha_k|})$ are all closed convex cones. Additionally, $X \in \Pi_E(\mathbb{D}^p)$ if and only if $X \in \mathbb{S}_E^p$ and $X_{\alpha_k \alpha_k} \in \Pi_{E_k}(\mathbb{D}^{|\alpha_k|})$ for $k = 1, \ldots, d$. Hence $\Pi_E(\mathbb{D}^p)$ is the intersection of closed convex sets, and therefore is itself a closed convex set.

Bakonyi and Johnson have formulated a clique decomposition theorem for EDM completion analogous to theorem 1 for PSD completion [BJ95].

Theorem 3 *Let E be a chordal sparsity pattern of order p, with cliques β_k, $k = 1, \ldots, m$. A matrix $X \in \mathbb{S}_E^p$ is in $\Pi_E(\mathbb{D}^p)$ if and only if*

$$X_{\beta_k, \beta_k} \in \mathbb{D}^{|\beta_k|}, \quad k = 1, \ldots, m. \tag{3.9}$$

Proof If X is an EDM for a set of points $u_i, i = 1, \ldots, m$ then $X_{\beta,\beta}$ is the EDM for the community of points $u_i, i \in \beta$. Therefore (3.9) is necessary, whether or not E is chordal. To show (3.9) is sufficient, in section 3.3 we will give direct methods for finding the EDM completion when E is chordal. □

If E is not chordal, the condition in the theorem is necessary for X to be in $\Pi_E(\mathbb{D}^p)$ but not sufficient. Bakonyi and Johnson offer a counterexample (which is very similar to the one provided by Grone *et al.* [GT84] for PSD matrices):

$$B = \begin{bmatrix} 0 & 0 & ? & 1 \\ 0 & 0 & 0 & ? \\ ? & 0 & 0 & 0 \\ 1 & ? & 0 & 0 \end{bmatrix}, \quad E = \{\{1,2\},\{2,3\},\{3,4\},\{4,1\}\}.$$

Here, each principal dense submatrix is either $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ which is an EDM for 2 points at the same location, or $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, which is an EDM for 2 points spaced 2 unit apart. However, as a whole B has no EDM completion, since $B_{12} = B_{23} = B_{34} = 0$ implies the four points are all overlapping, which is contradicted by $B_{14} > 0$.

3.2.3 Vertex elimination and ordering

In general, finding the maximal cliques in an undirected graph requires nonpolynomial time computation. (For example, the Bron-Kerbosch algorithm [BK73] has complexity $O(3^p)$.) However, efficient heuristics exist if we are willing to relax our constraints, and find the maximal cliques in a chordal extension.

For an arbitrary sparsity pattern E of a set of $p \times p$ matrices, we can find the maximal cliques of a chordal extension using the *vertex elimination algorithm* (Alg. 1). This is a very well-known algorithm and is described in the early paper [RTL76], as well as in many texts (for example, [BP93, VA15]). The algorithm results in a set of maximal cliques for a chordal extension $G' = (\{1,\ldots,p\}, E')$

where E' are the edges induced by each clique:

$$E' = \{\{i, j\} \mid \exists k, i \in \beta_k, j \in \beta_k\}.$$

The extra edges $E' \setminus E$ are added when, for some vertex v_k, $\mathbf{cadj}_{E^k}(v_k)$ is not a complete subgraph. The efficiency of the algorithm (characterized by having the least number of extra edges) depends on the chosen index ordering.

Algorithm 1 Vertex elimination.

```
1  Pick an index ordering v_1, ..., v_p. Define E^1 = E.
2  for  k = 1, ... p
3      Construct a clique. β_k = cadj_{E^k}(v_k).
4      Eliminate. E^{k+1} = {{i, j} ∈ E^k | i, j ≠ v_k}.
5  end
```

Perfect elimination ordering (PEO) An ordering that ensures that v_k is always simplicial in E^k (and thus β_k is a clique in E) is called a *perfect elimination ordering*. For example, consider the example in figure 3.1. The ordering $(3, 4, 2, 1, 6, 5)$ is a perfect elimination ordering, and the maximal cliques generated are

$$\beta_1 = \{3, 4, 2\}, \quad \beta_2 = \{4, 2, 5\}, \quad \beta_3 = \{2, 1, 5\}, \quad \beta_4 = \{1, 6, 5\}.$$

Fulkerson and Gross [FG65] show that the class of chordal graphs are exactly the class of graphs that have perfect elimination orderings. Computing the perfect elimination ordering of a chordal graph can be done in linear time, as demonstrated by several algorithms. The earliest linear complexity algorithm (in number of vertices) is the lexicographical breadth-first search by Rose, Tarjan, and Lueker [RTL76]. A simpler method, the maximum cardinality search, is given by Tarjan and Yannakakis [TY84]. Additionally, Lewis, Peyton, and Pothen [LPP89] showed that the maximum cardinality search is equivalent to Prim's algorithm for

finding the maximum weight spanning tree of a weighted undirected graph [Pri57].

Elimination ordering for nonchordal sparsity The situation for general (nonchordal) sparsity patterns is more complicated. In general, finding the ordering that minimizes fill-in is NP-complete [Yan81]. In practice, there are several tractable algorithms for finding heuristic orderings that reduce fill-in. For example, Hegerness [Heg06] gives several minimal degree orderings. (E' is a minimal chordal extension of E if there is no set $E'' \subsetneq E'$ that is also a chordal extension of E.) However, these schemes can still be computationally expensive and may not always lead to efficient chordal extensions. In practice, a popular heuristic is the minimum degree ordering, which greedily picks at each step the vertex with the smallest degree [Mar57, TW67, GL89]. An improvement upon that is the approximate minimum degree (AMD) ordering by Amestoy, Davis, and Duff [ADD96]. This scheme has low computational complexity ($O(p|E|)$) and typically yields fewer nonzeros than minimal degree reorderings.

Much of the research on the ordering schemes is motivated by solving large sparse linear systems $Ax = b$ where A is a PSD matrix with sparsity pattern E. The problem is usually solved by computing $LL^TQ^Tx = Q^Tb$ where Q is the permutation matrix for the reordering, LL^T is the Cholesky factorization for Q^TAQ, and $Q(L + L^T)Q^T \in \mathbb{S}^p_{E'}$ where E' is the chordal extension of E. If E is chordal and Q is the permutation matrix for a perfect elimination ordering, then $E' = E$. For surveys on chordal graphs and sparse matrices, see [BP93, VA15].

3.3 Minimum rank completion of chordal sparse matrices

In this section we offer a minimum rank completion algorithm for matrices A with chordal sparsity pattern E. The result is represented as factors $U \in \mathbb{R}^{p\times r}$ where r is the rank of the minimum rank completion and $A = \Pi_E(UU^T)$. This method is

extended to a minimum embedding dimension EDM completion of $A \in \Pi_E(\mathbb{D}^p)$, represented by $U = [u_1, \ldots, u_p]^T \in \mathbb{R}^{p \times r}$ and $A_{ij} = \|u_i - u_j\|_2^2$ for all $\{i, j\} \in E$.

In general, finding a PSD completion of an arbitrary sparsity pattern E is not straightforward. However, if the sparsity pattern E is chordal, finding the completions can be done through direct methods. Grone *et al.* [GJS84] and Dym and Gohberg [DG81] give PSD completion algorithms when E is chordal and block-banded, respectively. (See also [GHJ99].) Along the same lines, Bakonyi and Johnson [BJ95] give similar results for finding EDM completions when E is chordal. These methods can be implemented efficiently using clique-tree based methods such as in [Smi08],[VA15, Alg. 10.2], and are related to the supernodal Cholesky factorization techniques of [DR83, Liu92]. When the sparse matrix lies in the interior of $\Pi_E(\mathbb{S}_+^p)$ (that is, when every principal dense submatrix is full rank) then these methods yield the maximum determinant completion.

In some applications, it is desirable to find the completion with the lowest rank. For example, an SDP relaxation of a combinatorial problem is tight if the solution admits a rank-1 completion [GW95, Lov79]. Along the same lines, applications involving EDMs (such as sensor network node localization [BLW06] or protein conformation recovery [AKG13]) require a solution with embedding dimension 2 or 3 (corresponding to points in a physical space). The problem of finding the minimum rank PSD completion for general sparsity patterns is NP-hard [LV12, ELV13], and is discussed in many works (see surveys [Joh90, Lau01]. However, again, efficient methods exist when E is chordal. Dancis [Dan92] provides constructive proofs for the minimum rank completion of a chordal sparse matrix similar to that found in [GJS84]. The problem is also discussed in [LV12, HPR89, MAL14] for low rank PSD completions or [CH88, KW12] for low embedding dimension EDM completions.

To explain the novel parts of this algorithm, we present the minimum rank PSD (and minimum embedding dimension EDM) completions for a sparsity pattern

consisting of two overlapping diagonal blocks. The extension to general chordal sparsity patterns is straightforward, and follows from well-known properties of clique trees [VA15]. We discuss this briefly at the end of this section, and give an algorithm summarizing the two completions for chordal sparsity patterns.

Notation For nonsymmetric matrices $U \in \mathbb{R}^{p\times m}$ and a set $\beta \subset \{1,\ldots,p\}$, we use the notation $U_\beta = P_\beta U$ to represent the submatrix of U composed of the rows indexed in β.

3.3.1 Minimum-rank PSD completion

The existence of a minimum rank PSD completion is discussed in the following theorem by Dancis [Dan92, Th. 1.9].

Theorem 4 *For some matrix $A \in \Pi_E(\mathbb{S}^p_+)$, where E is a chordal sparsity pattern with maximal cliques $\beta_1,\ldots,\beta_m$, define*

$$r = \min_k \mathbf{rank}(A_{\beta_k,\beta_k}).$$

Then for all PSD completions X of A, it must be that $\mathbf{rank}(X) \geq r$. Additionally, there exists a completion X where $\mathbf{rank}(X) = r$.

Proof Clearly, the rank of X cannot be less than r. If $\mathbf{rank}(X) = \hat{r}$, then there exists a factorization $X = UU^T$ where U has $\hat{r}$ columns, and each maximal dense submatrix of A has a rank $\hat{r}$ factorization $A_{\beta_k,\beta_k} = U_{\beta_k}U_{\beta_k}^T$, and has rank $< r$, which is a contradiction. To show a rank-r completion exists, Alg. 2 gives a construction. □

We first consider the completion of a matrix with two overlapping diagonal

blocks, partitioned as

$$A = \begin{bmatrix} A_{11} & A_{21}^T & 0 \\ A_{21} & A_{22} & A_{32}^T \\ 0 & A_{32} & A_{33} \end{bmatrix} \in \Pi_E(\mathbb{S}_+^p).$$

Here, the two overlapping blocks are

$$H^{(1)} = \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} \quad \text{and} \quad H^{(2)} = \begin{bmatrix} A_{22} & A_{32}^T \\ A_{32} & A_{33}. \end{bmatrix}$$

Define s_1, s_2, and s_3 as the number of rows in A_{11}, A_{22}, and A_{33}, respectively. The $3,1$ block of A is not specified. (Its indices are not in E.) Define

$$r = \max\{\mathbf{rank}(H^{(1)}), \mathbf{rank}(H^{(2)})\},$$

and by theorem 4, the minimum rank completion of A must have rank r. Our goal is to find $U \in \mathbb{R}^{p \times r}$ such that UU^T is the rank-r PSD completion of A.

By the definition of r, there exists factorizations

$$H^{(1)} = U^{(1)}(U^{(1)})^T \text{ and } H^{(2)} = U^{(2)}(U^{(2)})^T$$

where $U^{(1)}$ and $U^{(2)}$ have r columns. A straightforward way to find these factors is to use an eigenvalue decomposition of $H^{(1)}$ and $H^{(2)}$. Note that the factors $U^{(1)}$ and $U^{(2)}$ may not have full column rank. We use the following partitioning of $U^{(1)}$ and $U^{(2)}$

$$H^{(1)} = \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} U_1^{(1)} \\ U_2^{(1)} \end{bmatrix} \begin{bmatrix} U_1^{(1)} \\ U_2^{(1)} \end{bmatrix}^T \quad H^{(2)} = \begin{bmatrix} A_{22} & A_{32}^T \\ A_{32} & A_{33} \end{bmatrix} = \begin{bmatrix} U_1^{(2)} \\ U_2^{(2)} \end{bmatrix} \begin{bmatrix} U_1^{(2)} \\ U_2^{(2)} \end{bmatrix}^T,$$

and it is clear that $A_{22} = U_2^{(1)}(U_2^{(1)})^T = U_1^{(2)}(U_1^{(2)})^T$.

We wish to find an $r \times r$ orthogonal matrix Q such that $U_2^{(1)} = U_1^{(2)} Q$. One way to find Q is to use the full singular value decompositions (SVDs) of the overlapping parts

$$U_2^{(1)} = W^{(1)} \Sigma^{(1)} (V^{(1)})^T, \quad U_1^{(2)} = W^{(2)} \Sigma^{(2)} (V^{(2)})^T.$$

Here, the dimensions of $W^{(1)}$ and $W^{(2)}$ are $s_2 \times s_2$, of $\Sigma^{(1)}$ and $\Sigma^{(2)}$ are $s_2 \times r$, and of $V^{(1)}$ and $V^{(2)}$ are $r \times r$. We assume that the singular values in $\Sigma^{(1)} = \Sigma^{(2)} = \Sigma$ are ordered, so that

$$\Sigma_{11} \geq \Sigma_{22} \geq \ldots \geq \Sigma_{r_0, r_0} \geq 0$$

and $r_0 = \mathbf{rank}(X_{22})$. Since $U_2^{(1)} (U_2^{(1)})^T = U_1^{(2)} (U_1^{(2)})^T$, then it must be that $W^{(1)} \Sigma^{(1)} = W^{(2)} \Sigma^{(2)}$, and $Q = V^{(2)} (V^{(1)})^T$. (It is important to ensure that the SVD is unique, in the sense that the singular vectors corresponding to the nonzero singular values are ordered and signed consistently for both $U_2^{(1)}$ and $U_1^{(2)}$. This can be done by imposing additional conventions on the SVD computation.) In the case that X_{22} is large and the rank of $X_{22} = r$, one can also compute this Q from the QR factorizations of $(U_2^{(1)})^T$ and $(U_1^{(2)})^T$.

The rank-r completion of A is

$$X = UU^T, \quad \text{where} \quad U = \begin{bmatrix} U_1^{(1)} \\ U_2^{(1)} \\ U_2^{(2)} V^{(2)} (V^{(1)})^T \end{bmatrix}.$$

To verify, it is clear that $X \succeq 0$ and has rank r. It can also be shown that $\Pi_E(X) = A$ by expanding X, and observing that the second principal dense

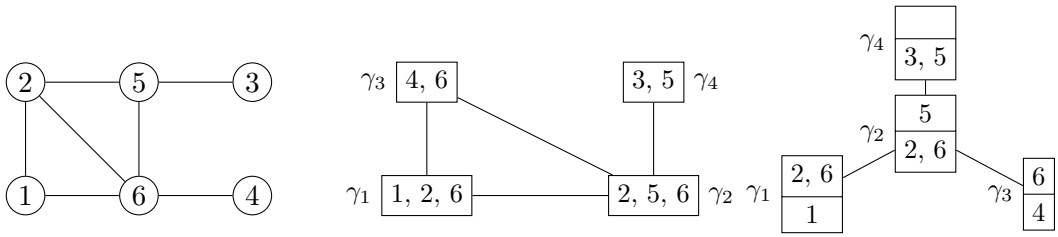

Figure 3.2: *Small example.* Left: a sparsity graph for a set of 6×6 matrices. Center: corresponding intersection graph. Right: a spanning tree of intersection graph where every topological ordering satisfies the running intersection property.

submatrix of X is

$$\begin{aligned}\begin{bmatrix} X_{22} & X_{32}^T \\ X_{32} & X_{33} \end{bmatrix} &= \begin{bmatrix} U_2^{(1)}(U_2^{(1)})^T & U_2^{(1)}V^{(1)}(V^{(2)})^T(U_2^{(2)})^T \\ U_2^{(2)}V^{(2)}(V^{(1)})^T(U_2^{(1)})^T & U_2^{(2)}(U_2^{(2)})^T \end{bmatrix} \\ &= \begin{bmatrix} U_1^{(2)}(U_1^{(2)})^T & U_1^{(2)}(U_2^{(2)})^T \\ U_2^{(2)}(U_1^{(2)})^T & U_2^{(2)}(U_2^{(2)})^T \end{bmatrix}\end{aligned}$$

which corresponds to the factorization of $H^{(2)}$.

Extension to chordal patterns To extend this method for general chordal sparsity patterns, we use a clique tree traversal method that is similar to the methods for efficiently computing a Cholesky factorization of PSD matrices [DR83, Liu92] and other computations associated with barrier methods [ADV13, VA15]. Specifically, for a chordal sparsity pattern E with maximal cliques $\beta_1, \ldots, \beta_l$, an *intersection graph* has vertices $1, \ldots, l$ and an edge between i and j whenever $\beta_i \cap \beta_j$ is nonempty. A *clique tree* is a spanning tree of the intersection graph. The tree has a *topological ordering* if all the non-root vertices k are numbered such that $k < \mathrm{par}(k)$. (See figure 3.2.)

We construct a clique tree T for a given chordal sparsity pattern E. We assume that E is connected. (If it is disconnected, a different clique tree can be constructed for each connected component, and the final completion can be

permuted to a block diagonal matrix where each diagonal block is PSD with rank r.) The choice of clique tree T determines for each clique β_k the contents of three sets

$$\alpha_k = \beta_k \cap \beta_{\mathrm{par}(k)}, \quad \eta_k = \beta_k \setminus \beta_{\mathrm{par}(k)}, \quad \nu_k = \{1, \ldots, p\} \setminus \beta_k.$$

that partition the entire index set

$$\{1, \ldots, p\} = \eta_k \cup \alpha_k \cup \nu_k.$$

(Here, $\beta_{\mathrm{par}(k)} = \emptyset$ whenever k is a root.) For each clique β_k, we partition a permutation of A and U as

$$\begin{bmatrix} A_{\nu_k\nu_k} & A_{\nu_k\alpha_k}^T & 0 \\ A_{\alpha_k\nu_k} & A_{\alpha_k\alpha_k} & A_{\alpha_k\eta_k}^T \\ 0 & A_{\eta_k\alpha_k} & A_{\eta_k\eta_k} \end{bmatrix} = \Pi_E \left(\begin{bmatrix} U_{\nu_k} \\ U_{\alpha_k} \\ U_{\eta_k} \end{bmatrix} \begin{bmatrix} U_{\nu_k} \\ U_{\alpha_k} \\ U_{\eta_k} \end{bmatrix}^T \right).$$

We then complete the 0 block using the same technique as for the two overlapping blocks, associating the sets ν_k, α_k, and η_k with the block indices 1,2,3. At each step, the rows U_{η_k} are computed using the elements in U_{α_k}, and everything else is left alone.

To ensure that local correctness implies global correctness, we need to guarantee that at each step, all rows in U_{α_k} have been computed, no rows in U_{η_k} have been computed, and the sets η_k, $k = 1, \ldots, l$ partition the index set $\{1, \ldots, p\}$. This is true if every topological ordering of T satisfying the *running intersection property (RIP)*

$$(\beta_k \cap \beta_{\mathrm{par}(k)}) \subseteq (\beta_{k+1}, \ldots, \beta_l) \tag{3.10}$$

and (equivalently) the *clique intersection property (CIP)*,

$$\beta_i \cap \beta_j \subseteq \beta_k$$

whenever β_k is in the path between β_i and β_j in T. Clearly, the RIP implies $\alpha_k \subseteq (\beta_{k+1}, \ldots, \beta_l)$, so going in reverse topological order ensures that U_{α_k} is fully computed before clique k is reached. From the CIP, it can be shown that the sets η_k, $k = 1, \ldots, l$, partition $\{1, \ldots, p\}$. It is known that chordal graphs E are exactly the set of graphs for which clique trees with the RIP and CIP can be constructed [Bun74, Gav74, BP93, BP93, VA15], and efficient methods for constructing such a T are discussed in [Pri57, LPP89, BP93, Tar83].

The entire algorithm for the minimum rank PSD completion is summarized in Alg. 2.

Algorithm 2 Minimum rank PSD completion.

Construct T the clique tree for a chordal pattern E.
Compute $r = \min_k \mathbf{rank}(A_{\beta_k\beta_k})$.
for k in reverse topological order of T
    Factor $ZZ^T = A_{\beta_k\beta_k}$ where Z has r columns.
    Partition the clique β_k into $\eta_k = \beta_k \setminus \beta_{\mathrm{par}(k)}$, $\alpha_k = \beta_k \cap \beta_{\mathrm{par}(k)}$
    Find $Q \in \mathbb{R}^{r\times r}$ where $Q^TQ = I$ and $U_{\alpha_k} = P_{\alpha_k} P_{\beta_k^T} ZQ$.
    Update $U_{\eta_k} = P_{\eta_k} P_{\beta_k}^T ZQ$.
end

3.3.2 Minimum embedding dimension EDM completion

From the close relation between EDMs and Gram matrices (given by the linear transformations (3.2) and (3.3)), we have the following extension of theorem 4 to low embedding dimension EDM completions.

Theorem 5 *For some matrix $A \in \Pi_E(\mathbb{D}^p)$, where E is a chordal sparsity pattern with maximal cliques $\beta_1, \ldots, \beta_m$, define*

$$r = \min_k \mathbf{emdim}(A_{\beta_k,\beta_k}).$$

Then for all EDM completions X

$$\textbf{emdim } X \geq r$$

and there exist a completion X where $\textbf{emdim } X = r$.

Proof Again it is clear that $\textbf{emdim } X$ cannot be less than r. If the embedding dimension of X is $\hat{r} < r$, then there exists p points $\hat{u}_1, \ldots, \hat{u}_p \in \mathbb{R}^{\hat{r}}$, and each dense principal submatrix A_{β_k,β_k} is an EDM for the community of points $\hat{u}_i$, for $i \in \beta_k$. This implies that for all k, $\textbf{emdim } A_{\beta_k,\beta_k} < r$, which is a contradiction. To show that a completion with embedding dimension r exists, we give a construction in Alg. 3. $\square$

We extend this completion method (Alg 2) to matrices $A \in \Pi_E(\mathbb{D}^p)$ where E is chordal by interpreting the PSD matrix in the previous section as a Gram matrix for a set of points with an EDM. Define the mapping from points to their EDM as

$$\textbf{edm}(U) = \textbf{diag}(UU^T)\mathbf{1}^T + \mathbf{1}\textbf{diag}(UU^T)^T - 2UU^T.$$

Our goal is to find

$$U = \begin{bmatrix} u_1^T \\ \vdots \\ u_p^T \end{bmatrix} \in \mathbb{R}^{p \times r} \text{ where } A = \Pi_E(\textbf{edm}(U)).$$

We again begin by completing a sparse matrix with two overlapping dense blocks

$$A = \begin{bmatrix} A_{11} & A_{21}^T & 0 \\ A_{21} & A_{22} & A_{32}^T \\ 0 & A_{32} & A_{33} \end{bmatrix} \in \Pi_E(\mathbb{D}^p), \quad H^{(1)} = \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix}, \quad H^{(2)} = \begin{bmatrix} A_{22} & A_{32}^T \\ A_{32} & A_{33} \end{bmatrix}.$$

The size of the diagonal blocks A_{11}, A_{22}, and A_{33} are $s_1 \times s_1$, $s_2 \times s_2$, and $s_3 \times s_3$ respectively.

We then find the matrices $U^{(1)} \in \mathbb{R}^{(s_1+s_2)\times r}$ and $U^{(2)} \in \mathbb{R}^{(s_2+s_3)\times r}$ for which $H^{(1)} = \mathbf{edm}(U^{(1)})$ and $H^{(2)} = \mathbf{edm}(U^{(2)})$. These factors can be calculated as the rank r factorizations of the Gram matrices computed using the transformation (3.3)

$$G^{(1)} = U^{(1)}(U^{(1)})^T = -\frac{1}{2}(I - \mathbf{1}c^T)H^{(1)}(I - c\mathbf{1}^T), \text{ for some } c : c^T\mathbf{1} = 1$$

and

$$G^{(2)} = U^{(2)}(U^{(2)})^T = -\frac{1}{2}(I - \mathbf{1}d^T)H^{(2)}(I - d\mathbf{1}^T) \text{ for some } d : d^T\mathbf{1} = 1.$$

Since EDMs are invariant to global translations of the points, there are multiple choices for $U^{(1)}$ and $U^{(2)}$, parametrized by c and d. The choice of these vectors will result in factors satisfying $(U^{(1)})^T c = 0$ and $(U^{(2)})^T d = 0$. In expanded form, the Gram matrices can be partitioned as

$$G^{(1)} = \begin{bmatrix} G_{11}^{(1)} & (G_{21}^{(1)})^T \\ G_{21}^{(1)} & G_{22}^{(1)} \end{bmatrix}, \qquad G^{(2)} = \begin{bmatrix} G_{11}^{(2)} & (G_{21}^{(2)})^T \\ G_{21}^{(2)} & G_{22}^{(2)} \end{bmatrix}$$

and if we can find $G^{(1)}$ and $G^{(2)}$ such that $G_{22}^{(1)} = G_{11}^{(2)}$, then we can find $U^{(1)}$ and $U^{(2)}$, and the full completion U using the same techniques as in the previous section. One choice is

$$c = \frac{1}{s_2}\begin{bmatrix} 0_{s_1} \\ \mathbf{1}_{s_2} \end{bmatrix}, \quad d = \frac{1}{s_2}\begin{bmatrix} \mathbf{1}_{s_2} \\ 0_{s_3} \end{bmatrix}, \tag{3.11}$$

which causes $G^{(1)}$ and $G^{(2)}$ to be the Gram matrices for points where the overlapping portion is centered at 0. We can verify that this satisfies the consistency

condition by expanding the definitions of $G^{(1)}$ and $G^{(2)}$:

$$\begin{bmatrix} G_{11}^{(1)} & (G_{21}^{(1)})^T \\ G_{21}^{(1)} & G_{22}^{(1)} \end{bmatrix} = -\frac{1}{2} \begin{bmatrix} I & -\frac{1}{s_2}\mathbf{1}\mathbf{1}^T \\ 0 & I - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T \end{bmatrix} \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} I & 0 \\ -\frac{1}{s_2}\mathbf{1}\mathbf{1}^T & I - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T \end{bmatrix},$$

$$\begin{bmatrix} G_{11}^{(2)} & (G_{21}^{(2)})^T \\ G_{21}^{(2)} & G_{22}^{(2)} \end{bmatrix} = -\frac{1}{2} \begin{bmatrix} I - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T & 0 \\ -\frac{1}{s_2}\mathbf{1}\mathbf{1}^T & I \end{bmatrix} \begin{bmatrix} A_{22} & A_{32}^T \\ A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} I - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T & -\frac{1}{s_2}\mathbf{1}\mathbf{1}^T \\ 0 & I \end{bmatrix},$$

and

$$-2G_{22}^{(1)} = -2G_{11}^{(2)} = (I - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T)A_{22}(I - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T).$$

We then pick $U^{(1)}$ and $U^{(2)}$ from the factorizations

$$G^{(1)} = U^{(1)}(U^{(1)})^T \text{ and } G^{(2)} = U^{(2)}(U^{(2)})^T,$$

and

$$U = \begin{bmatrix} U_1^{(1)} \\ U_2^{(1)} \\ U_2^{(2)}Q \end{bmatrix} \tag{3.12}$$

where $U_2^{(1)} = U_1^{(2)}Q$. To verify, it is clear that, if $Z = \mathbf{edm}(U)$, then $Z \in \mathbb{D}^p$ and $\mathbf{emdim}\ Z = r$. To see that $\Pi_E(Z) = A$, note that by construction,

$$H^{(1)} = \mathbf{edm}\left(\begin{bmatrix} U_1^{(1)} \\ U_2^{(1)} \end{bmatrix}\right),$$

and

$$H^{(2)} = \mathbf{edm}\left(\begin{bmatrix} U_1^{(2)} \\ U_2^{(2)} \end{bmatrix}\right) = \mathbf{edm}\left(\begin{bmatrix} U_2^{(1)} \\ U_2^{(2)}Q \end{bmatrix}\right).$$

The second equality comes from the property that EDMs are invariant to orthogonal transformations of their points.

Alternatively, we can consider a case where the parameter c in (3.11) is un-

known. In this case, it is not guaranteed that $U_2^{(1)}\mathbf{1} = 0$. As a result, using d as defined in (3.11), it is not guaranteed that $U_2^{(1)}(U_2^{(1)})^T = U_1^{(2)}(U_1^{(2)})^T$. To account for the mismatch in translation, we compute Q from the demeaned points

$$U_1^{(2)}Q = U_2^{(1)} - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T U_2^{(1)}.$$

Such a Q must exist; this follows from the observation that the *centered* Gram matrix corresponding to an EDM is unique, and therefore it must be that

$$U_1^{(2)}(U_1^{(2)})^T = (U_2^{(1)} - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T U_2^{(1)})(U_2^{(1)} - \frac{1}{s_2}\mathbf{1}\mathbf{1}^T U_2^{(1)})^T.$$

Then the completion is represented implicitly as

$$U = \begin{bmatrix} U_1^{(1)} \\ U_2^{(1)} \\ U_2^{(2)}Q + \frac{1}{s_2}\mathbf{1}\mathbf{1}^T U_2^{(1)} \end{bmatrix}. \tag{3.13}$$

To check that this yields the correct completion, note that U as defined in (3.13) is just a translated version of U as defined in (3.12), and EDMs are invariant to the translation of their points.

For general chordal sparsity patterns, we construct a clique tree from the cliques $\beta_1, \ldots, \beta_l$, with the same properties as in the previous section. For the most part, the completion method for EDMs mirrors the algorithm for PSD matrices, in that it traverses the clique tree T in reverse topological order, filling in all the rows in U_{η_k} using information in U_{α_k}. The full algorithm for the minimum embedding EDM completion is given in Alg. 3

Algorithm 3 Minimum embedding dimension EDM completion.

Construct T the clique tree for a chordal pattern E.

Compute $r = \min_k \mathbf{emdim}(A_{\beta_k\beta_k})$.

for k in reverse topological order of T

  if k is a root

    Update $U_{\beta_k} = Z$ where $Z \in \mathbb{R}^{|\beta_k|\times r}$ is from the factorization

$$ZZ^T = -\frac{1}{2}(I - \frac{1}{|\beta_k|}\mathbf{1}\mathbf{1}^T)A_{\beta_k\beta_k}(I - \frac{1}{|\beta_k|}\mathbf{1}\mathbf{1}^T).$$

  else

    Partition the set β_k as

$$\eta_k = \beta_k \setminus \beta_{\mathrm{par}(k)}, \qquad \alpha_k = \beta_k \cap \beta_{\mathrm{par}(k)}.$$

    Compute the Gram matrix parametrized by $c = P_{\alpha_k}P_{\beta_k}^T\mathbf{1}$

$$G = -\frac{1}{2}(I - \frac{1}{|\alpha_k|}\mathbf{1}c^T)A_{\beta_k\beta_k}(I - \frac{1}{|\alpha_k|}c\mathbf{1}^T),$$

    and factor $G = ZZ^T$ where $Z \in \mathbb{R}^{|\beta_k|\times r}$.

    Find $Q \in \mathbb{R}^{r\times r}$, $Q^TQ = I$ where $P_{\alpha_k}U = P_{\alpha_k}P_{\beta_k}^TZQ$.

    Update the new part

$$U_{\eta_k} = P_{\eta_k}P_{\beta_k}^TZQ + \frac{1}{|\alpha_k|}\mathbf{1}\mathbf{1}^TU_{\alpha_k}.$$

  end

end

CHAPTER 4

Decomposition methods for sparse matrix nearness problems

In this chapter[1] we discuss decomposition methods for finding the projection of a given matrix on the set of matrices that satisfy a certain property:

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|X - C\|_F^2 \\ \text{subject to} & X \in \mathcal{S} \end{array} \tag{4.1}$$

where $\mathcal{S}$ is one of the three sparse matrix cones: $\mathbb{S}^p_{E,+}$, $\Pi_E(\mathbb{S}^p_+)$, or $\Pi_E(\mathbb{D}^p)$, and E is a general (possibly nonchordal) sparsity pattern.

The problem (4.1) is a special instance of a *matrix nearness problem* [Hig88]. Matrix nearness problems arise in a wide range of applications, including statistics, machine learning, finance, and signal processing, and can take a variety of different forms. The matrix C may be symmetric or nonsymmetric, square or rectangular. The constraint set $\mathcal{S}$ may be convex or not, and the objective function may be a Frobenius norm or another norm, or a nonmetric distance function. We restrict our attention to problems involving symmetric matrix variables and convex constraints, using the Frobenius norm as the distance function. We also include an application of a nonsymmetric matrix nearness problem.

Matrix nearness problems with PSD and EDM constraints are among the most widely studied types of matrix nearness problems. The nearest PSD matrix and

[1] This chapter is largely based on a submitted journal paper; see [SV15].

nearest correlation matrix problems have applications in statistics, finance, and biology, and are discussed in [RJ99, Hig02, QS10, QS06, Mal04, AA12]. The nearest EDM problem is studied in multidimensional scaling [You13], and is used in chemistry and molecular biology to compute molecular structure [Wut89, GHH90, Tro97, AKG13], and for node localization in sensor networks [KW12, AW05].

These applications all involve fairly simple convex sets and, in principle, they can be solved by general-purpose convex optimization algorithms. However, due to the large number of variables (order p^2 for dense $p \times p$ matrix variables) the complexity of general-purpose methods grows rapidly with the matrix dimension p. Research on matrix nearness problems has therefore concentrated on specialized first-order algorithms, or, when higher accuracy is required, quasi-Newton algorithms. Examples of such first-order methods are alternating projection methods [GM89, Hig02, GHH90, AA12], dual first-order methods [BX05, HM11], augmented Lagrangian methods [QS11], and alternating direction methods [BHR10]. Algorithms based on quasi-Newton methods in combination with dual reformulations are described in [QS06, Mal04, QXY13].

In comparison, algorithms tailored to *sparse* matrix nearness problems, *i.e.*, problems with an additional sparsity constraint, have received less attention in the literature. Sparsity constraints arise naturally in large matrix nearness problems and can have a different meaning depending on the application and the interpretation of the data matrix C. We may be interested in recovering a sparse matrix with the properties represented as a set $\mathcal{S}$, from a noisy measurement or estimate of its nonzero elements. A typical example is the estimation of a sparse covariance or inverse covariance matrix from estimates of its nonzero entries. Or, the sparse matrix C represents a noisy estimate of a subset of the entries of a dense matrix Z, which is known to have certain structural properties, represented by a constraint $Z \in \hat{\mathcal{S}}$. In this case the matrices in $\hat{\mathcal{S}}$ that best fit the measurements are dense. This is an example of a problem with aggregate sparsity, and when the matrix

dimensions are large it is of interest to avoid working with a dense matrix variable and represent it implicitly as a sparse matrix X, with the added constraint that the matrix has a completion with the desired properties ($X \in \Pi_E(\hat{\mathcal{S}})$).

Many algorithms for sparse matrix optimization problems in the literature are extensions of algorithms for dense matrix optimization problems. When the problem has a sparse variable constraint $X \in \mathbb{S}^p_E$, one can use a dense matrix variable and impose the sparsity constraint by adding linear equality constraints. When the problem has aggregate sparsity, one can use a dense matrix variable and mask the irrelevant entries in the objective [Hig02, QS11, AKW99, Qi13, ADS13]. Extensions of this type are still computationally expensive. In particular, even if the linear constraints can be decoupled, they involve at least eigenvalue decompositions of order p, used for projecting on the set of PSD matrices or the set of EDMs.

In contrast, the approach taken in this chapter is to avoid eigenvalue decompositions of order p and only use eigenvalue decompositions of smaller dense matrices. By applying decomposition techniques for cones of sparse matrices with chordal structure, we write the three types of sparse matrix cones in terms of small dense PSD or EDM cones [GJS84, GT84, AHM88, BJ95]. In combination with first-order methods, these chordal decomposition techniques allow us to solve the matrix nearness problems without using eigenvalue decompositions of order p.

This chapter begins by discussing partially separable cones and their dual cones (section 4.1). We show that the matrix nearness problems of interest can be reformulated as

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \|P_\eta x - c\|_2^2 \\ \text{subject to} & x \in \mathcal{K} \end{array} \tag{4.2}$$

where $x \in \mathbb{R}^n$ and $\mathcal{K}$ is a partially separable cone. We then give decomposition methods for solving (4.2). These methods are the main contribution of this chapter. Section 4.2 gives dual decomposition methods when $\eta = \{1, \ldots, n\}$ and

the objective in (4.2) is the Euclidean projection on $\mathcal{K}$, and is strongly convex. Section 4.3 gives a Douglas-Rachford method for more general η.

We then give numerical results for the sparse matrix nearness problems in section 4.4, including two applications: the sparse projection of nonsymmetric matrices on the unit spectral norm ball, and the convex relaxation of the sensor network node localization problem. To demonstrate the performance gain, we compare the runtime of our proposed methods against a single eigenvalue decomposition (or singular value decomposition for nonsymmetric matrices) of order of the size of the matrix variable. The full eigenvalue decomposition represents the runtime of a single iteration when applying a typical first-order method to the matrix problems, but without using clique decomposition. For large problems ($p > 10,000$), our results compare favorably to preexisting methods, and in particular, we are able to solve very large sparse problems ($p = 100,000$) that previously would not be solvable using a standard Desktop computer and standard first-order methods.

4.1 Partially separable convex cones

A function $g : \mathbb{R}^n \to \mathbb{R}$ is *partially separable* if it can be expressed as

$$g(x) = \sum_{k=1}^{l} g_k(E_k x),$$

where each E_k has a nontrivial nullspace, *i.e.*, a rank substantially less than n. This concept was introduced by Griewank and Toint in the context of quasi-Newton algorithms [GT82, GT84][NW06, section 7.4]. Here we consider the simplest and most common example of partial separability and assume that $E_k = P_{\gamma_k}$ for some index set $\gamma_k \subseteq \{1, 2, \ldots, n\}$. This means that f can be written as a sum

of functions that depend only on subsets of the components of x:

$$g(x) = \sum_{k=1}^{l} g_k(P_{\gamma_k} x) = \sum_{k=1}^{l} g_k(x_{\gamma_k}).$$

Partial separability generalizes *separability* ($l = n$, $\gamma_k = \{k\}$) and *block-separability* (the sets γ_k form a partition of $\{1, 2, \ldots, n\}$).

Partially separable cone We call a cone $\mathcal{K} \subseteq \mathbb{R}^n$ *partially separable* if it can be expressed as

$$\mathcal{K} = \{x \mid P_{\gamma_k} x \in \mathcal{C}_k,\ k = 1, \ldots, l\} \tag{4.3}$$

where $\mathcal{C}_k$ is a convex cone in $\mathbb{R}^{|\gamma_k|}$. The terminology is motivated by the fact the indicator function $\delta_{\mathcal{K}}$ of $\mathcal{K}$ is a partially separable function:

$$\delta_{\mathcal{K}}(x) = \sum_{k=1}^{l} \delta_{\mathcal{C}_k}(P_{\gamma_k} x).$$

Note that $\mathcal{K}$ can also be expressed as an intersection of cones

$$\mathcal{K} = \mathcal{K}_1 \cap \cdots \cap \mathcal{K}_l$$

where $\mathcal{K}_k = \{x \mid P_{\gamma_k} x \in \mathcal{C}_k\}$. We assume that each $\mathcal{C}_k$ is a closed, nonempty convex cone. Then the cones $\mathcal{K}_k$ are nonempty convex cones, and closed by [Roc70, Th. 9.1], and $\mathcal{K}$ is a nonempty, closed convex cone. Additionally, we assume there exists a shared point in their relative interiors; that is

$$\mathbf{ri}\ \mathcal{K}_1 \cap \cdots \cap \mathbf{ri}\ \mathcal{K}_l \neq \emptyset.$$

Then the dual of $\mathcal{K}$ is the sum of the dual cones

$$\mathcal{K}^* = \left\{ \sum_{k=1}^{l} y \mid y \in \mathcal{K}_k^* \right\} = \left\{ \sum_{k=1}^{l} P_{\gamma_k}^T s_k \mid s_k \in \mathcal{C}_k^*, \quad k = 1, \ldots, l \right\}$$

and is also a closed convex cone [Roc70, Cor. 16.4.2].

Proper cones The following assumptions imply that $\mathcal{K}$ is proper,*i.e.*, closed, pointed, with nonempty interior.

- The index sets γ_k are distinct, *i.e.*, $\gamma_i \not\subseteq \gamma_j$ for $i \neq j$, and cover the entire index set $\{1, 2, \ldots, n\}$.
- The convex cones $\mathcal{C}_k$ are proper.
- There exists a point $\bar{x}$ with $P_{\gamma_k}\bar{x} \in \mathbf{int}\mathcal{C}_k$ for $k = 1, \ldots, l$.

These assumptions imply that $\mathcal{K}$ is itself a proper cone. As already shown, $\mathcal{K}$ is closed since each cone $\mathcal{C}_k$ is closed. The cone $\mathcal{K}$ is pointed because $x \in \mathcal{K}$, $-x \in \mathcal{K}$ implies $P_{\gamma_k}x \in \mathcal{C}_k$ and $-P_{\gamma_k}x \in \mathcal{C}_k$ for all k. Since the cones $\mathcal{C}_k$ are pointed, we have $P_{\gamma_k}x = 0$ for $k = 1, \ldots, l$. Since the index sets γ_k cover $\{1, 2, \ldots, n\}$, this implies $x = 0$. The cone $\mathcal{C}$ has nonempty interior because the point $\bar{x}$ is in its interior. It also follows that $\mathcal{K}^*$ is proper (since the dual cone of a proper cone is proper).

This implies the cone $\Pi_E(\mathbb{S}^p_+)$ is proper for chordal E, since each lower dimensional cone $\mathcal{C}_k = \mathbf{vec}(\mathbb{S}^{|\beta_k|}_+)$ is proper and $\bar{x} = 0$ satisfies $P_{\gamma_k}\bar{x} \in \mathbf{int}\ \mathcal{C}_k$ for all k. However, the sparse completable EDM cone is not proper, since the zero-diagonal restriction means it has no interior.

Example Take $n = 6$ and

$$\gamma_1 = \{1, 2, 6\}, \qquad \gamma_2 = \{2, 5, 6\}, \qquad \gamma_3 = \{3, 5\}, \qquad \gamma_4 = \{4, 6\}.$$

Let $\mathcal{C}_1 \subseteq \mathbb{R}^3$, $\mathcal{C}_2 \subseteq \mathbb{R}^3$, $\mathcal{C}_3 \subseteq \mathbb{R}^2$, $\mathcal{C}_4 \subseteq \mathbb{R}^2$ be proper convex cones. A vector $x \in \mathbb{R}^6$ is in the cone $\mathcal{K}$ defined in (4.3) if

$$(x_1, x_2, x_6) \in \mathcal{C}_1, \qquad (x_2, x_5, x_6) \in \mathcal{C}_2, \qquad (x_3, x_5) \in \mathcal{C}_3, \qquad (x_4, x_6) \in \mathcal{C}_4,$$

and a vector $s \in \mathbb{R}^6$ is in the dual cone $\mathcal{K}^*$ if

$$s = \begin{bmatrix} s_{11} \\ s_{12} \\ 0 \\ 0 \\ 0 \\ s_{13} \end{bmatrix} + \begin{bmatrix} 0 \\ s_{21} \\ 0 \\ 0 \\ s_{22} \\ s_{23} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ s_{31} \\ 0 \\ s_{32} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ s_{41} \\ 0 \\ s_{42} \end{bmatrix}$$

for some

$$s_1 = (s_{11}, s_{12}, s_{13}) \in \mathcal{C}_1^*, \quad s_2 = (s_{21}, s_{22}, s_{23}) \in \mathcal{C}_2^*,$$

$$s_3 = (s_{31}, s_{32}) \in \mathcal{C}_3^*, \quad s_4 = (s_{41}, s_{42}) \in \mathcal{C}_4^*.$$

4.1.1 Matrix nearness problems

When a sparsity pattern E is chordal, the matrix sets $\Pi_E(\mathbb{S}_+^p)$ and $\Pi_E(\mathbb{D}^p)$ are partially separable cones. To see this, we give transformations between matrix and vector cones. For a $p \times p$ symmetric matrix X, we define

$$\mathbf{vec}(X) = [X_{11}, \sqrt{2}X_{12}, \ldots \sqrt{2}X_{1p}, X_{22}, \sqrt{2}X_{23}, \ldots, \sqrt{2}X_{2p}, \ldots, X_{pp}]^T \in \mathbb{R}^{p(p+1)/2}$$

so that $\mathbf{tr}(XY) = \mathbf{vec}(X)^T\mathbf{vec}(Y)$. The inverse operation takes a vector in $\mathbb{R}^{p(p+1)/2}$ and transforms it to a matrix in $\mathbb{S}^p$:

$$\mathbf{mat}(x) = \begin{bmatrix} x_1 & x_2/\sqrt{2} & \dots & x_p/\sqrt{2} \\ x_2/\sqrt{2} & x_{p+1} & \dots & x_{p-1}/\sqrt{2} \\ \vdots & \vdots & \ddots & \vdots \\ x_p/\sqrt{2} & x_{2p-1}/\sqrt{2} & \dots & x_{p(p-1)/2} \end{bmatrix}$$

and $\mathbf{mat}(\mathbf{vec}(X)) = X$. Similarly, if $X \in \mathbb{S}^p_E$, we define $\mathbf{vec}_E(X)$ to contain the lower-diagonal elements of X whose indices are either along the diagonal or in E, scaled so that if $X \in \mathbb{S}^p_E$ and $Y \in \mathbb{S}^p_E$, then $\mathbf{tr}(XY) = \mathbf{vec}_E(X)^T\mathbf{vec}_E(Y)$. The inverse mapping $\mathbf{mat}_E(x)$ is defined such that $\mathbf{mat}_E(\mathbf{vec}_E(X)) = X$.

For sparse matrices with chordal pattern E, we define the sets $\gamma_1, \dots, \gamma_l$ so that

$$P_{\gamma_k}\mathbf{vec}_E(X) = \mathbf{vec}(P_{\beta_k} X P_{\beta_k}^T), \quad k = 1, \dots, l$$

where $\beta_1, \dots, \beta_l$ are the maximal cliques of E. Then

$$\mathbf{vec}_E(\Pi_E(\mathbb{S}^p_+)) = \{x \mid P_{\gamma_k} x \in \mathcal{C}_k\}, \quad \text{where} \quad u \in \mathcal{C}_k \iff \mathbf{mat}(u) \succeq 0,$$

and

$$\mathbf{vec}_E(\Pi_E(\mathbb{D}^p)) = \{x \mid P_{\gamma_k} x \in \mathcal{C}_k\}, \quad \text{where} \quad u \in \mathcal{C}_k \iff \mathbf{mat}(u) \in \mathbb{D}^{|\beta_k|}.$$

Therefore $\mathbf{vec}_E(\Pi_E(\mathbb{S}^p_+))$ and $\mathbf{vec}_E(\Pi_E(\mathbb{D}^p))$ are partially separable cones. When E is chordal, the matrix nearness problem (4.1) for $\mathcal{S} = \Pi_E(\mathbb{S}^p_+)$ or $\mathcal{S} = \Pi_E(\mathbb{D}^p)$ is a Euclidean projection on a partially separable cone. When E is nonchordal, then problem (4.1) can be reformulated as an optimization problem with an objective

that is convex, but not strongly convex:

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \sum_{\{i,j\}\in E} (X_{ij} - C_{ij})^2 \\ \text{subject to} & X \in \Pi_{E'}(\hat{\mathcal{S}}). \end{array}$$

Here, E' is a chordal extension of E, and $\hat{\mathcal{S}}$ corresponds to the dense matrix cones $\mathbb{S}_+^p$ or $\mathbb{D}^p$.

There is a rich literature on decomposition methods for computing the projection on an intersection of closed convex sets via a sequence of projections on each of the sets separately; see, for example, [Dyk83, HL88, Tse90, Tse91] and the books [BT97, CZ97]. We will discuss three approaches based on duality. In the first approach (section 4.2.2) the gradient projection method [Pol87, §7.2.1] or accelerated gradient projection method [Nes83, Nes04, BT09] are applied to the dual problem. These algorithms can be viewed as applications of Tseng's *alternating minimization method* for minimizing a strongly convex function over the intersection of convex sets [Tse90, Tse91, BT14]. The second approach (section 4.2.3) is the dual block coordinate ascent method [Tse93]. This method can be interpreted as a generalization of Dykstra's cyclic projection algorithm [Dyk83, BD86] or Han's successive projection algorithm [Han88], and also as a dual block coordinate gradient projection method [BT13]. In the third approach (section 4.3), we apply the Douglas-Rachford method [LM79] to alternately optimize over the objective and constraints.

Notation We use the shorthand

$$P = \begin{bmatrix} P_{\gamma_1}^T & P_{\gamma_2}^T & \cdots & P_{\gamma_l}^T \end{bmatrix}^T$$

Note that the matrix P^TP is diagonal and that the jth diagonal entry is the number of index sets γ_k that contain the index j:

$$(P^TP)_{jj} = |\{k \mid j \in \gamma_k\}|, \quad j = 1, \ldots, n. \tag{4.4}$$

We assume the sets γ_k cover the entire index set $\{1, \ldots, n\}$, and therefore P^TP is nonsingular. For a partially separable cone as defined in (4.3), we define the product of the lower dimensional cones as

$$\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2 \times \cdots \times \mathcal{C}_l, \qquad \mathcal{C}^* = \mathcal{C}_1^* \times \mathcal{C}_2^* \times \cdots \times \mathcal{C}_l^*.$$

Using this notation, we can write compactly

$$x \in \mathcal{K} \iff Px \in \mathcal{C}, \quad \text{and} \quad P^Tz \in \mathcal{K}^* \iff z \in \mathcal{C}^*.$$

4.2 Dual decomposition for partially separable cones

We consider the Euclidean projection of a vector c on a partially separable cone

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \|x - c\|_2^2 \\ \text{subject to} & x \in \mathcal{K} \end{array} \tag{4.5}$$

where $c \in \mathbb{R}^n$ is a given matrix and x is in a partially separable cone

$$x \in \mathcal{K} \iff Px \in \mathcal{C} \iff P_{\gamma_k}x \in \mathcal{C}_k, \quad k = 1, \ldots, l.$$

The problem has a unique solution $x^\star$ the orthogonal projection of c on $\mathcal{K}$. The dual of (4.5) is

$$\begin{array}{ll} \underset{s}{\text{maximize}} & -\|s + c\|_2^2 + \|c\|_2^2 \\ \text{subject to} & s \in \mathcal{K}^* \end{array} \tag{4.6}$$

The solution s of the dual is the projection of $-c$ on the polar cone $-\mathcal{K}^*$. By our assumption, there exists a point $\bar{x}$ in the intersection of the sets $\mathbf{ri}\,\mathcal{K}_k$, $k = 1, \ldots, l$. In this case the dual constraint can be written equivalently as

$$s = \sum_{k=1}^{l} P_{\gamma_k}^T z_k, \qquad \text{where} \qquad z_k \in \mathcal{C}_k^*.$$

The solutions $x^\star$ and $s^\star$ of the two projection problems are unique and related by the optimality conditions

$$c = x^\star - s^\star, \qquad x^\star \in \mathcal{K}, \qquad s^\star \in \mathcal{K}^*, \qquad s^{\star T} x^\star = 0,$$

and one can solve either the primal or the dual problem and compute the other solution from the relation $c = x^\star - s^\star$. The relationship is the same as that illustrated in 2.1, where x and $-s$ are orthogonal components of c, and $s = -\Pi_{-\mathcal{K}^*}(c)$.

We can write (4.6) equivalently using a change of variables $s = P^T z$, where $z = (z_1, \ldots, z_l)$

$$\begin{array}{ll} \underset{z}{\text{maximize}} & -\|P^T z + c\|_2^2 + \|c\|_2^2 \\ \text{subject to} & z \in \mathcal{C}^*. \end{array} \tag{4.7}$$

The key benefit of dual decomposition is that the dual constraint is separable:

$$z \in \mathcal{C}^* \iff z_k \in \mathcal{C}_k^*,\ k = 1, \ldots, l.$$

Extension to strongly convex objective The primal and dual problems corresponds to the projection of a matrix C on the cone of sparse completable matrices and its dual, when the sparsity pattern E is chordal. However, the methods

presented in this section generalize for the optimization problem

$$\begin{array}{lll} \underset{x}{\text{minimize}} & f(x) & \underset{s}{\text{maximize}} \quad -f^*(s) \\ \text{subject to} & x \in \mathcal{K}, & \text{subject to} \quad s \in \mathcal{K}^* \end{array} \tag{4.8}$$

where f is strongly convex with modulus μ, f^* is the conjugate of f and has $1/\mu$-Lipschitz continuous gradient. The extensions are straightforward and efficient if the gradient of f^* is easy to compute.

4.2.1 Projection on partially separable cones

The dual decomposition methods we discuss in the next two sections are descent methods for minimizing

$$g(z) = (1/2)\|P^T z + c\|_2^2 = (1/2)\|\sum_{k=1}^{l} P_{\gamma_k}^T z_k + c\|_2^2$$

where z is optimized over the product cone $\mathcal{C}^* = \mathcal{C}_1^* \times \mathcal{C}_2^* \times \cdots \times \mathcal{C}_l^*$. The methods generate a sequence of dual feasible suboptimal points z. From a dual feasible $z \in \mathcal{C}^*$, approximate primal and dual projections x and s are computed as $x = P^T z + c$ and $s = P^T z$. The distances to optimality $\|x - x^\star\|_2$ and $\|s - s^\star\|_2$ can be bounded in terms of the dual suboptimality $g(z) - f(z^\star)$. To see this, we note that for any dual optimal solution $z^\star$ and any $z \in \mathcal{C}^*$, we have

$$\begin{aligned} \|P^T z - P^T z^\star\|_2^2 &= \|P^T z + c\|_2^2 - \|P^T z^\star + c\|_2^2 - 2(Pz^\star + c)^T P^T (z - z^\star) \\ &\leq \|P^T z + c\|_2^2 - \|P^T z^\star + c\|_2^2. \end{aligned}$$

The inequality holds because

$$\nabla g(z^\star)^T (z - z^\star) = (P^T z^\star + c)^T P^T (z - z^\star) \geq 0$$

for all $z \in \mathcal{C}^*$ if $z^\star$ is optimal. Hence, if z is dual feasible and we define $s = P^T z$, $x = P^T z + c$, then

$$\begin{aligned}\frac{1}{2}\|x - x^\star\|_2^2 = \frac{1}{2}\|s - s^\star\|_2^2 &= \frac{1}{2}\|P^T(z - z^\star)\|_2^2 \\ &\leq \frac{1}{2}\|P^T z + c\|_2^2 - \frac{1}{2}\|P^T z^\star + c\|_2^2 \\ &= g(z) - g(z^\star).\end{aligned}$$

The Lipschitz constant of $\nabla g(x)$ is the largest eigenvalue of the Hessian $\nabla^2 g(z) = P^T P$:

$$L = \lambda_{\max}(P^T P) = \max_{j=1,\ldots,n} |\{k \mid j \in \gamma_k\}|. \tag{4.9}$$

(Recall that $P^T P$ is diagonal; see (4.4).)

4.2.2 Dual gradient projection method

The gradient projection method for minimizing the function (4.13) over $\mathcal{C}^*$ uses the iteration

$$z_k^i = \Pi_{\mathcal{C}_k^*}\left(z_k^{i-1} - tP_{\gamma_k}\nabla g(P^T z)\right)$$

where t is a positive step size. In terms of the block vectors,

$$\begin{aligned}z_k^i &= \Pi_{\mathcal{C}_k^*}(z_k^{i-1} - tP_{\gamma_k}(\sum_{j=1}^{l} P_{\gamma_j}^T z_j^{i-1} + c)) \\ &= \Pi_{\mathcal{C}_k^*}((1-t)z_k^{i-1} - tP_{\gamma_k}(\sum_{j\neq k} P_{\gamma_j}^T z_j^{i-1} + c)).\end{aligned}$$

On line 2 we use the fact that $P_{\gamma_k} P_{\gamma_k}^T = I$. The projections on $\mathcal{C}_k^*$ can also be expressed in terms of projections on $\mathcal{C}_k$ via the identity $u = \Pi_{\mathcal{C}_k}(u) - \Pi_{\mathcal{C}_k^*}(-u)$.

Recall from section 2.4.1 and equation (2.18) that if $t = 1/L$ for L given in (4.9)

then the sequence z^i converges to a minimizer of g over $\mathcal{C}^*$, and the duality gap decreases as $O(1/i)$. It follows that for the sequences $x^i = P^T z^i + c$ and $s^i = P^T z^i$ and their projections $x^\star$ and $s^\star$,

$$\|x^i - x^\star\|_2 = \|s^i - s^\star\|_2 = O(1/\sqrt{i}).$$

Similarly, from the $O(1/i^2)$ convergence rate in the objective of the accelerated gradient projection method, it follows that for the sequences $x^i = P^T z^i + c$ and $s^i = P^T z^i$:

$$\|x^i - x^\star\|_2 = \|s^i - s^\star\|_2 = O(1/i).$$

We also use Nesterov's first accelerated gradient projection method [Nes83], or, more generally, FISTA [BT09, Tse08]. With the assumption $z^{-1} = z^0$, FISTA applies a gradient projection update after an extrapolation step is:

$$z^i = \Pi_{\mathcal{C}^*}(v^i - t\nabla g(v^i)) \quad \text{where } v^i = z^{i-1} + \frac{i-2}{i+1}(z^{i-1} - z^{i-2}).$$

As shown in section 2.4.1, the convergence of the duality gap under acceleration is $O(1/i^2)$, which is an improvement over the $O(1/i)$ rate in the gradient projection method.

The gradient projection algorithm for the projection on $\mathcal{K}$ is summarized in Algorithm 4, and its accelerated version is summarized in Algorithm 5. It is important to keep in mind that the projection steps reduces to l projections on $\mathcal{C}_k^*$ or $\mathcal{C}_k$, and can be computed in parallel:

$$z_k^i = \Pi_{\mathcal{C}_k^*}(y_k^i) = y_k^i + \Pi_{\mathcal{C}_k}(-y_k^i), \quad k = 1, \ldots, l.$$

The stopping conditions used are given by (2.20).

Algorithm 4 Gradient projection method for projecting on $\mathcal{K}$.

Set $t = 1/L$ with L defined in (4.9).
Choose an initial $z^0 = (z_1^0, \ldots, z_l^0)$, and take $s^0 = P^T z^0$ and $x^0 = s^0 + c$.
for $i = 1, 2, \ldots$ until convergence
    Gradient step. Compute $y^i = z^{i-1} - tPx^{i-1}$.
    Projection step. Compute $z^i = \Pi_{\mathcal{C}^*}(y^i)$, $s^i = P^T z^i$, and $x^i = s^i + c$.
end

Algorithm 5 Fast gradient projection method for projecting on $\mathcal{K}$.

Set $t = 1/L$ with L defined in (4.9).
Choose an initial $z^{-1} = z^0 = (z_1^0, \ldots, z_l^0)$, and take $s^0 = P^T z^0$, $x^0 = s^0 + c$.
for $i = 1, 2, \ldots$ until convergence
    Extrapolation step. Compute $v^i = z^{i-1} + \frac{i-2}{i+1}(z^{i-1} - z^{i-2})$.
    Gradient step. Compute $y^i = v^i - tP(P^T v^i + c)$.
    Projection step. Compute $z^i = \Pi_{\mathcal{C}^*}(y^i)$, $s^i = P^T z^i$, and $x^i = s^i + c$.
end

4.2.3 Dual block coordinate ascent

We next apply block-coordinate ascent to the dual problem (4.7), which is similar to the *successive projection algorithm* [Han88],[CP11b, Pr. 11] for quadratic functions. At each step the problem reduces to

$$\begin{array}{ll} \underset{z_k}{\text{minimize}} & \|z_k + P_{\gamma_k}(\sum_{j \neq k} P_{\gamma_j}^T z_j + c)\|_2^2 \\ \text{subject to} & z_k \in \mathcal{C}_k^*. \end{array} \tag{4.10}$$

This is a Euclidean projection of the point $w = -P_{\gamma_k}\left(\sum_{j \neq k} P_{\gamma_j}^T z_j + c\right)$ on $\mathcal{C}_k^*$. Recall from (2.1) that the unique solution can be expressed in two equivalent forms:

$$z_k = \Pi_{\mathcal{C}_k^*}(w) = \Pi_{\mathcal{C}_k}(-w) + w. \tag{4.11}$$

We can view (4.10) as a block coordinate gradient projection update. The gradient of $g(z)$ with respect to z_k is

$$\nabla_{z_k} g(z) = P_{\gamma_k} \left(\sum_{j=1}^{l} P_{\gamma_j}^T z_j + c \right) = z_k + P_{\gamma_k} \left(\sum_{j \neq k} P_{\gamma_j}^T z_j + c \right),$$

because $P_{\gamma_k} P_{\gamma_k}^T = I$. Therefore the vector $w = z_k - \nabla_{z_k} g(z)$ and $\Pi_{\mathcal{C}_k^*}(w)$ is a block coordinate gradient projection step with step size one.

In general, the block coordinate ascent algorithm is not guaranteed to converge. Moreover, even when convergence is guaranteed, deciding when to terminate the block coordinate ascent method can be difficult, since the iterates may remain constant for several successive iterations [Pow73]. However, Tseng showed that the block coordinate descent method applied to the dual of (4.8) converges to the optimal solution if $f(x)$ (the primal objective) is strongly convex and $\mathcal{K}$ is a product of closed convex sets [Tse93]. This result is similar to the earlier results by Auslender [Aus76] for strictly convex $f(x)$ and Han [Han88, Cor. 4.3, Prop 4.4] for positive definite quadratic functions. Additionally, as shown in section (4.10), the block coordinate ascent applied to the projection on $\mathcal{K}$ is equivalent to Dykstra's algorithm [Dyk83, BD86], which is also proven to converge.

In Algorithm 6 we minimize over $z_1, \ldots, z_l$ cyclically, using the second expression in (4.11). We also maintain a primal variable $x^i = \sum_{j=1}^{l} P_{\gamma_j}^T z_j^i + c$. The simplest initialization is to take $z^0 = 0$ and $x^0 = a$. On line 4 we project the point

$$P_{\gamma_k} \left(\sum_{j \neq k} P_{\gamma_j}^T z_j^{i-1} + c \right) = P_{\gamma_k} x^{i-1} - z_k^{i-1}$$

on $\mathcal{C}_k$. The update on line 5 is easier to describe in words: x^i is equal x^{i-1} with the subvector $x_{\gamma_k}^{i-1}$ replaced by v. Line 6 can also be written as $z_k^i = z_k^{i-1} + v - P_{\gamma_k} x^{i-1}$.

Hence

$$z_k^i = z_k^{i-1} + v - P_{\gamma_k}\left(\sum_{j=1}^{l} P_{\gamma_j}^T z_j^{i-1} + c\right) = v - P_{\gamma_k}\left(\sum_{j\neq k} P_{\gamma_j}^T z_j^{i-1} + c\right).$$

We use the stopping condition proposed in section 2.3.4 (which is based on [Ray05, eq. (13)]).

Algorithm 6 Dual block coordinate ascent for projecting on $\mathcal{K}$.

Choose an initial $(z_1^0, \ldots, z_l^0) \in \mathcal{C}^*$ and set $x^0 = P^T z^0 + c$.
for $i = 1, 2, \ldots$ until convergence
  Select the next index: $k = (i-1) \bmod l + 1$.
  Projection step. Compute $v = \Pi_{\mathcal{C}_k}(P_{\gamma_k} x^{i-1} - z_k^{i-1})$.
  Update the primal variable. Compute $x^i = P_{\gamma_k}^T v + (I - P_{\gamma_k}^T P_{\gamma_k}) x^{i-1}$.
  Update dual variables. Compute

$$z_k^i = z_k^{i-1} + P_{\gamma_k}(x^i - x^{i-1}), \quad z_j^i = z_j^{i-1} \text{ for } j \neq k.$$

end

With the initialization $x^0 = c$, $z^0 = 0$, this is Dykstra's algorithm for computing the projection on $\bigcap_{k=1}^{l} \mathcal{K}_k$ [Dyk83]. Algorithm 6 is a special case of Tseng's *dual block coordinate ascent algorithm* [Tse93, §3] and convergence follows from [Tse93, Th. 3.1]. Recall also from section 2.3.4 that $f(z^i) - f(z^\star) \leq \tau/i$, where τ is a constant [BT13, theorem 6.3], and therefore x^i and $s^i = P^T z^i$ satisfy $\|x^i - x^\star\|_2 = \|z^i - z^\star\|_2 = O(1/\sqrt{i})$.

4.3 Douglas-Rachford for partially separable cones

We now solve the problem

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \|P_\eta x - c\|_2^2 \\ \text{subject to} & x \in \mathcal{K}, \end{array} \tag{4.12}$$

using decomposition methods based on the Douglas-Rachford splitting method [LM79, EB92, BC11]. Here, the objective is the squared distance between the subvector $P_\eta x = x_\eta$ and a given $|\eta|$-vector c:

$$\|P_\eta x - c\|_2^2 = \sum_{k=1}^{|\eta|} (x_{\eta(k)} - c_k)^2,$$

and is not strongly convex in general. The sets $\gamma_1, \ldots, \gamma_l$ are index sets (ordered subsets of $\{1, 2, \ldots, n\}$), and $\mathcal{C}_k$, $k = 1, \ldots, l$ are closed convex cones. If $\eta = \{1, 2, \ldots, n\}$ and $P_\eta = I$ then problem (4.12) is equivalent to the projection (4.5). When $\eta \neq \{1, 2, \ldots, n\}$ the problem is to project c on $P_\eta \mathcal{K}$, the projection of $\mathcal{K}$ on the subspace of vectors with support η. This problem appears in applications where data is incomplete; if $\hat{x}$ is an estimate of x, and only the elements $\hat{x}_i$ for $i \in \eta$ are reliable, then the best feasible prediction is the solution to (4.12) where $c = P_\eta \hat{x}$.

If for all $x \in \mathcal{K}$, $P_\eta x = 0 \Rightarrow x = 0$, then the set $P_\eta \mathcal{K}$ is closed. This is a special instance of Rockafellar's theorem 8.1 [Roc70], which states that if $\mathcal{K}$ is a closed convex cone and has a nontrivial intersection with the nullspace of B, then $B\mathcal{K}$ is a closed convex cone. In this case, the problem (4.12) has a unique optimal solution for the subvector $P_\eta x^\star$. However, the components of $x^\star$ outside η are not necessarily unique.

Problem (4.13) can be written equivalently by introducing $u = P_\eta P^T z$:

$$\begin{array}{ll} \underset{u,z}{\text{maximize}} & -\|u + c\|_2^2 + \|c\|_2^2 \\ \text{subject to} & P_\eta^T u = P^T z \\ & z \in \mathcal{C}^* \end{array} \tag{4.13}$$

where the solution $u^\star$ is the projection of $-c$ on the dual cone of $P_\eta \mathcal{K}$, which is given by $(P_\eta \mathcal{K})^* = \{u \mid P_\eta^T u \in \mathcal{K}^*\}$. The optimality conditions that relate the

two projections are

$$c = P_\eta x^\star - u^\star, \qquad x^\star \in \mathcal{K}, \qquad P_\eta^T u^\star \in \mathcal{K}^*, \qquad u^{\star T} P_\eta x^\star = 0.$$

Again, the primal and dual approaches are not completely equivalent. From the solution $x^\star$ of the primal problem (4.12) one obtains the dual solution $u^\star = P_\eta x^\star - c$. However from the dual solution $u^\star$ one only finds a partial primal solution $P_\eta x^\star$ and not the values $x_i^\star$ for $i \notin \eta$. The problem (4.13) without the conic constraint can be reformulated as the unconstrained minimization of

$$g(z) = -\|P^T z + P_\eta^T c\|_2^2 + \|c\|_2^2$$

and it is clear that g does not have a Lipschitz continuous gradient. For this reason, the dual decomposition methods cannot be applied to the nearness problem.

4.3.1 Consensus formulation for Douglas-Rachford method

We apply the Douglas-Rachford method to a reformulation of (4.12) and (4.13) that uses extra dummy variables and a consensus constraint. This similar to formulations mentioned in previous works (for example, problem 16 in [CP11b]).

Primal Douglas-Rachford We reformulate the primal in (4.12) as

$$\begin{array}{ll} \underset{x,y_k}{\text{minimize}} & (1/2)\|P_\eta x - c\|_2^2 + \sum_{k=1}^{l} \delta_{\mathcal{C}_k}(y_k) \\ \text{subject to} & P_{\gamma_k} x = y_k, \quad k = 1, \ldots, l. \end{array}$$

The variables are $x \in \mathbb{R}^n$ and an additional splitting variable $y = (y_1, y_2, \ldots, y_l)$. This problem has the form (2.23) if we take $\tilde{x} = (x, y)$, and

$$g(x, y) = (1/2)\|P_\eta x - c\|_2^2 + \sum_{k=1}^{l} \delta_{\mathcal{C}_k}(y_k), \qquad \mathcal{V} = \{(x, y) \mid y = Px\}.$$

The function g is separable with proximal operator

$$\mathbf{prox}_{tg}(x,y) = \begin{bmatrix} (I + tP_\eta^T P_\eta)^{-1}(x + tP_\eta^T c) \\ \Pi_{\mathcal{C}}(y) \end{bmatrix}. \tag{4.14}$$

Note that the inverse in the first block is the inverse of a strictly positive diagonal matrix, since $P_\eta^T P_\eta$ is diagonal with $(P_\eta^T P_\eta)_{ii} = 1$ if $i \in \eta$ and $(P_\eta^T P_\eta)_{ii} = 0$ otherwise. The projection on $\mathcal{C}$ in the second block reduces to l independent projections $\Pi_{\mathcal{C}}(y) = (\Pi_{\mathcal{C}_1}(y_1), \ldots, \Pi_{\mathcal{C}_l}(y_l))$. The projection on the subspace

$$\mathcal{V} = \{(x,y) \mid Px = y\}$$

is

$$\Pi_{\mathcal{V}}(x,y) = \begin{bmatrix} I \\ P \end{bmatrix} (I + P^T P)^{-1}(x + P^T y), \tag{4.15}$$

which is also simple to compute since $P^T P$ is diagonal; see (4.4). A summary of this method is given in Algorithm 7.

Algorithm 7 Douglas-Rachford method for primal problem (4.17).

Choose parameters $t > 0$, $\rho \in (0,2)$, initial z, w_1, $\ldots$, w_l.
for $i = 1, 2, \ldots$ until convergence
  Compute $x^i = \mathbf{prox}_{tf_1}(z^{i-1})$ using (4.14).
  Compute $u_k^i = \Pi_{\mathcal{C}_k^*}(w_k^{i-1})$ for $k = 1, \ldots, l$.
  Compute $(y^i, v^i) = \Pi_{\mathcal{V}}(2x^i - z^{i-1}, 2u^i - w^{i-1})$ using (4.15).
  Update z^i and w^i:

$$\begin{aligned} z^i &= z^{i-1} + \rho(y^i - x^i), \\ w_k^i &= w_k^{i-1} + \rho(v_k^i - u_k^i), \quad k = 1, \ldots, l. \end{aligned}$$

end

Dual Douglas-Rachford To apply the Douglas-Rachford method to the dual problem (4.13) we write it as

$$\begin{array}{ll} \underset{s,z_k}{\text{minimize}} & (1/2)\|s+c\|_2^2 + \sum_{k=1}^{l} \delta_{\mathcal{C}_k^*}(z_k) \\ \text{subject to} & \sum_{k=1}^{l} P_{\gamma_k}^T z_k = P_\eta^T s. \end{array}$$

This is in the form (2.23) with $\tilde{x} = (s, z)$,

$$g(s,z) = (1/2)\|s+c\|_2^2 + \sum_{k=1}^{l} \delta_{\mathcal{C}_k^*}(z_k), \quad \text{and} \quad \mathcal{V} = \{(s,z) \mid P_\eta^T s = P^T z\}.$$

The proximal operator of g is

$$\mathbf{prox}_{tg}(s,z) = \begin{bmatrix} (1+t)^{-1}(s-tc) \\ \Pi_{\mathcal{C}^*}(z) \end{bmatrix}.$$

The projection on $\mathcal{V}$ can be written as

$$\Pi_{\mathcal{V}}(s,z) = \begin{bmatrix} s \\ z \end{bmatrix} + \begin{bmatrix} -P_\eta \\ P \end{bmatrix} (P_\eta^T P_\eta + P^T P)^{-1}(P_\eta^T s - P^T z). \tag{4.16}$$

Here, $P_\eta^T P_\eta + P^T P$ is diagonal and positive definite, since by assumption the sets γ_k cover the entire index set $\{1, \ldots, n\}$. The Douglas-Rachford method applied to the dual problem is given in Alg. 8.

Algorithm 8 Douglas-Rachford method for alternate dual problem (4.13).

Choose parameters $t > 0$, $\rho \in (0, 2)$, initial u^0, v_1^0, $\ldots$, v_l^0.

for $i = 1, 2, \ldots$ until convergence

  Compute $u^i = (w^{i-1} + tc)/(1 + t)$

  Compute $z_k^i = \Pi_{\mathcal{C}_k^*}(r_k^{i-1})$ for $k = 1, \ldots, l$.

  Compute $(v^i, q^i) = \Pi_{\mathcal{V}'}(2u^i - w^{i-1}, 2z^i - r^{i-1})$ using (4.16).

  Update w^i and r^i:

$$\begin{aligned} w^i &= w^{i-1} + \rho(v^i - u^i) \\ r_k^i &= r_k^{i-1} + \rho(q_k^i - z_k^i), \quad k = 1, \ldots, l. \end{aligned}$$

end

4.3.2 Extensions

Problems (4.12) and (4.13) are instances of a more general optimization problem

$$\begin{array}{lll} \underset{x}{\text{minimize}} & f(x) & \\ \text{subject to} & Ax = b & \\ & x \in \mathcal{K} & \end{array} \qquad \begin{array}{lll} \underset{z,u}{\text{maximize}} & -f^*(P^T z + A^T u) + b^T u & \\ \text{subject to} & P^T z \in \mathcal{K}^* & \end{array} \tag{4.17}$$

where the primal variable is $x \in \mathbb{R}^n$, and the dual variables are $u \in \mathbb{R}^m$ and $z = (z_1, \ldots, z_l)$ are multipliers for the affine and conic constraints. Here, the matrix $A \in \mathbb{R}^{m \times n}$, and m is small. The function $f(x)$ is convex, but may not be strongly convex, and is assumed to have an easy-to-compute proximal operator. We also assume that $f(x)$ is a separable function, *i.e.* $f(x) = \sum_{i=1}^{n} f_i(x_i)$ and $f_1, \ldots, f_n$ are all convex functions with easy proximal operators. (An example of a separable f that we have not yet discussed is the one-norm: $f(x) = \|x\|_1 = \sum_i |x_i|$, in which $\mathbf{prox}_{tf}$ is the well-studied shrinkage operator.)

As an example, the projection on $\Pi_E(\mathbb{D}^p)$ can be written in the primal form, where $Ax = b$ captures the 0-diagonal constraint, and $\mathcal{K}$ is a partially separable cone with $\mathcal{C}_k$ the vectorized lower dimensional proper cones $\mathbb{D}_0^{|\beta_k|}$.

Primal extension There are two ways of handling the affine constraint in the primal problem. One is to incorporate it as an additional conic constraint. In this case, $\mathcal{K}$ is no longer proper, but is still closed, and the methods in this section can be easily extended if the proximal operators of f and f^* are easy to compute. Another way is to incorporate it in the objective, reformulating the primal of (4.17) as

$$\begin{array}{ll} \underset{x,u}{\text{minimize}} & f_1(x) + f_2(u) \\ \text{subject to} & Px = u \end{array}$$

where $f_1(x) = f(x) + \delta_{\{0\}}(Ax - b)$ and $f_2(u) = \delta_{\mathcal{C}}(u)$. The Douglas-Rachford method at each iteration computes the proximal operator of $f_1(x) + f_2(u)$ (which is separable) and the projection on the consensus constraint. This extension is similar in computational complexity to Algs. 7 and 8 if $f(x)$ and A are such that the optimization problem

$$\begin{array}{ll} \underset{\xi}{\text{minimize}} & f(\xi) + 1/(2t)\|\xi - x\|_2^2 \\ \text{subject to} & A\xi = b \end{array} \tag{4.18}$$

is easy to solve. For example, if $Ax = b$ is a sparsity constraint $P_\nu x = 0$ (as in a reformulation of the EDM projection) then the solution to (4.18) is simply $\Pi_\nu(\mathbf{prox}_{tf}(x))$ (and is cheap if $\mathbf{prox}_{tf} f(x)$ is cheap).

Dual extension The Douglas-Rachford method applied to the dual problem in (4.17) can then be rewritten as a minimization of $f_1(z) + f_2(y) + f_3(z)$ over the dual consensus constraint

$$\underset{s,y,z}{\text{maximize}} \quad -f^*(s) + b^T y + \delta_{\mathcal{C}^*}(z) \qquad \text{subject to} \quad P^T z + A^T y = s.$$

Solving this using the Douglas-Rachford method is computationally cheap if the proximal operator of f^* is easy to compute and the matrix $I + AA^T$ is easy to

factor. This is because the projection on the subspace

$$\mathcal{V} = \{(s, y, z) \mid P^T z + A^T y = u\}$$

is

$$\Pi_{\mathcal{V}}(s, y, z) = \begin{bmatrix} s \\ y \\ z \end{bmatrix} + \begin{bmatrix} -I \\ A \\ P \end{bmatrix} (I + P^T P + A^T A)^{-1}(s - P^T z - A^T y),$$

where, from the Sherman-Morrison-Woodbury inversion lemma,

$$(I + P^T P + A^T A)^{-1} = D + DA^T(I + AA^T)^{-1}AD$$

and $D = (I + P^T P)^{-1}$ is diagonal. In other words, if $(I + AA^T)$ is easy to factor, then $\Pi_{\mathcal{V}}$ is easy to compute.

In our numerical experiments, we do not use these extensions. Instead, we represent the affine constraints ($\mathbf{diag}(X) = 0$ for EDMs and $\mathbf{diag}(X) = \mathbf{1}$ for spectral norm constraints) as additional (non-proper) conic constraints. However, the extension may be interesting for different instances of $f(x)$.

4.4 Matrix nearness problems

We now apply the methods discussed in this section to solve the matrix nearness problem (4.1). We test the proposed algorithms on problems with sizes ranging from 1000 to $100,000$. The problems in the first set of experiments are constructed from thirteen symmetric sparsity patterns in the University of Florida sparse matrix collection [DH11]. In the second set of experiments, we consider a family of randomly generated sensor network node localization problems [BY04b, KW12, AW05]. The experiments are performed on an Intel Xeon CPU E31225 processor with 32 GB RAM, using MATLAB version 8.3.0 (2014a).

4.4.1 Matrix nearness with chordal sparsity

We first present the numerical results for solving the matrix nearness problems when $\mathcal{C} = \Pi_E(\mathbb{S}^p_+)$ and $\mathcal{C} = \Pi_E(\mathbb{D}^p)$, where E is chordal. Recall that in this case $\mathcal{C}$ can be interpreted as a partially separable cone. We apply the dual decomposition methods discussed in section 4.2 and the Douglas-Rachford method for the consensus formulation, discussed in section 4.3. (In the latter case, we use Alg. 7 where $\eta = \{1, \ldots, n\}$.)

In the dual gradient projection and Douglas-Rachford algorithms the main step per iteration is the projection on l dense PSD or EDM cones, and these projections can be done in parallel. In the dual block coordinate ascent method, some of the projections can be computed in parallel, if they are scheduled using a topological ordering on a clique tree [BP93]. Since the projections are the most expensive part of the algorithms, exploiting parallelism would result in a significant speedup. This possibility was not exploited in the code used for the experiments, which computes the projections sequentially.

Problem generation The patterns generated in this section are chordal extensions of thirteen nonchordal sparsity patterns from the University of Florida sparse matrix collection [DH11]. Table 4.1 gives some statistics for the patterns before the extension. The sparsity graphs of three of the larger patterns (`mario001`, `c-60`, `c-67`) are not connected, but since the largest connected component contains almost all the vertices (as shown in the table), we did not remove the smaller connected components.

The chordal extensions are computed in two steps. We first use graph elimination (or symbolic Cholesky factorization) using a fill-reducing reordering (the MATLAB `amd` reordering) to generate a first chordal extension. We then merge some of the smaller cliques of this extension according to heuristics discussed in [SAV14, §6.2]. Table 4.2 gives the statistics of the chordal extensions without

pattern	p	density
ex4	1601	1.24e-2
g3rmt3m3	5357	7.24e-3
barth4	6019	1.13e-3
c-37	8204	1.11e-3
tuma2	12992	2.92e-4
crack_dual	20141	1.98e-4
biplane-9	21701	1.79e-4
mario001	38434	1.39e-4
c-60	43640	1.57e-4
c-67	57975	1.58e-4
rail_79841	79841	8.69e-5
luxembourg_osm	114599	1.82e-5

Table 4.1: *Sparse matrix problems.* Thirteen symmetric sparsity patterns from the University of Florida sparse matrix collection. For each pattern we give the matrix order p and the density, defined as $(p + 2|E|)/p^2$.

clique merging, and table 4.3 gives the statistics after clique merging. This step is clearly advantageous, as the sparsity is not much increased, but the number of cliques drastically decreases. The average clique size also increases, but this is not a disadvantage, since the runtime of a eigenvalue decomposition does not differ significantly when $p < 50$.

For each pattern E, we compute five instances of C with lower triangular nonzeros generated from independent normalized Gaussian distributions. The numerical results in the following tables are averages over the five instances. An example of a sparsity pattern (before and after permutation and chordal embedding) for a large matrix ($p = 38,434$) is given in figure 4.1. Also given is the histogram of clique sizes, which can be used to analyze the efficiency of the decomposition methods on a particular problem.

Sparse PSD cones We first consider projections on the sparse PSD and PSD completable cones. When E is a chordal pattern, the projection of C on $\Pi_E(\mathbb{S}^p_+)$

p	density	m	avg. clique size	max. clique size
1601	3.24e-2	598	18.0	74
5357	2.97e-2	577	52.9	261
6019	6.12e-3	3637	11.1	89
8204	4.05e-3	7158	7.1	257
12992	2.90e-3	11051	5.7	241
20141	1.21e-3	17053	6.5	168
21701	1.41e-3	16755	8.0	147
38434	5.36e-4	30917	6.0	188
43640	1.37e-3	39468	6.2	954
57975	2.45e-4	52404	5.5	132
79841	5.31e-4	61059	8.7	337
114599	4.34e-5	113361	2.9	45

Table 4.2: *Chordal extensions.* The table shows the density, the number of cliques (m), and the average and maximum clique size, after a chordal extension, with no clique merging, for the patterns listed in table 4.1.

p	density	m	avg. clique size	max. clique size
1601	4.94e-2	94	46.3	74
5357	3.27e-2	267	80.8	261
6019	1.11e-2	317	42.2	89
8204	9.54e-3	1121	21.5	257
12992	5.22e-3	704	37.2	241
20141	2.80e-3	1098	35.7	168
21701	2.99e-3	1099	40.7	147
38434	1.25e-3	2365	28.1	188
43640	2.56e-3	6175	19.5	954
57975	9.04e-4	8875	14.9	132
79841	9.71e-4	4247	44.4	337
114599	2.02e-4	7035	18.9	58

Table 4.3: *Clique merging.* The table shows the density, the number of cliques (m), and the average and maximum clique size, after a chordal extension and clique merging, for the patterns listed in table 4.1.

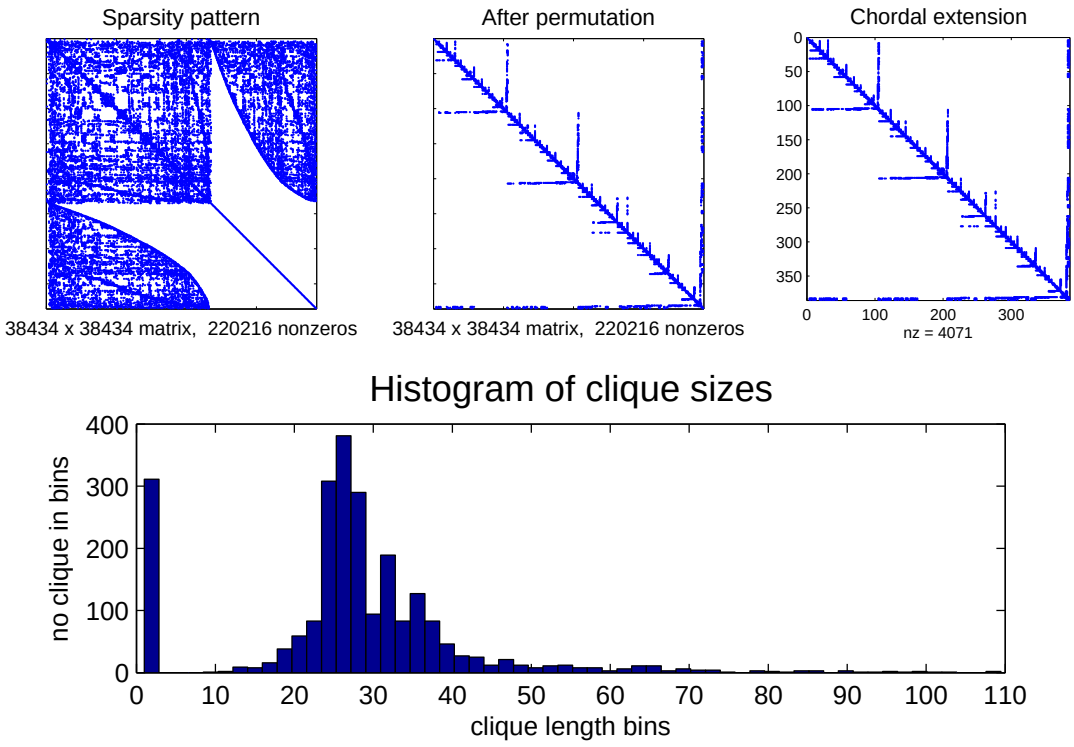

Figure 4.1: *UF matrices example.* Top: sparsity patterns for a matrix of order $p = 38,434$, with 0.014% nonzeros (`GHS_indef/mario001` in the UF matrix collection). The original pattern (left), the pattern after AMD permutation (center), and the chordal extension of the permuted pattern (right) are given. Bottom: histogram of the clique sizes in the decomposed problem. This is representative of most of the examples in which our decomposition methods are successful; the average clique size is around 25-50, and the maximum clique size is around 100-200.

simplifies to

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|X - C\|_F^2 \\ \text{subject to} & P_{\beta_k} X P_{\beta_k}^T \succeq 0, \quad k = 1, \ldots, m, \end{array} \tag{4.19}$$

where $\beta_1, \ldots, \beta_m$ are the cliques of the sparsity pattern. The dual of this problem is

$$\begin{array}{ll} \underset{Z_1,\ldots,Z_m}{\text{maximize}} & -\left\| \sum_{k=1}^m P_{\beta_k}^T Z_k P_{\beta_k} + C \right\|_F^2 + \|C\|_F^2 \\ \text{subject to} & Z_k \succeq 0, \quad k = 1, \ldots, m, \end{array} \tag{4.20}$$

and is equivalent to the projection of $-C$ on $\mathbb{S}_{E,+}^p$.

We apply the fast projected gradient and block coordinate ascent methods to the dual problem (4.20), and the Douglas-Rachford method to the primal problem (4.19) and the dual problem (4.20). The step size in the fast projected gradient method is $t = 1/L$. In the Douglas-Rachford methods we used $t = 1$, $\rho = 1.75$. (A different choice of t and ρ may accelerate the Douglas-Rachford methods.) A sample evolution plot comparing the three methods is given in figure 4.2. A tolerance $\epsilon = 10^{-3}$ was used in the the various stopping conditions given in chapter 2. The algorithms were terminated when the CPU time exceeded a maximum four hours. The runtimes are averages over five instances; we did not observe a significant variance in runtime or number of iterations for the different instances. To reduce computational overhead, the stopping conditions are tested every 25 iterations. The results are given in table 4.4.

Sparse EDM cones Table 4.5 shows the results of a similar experiment for the projection on the EDM completable cone by solving

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \|X - C\|_F^2 \\ \text{subject to} & P_{\beta_k} X P_{\beta_k}^T \in \mathbb{D}_0^{|\beta_k|}, \ k = 1, \ldots, m \\ & \mathbf{diag}(X) = 0 \end{array} \tag{4.21}$$

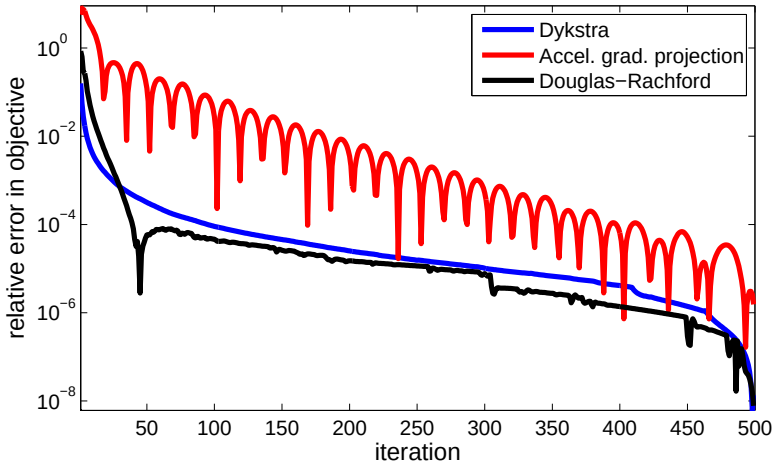

Figure 4.2: *Convergence.* Relative error in objective $|f(X^i) - f(X^\star)|/|f(X^\star)|$ for the projection of a matrix C on the cone of sparse matrices with PSD completion. The plot compares the two dual decomposition methods and the Douglas-Rachford method. The matrix C is 1601×1601, with chordal sparsity, and with 1.24% nonzeros.

p	total runtime				time/iteration				eig.
	F-PG	BCD	P-DR	D-DR	F-PG	BCD	P-DR	D-DR	dcp.
1601	2.7e1	3.5	4.4	5.0	1.2e-1	1.4e-1	1.8e-1	2.0e-1	1.5
5357	4.8e2	8.4e1	4.6e1	5.2e1	1.3	1.7	1.8	2.1	6.7e1
6019	1.2e2	9.6	1.3e1	1.6e1	3.4e-1	3.8e-1	5.1e-1	6.2e-1	9.4e1
8204	2.6e3	7.1e1	1.1e2	1.1e2	9.5e-1	1.4	1.4	1.5	2.4e2
12992	2.4e2	6.2e1	3.6e1	4.2e1	9.5e-1	1.2	1.4	1.7	9.2e2
20141	2.5e2	3.9e1	3.8e1	4.6e1	9.9e-1	1.6	1.5	1.9	3.4e3
21701	3.3e2	3.4e1	4.6e1	5.8e1	1.2	1.4	1.8	2.3	4.2e3
38434	4.7e2	4.7e1	6.2e1	7.8e1	2.1	1.9	2.5	3.1	2.3e4
43640	$>$ 4hr	1.9e3	1.6e3	1.5e3	1.0e1	1.9e1	1.6e1	1.5e1	3.5e4
57975	$>$ 4hr	1.4e2	1.1e3	1.1e3	3.5	5.7	6.4	6.2	OOM
79841	2.4e3	3.0e2	2.4e2	3.0e2	6.3	7.6	9.7	1.2e1	OOM
114599	5.3e2	5.5e1	1.0e2	1.2e2	2.6	2.2	4.0	4.6	OOM

Table 4.4: *Projection on chordal PSD completable matrices.* CPU times (in seconds) for the projection on $\Pi_E(\mathbb{S}^p_+)$. The total runtimes and times per iteration are given. The algorithms are: dual fast projected gradient method (F-PG), dual block coordinate ascent (BCD), primal Douglas-Rachford method (P-DR), and dual Douglas-Rachford method (D-DR).

p	total runtime				time/iteration			
	F-PG	BCD	P-DR	D-DR	F-PG	BCD	P-DR	D-DR
1601	7.6e1	1.2e1	5.7	5.2	1.5e-1	1.6e-1	2.3e-1	2.1e-1
4307	1.9e3	3.2e1	2.1e1	2.0e1	2.9e-1	4.1e-1	4.2e-1	3.9e-1
5357	1.1e3	2.3e2	5.6e1	5.3e1	1.5	1.9	2.2	2.1
6019	3.9e2	2.7e1	1.7e1	1.6e1	4.4e-1	5.0e-1	6.9e-1	6.2e-1
8204	8.6e3	2.7e2	8.5e1	8.3e1	1.2	2.7	1.7	1.7
12992	6.8e2	1.8e2	4.5e1	4.1e1	1.2	1.5	1.8	1.6
20141	9.1e2	1.3e2	5.0e1	4.4e1	1.3	1.7	2.0	1.8
21701	1.1e3	1.6e2	6.2e1	5.4e1	1.6	2.1	2.5	2.2
38434	1.4e3	8.6e2	8.3e1	7.2e1	2.1	6.1	3.3	2.9
43640	$>$ 4hr	1.1e4	9.1e2	9.3e2	1.4e1	7.1e1	1.8e1	1.9e1
57975	$>$ 4hr	4.6e3	5.5e2	5.6e2	4.8	3.7e1	7.3	7.5
79841	8.8e3	9.8e2	3.2e2	2.8e2	9.5	9.8	1.3e1	1.1e1
114599	2.5e3	1.3e2	1.3e2	1.1e2	3.3	5.4	5.1	4.5

Table 4.5: *Projection on chordal EDM completable matrices.* CPU times (in seconds) for the projection on $\Pi_E(\mathbb{D}^p)$. The total runtimes and times per iteration are given. The algorithms are: the dual fast projected gradient method (F-PG), the dual block coordinate ascent (BCD), the primal Douglas-Rachford method (P-DR), and the dual Douglas-Rachford method (D-DR).

where $\beta_1, \ldots, \beta_m$ are the cliques in the chordal pattern E. The strictly lower-triangular nonzero values of C are assigned according to a uniform distribution in $[0, 1]$. The diagonal of C is set to zero. The constraint set is treated as a partially separable set composed of $m+1$ lower dimensional constraints. The first m cones are the proper cones $\mathcal{C}_k = \mathbb{D}_0^{|\beta_k|}, k = 1, \ldots, m$. The $m+1$th cone is the zero-diagonal constraint $\mathcal{C}_{m+1} = \{X \mid \mathbf{diag}(X) = 0\}$. This partially separable cone $\mathcal{K}$ is closed but not proper.

In all cases when the method converged, the final objective values for the different algorithms are equal to two or three significant digits. (The unaccelerated projected gradient method in general took much longer than all other methods to converge, and the results are not included.) In general, the fast projected gradient method converged more slowly than the other methods. In all but four instances, the dual fast projected gradient method took between 200 and 1000 iterations and in two instances exceeded the time limit. In comparison, the dual block coordinate

ascent and Douglas-Rachford algorithms took between 25 and 150 iterations to converge. (For the block coordinate ascent algorithm, we count one cycle through all l cones as one iteration.)

4.4.2 Matrix nearness with nonchordal sparsity

In the next two sets of experiments we consider problems with nonchordal sparsity patterns. We first consider a reformulation of the projection on the PSD completable cone

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \displaystyle\sum_{\{i,j\}\in E} (X_{ij} - C_{ij})^2 \\ \text{subject to} & P_{\beta_k} X P_{\beta_k}^T \succeq 0 \end{array} \tag{4.22}$$

where $\beta_1, \ldots, \beta_m$ are the cliques of E', the chordal extension of E. Here, the parameter C has sparsity pattern E and the primal variable X is sparse with pattern E'. The formulation (4.22) can also be used in cases where it is desirable to compute some elements of X_{ij} where $\{i,j\} \notin E$, but the full completion of X is not needed. In this case, E' is the chordal extension of E plus the indices of the additional desired elements.

The dual of (4.22) is

$$\begin{array}{ll} \underset{S,Z_k}{\text{maximize}} & -\|S + C\|_F^2 + \|C\|_F^2 \\ \text{subject to} & S = \displaystyle\sum_{k=1}^m P_{\beta_k}^T Z_k P_{\beta_k} \\ & S_{ij} = 0, \quad \forall \{i,j\} \in E' \setminus E \\ & Z_k \succeq 0, \quad k = 1, \ldots, m \end{array}$$

Here, the dual solution $S^\star$ is also the projection of $-C$ on the sparse PSD cone, and $\Pi_E(X) - S = C$.

We use the patterns listed in table 4.1 as E and the chordal extensions listed in table 4.3 as E'. For each sparsity pattern E, we consider five randomly generated

p	total runtime		time/iteration		
	P-DR	D-DR	P-DR	D-DR	Eig. dcp.
1601	3.5e1	2.8e1	1.6e-1	1.6e-1	1.5
5357	4.4e2	3.8e2	1.6	1.7	6.7e1
6019	1.3e2	1.1e2	4.5e-1	4.8e-1	9.5e1
8204	4.1e2	3.6e2	1.3	1.3	2.4e2
12992	1.9e2	1.3e2	1.2	1.3	9.2e2
20141	2.6e2	2.1e2	1.3	1.4	3.4e3
21701	5.2e2	4.3e2	1.6	1.7	4.2e3
38434	2.4e2	1.7e2	2.4	2.3	2.3e4
43640	4.1e3	3.6e3	1.3e1	1.3e1	3.5e4
57975	1.4e3	1.2e3	5.6	5.3	OOM
79841	2.1e3	1.8e3	8.6	9.1	OOM
114599	7.1e2	4.5e2	3.7	3.6	OOM

Table 4.6: *Projection on nonchordal PSD completable matrices.* CPU times (in seconds) for the projection on $\Pi_E(\mathbb{S}^p_+)$. The Douglas-Rachford method is applied to the primal and dual problem form (P-DR and D-DR). The total runtimes and times per iteration are given, and compared against the runtime for a single eigenvalue decomposition of the full $p \times p$ matrix C.

matrices $C \in \mathbb{S}^p_E$, with lower-diagonal nonzero values chosen from a normal Gaussian distribution. The results for the primal and dual Douglas-Rachford methods (Algorithms 7 and 8) are given in table 4.6. Both the primal and dual method equivalently solves for $S^\star$. However, while the primal method also solves for $X^\star$, the dual method can only solve for $\Pi_E(X^\star)$.

Compared to the chordal problems (4.19), solving the nonchordal problems in general took more iterations (between 75 to 300 iterations). The final objective values for the primal and dual Douglas-Rachford methods are equal to around 2 or 3 significant digits.

4.4.3 Example: Sensor network node localization

We consider the sensor network node localization problem [BY04b, KW12, AW05] as an application of a matrix nearness problem. In this scenario, low-cost sensors are randomly scattered through space, and communicate to nearby sensors that

are within a radio range to estimate pairwise distances. Since the sensors are cheap, the pairwise distance estimates are noisy. The sensor network node localization problem is to recover the position of the sensors using the partial noisy distance readings, and can be written as an optimization problem

$$\begin{array}{ll} \underset{D}{\text{minimize}} & \displaystyle\sum_{\{i,j\}\in E} (D_{ij} - \hat{D}_{ij})^2 \\ \text{subject to} & D \in \mathbb{D}^p \\ & \mathbf{emdim}(D) = r \end{array} \tag{4.23}$$

where the $p \times p$ matrix $\hat{D}$ contains the partial squared pairwise distance measurements, and $\{i, j\} \in E$ if node i and j are within radio range. The final solution D is the EDM for the recovered positions $U = [u_1, \ldots, u_p]^T$, which can be computed from a factorization

$$(1/2)JDJ = UU^T \quad \text{where} \quad J = I - (1/p)\mathbf{1}\mathbf{1}^T.$$

The points $u_1, \ldots, u_p$ are possibly translated or rotated estimates of the true configuration. Finally, r is the dimension of the node positions, and is usually 2 or 3, corresponding to physical space.

Problem (4.23) is nonconvex because of the constraint on the embedding dimension. (For interesting problems, $p \gg r$.) We solve the convex relaxation of (4.23), where the embedding dimension constraint is dropped.) We construct problems with p varying from 1000 to $100,000$ and uniformly distributing p sensors in 3-D space. We then link nodes within radio range, with a decreasing range so that the sparsity pattern remains manageable. For some problems with $p \geq 10,000$, this resulted in graphs that were not connected. However the largest connected component contained over 95% of the vertices, so we did not remove these instances. Next we add noise to the nonzero elements of D to construct a

p	range (R)	density	density (extens.)	avr. # cliques	avg. clique size
1000	5.00e-2	3.67e-2	2.56e-1	46.2	122.8
1000	1.00e-1	9.15e-2	4.39e-1	36.6	206.3
5000	1.00e-2	3.93e-3	6.89e-2	228.0	121.6
5000	2.00e-2	1.03e-2	1.58e-1	191.4	240.7
10000	5.00e-3	1.47e-3	3.20e-2	474.4	102.1
25000	2.00e-3	3.96e-4	9.22e-3	1277.4	71.3
50000	1.00e-3	1.48e-4	2.70e-3	2821.4	46.3
75000	6.67e-4	8.34e-5	1.17e-3	4779.4	32.9
100000	5.00e-4	5.56e-5	6.04e-4	7214.4	24.4

Table 4.7: *Sensor network localization problem parameters.* Average problem statistics (density of the patterns E and E', number of cliques and average size of cliques) are given.

matrix $C \in \mathbb{S}^p_E$:

$$C_{ij} = \begin{cases} D_{ij} + N_{ij}, & \{i,j\} \in E, \\ 0, & \text{otherwise} \end{cases}$$

where for each i, j, the noise $N_{ij} = N_{ji}$ is drawn from a Gaussian distribution with mean zero and standard deviation 0.1. As a 2-D example, figure 4.3 shows a zoomed-in view of 5000 nodes connected in a unit square, along with the correspondng (permuted) sparsity pattern, and the histogram of clique sizes. We observe that though the network looks dense, the resulting sparsity level is very low; this is often true in applications where the sparsity pattern is determined by a network topology.

We solve problem (4.21) using the Douglas-Rachford method for the primal formulation (Algorithm 7). Table 4.7 gives the problem statistics and table 4.8 gives the runtime results, averaged over five instances. In general, each problem converged after 200 to 400 iterations.

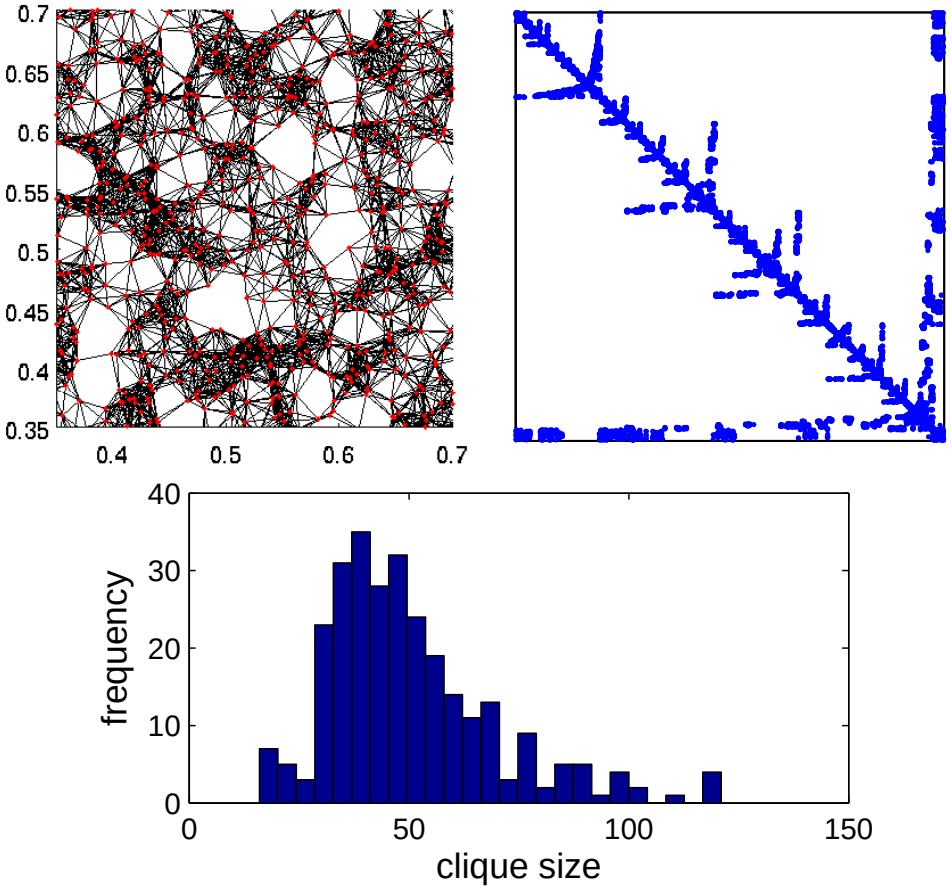

Figure 4.3: *Sensor network example.* Top left: zoomed in version of a 2-D sensor network with 5000 sensors in a unit square, with a radio range of 0.001. Top right: corresponding (permuted) sparsity pattern. Bottom: histogram of clique sizes for a chordal extension of the sparsity pattern.

p	range (R)	total runtime	time/ iteration	eigen. decomp.
1000	5.00e-2	1.5e2	6.9e-1	1.4
1000	1.00e-1	4.1e2	1.6	1.5
5000	1.00e-2	1.6e3	5.3	1.4e2
5000	2.00e-2	5.0e3	1.9e1	1.4e2
10000	5.00e-3	3.6e3	1.0e1	9.8e2
25000	2.00e-3	6.4e3	1.8e1	1.4e4
50000	1.00e-3	5.4e3	1.8e1	OOM
75000	6.67e-4	3.7e3	1.6e1	OOM
100000	5.00e-4	2.6e3	1.3e1	OOM

Table 4.8: *Sensor network localization runtimes.* Average runtime of decomposition methods, compared against a single eigenvalue decomposition.

4.4.4 Example: Projection on unit spectral norm ball

The *spectral norm* of a nonsymmetric matrix U is the absolute value of its maximum singular value (denoted as $\|U\|_2$). In terms of vector norms, the spectral norm can be defined as

$$\|U\|_2 = \max_{x:\|x\|_2=1} \|Ux\|_2.$$

We solve the following matrix nearness problems for nonsymmetric matrices C in $\mathbb{R}^{m\times n}$:

$$\begin{array}{ll} \underset{U}{\text{minimize}} & \|U-C\|_F^2 \\ \text{subject to} & U_{ij}=0, \quad \forall (i,j)\notin F \\ & \|U\|_2 \leq 1. \end{array} \tag{4.24}$$

The sparsity pattern $F \subseteq \{1,\ldots,m\}\times\{1,\ldots,n\}$ is a set of *ordered* index pairs and C is a noisy measurement. In the absence of the sparsity constraint (where $F = \{1,\ldots,m\}\times\{1,\ldots,n\}$), the solution can be computed using a singular-value decomposition (SVD)

$$U^* = \sum_{i=1}^{r} \hat{\sigma}_i q_i v_i^T, \qquad \text{where } \hat{\sigma}_i = \begin{cases} \sigma_i, & |\sigma_i| \leq 1, \\ \frac{\sigma_i}{|\sigma_i|}, & \text{else}, \end{cases}$$

where σ_i are the singular values of C, q_i and v_i the corresponding left and right singular vectors of C, and r is the rank of C. With the sparsity pattern, the problem can be solved using Dykstra's method, by alternating the projection on the spectral and sparsity constraint. However, this involves a sequence of $m \times n$ singular value decompositions, which may be costly if both m and n are large.

Alternatively, the problem can be framed as a problem with a PSD constraint

$$\begin{aligned} \underset{U}{\text{minimize}} \quad & \|U - C\|_F^2 \\ \text{subject to} \quad & U_{ij} = 0, \quad \forall (i,j) \notin F \\ & \begin{bmatrix} I & U \\ U^T & I \end{bmatrix} \succeq 0. \end{aligned} \tag{4.25}$$

The equivalence of these two formulations (4.24) and (4.25) can be shown by taking Schur-complements: for all $t > 0$,

$$\begin{bmatrix} tI & U \\ U^T & tI \end{bmatrix} \succeq 0 \iff tI - \frac{1}{t} U^T U \succeq 0 \iff \|U\|_2^2 \leq t.$$

Without the sparsity constraint, solving (4.25) (which requires an eigenvalue decomposition of order $m + n$) can be much more expensive than solving (4.24) (requiring only an order $\min\{m, n\}$ singular value decomposition). However, with the sparsity constraint, problem (4.25) is a matrix nearness problem with an affine constraint (unit diagonal), and can be solved using the Douglas-Rachford method applied to the consensus formulation (Algs 7 and 8).

We test the runtime of our algorithms in solving the unit spectral norm projection problem using sample nonsymmetric sparsity patterns from the University of Florida sparse matrix collection. An example of a sparsity pattern and resulting histogram of clique sizes is given in figure 4.4. The problem parameters and runtimes are given in table 4.9, and is compared (favorably) against the runtime

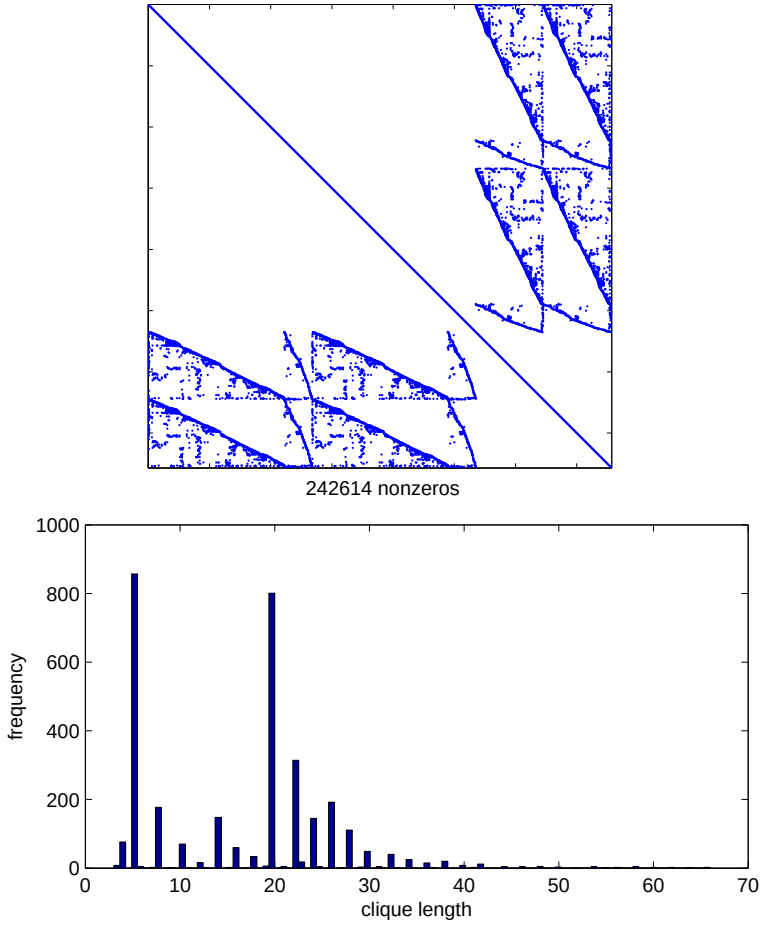

Figure 4.4: *Spectral norm example.* Top: aggregate sparsity pattern of problem (4.25), where the pattern of U is $26,722 \times 11,028$ with sparsity level 0.035% (labeled `psse0` in the UF sparse matrix collection). Bottom: corresponding histogram of clique sizes.

pattern	$p \times q$	density	no. cliques	avg. cl. size	total time	SVD
model8	2896×6464	1.4e-3	1035	30.0	1.5e3	4.4e1
deter4	3235×9133	6.5e-4	1183	16.1	9.0e1	7.2e1
deter8	3831×10905	5.3e-4	548	36.2	1.4e2	1.2e2
deter6	4255×12113	4.8e-4	608	36.2	1.6e2	1.7e2
fxm3_6	6200×12625	7.4e-4	2529	14.5	2.5e2	3.9e2
deter7	6375×18153	3.2e-4	908	36.3	2.4e2	5.2e2
psse0	26722×11028	3.5e-4	3269	16.5	1.7e2	2.2e3
scsd8-2c	5130×35910	6.1e-4	8711	11.1	1.3e3	2.9e2
stormg2-27	14441×37485	1.7e-4	6460	16.8	1.2e3	4.7e3
stormg2-125	66185×172431	3.8e-5	28661	17.9	1.1e4	OOM

Table 4.9: *Projection on the unit spectral norm ball.* Problem statistics and total CPU runtimes for the primal Douglas-Rachford method are given. The runtimes are compared against a single SVD, which represents a single iteration in a first-order method that does not use chordal decomposition.

of a single singular value decomposition of C. In general, if U is either very tall or very wide, then finding the SVD of U may be very quick compared to finding the eigenvalue decomposition of the augmented matrix in (4.25). For this reason, the runtimes in the SVD columns seem much smaller than in previous experiments. However, keeping in mind that the SVD runtime represents only one iteration in a method that does not use decomposition, then the decomposition methods still offer noticeable runtime reduction.

4.5 Discussion

We have presented decomposition methods for projections on sparse PSD, PSD completable, and EDM completable cones. By combining clique decomposition theorems for chordal sparsity patterns and first-order convex optimization algorithms, we are able to solve large problems with sizes ranging from 1000 to $100,000$. The key feature of the algorithms is that they involve only small dense eigenvalue decompositions, corresponding to the cliques in the chordal extension of the sparsity pattern. To underscore the importance of this property, we briefly

outline some alternative first-order methods that do not use the clique decomposition. The first problem (a projection on the sparse PSD cone) can be viewed as projection on the intersection $\mathbb{S}^p_E \cap \mathbb{S}^p_+$ of two sets. One can apply Dykstra's algorithm and solve the problem by alternating projection on the set of sparse matrices $\mathbb{S}^p_E$ and the dense PSD cone $\mathbb{S}^p_+$. Similarly, the projections on the PSD completable and EDM completable cones (problems II and III) can be viewed as minimizing a sum $f(X) + g(X)$, where

$$f(X) = \sum_{\{i,j\}\in E} (X_{ij} - C_{ij})^2, \qquad g(X) = \delta_{\mathcal{C}}(X)$$

where $\mathcal{C} = \mathbb{S}^p_+$ or $\mathbb{D}^p$. The Douglas-Rachford method for minimizing the sum requires projections on $\mathcal{C}$. Hence, at every iteration of these algorithms, a single eigenvalue decomposition of order p is needed. However, a full eigenvalue decomposition quickly becomes impractical for p greater than 10,000. On the machine used for our experiments, a single dense eigenvalue decomposition of order $p = 20,000$ takes 15 minutes to compute, and exceeds memory for $p \geq 50,000$. Sparse eigenvalue decomposition methods also pose difficulties. Even when the initial and final matrix variables in the algorithms are sparse, the intermediate variables (and in particular, the arguments to the projections on $\mathbb{S}^p_+$ and $\mathbb{D}^p$) are dense and, unless the method has almost converged, have close to $p/2$ positive and negative eigenvalues. On the same machine, a single full matrix projection using a sparse eigenvalue decomposition took more than an hour for a problem of size $p = 8204$ and more than eight hours for a problem of size $p = 12,992$. In comparison, the runtimes of the decomposition methods discussed in the paper depend less strongly on p and more on the sparsity pattern and density; a test problem with order $p = 21,701$ converged in 3 minutes, using about 2 seconds per iteration. In none of the test problems were we close to running out of memory.

There are also some interesting differences among the decomposition methods.

The dual block-coordinate ascent method (Dykstra's method) and the Douglas-Rachford method seem to converge in fewer iterations than the accelerated dual projection method. This is perhaps due to the fact that the Lipschitz constant L as defined in (4.9) is usually very large. In addition, the Douglas-Rachford methods are more general and can be applied to the problems with nonchordal sparsity patterns. However they converged more slowly on the test problems with nonchordal patterns. This is perhaps a result of the loss of strong convexity in the objective when E is nonchordal. We did not observe a difference in efficiency between the primal and dual Douglas-Rachford methods.

A general difficulty when applying the Douglas-Rachford algorithm is the sensitivity to the choice of the problem parameters t and ρ. We used the same fixed values for all the experiments, and it is possible that the performance can be further improved by tuning the parameters.

CHAPTER 5

Decomposition methods for sparse linear SDPs

In this chapter[1] we discuss the optimization problem

$$\begin{array}{ll} \underset{X}{\text{minimize}} & \mathbf{tr}(CX) \\ \text{subject to} & \mathbf{tr}(A_i X) = b_i, \quad i = 1, \ldots, m \\ & X \succeq 0. \end{array} \tag{5.1}$$

Here, the primal variable is a symmetric matrix $X \in \mathbb{S}^p$, and the problem data are the vector $b \in \mathbb{R}^m$ and the matrices C, $A_i \in \mathbb{S}^p$. The main difference between this problem and those discussed in the previous chapter is that the affine constraints $\mathbf{tr}(A_i X) = b_i$ may be nontrivial. (That is, projecting on the affine set defined by those equalities is computationally expensive.) However, we assume that the coefficient matrices C and $A_1, \ldots, A_m$ have some sparsity; we refer to the union of their sparsity patterns as the problem's *aggregate sparsity*. As described in the introduction, SDPs with aggregate sparsity appear in many applications, for example, in the relaxations of combinatorial problems [GR00, GLS88, LS91], polynomial optimization [Las01, Par00], and more. As an example, in section 5.5.2 we explore an SDP reformulation of the nearest EDM completion, using a ℓ_1 norm distance penalty. The aggregate sparsity here corresponds to the network topology of the nodes.

Problem (5.1) can be reformulated as an optimization problem over partially separable cones using similar techniques demonstrated in the previous chapter.

[1] This chapter is largely based on a previous journal paper; see [SAV14].

Additionally, we integrate the techniques similar to the conversion methods of [KKM11, FKM00]. These methods use a reformulation of the sparse SDPs which is easier to handle by interior-point algorithms. The decomposition method discussed in this paper follows this conversion approach and solves the reformulated problem using Spingarn's method.

In section 5.1 we begin by discussing SDPs with aggregate sparsity, and formulate the problem over a partially separable cone. When the sparsity pattern is chordal, the subsets corresponding to the lower dimensional cones can be arranged as vertices in a tree satisfying the running intersection property. In section 5.2, we describe the clique conversion method of [KKM11, FKM00] for linear conic optimization over partially separable cones. This method is used to exploit sparsity in the Schur complement of the KKT system, but can introduce many additional affine constraints. To mitigate this, in section 5.3 we apply a Douglas-Rachford method in which each subproblem is solved using a customized interior-point solver. In section 5.4 we use this method for linear semidefinite optimization, and give numerical results, with a short discussion in section 5.6.

5.1 Sparse SDPs

Problem (5.1) has aggregate sparsity pattern E if the matrices $C, A_i \in \mathbb{S}^p_E$. The dual of (5.1)

$$\begin{array}{ll} \underset{y,S}{\text{maximize}} & b^T y \\ \text{subject to} & \sum_{i=1}^m y_i A_i + S = C \\ & S \succeq 0 \end{array} \tag{5.2}$$

with variables $y \in \mathbb{R}^m$ and slack matrix $S \in \mathbb{S}^p$. The dual variable S is necessarily sparse at any dual feasible point, with the same sparsity pattern E. The primal variable X in (5.1) on the other hand, is dense in general, but one can note that the cost function and the equality constraints only depend on the entries of X in

the positions of the nonzeros of the sparsity pattern E. The other entries of X are arbitrary, as long as the matrix is positive semidefinite.

The primal (5.1) and dual problems (5.2) can therefore be viewed alternatively as conic linear optimization problems with respect to a pair of non-self-dual cones:

$$\begin{array}{lll} \underset{X}{\text{min.}} & \mathbf{tr}(CX) & \\ \text{s.t.} & \mathbf{tr}(A_i X) = b_i, \quad i = 1, \ldots, m & \\ & X \in \Pi_{E'}(\mathbb{S}^p_+) & \end{array} \qquad \begin{array}{lll} \underset{y,S}{\text{max.}} & b^T y & \\ \text{s.t.} & \sum_{i=1}^m y_i A_i + S = C & \\ & S \in \mathbb{S}^p_{E',+} & \end{array} \tag{5.3}$$

where E' is a chordal extension of E. Here, the primal and dual variables X and S are both matrices in $\mathbb{S}^p_{E'}$, though a feasible S has the added restriction of being in $\mathbb{S}^p_E$. The clique decomposition theorems from section 3.2 show that since E' is chordal, then $\Pi_{E'}(\mathbb{S}^p_+)$ is a partially separable cone. Using the same notation as in chapter 4, we denote

$$\mathcal{K} = \mathbf{vec}_{E'}(\Pi_{E'}(\mathbb{S}^p_+)) = \{\mathbf{vec}_{E'}(X) \mid X_{\beta_k,\beta_k} \succeq 0\}$$

the partially separable vector cone and

$$\mathcal{K}^* = \mathbf{vec}_{E'}(\Pi_{E'}(\mathbb{S}^p_+)) = \{\sum_{k=1}^{l} P_{\beta_k}^T Z_k P_{\beta_k} \mid Z_k \succeq 0\}$$

its dual, where $\beta_1, \ldots, \beta_l$ are the maximal cliques of E'. The cones $\mathcal{K}$ and $\mathcal{K}^*$ can also be expressed in terms of their lower dimensional cones

$$\mathcal{K} = \{x \mid P_{\gamma_k} x \in \mathcal{C}_k,\ k = 1, \ldots, l\}, \quad \mathcal{K}^* = \{\sum_{k=1}^{l} P_{\gamma_k}^T z_k \mid z_k \in \mathcal{C}_k^*,\ k = 1, \ldots, l\}$$

where the index sets γ_k are chosen such that

$$P_{\gamma_k} \mathbf{vec}_{E'}(X) = \mathbf{vec}(X_{\beta_k,\beta_k}), \tag{5.4}$$

and $\mathcal{C}_k = \mathbf{vec}(\mathbb{S}_+^{|\beta_k|}) = \mathcal{C}_k^*$. For this choice of $\mathcal{C}_k$, as shown in section 4.1, $\mathcal{K}$ and $\mathcal{K}^*$ are proper convex cones: closed, pointed, and with nonempty interior. (We do not consider the sparse EDM cone in this chapter.)

Tree representation As discussed in section 3.3, if E' is chordal with cliques $\beta_1, \ldots, \beta_l$, then there is a clique tree T such that every topological ordering satisfies the running intersection property (3.10). The sets β_k can then be partitioned as $\eta_k \cup \alpha_k$, where

$$\alpha_k = \beta_k \cap \beta_{\mathrm{par}(k)}, \quad \eta_k = \beta_k \setminus \beta_{\mathrm{par}(k)}.$$

Similarly, the sets γ_k as defined in (5.4) have a one-to-one relation with the cliques β_k. Therefore we do not have to distinguish between a clique tree T for E' and a spanning tree with the running intersection property spanning the intersection graph of the sets γ_k. The sets

$$\sigma_k = \gamma_k \cap \gamma_{\mathrm{par}(k)}, \quad \nu_k = \gamma_k \setminus \gamma_{\mathrm{par}(k)} \tag{5.5}$$

are in a one-to-one relation to the sets α_k and η_k respectively; for any $Z \in \Pi_{E'}(\mathbb{S}_+^p)$:

$$P_{\sigma_k}(\mathbf{vec}_{E'}(Z)) = \mathbf{vec}(Z_{\alpha_k\alpha_k}),$$

and $P_{\nu_k}(\mathbf{vec}_{E'}(Z))$ contains the elements in $Z_{\beta_k\beta_k}$ not included in Z_{α_k,α_k}.

5.2 The conversion method

We consider the linear optimization problem over partially separable cones

$$\begin{array}{ll} \underset{x}{\text{minimize}} & c^T x \\ \text{subject to} & Ax = b \\ & P_{\gamma_k} x \in \mathcal{C}_k, \quad k = 1, \ldots, l \end{array} \tag{5.6}$$

Here, $A \in \mathbb{R}^{m\times n}$, $c \in \mathbb{R}^n$, and $b \in \mathbb{R}^m$ are the problem parameters. The index sets γ_k are subsets of $\{1,\ldots,n\}$. We assume the index sets cover the entire index set; that is, $\bigcup_{k=1}^{l} \gamma_k = \{1,\ldots n\}$. The primal variable is $x \in \mathbb{R}^n$, and the primal conic constraint can be written compactly as

$$Px \in \mathcal{C}, \quad P = \begin{bmatrix} P_{\gamma_1} \\ \vdots \\ P_{\gamma_l} \end{bmatrix}, \quad \mathcal{C} = \mathcal{C}_1 \times \cdots \times \mathcal{C}_l.$$

The dual of (5.6) is

$$\begin{array}{ll} \underset{y,z}{\text{maximize}} & b^T y \\ \text{subject to} & P^T z + A^T y = c \\ & z_k \in \mathcal{C}_k^*, \quad k = 1,\ldots,l. \end{array}$$

The dual variables are $y \in \mathbb{R}^m$ and $z = (z_1,\ldots,z_l)$ where the subvectors z_k are of length $|\gamma_k|$. Equivalently, we write the dual conic constraint as

$$z \in \mathcal{C}^*, \quad \mathcal{C}^* = \mathcal{C}_1^* \times \cdots \times \mathcal{C}_l^*.$$

5.2.1 Problem reformulation

The objective of the conversion method is to reformulate (5.6) over "converted variables" $\tilde{x} \in \mathcal{C}$, under certain equality constraints, in order to decouple the variables and exploit sparsity in the Schur complement system of the KKT equations. The reformulation introduces a change of variables $\tilde{x} = Px$ and the primal

problem in (5.6) is rewritten as

$$\begin{array}{ll} \underset{\tilde{x}}{\text{minimize}} & \tilde{c}^T\tilde{x} \\ \text{subject to} & \tilde{A}\tilde{x} = b \\ & \tilde{x} \in \mathcal{V} \\ & \tilde{x} \in \mathcal{C}. \end{array} \tag{5.7}$$

Here, $\mathcal{V} = \mathbf{range}(P)$ and the subspace constraint $\tilde{x} \in \mathcal{V}$ ensures that there exists x where $Px = \tilde{x}$. The vector $\tilde{c}$ and matrix $\tilde{A}$ are chosen such that $P^T\tilde{c} = c$ and $\tilde{A}P = A$, and can be partitioned as

$$\tilde{A} = \left[\begin{array}{cccc} \tilde{A}_1 & \tilde{A}_2 & \cdots & \tilde{A}_l \end{array} \right], \qquad \tilde{c}^T = \left[\begin{array}{cccc} \tilde{c}_1^T & \tilde{c}_2^T & \cdots & \tilde{c}_l^T \end{array} \right] \tag{5.8}$$

where $\tilde{A}_k$ and $\tilde{c}_k$ are blocks of size $|\gamma_k|$. It is straightforward to find $\tilde{A}$ and $\tilde{c}$ that satisfy these conditions. For example, one can take $\tilde{A} = AJ$, $\tilde{c} = J^Tc$ where J is a left inverse of P; for example,

$$J = \left[\begin{array}{ccc} P_{\nu_1}^T P_{\nu_1} P_{\gamma_1}^T & P_{\nu_2}^T P_{\nu_2} P_{\gamma_2}^T & \cdots \quad P_{\nu_l}^T P_{\nu_l} P_{\gamma_l}^T \end{array} \right] \tag{5.9}$$

where $\nu_k \subseteq \gamma_k$, $k = 1, \ldots, l$ are chosen to partition $\{1, \ldots, n\}$. (For example, $\nu_k = \gamma_k \setminus \gamma_{\mathrm{par}(k)}$ where $\mathrm{par}(k)$ is the parent of vertex k in the clique tree T.) We will see later that different constructions of $\tilde{A}$ may offer certain advantages.

5.2.2 Example: conversion

Consider the linear optimization problem over $x \in \mathbb{R}^5$

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & \begin{bmatrix} a_{11} & a_{12} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 0 & 0 & a_{44} & a_{45} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \\ & (x_1, x_2) \in \mathcal{C}_1, \quad (x_2, x_3, x_4) \in \mathcal{C}_2, \quad (x_4, x_5) \in \mathcal{C}_3. \end{aligned} \tag{5.10}$$

One possible reformulation of (5.10) is

$$\begin{aligned} \text{minimize} \quad & \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}^T \begin{bmatrix} (\tilde{x}_1)_1 \\ (\tilde{x}_1)_2 \end{bmatrix} + \begin{bmatrix} 0 \\ c_3 \\ c_4 \end{bmatrix}^T \begin{bmatrix} (\tilde{x}_2)_1 \\ (\tilde{x}_2)_2 \\ (\tilde{x}_2)_3 \end{bmatrix} + \begin{bmatrix} 0 \\ c_5 \end{bmatrix}^T \begin{bmatrix} (\tilde{x}_3)_1 \\ (\tilde{x}_3)_2 \end{bmatrix} \\ \text{subject to} \quad & \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \tilde{x}_1 + \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & a_{24} \end{bmatrix} \tilde{x}_2 = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} && (5.11) \\ & \begin{bmatrix} a_{32} & a_{33} & a_{34} \end{bmatrix} \tilde{x}_2 = b_3 && (5.12) \\ & \begin{bmatrix} a_{44} & a_{45} \end{bmatrix} \tilde{x}_3 = b_4 && (5.13) \\ & (\tilde{x}_1)_2 = (\tilde{x}_2)_1, \quad (\tilde{x}_2)_3 = (\tilde{x}_3)_1 && (5.14) \\ & \tilde{x}_1 \in \mathcal{C}_1, \quad \tilde{x}_2 \in \mathcal{C}_2, \quad \tilde{x}_3 \in \mathcal{C}_3. && (5.15) \end{aligned}$$

The objective can be written more compactly as $\tilde{c}^T \tilde{x}$ where

$$\tilde{c}_1 = (c_1, c_2), \quad \tilde{c}_2 = (0, c_3, c_4), \quad \tilde{c}_3 = (0, c_5)$$

and $\tilde{c} = (\tilde{c}_1, \tilde{c}_2, \tilde{c}_3)$. The constraints (5.11), (5.12), and (5.13) correspond to the affine constraint $\tilde{A}\tilde{x} = b$, where

$$\tilde{A}_1 = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad \tilde{A}_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & a_{24} \\ a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \tilde{A}_3 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ a_{44} & a_{45} \end{bmatrix}$$

and $\tilde{A} = \begin{bmatrix} \tilde{A}_1 & \tilde{A}_2 & \tilde{A}_3 \end{bmatrix}$. The constraint (5.14) are the consistency constraints that ensures there exists a feasible x to (5.10) where $Px = \tilde{x}$. The final constraint (5.15) decouples the overlapping conic constraints in (5.10) into three separate conic constraints.

5.2.3 Representation of consistency set

For $\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_l)$, the consistency constraint $\tilde{x} \in \mathcal{V}$ is needed to ensure that the variables $\tilde{x}_k$ can be interpreted as copies $\tilde{x}_k = P_{\gamma_k} x$ of overlapping subvectors of some $x \in \mathbb{R}^n$. One way to do this is to enforce

$$\tilde{x} \in \mathcal{V} \iff P_\sigma(P_{\gamma_j}^T \tilde{x}_j - P_{\gamma_k}^T \tilde{x}_k) = 0$$

for each pair of sets γ_j, γ_k whenever $\sigma = \gamma_j \cap \gamma_k$ is nonempty. However, enforcing consistency for every pair of overlapping cliques may result in many redundant constraints. Since γ_k also have the special property that they can be arranged as vertices of a tree T satisfying the RIP, it suffices to only enforce consistency between adjacent cliques in T. With the sets ν_k and σ_k as defined in (5.5), then,

$$\tilde{x} \in \mathcal{V} \iff P_{\sigma_j}(P_{\gamma_j}^T \tilde{x}_j - P_{\gamma_k}^T \tilde{x}_k) = 0, \quad k = 1, \ldots, l, \quad \gamma_j \in \mathrm{ch}(\gamma_k). \tag{5.16}$$

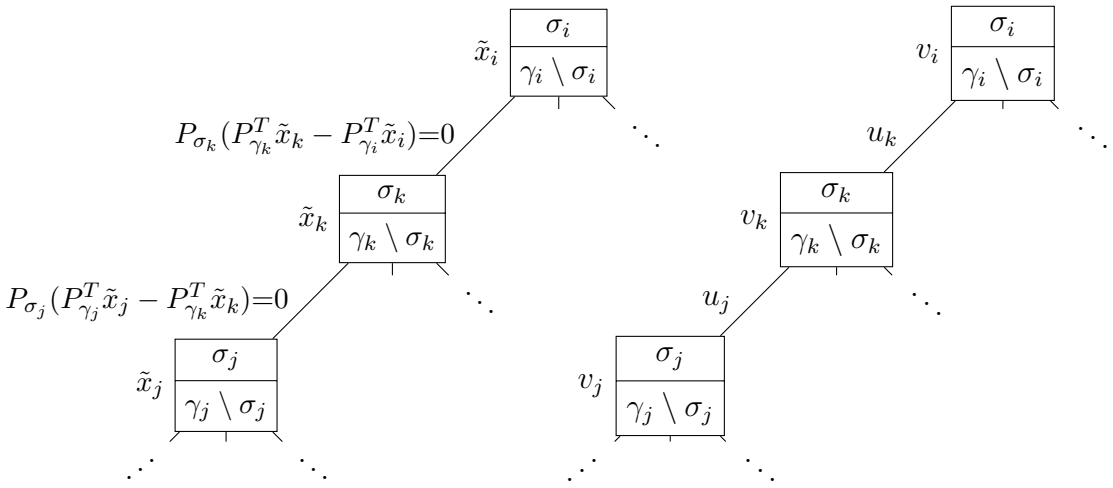

Figure 5.1: *The subspaces $\mathcal{V}$ and $\mathcal{V}^\perp$.* The figures show three vertices of the intersection tree. The left-hand figure illustrates $\mathcal{V}$. We associate the subvector $\tilde{x}_k$ of $\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_l)$ with vertex k in the tree and associate a consistency constraint $P_{\sigma_j}(P_{\gamma_j}^T \tilde{x}_j - P_{\gamma_k}^T \tilde{x}_k) = 0$ with the edge between vertex j and its parent k. Then $(\tilde{x}_1, \ldots, \tilde{x}_l)$ is in $\mathcal{V}$ if and only if the consistency constraints are satisfied. The right-hand figure illustrates $\mathcal{V}^\perp$. Here we associate the subvector v_k of $v = (v_1, \ldots, v_l)$ with vertex k in the tree and a vector $u_j \in \mathbb{R}^{|\sigma_j|}$ with the edge between vertex j and its parent k. Then $v \in \mathcal{V}^\perp$ if and only there exist values of u_k such that $v_k = P_{\gamma_k}(P_{\sigma_k}^T u_k - \sum_{\gamma_j \in \mathrm{ch}(\gamma_k)} P_{\sigma_j}^T u_j)$.

Likewise, a vector $v = (v_1, \ldots, v_l)$ is in $\mathcal{V}^\perp$ if and only if there exist $u_j \in \mathbb{R}^{|\sigma_j|}$, $j = 1, \ldots, l$, such that

$$v_k = P_{\gamma_k}(P_{\sigma_k}^T u_k - \sum_{\gamma_j \in \mathrm{ch}(\gamma_k)} P_{\sigma_j}^T u_j), \quad k = 1, \ldots, l. \tag{5.17}$$

The spanning trees for both the primal and dual consistency constraints are illustrated in the left and right hand side of figure 5.1, respectively.

We can express (5.16) in terms of matrices as follows. Define an $l \times l$ matrix K with elements

$$K_{ij} = \begin{cases} 1 & i = j \\ -1 & i = \mathrm{par}(j) \\ 0 & \text{otherwise.} \end{cases}$$

The matrix K is the transpose of the vertex-arc incidence matrix of the spanning

tree T, if we direct the arcs from children to parents. Then the equations (5.16) and (5.17) can be written succinctly as

$$B\tilde{x} = 0, \qquad v = B^T u,$$

for some $u = (u_1, \ldots, u_l) \in \mathbb{R}^{|\sigma_1|} \times \cdots \times \mathbb{R}^{|\sigma_l|}$ and

$$B = \begin{bmatrix} P_{\sigma_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & P_{\sigma_l} \end{bmatrix} (K^T \otimes I_n) \begin{bmatrix} P_{\gamma_1}^T & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & P_{\gamma_l}^T \end{bmatrix}. \tag{5.18}$$

Here, I_n is the $n \times n$ identity matrix and $K \otimes I_n$ is the Kronecker product. The matrix B is an $l \times l$ block matrix with diagonal blocks $P_{\sigma_k} P_{\gamma_k}^T$, $k = 1, \ldots, l$, and $-P_{\sigma_j} P_{\gamma_k}^T$ in block row j and block column i, where $k = \mathrm{par}(j)$. The rest of the matrix B is zero. If the vertices of the intersection tree are numbered in a topological ordering (*i.e.*, each vertex receives a lower number than its parent), then the matrix K is lower triangular. Note that the sparsity pattern of $B^T B$ can be embedded in the sparsity pattern of PP^T. The matrix BB^T, on the other hand, is not necessarily sparse.

5.2.4 Interior-point method

The conversion method of [KKM11, FKM00] solves the reformulated problem (5.7) using an interior-point method. In this section we explain the main advantage of this reformulation. We also describe its limitations, and in the next section, propose a first-order splitting method that overcomes this limitation.

Recall from section 2.5 that the main step per iteration of an interior-point

method is to solve the KKT system, which for (5.6) is

$$\begin{bmatrix} H & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} d_x \\ d_y \end{bmatrix}.$$

Here we will assume that $H = \phi(w)$ where ϕ is a logarithmic barrier function for $\mathcal{K}$, and w is some point in **int** $\mathcal{K}$. In most implementations, the KKT equation (5.20) is solved by eliminating $\Delta\tilde{x}$ and solving the Schur complement system

$$AH^{-1}A^T\Delta y = AH^{-1}d_x - d_y. \tag{5.19}$$

For nonpolyhedral cones (and specifically, for the vectorized PSD cone) the Hessian of the logarithmic barrier is dense. Therefore, the Hessian H of the logarithmic barrier for the partially separable cone $\mathcal{K}$ consists of overlapping principal dense submatrices, and H^{-1} is in general dense.

In contrast, after conversion, the KKT system for problem (5.7) solves a larger system

$$\begin{bmatrix} \tilde{H} & \tilde{A}^T & B^T \\ \tilde{A} & 0 & 0 \\ B & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta\tilde{x} \\ \Delta y \\ \Delta u \end{bmatrix} = \begin{bmatrix} d_{\tilde{x}} \\ d_y \\ d_u \end{bmatrix} \tag{5.20}$$

where now, $\tilde{H} = \mathbf{diag}(H_1, \ldots, H_l)$ is a positive definite block-diagonal scaling matrix and $\tilde{H}^{-1}$ is block diagonal. Here, B is as defined in (5.18). The Schur complement system of (5.20) after eliminating $\Delta\tilde{x}$ is

$$\begin{bmatrix} \tilde{A}\tilde{H}^{-1}\tilde{A}^T & \tilde{A}\tilde{H}^{-1}B^T \\ B\tilde{H}^{-1}\tilde{A}^T & B\tilde{H}^{-1}B^T \end{bmatrix} \begin{bmatrix} \Delta y \\ \Delta u \end{bmatrix} = \begin{bmatrix} \tilde{A}\tilde{H}^{-1}d_{\tilde{x}} - d_y \\ B\tilde{H}^{-1}d_{\tilde{x}} - d_u \end{bmatrix}. \tag{5.21}$$

The sparsity of $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$ depends on the choice of $\tilde{A}$, and as will be shown, motivates different choices of $\tilde{A}$. The 2,2 block $B\tilde{H}^{-1}B^T$ of (5.21) on the other hand may be quite dense.

An interior-point method for the reformulated problem can exploit the sparsity of $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$ by solving (5.21) using a sparse Cholesky factorization method. This matrix has the same sparsity pattern as the system obtained by eliminating Δu in (5.21), *i.e.*,

$$M_1 = \tilde{A}(\tilde{H}^{-1} - \tilde{H}^{-1}B^T(B\tilde{H}^{-1}B^T)^{-1}B\tilde{H}^{-1})\tilde{A}^T.$$

The matrix M_1 is often dense (due to the $B\tilde{H}^{-1}B^T$ term), even if $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$ is sparse. Alternatively, if $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$ is nonsingular, one can also explicitly eliminate Δy and reduce it to a dense linear equation in Δu with coefficient matrix

$$M_2 = B(\tilde{H}^{-1} - \tilde{H}^{-1}\tilde{A}^T(\tilde{A}\tilde{H}^{-1}\tilde{A}^T)^{-1}\tilde{A}\tilde{H}^{-1})B^T.$$

To form matrix M_2 one can take advantage of sparsity in $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$. (This is the approach taken in [KKK08].) Whichever method is used for solving (5.21), the advantage of the enhanced sparsity resulting from the sparse 1,1 block $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$ must be weighed against the increased size of the reformulated system (5.21). This is especially important for semidefinite programming, where the extra variables Δu are vectorized matrices, so the difference in size of the two Schur complement systems is very substantial.

In the regime where the number of rows in B is small compared to than that in $\tilde{A}$, an interior-point method applied to (5.7) will be much more efficient than when applied to (5.6). In this chapter we consider the opposite regime, where we show in section 5.4 that even for modest amounts of overlap in the sets γ_k, solving (5.21) becomes expensive, even when compared to solving the original, unconverted problem. However, the per-iteration cost of solving (5.7) using an interior-point method is significantly reduced if we remove the consistency constraint.

5.3 The Spingarn-IPM method

Motivated by the high cost of solving the KKT equations (5.21) of the converted problem we now examine the alternative of using a first-order splitting method to exploit the sparsity of $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$. The converted problem (5.7) can be written as

$$\begin{array}{ll} \underset{\tilde{x}}{\text{minimize}} & f(\tilde{x}) = \tilde{c}^T\tilde{x} + \delta(\tilde{A}\tilde{x} - b) + \delta_{\mathcal{C}}(\tilde{x}) \\ \text{subject to} & \tilde{x} \in \mathcal{V} \end{array} \tag{5.22}$$

where δ and $\delta_{\mathcal{C}}$ are the indicator functions for $\{0\}$ and $\mathcal{C}$, respectively. The function $f(\tilde{x})$ is closed if there exists $\tilde{x} \in \mathcal{C}$ where $\tilde{A}\tilde{x} = b$; see [Roc70, theorem 9.3]. We apply Spingarn's method of partial inverses (section 2.4.3). Starting at some z^0, the following three steps are repeated:

$$\begin{aligned} \tilde{x}^i &= \mathbf{prox}_{tf}(w^{i-1}) \\ u^i &= \Pi_{\mathcal{V}}(2\tilde{x}^i - w^{i-1}) \\ w^i &= w^{i-1} + \rho(u^i - \tilde{x}^i). \end{aligned}$$

The algorithm depends on two parameters: a positive steplength t and a *relaxation parameter* ρ, which must remain in the interval $0 < \rho < 2$. We refer to section 2.4.3 for more analysis on Spingarn's method in general (*i.e.* convergence properties and stopping conditions).

5.3.1 Projection on consistency constraints

It can be shown that the Euclidean projection of $\tilde{x}$ on $\mathcal{V}$ is

$$\Pi_{\mathcal{V}}(\tilde{x}) = P(P^TP)^{-1}P^T\tilde{x},$$

which is easy to compute; as in the previous chapter, it simply averages the overlapping terms. (The matrix P^TP is a diagonal matrix where $(P^TP)_{ii}$ is the number of cliques containing the index i.)

For each $i \in \{1, 2, \ldots, n\}$, define $M(i) = \{k \mid i \in \gamma_k\}$ the set of cliques that include i. The sets γ_k for $k \in M(i)$ define a subtree (this is a consequence of the running intersection property). Then $\bar{x} = (P^TP)^{-1}P^T\tilde{x}$ is the n-vector with components

$$\bar{x}_i = \frac{(\sum_{k \in M(i)} P_{\gamma_k}^T \tilde{x}_k)_i}{|M(i)|}, \quad i = 1 \ldots, n,$$

a simple average of the corresponding components of $\tilde{x}_k$, for the sets γ_k that contain i. In other words, each element of $\bar{x}_i$ can be computed using only the subvectors $\tilde{x}_k$ where $k \in M(i)$ (which can be significantly fewer than the total number of cliques). This property may be exploited in distributed computing.

5.3.2 Evaluation of the proximal operator

The value $\tilde{x} = \mathbf{prox}_{tf}(w)$ of the prox-operator of the function f in (5.22) is the primal solution in the pair of conic quadratic optimization problems (conic QPs)

$$\begin{array}{lll} \min\limits_{\tilde{x}} & \tilde{c}^T\tilde{x} + \dfrac{1}{2t}\|\tilde{x} - w\|_2^2 & \max\limits_{y,\tilde{s}} \quad b^Ty - \dfrac{t}{2}\|\tilde{c} - \tilde{A}^Ty - t^{-1}w - \tilde{s}\|_2^2 \\ \text{s.t.} & \tilde{A}\tilde{x} = b, \quad \tilde{x} \in \mathcal{C} & \text{s.t.} \quad \tilde{s} \in \mathcal{C}^* \end{array} \tag{5.23}$$

with primal variables $\tilde{x}$ and dual variables y, $\tilde{s}$. Equivalently, $\tilde{x}$, y, $\tilde{s}$ satisfy

$$\tilde{A}\tilde{x} = b, \qquad \tilde{A}^Ty + \tilde{s} + t^{-1}(w - \tilde{x}) = \tilde{c}, \qquad \tilde{x} \in \mathcal{C}, \qquad \tilde{s} \in \mathcal{C}^*, \qquad \tilde{x}^T\tilde{s} = 0. \tag{5.24}$$

We will assume that the proximal operator of f is computed exactly, *i.e.*, we do not explore the possibility of speeding up the algorithm by using inexact proximal operator evaluations. This is justified if an interior-point method is used for solving (5.23), since interior-point methods achieve a high accuracy and offer only

a modest gain in efficiency if solutions with low accuracy are acceptable. Using the optimality conditions (5.24) that characterize $\tilde{x}^i = \mathbf{prox}_{tf}(w^{i-1})$ we can then be more specific about the accuracy of $\tilde{x}^i$ as an approximate solution of the conic LP (5.7). By solving the primal and dual conic QPs we find $\tilde{x}^i$, y^i, $\tilde{s}^i$ that satisfy $\tilde{x}^i \in \mathcal{C}$, $\tilde{s}^i \in \mathcal{C}^*$, and

$$\tilde{A}\tilde{x}^i = b, \qquad \tilde{A}^T y^i + \tilde{s}^i + v^i = \tilde{c}, \qquad (\tilde{x}^i)^T \tilde{s}^i = 0 \tag{5.25}$$

where $v^i = t^{-1}(w^{i-1} - \tilde{x}^i)$. These are exactly the primal-dual optimality conditions for the conic LPs (5.7), except for the constraints involving $\mathcal{V}$ and $\mathcal{V}^\perp$. The primal and dual residuals $r_{\mathrm{p}}^i = P_{\mathcal{V}}(\tilde{x}^i) - \tilde{x}^i$ and $r_{\mathrm{d}}^i = -P_{\mathcal{V}}(v^i)$ measure the deviation of $\tilde{x}^i$ from $\mathcal{V}$ and of $\tilde{v}^i$ from $\mathcal{V}^\perp$.

We now comment on the cost of evaluating the proximal operator by solving the conic QPs (5.23). An interior-point method applied to this problem requires the solution of KKT systems of the form

$$\begin{bmatrix} t^{-1}I + H & \tilde{A}^T \\ \tilde{A} & 0 \end{bmatrix} \begin{bmatrix} \Delta\tilde{x} \\ \Delta y \end{bmatrix} = \begin{bmatrix} d_{\tilde{x}} \\ d_y \end{bmatrix}$$

where H is a block-diagonal positive definite scaling matrix. As before, we assume that the diagonal blocks of H are of the form $H_k = \nabla^2\phi_k(w_k)$ where ϕ_k is a logarithmic barrier of $\mathcal{C}_k$. The cost per iteration of evaluating the proximal operator is dominated by the cost of assembling the coefficient matrix

$$\tilde{A}(t^{-1}I + H)^{-1}\tilde{A}^T = \sum_{k=1}^{l} \tilde{A}_k(t^{-1}I + H_k)^{-1}\tilde{A}_k^T \tag{5.26}$$

in the Schur complement equation

$$\tilde{A}(t^{-1}I + H)^{-1}\tilde{A}^T\Delta y = \tilde{A}(t^{-1}I + H)^{-1}d_x - d_y$$

and the cost of solving the Schur complement system. For many types of conic LPs the extra term $t^{-1}I$ in (5.26) can be handled by simple changes in the interior-point algorithm. This is true in particular when H_k is diagonal or diagonal-plus-low-rank, as is the case when $\mathcal{C}_k$ is a nonnegative orthant or second-order cone. For positive semidefinite cones the modifications are more involved and will be discussed in section 5.4. In general, it is therefore fair to assume that in most applications the cost of assembling the Schur complement matrix in (5.26) is roughly the same as the cost of computing $\tilde{A}^T H^{-1} \tilde{A}^T$. Since the Schur complement matrix in (5.26) is sparse (it has the same sparsity as $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$), it can be factored at a smaller cost than its counterpart (5.21) for the reformulated conic LPs. Depending on the sparsity level of the Schur complement system, one evaluation of the proximal operator via an interior-point method can be substantially less expensive than solving the reformulated problems by an interior-point method.

5.3.3 Sparsity in the Schur complement system

The efficiency in both the conversion method of [KKM11] and in our splitting method depends on the sparsity level of the matrix $\tilde{A}\tilde{H}^{-1}\tilde{A}^T$. In the conversion method, it is the 1,1 block in the Schur complement system (5.21), and in the splitting method, it is representative of the sparsity in the Schur complement system of the proximal operator (5.26). Recall that $\tilde{H}$ and $\tilde{H}^{-1}$ are block diagonal matrices, where each block is of the order of $|\gamma_k|$.

As an example, consider a small conic LP with $m = 4$, $n = 6$, with index sets

$$\gamma_1 = \{1, 2, 6\}, \quad \gamma_2 = \{2, 5, 6\}, \quad \gamma_3 = \{4, 6\}, \quad \gamma_4 = \{3, 5\}$$

and clique tree as given in figure 3.2. Specifically, 2 = par(1), 2 = par(3), and

$4 = \text{par}(2)$, and the sets

$$\nu_1 = \{1\}, \quad \nu_2 = \{2, 6\}, \quad \nu_3 = \{4\}, \quad \nu_4 = \{3, 5\}$$

partition the space $\{1, \ldots, 6\}$. Consider a constraint matrix A with zeros in the following positions:

$$A = \begin{bmatrix} A_{11} & A_{12} & 0 & 0 & 0 & A_{16} \\ 0 & A_{22} & 0 & 0 & A_{25} & A_{26} \\ 0 & 0 & 0 & A_{34} & 0 & A_{36} \\ 0 & 0 & A_{43} & 0 & A_{45} & 0 \end{bmatrix}.$$

In other words, in this example, equality i in $Ax = b$ involves only variables x_k for $k \in \gamma_i$. The primal reformulated problem has a variable $\tilde{x} = (\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4)$ in $\mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^2 \times \mathbb{R}^2$. If we define $\tilde{A}$ (following (5.8) and (5.9)), we obtain

$$\tilde{A}_1 = \left[\begin{array}{ccc|ccc|cc|cc} A_{11} & 0 & 0 & A_{12} & 0 & A_{16} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{22} & 0 & A_{26} & 0 & 0 & 0 & A_{25} \\ 0 & 0 & 0 & 0 & 0 & A_{36} & A_{34} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & A_{43} & A_{45} \end{array}\right].$$

With this choice the 4×4 matrix $\tilde{A}_1 \tilde{H}^{-1} \tilde{A}_1^T$ has sparsity pattern shown on the left of figure 5.2. On the other hand, if we choose

$$\tilde{A}_2 = \left[\begin{array}{ccc|ccc|cc|cc} A_{11} & A_{12} & A_{16} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{22} & A_{25} & A_{26} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & A_{34} & A_{36} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & A_{43} & A_{45} \end{array}\right],$$

then the sparsity pattern of $\tilde{A}_2 \tilde{H}^{-1} \tilde{A}_2^T$ is diagonal, as shown on the right of figure 5.2. In cases where the matrix $\tilde{A} H^{-1} \tilde{A}^T$ is block diagonal with b blocks,

$$\begin{bmatrix} \bullet & \bullet & \bullet & 0 \\ \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & 0 \\ 0 & \bullet & 0 & \bullet \end{bmatrix} \qquad \begin{bmatrix} \bullet & 0 & 0 & 0 \\ 0 & \bullet & 0 & 0 \\ 0 & 0 & \bullet & 0 \\ 0 & 0 & 0 & \bullet \end{bmatrix}$$

Figure 5.2: *Sparsity of Schur complement matrix.* Left: sparsity pattern of $\tilde{A}_1\tilde{H}^{-1}\tilde{A}_1$, where $\tilde{A}_1 = AJ$, and J is as defined in (5.9). For this particular problem, if $\tilde{A}_1$ is picked this way, the Schur complement matrix of the converted problem is not very sparse. Right: sparsity pattern of $\tilde{A}_2\tilde{H}^{-1}\tilde{A}_2$ where A_2 is chosen to optimize the sparsity in the Schur complement matrix. Note that for general problems, the optimal converted coefficient may not be so obvious.

then the evaluation of the proximal operator (5.23) decomposes into b separate problems, which can be solved in parallel.

In general, an automated way of finding the optimal choice of $\tilde{A}$ remains an open problem. However, as illustrated in this small example, sometimes there are obvious solutions. It may be advantageous to use a heuristic algorithm, which for each constraint, greedily finds cliques that best covers the nonzeros in that constraint.

5.4 Linear semidefinite optimization

We now return to the sparse semidefinite optimization problems (5.3). The sparsity pattern E' is the chordal extension of the problem's aggregate sparsity pattern E, and E' has maximal cliques $\beta_1, \ldots, \beta_l$. We construct a clique tree T such that every topological ordering of T satisfies the RIP (3.10). The subsets of β_k, $k = 1, \ldots l$ are defined as

$$\alpha_k = \beta_k \cap \beta_{\mathrm{par}(k)}, \qquad \eta_k = \beta_k \setminus \beta_{\mathrm{par}(k)}$$

and $\mathrm{par}(k)$ and $\mathrm{ch}(k)$ are the parent and children of k in T.

The reformulated primal problem is

$$\begin{array}{ll} \underset{\tilde{X}_k}{\text{min.}} & \sum_{k=1}^{l} \mathbf{tr}(\tilde{C}_k \tilde{X}_k) \\ \text{s.t.} & \sum_{k=1}^{l} \mathbf{tr}(\tilde{A}_{ik} \tilde{X}_k) = b_i, \quad i = 1, \ldots, m \\ & P_{\alpha_j}(P_{\beta_k}^T \tilde{X}_k P_{\beta_k} - P_{\beta_j}^T \tilde{X}_j P_{\beta_j}) P_{\alpha_j}^T = 0, \quad k = 1, \ldots, l, \quad j \in \mathrm{ch}(k) \\ & \tilde{X}_k \succeq 0, \quad k = 1, \ldots, l \end{array} \tag{5.27}$$

with variables $\tilde{X}_k \in \mathbb{S}^{|\beta_k|}$, $k = 1, \ldots, l$. The second set of equality constraints in (5.27) are the consistency constraints that ensure that the entries of $\tilde{X}_k$ agree when they refer to the same entry of X. The coefficient matrices $\tilde{C}_k$ and $\tilde{A}_{ik}$ are chosen so that

$$C = \sum_{k=1}^{l} P_{\beta_k}^T \tilde{C}_k P_{\beta_k}, \qquad A_i = \sum_{k=1}^{l} P_{\beta_k}^T \tilde{A}_{ik} P_{\beta_k} \tag{5.28}$$

which forces $\mathbf{tr}(CW) = \sum_{k=1}^{l} \mathbf{tr}(\tilde{C}_k W_{\beta_k \beta_k})$ and $\mathbf{tr}(A_i W) = \sum_{k=1}^{l} \mathbf{tr}(\tilde{A}_{ik} W_{\beta_k \beta_k})$ for all $W \in \mathbb{S}_{E'}^p$. One choice, which is analogous to the technique suggested in section 5.2.1 is

$$\tilde{C}_k = P_{\beta_k} \left(C - P_{\alpha_k} C P_{\alpha_k}^T \right) P_{\beta_k}^T, \qquad \tilde{A}_{ik} = P_{\beta_k} \left(A_i - P_{\alpha_k} A_i P_{\alpha_k}^T \right) P_{\beta_k}^T. \tag{5.29}$$

Using this choice, the definitions (5.28) simplify as

$$\tilde{C}_k = \begin{bmatrix} C_{\eta_k, \eta_k} & C_{\eta_k, \alpha_k} \\ C_{\alpha_k, \eta_k} & 0 \end{bmatrix}, \qquad \tilde{A}_{ik} = \begin{bmatrix} (A_i)_{\eta_k, \eta_k} & (A_i)_{\eta_k, \alpha_k} \\ (A_i)_{\alpha_k, \eta_k} & 0 \end{bmatrix}.$$

The converted dual problem is

$$\begin{array}{ll} \underset{y,U_k,\tilde{S}_k}{\text{max.}} & b^T y \\ \text{s.t.} & \sum_{i=1}^m y_i \tilde{A}_{ik} + P_{\beta_k}(P_{\alpha_k}^T U_k P_{\alpha_k} - \sum_{j \in \text{ch}(k)} P_{\alpha_j}^T U_j P_{\alpha_j}) P_{\beta_k}^T + \tilde{S}_k = \tilde{C}_k, \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad k = 1, \ldots, l \\ & \tilde{S}_k \succeq 0, \quad k = 1, \ldots, l, \end{array} \tag{5.30}$$

with variables y, $\tilde{S}_k \in \mathbb{S}^{|\beta_k|}$, and $U_k \in \mathbb{S}^{|\alpha_k|}$, $k = 1, \ldots, l$. The reformulations (5.27) and (5.30) also follow from the clique-tree conversion methods proposed in [KKM11, FKM00]. Note that the matrices U_k in the dual problem play an identical role as the update matrices in a multifrontal supernodal Cholesky factorization [ADV13].

Consistency constraints The variables $\tilde{X}_k$ in (5.27) are interpreted as copies of the dense submatrices $X_{\beta_k\beta_k}$. The second set of equality constraints in (5.27) are the consistency constraints that ensure that the entries of $\tilde{X}_k$ agree when they refer to the same entry of X. If we partition $\tilde{X}_k$ and $X_{\beta_k\beta_k}$ conformally as

$$\tilde{X}_k = \begin{bmatrix} \tilde{X}_{k,11} & \tilde{X}_{k,21}^T \\ \tilde{X}_{k,21} & \tilde{X}_{k,22} \end{bmatrix}, \qquad X_{\beta_k\beta_k} = \begin{bmatrix} X_{\eta_k,\eta_k} & X_{\eta_k,\alpha_k} \\ X_{\alpha_k,\eta_k} & X_{\alpha_k\alpha_k} \end{bmatrix}$$

then the consistency equations reduce to

$$\tilde{X}_{j,22} - P_{\alpha_j}(P_{\beta_k}^T \tilde{X}_k P_{\beta_k}) P_{\alpha_j}^T = 0, \quad k = 1, \ldots, l, \quad j \in \text{ch}(k).$$

5.4.1 Proximal operator

In the clique tree conversion methods of [NFF03, KKM11] the converted SDP (5.27) is solved by an interior-point method. A limitation to this approach is the large number of equality constraints added in the primal problem or, equivalently,

the large dimension of the auxiliary variables U_k in the dual problem. In section 5.3 we proposed an operator-splitting method to address this problem. The key step in each iteration of the splitting method is the evaluation of a proximal operator, by solving the quadratic conic optimization problem (QP)

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^{l} \mathbf{tr}(\tilde{C}_k \tilde{X}_k) + 1/(2t) \sum_{k=1}^{l} \|\tilde{X}_k - Z_k\|_F^2 \\ \text{subject to} & \sum_{k=1}^{l} \mathbf{tr}(\tilde{A}_{ik} \tilde{X}_k) = b_i, \quad i = 1, \ldots, m \\ & \tilde{X}_k \succeq 0, \quad k = 1, \ldots, l. \end{array} \tag{5.31}$$

Solving this problem by a general-purpose solver can be quite expensive and most solvers require a reformulation to remove the quadratic term in the objective by adding second-order cone constraints. However the problem can be solved efficiently via a customized interior-point solver, as we now describe. A similar technique was used for handling variable bounds in SDPs in [NWV08, TTT07].

The Newton equation or KKT system that must be solved in each iteration of an interior-point method for the conic QP (5.31) has the form

$$(1/t)\Delta\tilde{X}_k + W_k \Delta\tilde{X}_k W_k + \sum_{i=1}^{m} \Delta y_i \tilde{A}_{ik} = R_k, \quad k = 1, \ldots, l \tag{5.32}$$

$$\sum_{k=1}^{l} \mathbf{tr}(\tilde{A}_{ik}\, \Delta X_k) = r_i, \quad i = 1, \ldots, m, \tag{5.33}$$

with variables $\Delta\tilde{X}_k$, Δy, where W_k is a positive definite scaling matrix. The first term $(1/t)\Delta\tilde{X}_k$ results from the quadratic term in the objective. Without this term it is straightforward to eliminate the variable $\Delta\tilde{X}_k$ from first equation, to obtain an equation in the variable Δy. To achieve the same goal at a similar cost with a customized solver we first compute eigenvalue decompositions $W_k = Q_k \mathbf{diag}(\lambda_k) Q_k^T$ of the l scaling matrices, and define l matrices $S_k \in \mathbb{S}^{|\beta_k|}$

with entries

$$(S_k)_{ij} = \frac{t}{1 + t\lambda_{ki}\lambda_{kj}}, \quad i, j = 1, \ldots, |\beta_k|.$$

We can now use the first equation in (5.32) to express $\Delta\tilde{X}_k$ in terms of Δy:

$$\Delta\tilde{X}_k = Q_k \left(S_k \circ (\widehat{R}_k - \sum_{i=1}^{m} \Delta y_i \widehat{A}_{ik})\right) Q_k^T$$

with $\widehat{R}_k = Q_k^T R_k Q_k$, $\widehat{A}_{ik} = Q_k^T \tilde{A}_{ik} Q_k$, and where $\circ$ denotes the Hadamard (component-wise) product. Substituting the expression for $\Delta\tilde{X}_k$ in the second equation of (5.33) gives an equation $H\Delta y = g$ with

$$H_{ij} = \sum_{k=1}^{l} \mathbf{tr}(\widehat{A}_{ik}(S_k \circ \widehat{A}_{jk})), \qquad g_i = r_i - \sum_{k=1}^{l} \mathbf{tr}(\widehat{A}_{ik}(S_k \circ R_k)), \qquad i, j = 1, \ldots, m. \tag{5.34}$$

The cost of this solution method for the KKT system (5.32)–(5.33) is comparable to the cost of solving the KKT systems in an interior-point method applied to the conic optimization problem (5.31) without the quadratic term. The proximal operator can therefore be evaluated at roughly the same cost as the cost of solving the converted SDP (5.27) with the consistency constraints removed.

To illustrate the value of this technique, we compare in figure 5.3 the time needed to solve the semidefinite QP (5.31) using several methods: the general-purpose conic solver SDPT3 for MATLAB, called directly or via CVX (version 2.0 beta) [GB12, GB08], and an implementation of the algorithm described above in CVXOPT [ADV10a, ADL12]. The problems are dense and randomly generated with $l = 1$, $m = p = |\beta_1|$, and $t = 1$. The figure shows CPU time versus the order p of the matrix variable, computed on an Intel Xeon CPU E3-1225 processor with 4 cores, 3.10 GHz clock speed, and 32 GB RAM. For the fast proximal operator implementation, CPU time is measured using the Python module psutils (available at `code.google.com/p/psutil`). For CVX and SDPT3, we used the CPU times reported by the solver.

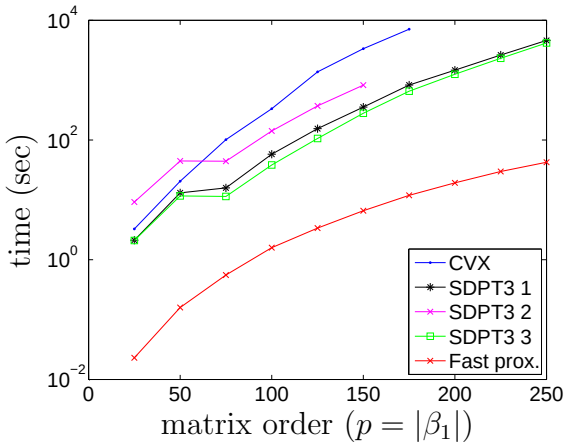

Figure 5.3: *Custom proximal operator.* Runtime required for a single proximal operator evaluation (5.31) on a dense subproblem with one clique ($l = 1$) of size $p = |\beta_1|$ and $m = p$ constraints (averaged over 10 trials). The CPU time of the general-purpose solver SDPT3, called directly or via CVX, is compared against the CPU time of a customized fast proximal operator.

The fast algorithm uses the cone QP solver in CVXOPT with the default termination criteria. In the CVX code the function `sum_square()` was used to represent the quadratic term in the objective. The three SDPT3 curves correspond to different ways of converting the problem to a conic LP using the equivalence

$$u^T u \leq v \qquad \Longleftrightarrow \qquad \left\| \begin{bmatrix} 2u \\ v-1 \end{bmatrix} \right\|_2 \leq v+1.$$

In the first formulation ('SDPT3 1') we replace the squared norm in the objective with a variable w and add the constraint $\|\tilde{X} - Z\|_F^2 \leq w$ via a single second-order cone constraint of order $p(p+1)/2+1$. In the second formulation ('SDPT3 2'), we replace the quadratic term with a sum of $p(p+1)/2$ auxiliary variables w_{ij}, for $1 \leq i \leq j \leq p$, and add $p(p+1)/2$ second-order cone constraints of dimension two to express $(\tilde{X}_{ij} - Z_{ij})^2 \leq w_{ij}$. In the third formulation ('SDPT3 3') we replace the quadratic term with the sum of p variables w_i, subject to the constraint that w_i is an upper bound on the Euclidean norm of the lower-triangular part of the ith

column of $\tilde{X} - Z$. As can be seen, the choice between the conic LP reformulations has an effect on the efficiency of a general-purpose solver. However the experiment also confirms that the fast proximal operator evaluation is orders of magnitude faster than general-purpose solvers applied to equivalent conic LPs.

Block diagonal correlative sparsity The efficiency of the decomposition method depends crucially on the cost of the proximal operator evaluations, which is determined by the sparsity pattern of the Schur complement matrix H (5.34), *i.e.*, the *correlative sparsity* pattern of the reformulated problems [KKK08]. Note that in general the scaled matrices $\widehat{A}_{ik}$ used to assemble H will be either completely dense (if $\tilde{A}_{ik} \neq 0$) or zero (if $\tilde{A}_{ik} = 0$). Therefore $H_{ij} = 0$ if for each k at least one of the coefficient matrices $\tilde{A}_{ik}$ and $\tilde{A}_{jk}$ is zero. This rule characterizes the correlative sparsity pattern.

As pointed out in section 5.3.3, the sparsity of the Schur complement matrix can be enhanced by exploiting the flexibility in the choice of parameters of the reformulated problem (the matrices $\tilde{A}_{ik}$). The definition (5.29) is one possible choice, but any set of matrices that satisfy (5.30) can be used instead. While the optimal choice is not clear in general, it is straightforward in the important special case when the index set $\{1, \ldots, m\}$ can be partitioned in l sets $\upsilon_1, \ldots, \upsilon_l$, with the property that if $i \in \upsilon_j$, then all the nonzero entries of A_i belong to the principal submatrix $(A_i)_{\beta_j \beta_j}$. In this case,

$$A_i = P_{\beta_j}^T P_{\beta_j} A_i P_{\beta_j}^T P_{\beta_j} \text{ for all } i \in \upsilon_j,$$

and a valid choice for the coefficient matrices $\tilde{A}_{ik}$ is to take

$$\tilde{A}_{ij} = P_{\beta_j} A_i P_{\beta_j}^T, \qquad \tilde{A}_{ik} = 0, \quad k \neq j,$$

when $i \in \upsilon_j$. With this choice, the matrix H can be re-ordered to be block-diagonal

with dense blocks $H_{v_i v_i}$. Moreover the QP (5.31) is separable and equivalent to l independent subproblems

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(C_k\tilde{X}_k) + 1/(2t)\|\tilde{X}_k - Z_k\|_F^2 \\ \text{subject to} & \mathbf{tr}(\tilde{A}_{ik}\tilde{X}_k) = b_i, \quad i \in v_k \\ & \tilde{X}_k \succeq 0. \end{array}$$

5.5 Numerical results

In this section we present the results of numerical experiments with the decomposition method applied to SDPs. First, we describe how steplength selection can significantly affect (and impair) convergence speed and show how a simple adaptive steplength scheme can make the method more robust. Then, we apply the decomposition method to an approximate Euclidean distance matrix completion problem, motivated by an application in sensor network node localization, and illustrate the convergence behavior of the method in practice. The problem involves a sparse matrix variable whose sparsity pattern is characterized by the sensor network topology, and is interesting because in the converted form the system has block diagonal correlative sparsity regardless of the network topology. Finally, we present extensive runtime results for a family of problems with block-arrow aggregate sparsity and block-diagonal correlative sparsity. By comparing the CPU times required by general-purpose interior-point methods and the decomposition method, we are able to characterize the regime in which each method is more efficient.

Software specifications The decomposition method is implemented in Python (version 2.6.5), using the conic quadratic optimization solver of CVXOPT (version 1.1.5) [ADL12] for solving the conic QPs (5.31) in the evaluation of the proximal operators. SEDUMI (version 1.1) [Stu99] and SDPT3 (version 4.0) in MATLAB

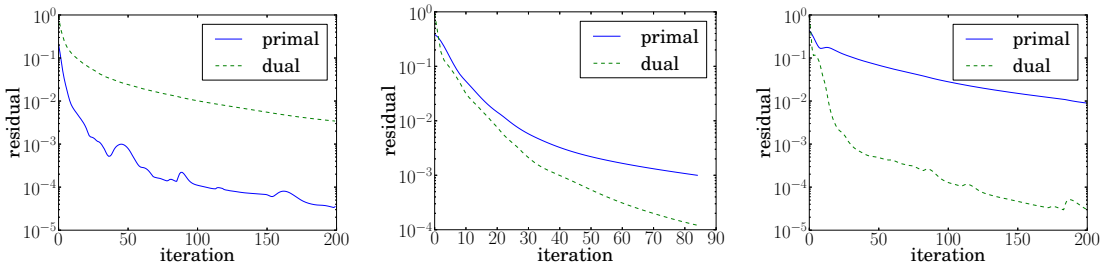

Figure 5.4: *Choice of constant step size.* Primal residual $\|r_{\mathrm{p}}^i\|_2/\|\tilde{x}^i\|_2$ and dual residual $\|r_{\mathrm{d}}^i\|_2/\|v^i\|_2$ versus iteration number i for three constant values of t^i: $t^i = 10$ (left), $t^i = 100$ (center), and $t^i = 1000$ (right).

(version 2011 b) are used as the general-purpose solver.

5.5.1 Adaptive step length selection

The first experiment illustrates the effect of the choice of the steplength parameter t_k and explains the motivation behind the adaptive strategy (2.29). We pick a randomly generated SDP with a block-banded sparsity pattern E of order $p = 402$ with $l = 50$ cliques of size $|\beta_k| = 10$. The cliques correspond to overlapping diagonal blocks of order 10, with an overlap of size 2. The correlative sparsity pattern in the converted SDP has a block-arrow structure with 50 diagonal blocks of size 10×10 and 10 dense rows and columns at the end. The number of primal constraints and dual variables is $m = 510$.

Figure 5.4 shows the primal and dual residuals $\|r_{\mathrm{p}}^i\|_2/\|x^i\|_2$ and $\|r_{\mathrm{d}}^i\|_2/\|v^i\|_2$ for three constant values of the steplength parameter: $t^i = 10$, 100, and 1000. As can be seen, the choice of t^i has a strong effect on the speed of convergence. The figures suggest that when t^i is too small the dual residual decreases more slowly than the primal residual, and when t^i is too large, the primal residual decreases more slowly. For a good value of t^i in between, the two residuals decrease at about the same rate. This observation motivates the adaptive strategy (2.29). Figure 5.5 shows the residuals if the adaptive strategy is used, with $\mu = 2$, $\tau^i = 1 + 0.9^i$,

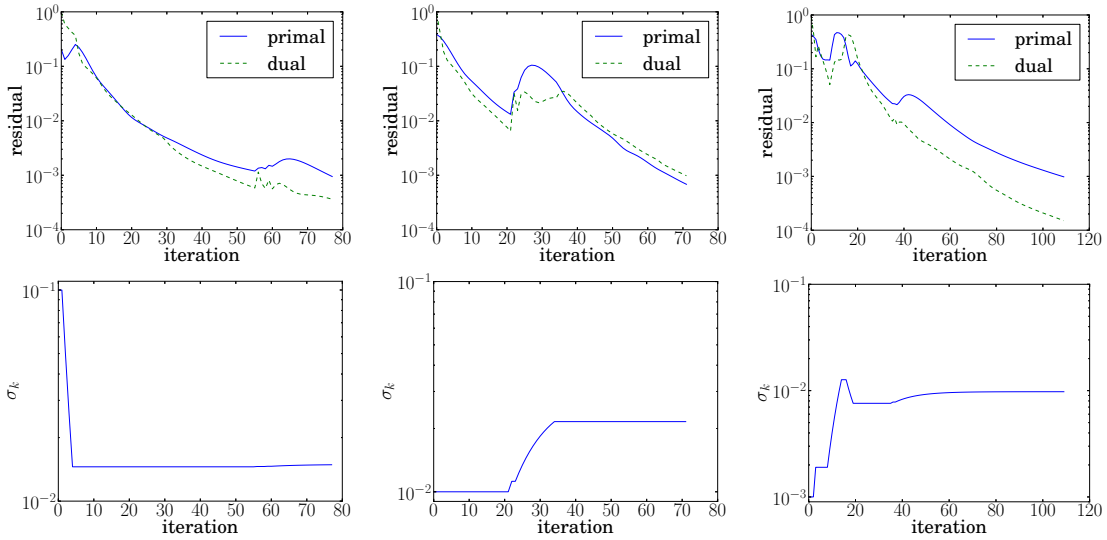

Figure 5.5: *Adaptive step size.* Primal and dual residuals versus iteration number with adaptive selection of t^i, starting with a value 10 (top left), 100 (center), 1000 (right). The graphs on the bottom row show the values of t^i during the three runs of the algorithm.

and starting at three different values of t^i (10, 100, 1000). Figure 5.5 shows the resulting values of t_k versus the iteration number i.

The convergence graphs indicate that a simple heuristic for adapting the steplength can improve the speed of convergence and make it less dependent on the initial steplength. The specific convergence behavior depends on the parameter μ and the decay rule for τ^i, but is much less sensitive to the choice of t^0. While in general the convergence with adaptive steplength is not faster than with a carefully tuned constant steplength, the adaptive strategy is more robust than picking an arbitrary constant steplength.

5.5.2 Approximate Euclidean distance matrix completion

In this section, we consider the problem of fitting a EDM to measurements $\widehat{D}_{ij}$ of a subset of its entries. This and related problems arise in many applications, including, for example, the sensor network node localization problem [BY04b,

CY07, KKW09, KW12].

From chapter 3 it is shown that D is a $p \times p$ EDM if and only if there exists a PSD matrix $X \in \mathbb{S}^p$ such that

$$D_{ij} = \mathbf{tr}(A_{ij}X), \quad A_{ij} = (e_i - e_j)(e_i - e_j)^T, \quad i, j = 1, \ldots, p$$

where $X \succeq 0$ is the Gram matrix ($X_{ij} = x_i^T x_j$) and e_i denotes the ith unit vector in $\mathbb{R}^p$.

In the EDM approximation problem we are given a set of measurements $\widehat{D}_{ij}$ for entries in a sparsity pattern $\{i, j\} \in E$. The problem of fitting a EDM to the measurements can be posed as

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{\{i,j\}\in E} |\mathbf{tr}(A_{ij}X) - \widehat{D}_{ij}| \\ \text{subject to} & X \succeq 0, \end{array} \tag{5.35}$$

with variable $X \in \mathbb{S}^p$. Now let $E' \supseteq E$ be the chordal embedding of the aggregate sparsity pattern E of the matrices A_{ij}. Then, without loss of generality, we can restrict the variable X in (5.35) to be a sparse matrix in $\mathbb{S}^p_{E'}$ and we obtain the equivalent problem

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{\{i,j\}\in E} |\mathbf{tr}(A_{ij}X) - \widehat{D}_{ij}| \\ \text{subject to} & X \in \Pi_{E'}(\mathbb{S}^p_+). \end{array} \tag{5.36}$$

This problem is readily converted into a standard conic LP of the form (5.1), which can then be solved using the proposed decomposition method. An interesting feature of this application is that the correlative sparsity associated with the converted problem is block-diagonal.

The conversion method and the block-diagonal correlative sparsity can also be

explained directly in terms of the problem (5.36). Suppose E has l cliques β_k, $k = 1, \ldots, l$. Suppose we partition the set E in l sets E_k with the property that if $\{i, j\} \in E_k$ then $i, j \in \beta_k$. Then (5.36) is equivalent to

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^{l} \sum_{\{i,j\} \in E_k} |\mathbf{tr}(A_{ij} P_{\beta_k}^T \tilde{X}_k P_{\beta_k}) - \widehat{D}_{ij}| \\ \text{subject to} & P_{\alpha_j}(P_{\beta_k}^T \tilde{X}_k P_{\beta_k} - P_{\beta_j}^T \tilde{X}_j P_{\beta_j}) P_{\alpha_j}^T = 0, \quad k = 1, \ldots, l, \quad \beta_j \in \mathrm{ch}(\beta_k) \\ & \tilde{X}_k \succeq 0, \quad k = 1, \ldots, l, \end{array} \tag{5.37}$$

with variables $\tilde{X}_k \in \mathbb{S}^{|\beta_k|}$, $k = 1, \ldots, l$. This problem can be solved using Spingarn's method. At each iteration we alternate between projection on the subspace defined by the consistency equations in (5.37) and evaluation of a prox-operator, via the solution of

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^{l} \sum_{\{i,j\} \in E_k} |\mathbf{tr}(A_{ij} P_{\beta_k}^T \tilde{X}_k P_{\beta_k}) - \widehat{D}_{ij}| + 1/(2t) \sum_{k=1}^{l} \|\tilde{X}_k - Z_k\|_F^2 \\ \text{subject to} & \tilde{X}_k \succeq 0, \quad k = 1, \ldots, l. \end{array} \tag{5.38}$$

Note that this problem is separable because if $\{i, j\} \in E_k$, A_{ij} is nonzero only in positions that are included in $\beta_k \times \beta_k$. The problems (5.38) can be solved efficiently via a straightforward modification of the interior-point method described in section 5.4.

We now illustrate the convergence of the decomposition method on two randomly generated networks. An example of a network topology is shown in figure 5.6 (top) for a problem with 500 nodes. The network edges are assigned using the following rule: a pair $\{i, j\}$ is in the sparsity pattern E if one of the nodes is among the five nearest neighbors of the other node.

To compute a chordal embedding E', we use an approximate minimum degree (AMD) reordering, which gives a permutation of the sparsity pattern that reduces

fill-in (figure 5.6, bottom left). Often, the resulting embedding contains many small cliques and for our purposes it is more efficient to merge some neighboring cliques, using algorithms similar to those in [RS09, HS10]. Specifically, traversing the tree in a topological order, we greedily merge clique k with its parent if

$$(|\beta_{\mathrm{par}(k)}| - |\alpha_k|)(|\eta_k|) \leq t_{\mathrm{fill}} \quad \text{or} \quad \max(|\eta_k|, |\eta_{\mathrm{par}(k)}|) \leq t_{\mathrm{size}}$$

where t_{fill} is a threshold based on the amount of fill that results from merging clique k with its parent, and t_{size} is a threshold based on the cardinality of the sets $\eta_{\mathrm{par}(k)}$ and η_k. In figure 5.6 (bottom right) we show the result of this clique-merging technique using the values $t_{\mathrm{fill}} = t_{\mathrm{size}} = 5$. This reduced the 359 original cliques with an average of 5 nodes each to 79 cliques with an average of 10 nodes.

A typical convergence plot of the resulting problem is given in figure 5.7 for a network with 500 nodes (left) and 2000 nodes (right). A constant value $t_k = 0.2$ is used for the steplength parameter. The greedy clique merging strategy described above was used, with the same threshold values.

5.5.3 Block-arrow sparsity

In this experiment we compare the efficiency of the splitting method with general-purpose SDP solvers. We consider a family of randomly generated SDPs with a block-arrow aggregate sparsity pattern E and a block-diagonal correlative sparsity pattern. The sparsity pattern E and corresponding clique tree are defined in figure 5.8. It consists of l diagonal blocks of size $d \times d$, plus w dense final rows and columns. We take the clique

$$\beta_l = \{(l-1)d + 1, \ldots, ld, ld + 1, \ldots, ld + w\}$$

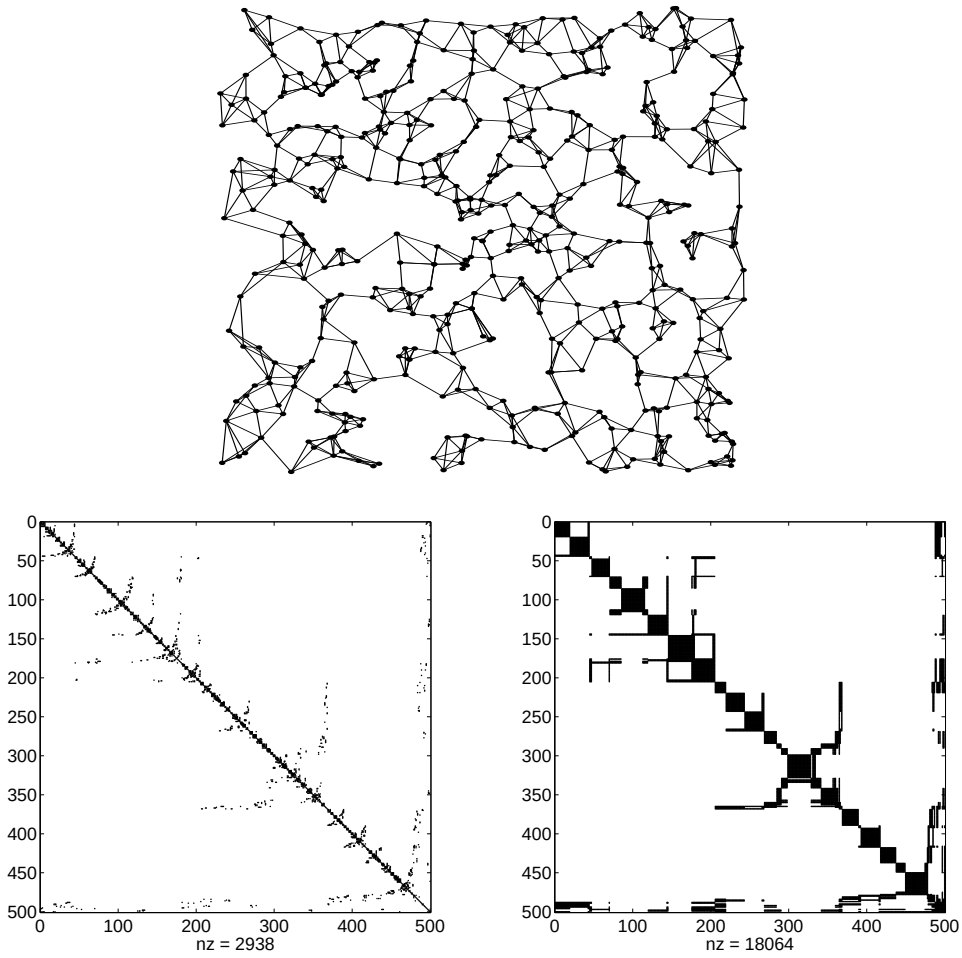

Figure 5.6: *EDM example.* Top: nearest-neighbor network for a problem with 500 nodes in two dimensions. Two nodes are connected if one of the two is among the 5 nearest neighbors of the other node. Bottom left: corresponding sparsity pattern after AMD permutation and chordal extension. Bottom right: corresponding sparsity pattern after clique merging. Before clique merging, there are 359 cliques with an average of 5 elements. After clique merging, there are 79 cliques with an average of 5 elements.

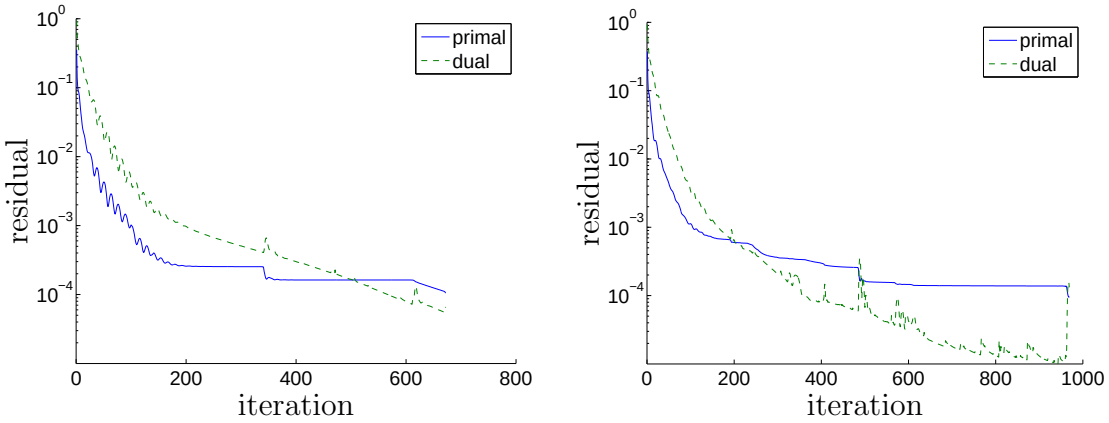

Figure 5.7: *Convergence.* Relative primal and dual residuals versus iteration number for networks with 500 (left) and 2000 (right) nodes. For $n = 500$, there are 82 cliques, and for $n = 2000$, there are 310 cliques. A constant steplength parameter $t_k = 0.2$ is used.

(with $\alpha_l = \emptyset$) as root of the clique tree. The other $l - 1$ cliques β_k and the intersections $\alpha_k = \beta_k \cap \mathrm{par}(\beta_k)$ with their parent cliques are

$$\beta_k = \{(k-1)d+1, \ldots, kd\} \cup \alpha_k, \quad \alpha_k = \{ld+1, ld+2, \ldots, ld+w\}, \quad k = 1, \ldots, l-1.$$

We generate matrix cone programs with $m = ls$ primal equality constraints, partitioned in l sets

$$\upsilon_k = \{(k-1)s+1, (k-1)s+2, \ldots, ks\}, \qquad k = 1, \ldots, l,$$

of equal size $|\upsilon_k| = s$. If $i \in \upsilon_k$, then the coefficient matrix A_i contains a dense $\beta_k \times \beta_k$ block, and is otherwise zero. We will use the notation

$$(A_i)_{\beta_k \beta_k} = \begin{bmatrix} U_i & V_i \\ V_i^T & W_i \end{bmatrix},$$

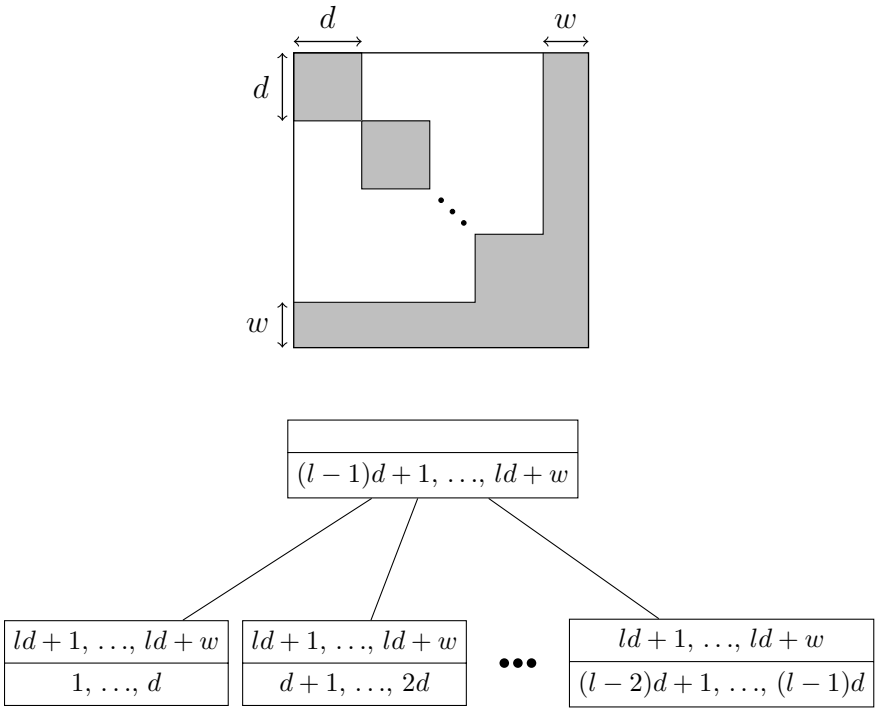

Figure 5.8: *Block-arrow example.* Top: illustration of block-arrow sparsity pattern for l cliques. The order of the matrix is $ld + w$. The first l diagonal blocks in the matrix have size d, the last block column and block row have width w. The cliques therefore have size $d + w$. Bottom: corresponding clique tree. Each clique in the clique tree is partitioned in two sets: the top row shows $\alpha_k = \beta_k \cap \mathrm{par}(\beta_k)$; the bottom row shows $\eta_k = \beta_k \setminus \alpha_k$.

for the nonzero block of A_i if $i \in \upsilon_k$. The primal and dual SDPs can therefore be expressed as

$$\begin{array}{lll} \text{minimize} & \mathbf{tr}(CX) & \text{maximize} \quad b^T y \\ \text{subject to} & \mathcal{A}(X) = b & \text{subject to} \quad \mathcal{A}^*(y) + S = C \\ & X \succeq 0 & \qquad\qquad S \succeq 0 \end{array} \tag{5.39}$$

with a linear mapping $\mathcal{A} : \mathbb{S}^{(ld+w)\times(ld+w)} \to \mathbb{R}^{ls}$ defined as

$$\mathcal{A}(X)_i = \mathbf{tr}\left(\begin{bmatrix} U_i & V_i \\ V_i^T & W_i \end{bmatrix} \begin{bmatrix} X_{kk} & X_{k,l+1} \\ X_{l+1,k} & X_{l+1,l+1} \end{bmatrix} \right), \quad i \in \upsilon_k, \quad k = 1, \ldots, l,$$

where X_{ij} denotes the i, j block of X. (These blocks have dimensions $X_{ii} \in \mathbb{S}^d$ for $i = 1, \ldots, l$, $X_{l+1,l+1} \in \mathbb{S}^w$, $X_{l+1,i} \in \mathbb{R}^{w\times d}$ for $i = 1, \ldots, l$.) The adjoint $\mathcal{A}^* : \mathbb{R}^{ls} \to \mathbb{S}^{(ld+w)\times(ld+w)}$ is

$$\mathcal{A}^*(y) = \begin{bmatrix} \sum_{i\in\upsilon_1} y_i U_i & 0 & \cdots & 0 & \sum_{i\in\upsilon_1} y_i V_i \\ 0 & \sum_{i\in\upsilon_2} y_i U_i & \cdots & 0 & \sum_{i\in\upsilon_2} y_i V_i \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \sum_{i\in\upsilon_l} y_i U_i & \sum_{i\in\upsilon_l} y_i V_i \\ \sum_{i\in\upsilon_1} y_i V_i^T & \sum_{i\in\upsilon_2} y_i V_i^T & \cdots & \sum_{i\in\upsilon_l} y_i V_i^T & \sum_{i=1}^{m} y_i W_i \end{bmatrix}.$$

In the reformulated problem, the variable X is replaced with l matrices $\tilde{X}_k = X_{\beta_k\beta_k}$, *i.e.*, defined as

$$\tilde{X}_k = \begin{bmatrix} (\tilde{X}_k)_{11} & (\tilde{X}_k)_{12} \\ (\tilde{X}_k)_{21} & (\tilde{X}_k)_{22} \end{bmatrix} = \begin{bmatrix} X_{kk} & X_{k,l+1} \\ X_{k,l+1}^T & X_{l+1,l+1} \end{bmatrix}, \quad k = 1, \ldots, l,$$

and the primal SDP is converted to

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^{l} \mathbf{tr}(\tilde{C}_k \tilde{X}_k) \\ \text{subject to} & \mathbf{tr}\left(\begin{bmatrix} U_i & V_i \\ V_i^T & W_i \end{bmatrix} \begin{bmatrix} (\tilde{X}_k)_{11} & (\tilde{X}_k)_{12} \\ (\tilde{X}_k)_{21} & (\tilde{X}_k)_{22} \end{bmatrix} \right) = b_i, \quad i \in v_k, \quad k = 1, \ldots, l \\ & (\tilde{X}_k)_{22} = (\tilde{X}_l)_{22}, \quad k = 1, \ldots, l-1 \\ & \tilde{X}_k \succeq 0, \quad k = 1, \ldots, l \end{array} \tag{5.40}$$

where

$$\tilde{C}_k = \begin{bmatrix} C_{kk} & C_{k,l+1} \\ C_{k,l+1}^T & 0 \end{bmatrix}, \quad k = 1, \ldots, l-1, \qquad \tilde{C}_l = \begin{bmatrix} C_{ll} & C_{l,l+1} \\ C_{l,l+1}^T & C_{l+1,l+1} \end{bmatrix}.$$

With this choice of parameters, the correlative sparsity pattern of the converted SDP (5.40) is block-diagonal, *i.e.*, except for the consistency constraints $(\tilde{X}_k)_{22} = (\tilde{X}_l)_{22}$ the problem is separable with independent variables $\tilde{X}_k \in \mathbb{S}^{p+w}$. This allows us to compute the proximal operator by solving l independent conic QPs.

Problem generation The problem data are randomly generated as follows. First, the entries of U_k, V_k, W_k are drawn independently from a unit-variance normal distribution. A strictly primal feasible X is constructed as $X = Z + \alpha I$ where $Z \in \mathbb{S}_E^p$ is randomly generated with unit-variance normal distribution i.i.d. entries and α is chosen so that $X_{\beta_k \beta_k} = Z_{\beta_k \beta_k} + \alpha I \succ 0$ for $k = 1, \ldots, l$. The right-hand side b in the primal constraint is computed as $b_i = \mathbf{tr}(A_i X)$, $i = 1, \ldots, m$.

Next, strictly dual feasible $y \in \mathbb{R}^m$, $S \in \mathbb{S}_E^p$ are constructed. The vector y has Gaussian i.i.d. entries and S is constructed as $S = \sum_{k=1}^{l} P_{\beta_k}^T \tilde{S}_k P_{\beta_k}$, with $\tilde{S}_k = Z_k + \alpha I$, where $Z_k \in \mathbb{S}^{|\beta_k|}$ has Gaussian distribution generated entries, and α chosen so that $\tilde{S}_k \succ 0$. Finally, the matrix C is constructed as $C = S + \sum_i y_i A_i$.

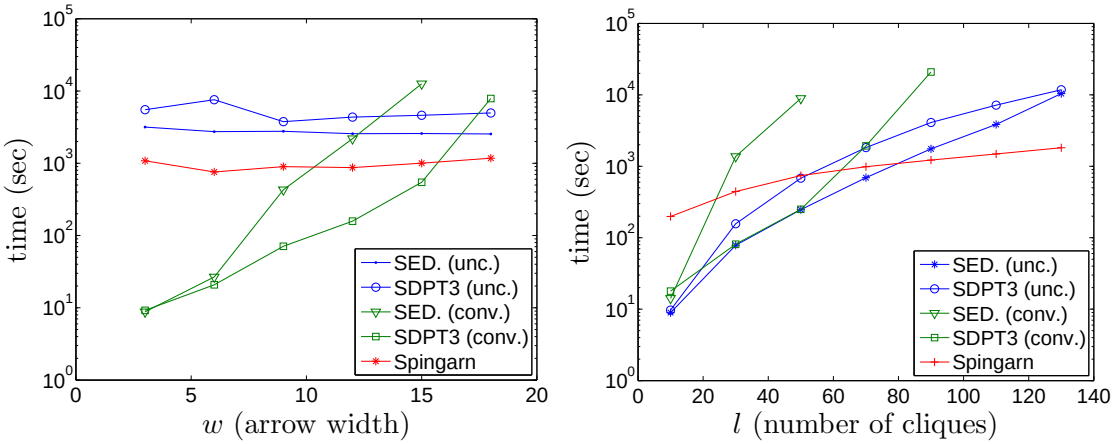

Figure 5.9: *Performance.* Solution time for randomly generated SDPs with block--arrow sparsity patterns. Times are reported for SEDUMI (SED.) and SDPT3 applied to the original ('unc.') and converted ('conv') SDPs, and the Spingarn method applied to the converted SDP. The figure on the left shows the times as function of arrow width w, for fixed dimensions $l = 100$, $d = 20$, $s = 10$. The figure on the right shows the times versus number of cliques l, for fixed dimensions $w = 20$, $d = 20$, $s = 10$.

Comparison with general-purpose SDP solvers In figure 5.9 we compare the solution time of Spingarn's algorithm with the general-purpose interior-point solvers SEDUMI and SDPT3, applied to the unconverted and converted SDPs (5.1) and (5.27). In the decomposition method we use a constant steplength parameter $t_k = 0.2$ and relaxation parameter $\rho_k = 1.75$. The stopping criterion is (2.28) with $\epsilon_{\mathrm{p}} = \epsilon_{\mathrm{d}} = 10^{-4}$. For each data point we report the average CPU time over 5 instances.

To interpret the results, it is useful to consider the linear algebra complexity per iteration of each method. The unconverted SDP (5.39) has a single matrix variable X of order $p = ld + w$. The cost per iteration of an interior-point method is dominated by the cost of forming and solving the Schur complement equation, which is dense and of size $m = sl$. For the problem sizes used in the figures (w small compared to ld) the cost of solving the Schur complement dominates the overall complexity. This explains the nearly constant solution time in the first figure (fixed l, s, p, varying w) and the increase with l shown in the second figure.

The converted SDP (5.40) has l variables $\tilde{X}_k$ of order $d + w$. The Schur complement equation in an interior-point method has the general structure (5.19) with a leading block-diagonal matrix (l blocks of size $s \times s$) augmented with a dense block row and block column of width proportional to lw^2. For small w, exploiting the block-diagonal structure in the Schur complement equation, allows one to solve the Schur complement equation very quickly and reduces the cost per iteration to a fraction of a cost of solving the unconverted problem, despite the increased size of the problem. However the advantage disappears with increasing w (figure 5.9 left).

The main step in each iteration of the Spingarn method applied to the converted problem is the evaluation of the prox-operators via an interior-point method. The Schur complement equations that arise in this computation are block-diagonal (l blocks of order s) and therefore the cost of solving them is independent of w and linear in l. As an additional advantage, since the correlative sparsity pattern is block-diagonal, the proximal operator can be evaluated by solving l independent conic QPs that can be solved in parallel. This was not implemented in the experiment, but could reduce the solution time by a factor of roughly l.

Accuracy and steplength selection The principal disadvantage of the splitting method, compared with an interior-point method, is the more limited accuracy and the higher sensitivity to the choice of algorithm parameters. Figure 5.10 (left) shows the number of iterations versus l for different values of the tolerance ϵ used in the stopping criterion. The right-hand plot shows the number of iterations versus l for two different constant values of the steplength parameter t_k ($t_k = 1.0$ and $t_k = 0.2$) and for an adaptively adjusted steplength.

So far we have assumed that the error in the proximal operator evaluations is negligible compared with the exit tolerances used in Spingarn's method. In the last experiment we examine the effect of inexactness of the proximal operator

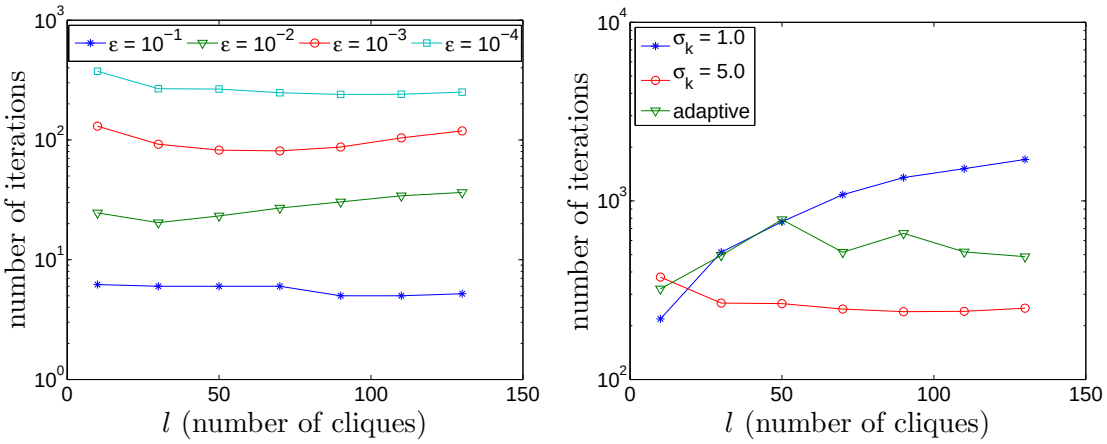

Figure 5.10: *Accuracy and steplength.* Left: number of iterations for of Spingarn's method based on the desired accuracy, for a problem instance with $d = 20$, $w = 20$, $s = 10$, and using a fixed steplength parameter $t_k = 0.2$. Right: number of iterations for the same problem with $\epsilon = 10^{-4}$ and different choices of steplength.

evaluations. As test problem we use an instance of the family of block-arrow problems, with $d = 10$, $l = 25$, $w = 10$, $s = 10$. The accuracy of the conic QP solver used to evaluate the prox-operators is controlled by three tolerances that bound the error in the optimality conditions (5.25). The tolerance ϵ_{feas} is an upper bound on the relative error in the primal and dual equality constraints, ϵ_{abs} is an upper bound on the duality gap, and ϵ_{rel} is an upper bound on the relative duality gap. In the experiment we set these tolerances to $\epsilon_{\text{feas}} = \epsilon_{\text{abs}} = \epsilon_{\text{rel}}/10 = \epsilon_{\text{prox}}/10$, for three values of ϵ_{prox}: the CVXOPT default value $\epsilon_{\text{prox}} = 10^{-6}$, and two smaller values, 10^{-8} and 10^{-10}. For each value, we run Spingarn's method with different tolerances ϵ_{p} and ϵ_{d} in the stopping condition (2.28). The results are shown in Table 5.1. In columns 3–5 we compare the solution from Spingarn's method with the answer returned by an interior-point solver (SDPT3). The entries in these columns are defined as

$$r_X = \frac{\|\tilde{X} - \tilde{X}_{\text{ipm}}\|_F}{\|\tilde{X}_{\text{ipm}}\|_F}, \qquad r_y = \frac{\|y - y_{\text{ipm}}\|_2}{\|y_{\text{ipm}}\|_2}, \qquad r_{\text{obj}} = \frac{|f(\tilde{X}) - f(\tilde{X}_{\text{ipm}})|}{|f(\tilde{X}_{\text{ipm}})|}$$

where $\tilde{X}_{\text{ipm}}$ and y_{ipm} are the optimal primal and dual variables of the converted

ϵ_{prox}	$\epsilon_{\text{p}} = \epsilon_{\text{d}}$	r_X	r_y	r_{obj}	#iterations
10^{-6}	10^{-4}	$9.4\ 10^{-3}$	$7.6\ 10^{-4}$	$1.6\ 10^{-5}$	179
10^{-8}	10^{-4}	$9.1\ 10^{-3}$	$7.2\ 10^{-4}$	$1.5\ 10^{-5}$	180
	10^{-6}	$1.2\ 10^{-4}$	$8.6\ 10^{-6}$	$5.7\ 10^{-8}$	443
	10^{-8}	$2.1\ 10^{-5}$	$2.1\ 10^{-6}$	$1.8\ 10^{-8}$	2376
10^{-10}	10^{-4}	$9.1\ 10^{-3}$	$7.2\ 10^{-4}$	$1.5\ 10^{-5}$	180
	10^{-6}	$1.2\ 10^{-4}$	$8.2\ 10^{-6}$	$4.3\ 10^{-8}$	444
	10^{-8}	$1.1\ 10^{-5}$	$2.8\ 10^{-7}$	$1.9\ 10^{-8}$	759

Table 5.1: *Accuracy.* Relative differences between solutions, computed by Spingarn's method and an interior-point method, and the number of iterations in Spingarn's method, for varying exit conditions in Spingarn's method and the proximal operator evaluations. The first column is the tolerance in the proximal operator evaluations. The second column shows the tolerances in Spingarn's method.

problem computed by SDPT3, and the function $f(\tilde{X})$ is the primal objective value. The last column in the table is the number of iterations in Spingarn's method. (For $\epsilon_{\text{prox}} = 10^{-6}$ and $\epsilon_{\text{p}} = \epsilon_{\text{d}} = 10^{-6}$, the method did not converge in 10000 iterations.)

From the table we can make a few observations. First, when $\epsilon_{\text{prox}} \ll \epsilon_{\text{p}} = \epsilon_{\text{d}}$, the relative error in the solution seems to be comparable in magnitude with the primal and dual residuals. Second, when running Spingarn's method with a moderate accuracy (for example, with $\epsilon_{\text{p}} = \epsilon_{\text{d}} = 10^{-4}$), increasing the accuracy of the proximal operator evaluation does not improve the convergence or the accuracy of the result, and the accuracy of an interior-point method with typical default settings is adequate.

5.6 Discussion

We have described a decomposition method that exploits partially separable structure in linear conic optimization problems. The basic idea is straightforward: by replicating some of the variables, we reformulate the problem as an equivalent linear optimization problem with block-separable conic inequalities and an equality constraint that ensures that the replicated variables are consistent. We can

then apply Spingarn's method of partial inverses to this equality-constrained convex problem. Spingarn's method is a generalized alternating projection method for convex optimization over a subspace. It alternates orthogonal projections on the subspace with the evaluation of the proximal operator of the cost function. In the method described in the paper, these prox-operators are evaluated by an interior-point method for conic quadratic optimization.

When applied to sparse SDPs, the reformulation coincides with the clique conversion methods which were introduced in [KKM11, FKM00] with the purpose of exploiting sparsity in interior-point methods for semidefinite programming. By solving the converted problems via a splitting algorithm instead of an interior-point algorithm we extend the applicability of the conversion methods to problems for which the converted problem is too large to handle by interior-point methods. As a second advantage, if the correlative sparsity is block-diagonal, the most expensive step of the decomposition algorithm (evaluating the proximal operator) is separable and can be parallelized. The numerical experiments indicate that the approach is effective when a moderate accuracy (compared with interior-point methods) is acceptable. However the convergence can be quite slow and strongly depends on the choice of steplength.

A critical component in the decomposition algorithm for semidefinite programming is the use of a customized interior-point method for evaluating the proximal operators. This technique allows us to evaluate the proximal operator at roughly the same cost of solving the reformulated SDP without the consistency constraints. As a further improvement one could extend this technique to exploit sparsity in the coefficient matrices of the reformulated problem, using techniques developed for interior-point methods for sparse matrix cones [ADV13].

CHAPTER 6

Conclusion

This thesis presents a set of decomposition methods for large-scale sparse semidefinite and EDM optimization problems. The main contribution is to combine results of sparse matrix theory with first-order methods and interior-point methods in order to exploit sparsity more effectively. A key observation is that the sparse completable PSD and EDM cones can be written in terms of smaller, overlapping matrix inequality constraints. By applying first-order splitting methods, we reduce the problem into several smaller subproblems that often can be solved independently. Specifically, we use dual decomposition methods and Douglas-Rachford splitting methods to solve matrix nearness problems, and a hybrid proximal splitting method and interior-point method to solve large linear SDPs. To demonstrate that our methods successfully exploit sparsity, we solve problems involving sparse matrices as large as $100,000 \times 100,000$. We compare runtime results against first-order and interior-point methods that do not use chordal decomposition, showing a significant reduction in runtime and memory usage.

The goal of the thesis is to offer a small contribution toward the wider set of large-scale optimization problems. The excitement over "Big Data problems" seems to have permeated every industry, from software companies like Facebook [Mar14] and Twitter [Edw14], to the healthcare industry mining patient data [Der15], to optimally running political campaigns [Lam13]. In optimization, Big Data applications are driving current research in very large-scale and distributed optimization, and have been the subject of recent dedicated workshops

and a major theme of optimization conferences.

The interest in large-scale optimization has caused a resurgence of classical methods (*e.g.* splitting methods, coordinate descent methods, stochastic methods, *etc.*) that that are less accurate than interior-point methods, but use very inexpensive iterations and are therefore better suited for very large problems. (See, for example, the survey [CBS14].) However, despite the advances in first-order methods for many types of large-scale nonsmooth convex optimization problems, success in applying similar methods to sparse SDPs has been more limited. As we have shown in this thesis, without using decomposition, handling this constraint usually involves full eigenvalue decompositions, which is expensive in both computations and memory usage. The issue is not improved when using a Lanczos-type method, since the intermediate variables are usually dense, and computing a PSD projection often involves computing much more than a few eigenvectors.

In this thesis we have applied chordal decomposition to solve large sparse SDPs for two classes of problems: linear conic optimization and matrix nearness problems. We have described techniques that exploit chordal decomposition to solve very large, sparse problems. Hopefully, we have illustrated the versatility of the methods; there are many more first-order methods we did not explore, but which can also exploit chordal decomposition of SDPs in very similar ways. Similarly, the extensions to stochastic methods (like stochastic gradient descent and stochastic coordinate descent) may lead to exciting results.

In almost all the methods presented, the most expensive computations (evaluations of proximal operators) can be parallelized. We explored this in a few informal experiments, and found that in most cases, it is possible to achieve nearly linear speedup on a multicore machine. However, when extended to distributed computing across multiple machines, we found that the interprocessor communication lag diminished any parallelization benefit. A fully distributed implementation of these methods remains a promising but difficult direction for further research.

References

[AA12] S. Al-Homidan and M. AlQarni. "Structure methods for solving the nearest correlation matrix problem." *Positivity*, **16**(3):497–508, 2012.

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

[ADL12] M. S. Andersen, J. Dahl, Z. Liu, and L. Vandenberghe. "Interior-point methods for large-scale cone programming." In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*, pp. 55–83. MIT Press, 2012.

[ADS13] C. M. Alaíz, F. Dinuzzo, and S. Sra. "Correlation matrix nearness and completion under observation uncertainty." *IMA Journal of Numerical Analysis*, 2013.

[ADV10a] M. Andersen, J. Dahl, and L. Vandenberghe. *CVXOPT: A Python Package for Convex Optimization*. `www.cvxopt.org`, 2010.

[ADV10b] M. Andersen, J. Dahl, and L. Vandenberghe. *SMCP: Python Extension for Sparse Matrix Cone Programs*, 2010. `abel.ee.ucla.edu/smcp`.

[ADV10c] 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.

[AHM88] J. Agler, J. W. Helton, S. McCullough, and L. Rodman. "Positive semidefinite matrices with a given sparsity pattern." *Linear Algebra and Its Applications*, **107**:101–149, 1988.

[AHV14] M. S. Andersen, A. Hansson, and L. Vandenberghe. "Reduced-complexity semidefinite relaxations of optimal power flow problems." *Power Systems, IEEE Transactions on*, **29**(4):1855–1863, 2014.

[AKG13] B. Alipanahi, N. Krislock, A. Ghodsi, H. Wolkowicz, L. Donaldson, and M. Li. "Determining protein structures from NOESY distance constraints by semidefinite programming." *Journal of Computational Biology*, **20**(4):296–310, 2013.

[AKW99] A. Y. Alfakih, A. Khandani, and H. Wolkowicz. "Solving Euclidean distance matrix completion problems via semidefinite programming." *Computational Optimization and Applications*, **12**(1-3):13–30, 1999.

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

[Aus76] A. Auslender. *Optimisation: Méthodes Numériques*. Masson, 1976.

[AW05] S. Al-Homidan and H. Wolkowicz. "Approximate and exact completion problems for Euclidean distance matrices using semidefinite programming." *Linear Algebra and Its Applications*, **406**:109–141, 2005.

[BB96] H. H. Bauschke and J. M. Borwein. "On projection algorithms for solving convex feasibility problems." *SIAM review*, **38**(3):367–426, 1996.

[BC11] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.

[BD86] J. P. Boyle and R. L. Dykstra. "A method for finding projections onto the intersection of convex sets in Hilbert spaces." In R. Dykstra, T. Robertson, and F. T. Wright, editors, *Advances in Order Restricted Statistical Inference*, volume 37 of *Lecture Notes in Statistics*, pp. 28–47. Springer-Verlag, 1986.

[BHF08] X. Bai, W. Huang, K. Fujisawa, and Y. Wang. "Semidefinite programming for optimal power flow problems." *International Journal of Electrical Power & Energy Systems*, **30**(6):383–392, 2008.

[BHR10] R. Borsdorf, N. J. Higham, and M. Raydan. "Computing a nearest correlation matrix with factor structure." *SIAM Journal on Matrix Analysis and Applications*, **31**(5):2603–2622, 2010.

[BJ95] M. Bakonyi and C. R. Johnson. "The Euclidian distance matrix completion problem." *SIAM Journal on Matrix Analysis and Applications*, **16**(2):646–654, 1995.

[BK73] C. Bron and J. Kerbosch. "Algorithm 457: Finding all cliques of an undirected graph." *Communications of the ACM*, **16**(9):575–577, 1973.

[BL08] P. J. Bickel and E. Levina. "Covariance regularization by thresholding." *The Annals of Statistics*, pp. 2577–2604, 2008.

[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.

[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. 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.

[BT97] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, Belmont, Mass., 1997.

[BT09] 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.

[BT13] A. Beck and L. Tetruashvili. "On the convergence of block coordinate descent type methods." *SIAM Journal on Optimization*, **23**(4):2037–2060, 2013.

[BT14] A. Beck and M. Teboulle. "A fast dual proximal gradient algorithm for convex minimization and applications." *Operations Research Letters*, **42**:1–6, 2014.

[Bun74] P. Buneman. "A characterization of rigid circuit graphs." *Discrete Mathematics*, **9**:205–212, 1974.

[BV96] S. Boyd and L. Vandenberghe. "Semidefinite programming relaxations of non-convex problems in control and combinatorial optimization." In *Mathematical Engineering: A Kailath Festschrift*, chapter 1, pp. 1–10. Kluwer, 1996.

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

[BV08] S. Burer and D. Vandenbussche. "A finite branch-and-bound algorithm for nonconvex quadratic programming via semidefinite relaxations." *Mathematical Programming*, **113**(2):259–282, 2008.

[BX05] S. Boyd and L. Xiao. "Least-squares covariance matrix adjustment." *SIAM Journal on Matrix Analysis and Applications*, **27**(2):532–546, 2005.

[BY04a] S. J. Benson and Y. Ye. "DSDP5: A software package implementing the dual-scaling algorithm for semidefinite programming." Technical Report ANL/MCS-TM-255, Mathematics and Computer Science Division, Argonne National Laboratory, 2004.

[BY04b] P. Biswas and Y. Ye. "Semidefinite programming for ad hoc wireless sensor network localization." In *Third International Symposium on Information Processing in Sensor Networks, IPSN'04*, pp. 46–54, 2004.

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

[Cau47] A. Cauchy. "Méthode générale pour la résolution des systemes d'équations simultanées." *Comp. Rend. Sci. Paris*, **25**(1847):536–538, 1847.

[CBS14] V. Cevher, S. Becker, and M. Schmidt. "Convex optimization for big data: Scalable, randomized, and parallel algorithms for big data analytics." *Signal Processing Magazine, IEEE*, **31**(5):32–43, 2014.

[CD14] A. Chambolle and C. Dossal. "On the convergence of the iterates of "FISTA"." preprint, `https://hal.inria.fr/hal-01060130`, 2014.

[CH88] G. M. Crippen and T. F. Havel. *Distance Geometry and Molecular Conformation*, volume 74. Research Studies Press Taunton, England, 1988.

[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.

[CP11b] P. L. Combettes and J.-C. Pesquet. "Proximal splitting methods in signal processing." In H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pp. 185–212. Springer, 2011.

[CY00] C. Choi and Y. Ye. "Solving sparse semidefinite programs using the dual scaling algorithms with an iterative solver." *Manuscript, Department of Management Sciences, University of Iowa, Iowa City, IA*, **52242**, 2000.

[CY07] A. M.-C. Cho and Y. Ye. "Theory of semidefinite programming for sensor network localization." *Mathematical Programming, Series B*, **109**:367–384, 2007.

[CZ97] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 1997.

[Dan63] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963.

[Dan92] J. Dancis. "Positive semidefinite completions of partial hermitian matrices." *Linear Algebra and Appl.*, **175**:97–114, 1992.

[Dat10] J. Dattorro. *Convex Optimization and Euclidean Distance Geometry.* Lulu. com, 2010.

[dBE08] A. d'Aspremont, O. Banerjee, and L. El Ghaoui. "First-order methods for sparse covariance selection." *SIAM Journal on Matrix Analysis and Applications*, **30**(1):56–66, 2008.

[dE59] D.A. d'Esopo. "A convex programming procedure." *Naval Research Logistics Quarterly*, **6**(1):33–42, 1959.

[Dem72] A. P. Dempster. "Covariance selection." *Biometrics*, **28**:157–175, 1972.

[Der15] M. Derman. "IBM Watson Health, Epic and Mayo Clinic to unlock new insights from electronic health records." `https://www-03.ibm.com/press/us/en/pressrelease/46768.wss`, May 2015. Accessed: 2015-8-19.

[DG81] H. Dym and I. Gohberg. "Extensions of band matrices with band inverses." *Linear Algebra and Appl.*, **36**:1–24, 1981.

[DH73] W. E. Donath and A. J. Hoffman. "Lower bounds for the partitioning of graphs." *IBM Journal of Research and Development*, **17**:420–425, 1973.

[DH11] T. A. Davis and Y. Hu. "The University of Florida sparse matrix collection." *ACM Transactions on Mathematical Software*, **38**:1 – 25, 2011.

[DR83] I. S. Duff and J. K. Reid. "The multifrontal solution of indefinite sparse symmetric linear equations." *ACM Transactions on Mathematical Software*, **9**(3):302–325, 1983.

[DY14] D. Davis and W. Yin. "Convergence rate analysis of several splitting schemes.", 2014. arXiv preprint arXiv:1406.4834.

[Dyk83] R. L. Dykstra. "An algorithm for restricted least squares regression." *Journal of the Americal Statistical Association*, **78**:837–842, December 1983.

[EB92] J. Eckstein and D. Bertsekas. "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators." *Mathematical Programming*, **55**:293–318, 1992.

[Eck94] J. Eckstein. "Parallel alternating direction multiplier decomposition of convex programs." *Journal of Optimization Theory and Applications*, **80**(1):39–62, 1994.

[Edw14] J. Edwards. "Twitter is quietly building a $100 million business In Big Data." `http://www.businessinsider.com/twitter-100-million-big-data-business-2014-4`, April 2014. Accessed: 2015-8-19.

[ELV13] M. Eisenberg-Nagy, M. Laurent, and A. Varvitsiotis. "Complexity of the positive semidefinite matrix completion problem with a rank constraint." In *Discrete Geometry and Optimization*, pp. 105–120. Springer, 2013.

[FG65] D. R. Fulkerson and O. Gross. "Incidence matrices and interval graphs." *Pacific Journal of Mathematics*, **15**(3):835–855, 1965.

[FKK09] K. Fujisawa, S. Kim, M. Kojima, Y. Okamoto, and M. Yamashita. "User's manual for SparseCoLO: Conversion methods for sparse conic-form linear optimization problems." Technical report, Research Report B-453, Dept. of Math. and Comp. Sci., Tokyo Institute of Technology, Tokyo 152-8552, Japan, 2009.

[FKM00] 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**:647–674, 2000.

[Gav74] F. Gavril. "The intersection graphs of subtrees in trees are exactly the chordal graphs." *Journal of Combinatorial Theory Series B*, **16**:47–56, 1974.

[GB08] M. Grant and S. Boyd. "Graph implementations for nonsmooth convex programs." In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control (a tribute to M. Vidyasagar)*, pp. 95–110. Springer, 2008.

[GB12] M. Grant and S. Boyd. *CVX: Matlab Software for Disciplined Convex Programming, version 2.0 (beta)*. `cvxr.com`, 2012.

[GHH90] W. Glunt, T. L. Hayden, S. Hong, and J. Wells. "An alternating projection algorithm for computing the nearest Euclidean distance matrix." *SIAM Journal on Matrix Analysis and Applications*, **11**(4):589–600, 1990.

[GHJ99] W. Glunt, T.L. Hayden, C.R. Johnson, and P. Tarazaga. "Positive definite completions and determinant maximization." *Linear algebra and its applications*, **288**:1–10, 1999.

[GJS84] R. Grone, C. R. Johnson, E. M Sá, and H. Wolkowicz. "Positive definite completions of partial Hermitian matrices." *Linear Algebra and Appl.*, **58**:109–124, 1984.

[GL89] A. George and J. W. H. Liu. "The evolution of the minimum degree ordering algorithm." *Siam review*, **31**(1):1–19, 1989.

[GLS88] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer-Verlag, 1988.

[GM89] N. Gaffke and R. Mathar. "A cyclic projection algorithm via duality." *Metrika*, **36**:29–54, 1989.

[Gol62] A. A. Goldstein. "Cauchy's method of minimization." *Numerische Mathematik*, **4**(1):146–150, 1962.

[Gol64] A. A. Goldstein. "Convex programming in Hilbert space." *Bulletin of the American Mathematical Society*, **70**:709–710, 1964.

[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.

[GS00] L. Grippo and M. Sciandrone. "On the convergence of the block nonlinear Gauss–Seidel method under convex constraints." *Operations Research Letters*, **26**(3):127–136, 2000.

[GT82] A. Griewank and Ph. L. Toint. "Partitioned variable metric updates for large structured optimization problems." *Numerische Mathematik*, **39**:119–137, 1982.

[GT84] A. Griewank and Ph. L. Toint. "On the existence of convex decompositions of partially separable functions." *Mathematical Programming*, **28**:25–49, 1984.

[GW95] M. X. Goemans and D. P. Williamson. "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming." *Journal of the Association for Computing Machinery*, **42**(6):1115–1145, 1995.

[Han88] S.-P. Han. "A successive projection method." *Mathematical Programming*, **40**:1–14, 1988.

[Heg06] P. Heggernes. "Minimal triangulation of graphs: a survey." *Discrete Mathematics*, **306**:297–317, 2006.

[Hig88] N.J. Higham. "Computing a nearest symmetric positive semidefinite matrix." *Linear algebra and its applications*, **103**:103–118, 1988.

[Hig02] N. J. Higham. "Computing the nearest correlation matrix—a problem from finance." *IMA Journal of Numerical Analysis*, **22**(3):329–343, 2002.

[HL88] S.-P. Han and G. Lou. "A parallel algorithm for a class of convex programs." *SIAM Journal on Control and Optimization*, **26**(2):345–355, 1988.

[HLW03] B. S. He, L. Z. Liao, and S. L. Wang. "Self-adaptive operator splitting methods for monotone variational inequalities." *Numerische Mathematik*, **94**:715–737, 2003.

[HM11] D. Henrion and J. Malick. "Projection methods for conic feasibility problems: applications to polynomial sum-of-squares decompositions." *Optimization Methods & Software*, **26**(1):23–46, 2011.

[HM12] D. Henrion and J. Malick. "Projection methods in conic optimization." In M. F. Anjos and J. B. Lasserre, editors, *Handbook on Semidefinite, Conic and Polynomial Optimization*, pp. 565–600. Springer, 2012.

[HPR89] J. Helton, S. Pierce, and L. Rodman. "The ranks of extremal positive semidefinite matrices with given sparsity pattern." *SIAM Journal on Matrix Analysis and Applications*, **10**:407–423, 1989.

[HS10] J. D. Hogg and J. A. Scott. *A Modern Analyse Phase for Sparse Tree-Based Direct Methods*. Science and Technology Facilities Council, 2010.

[Hu06] S. Hu. *Semidefinite Relaxation Based Branch-and-Bound Method for Nonconvex Quadratic Programming*. PhD thesis, Massachusetts Institute of Technology, 2006.

[HW88] T.L. Hayden and J. Wells. "Approximation by matrices positive semidefinite on a subspace." *Linear Algebra and its Applications*, **109**(0):115 – 130, 1988.

[HYW00] B. S. He, H. Yang, and S. L. Wang. "Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities." *Journal of Optimization Theory and Applications*, **106**:337–356, 2000.

[Jab12] R. A. Jabr. "Exploiting sparsity in SDP relaxations of the OPF problem." *IEEE Transactions on Power Systems*, **27**(2):1138–1139, 2012.

[Joh90] C. R. Johnson. "Matrix completion problems: a survey." In C. R. Johnson, editor, *Matrix Theory and Applications*, volume 40 of *Proceedings of Symposia in Applied Mathematics*, pp. 171–189. American Mathematical Society, 1990.

[JT95] C. R. Johnson and P. Tarazaga. "Connections between the real positive semidefinite and distance matrix completion problems." *Linear Algebra and Its Applications*, **223/224**:375–391, 1995.

[Kan39] L. V. Kantorovich. "The mathematical method of production planning and organization." *Management Science*, **6**:363–422, 1939.

[Kar84] N. Karmarkar. "A new polynomial-tme algorithm for linear programming." *Combinatorica*, **4**(4):373–395, 1984.

[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**: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.

[KMR14] N. Krislock, J. Malick, and F. Roupin. "Improved semidefinite bounding procedure for solving Max-Cut problems to optimality." *Mathematical Programming*, **143**(1-2):61–86, 2014.

[KR98] S. E. Karisch and F. Rendl. "Semidefinite programming and graph equipartition." In P. M. Pardalos and H. Wolkowicz, editors, *Topics in Semidefinite and Interior-Point Methods*, volume 18 of *Fields Institute Communications*, pp. 77–95. The American Mathematical Society, 1998.

[Kru64] J. B. Kruskal. "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis." *Psychometrika*, **29**(1):1–27, 1964.

[KW12] N. Krislock and H. Wolkowicz. "Euclidean distance matrices and applications." In M. F. Anjos and J. B. Lasserre, editors, *Handbook on Semidefinite, Conic and Polynomial Optimization*, chapter 30, pp. 879–914. Springer, 2012.

[Lam13] A. Lampitt. "The real story of how big data analytics helped Obama win." `http://www.infoworld.com/article/2613587/big-data/the-real-story-of-how-big-data-analytics-helped-obama-win.html`, February 2013. Accessed: 2015-8-19.

[Las01] J. B. Lasserre. "Global optimization with polynomials and the problem of moments." *SIAM Journal on Optimization*, **11**(3):796–817, 2001.

[Lau01] M. Laurent. "Matrix completion problems." In C. A. Floudas and P. M. Pardalos, editors, *Encyclopedia of Optimization*, volume III, pp. 221–229. Kluwer, 2001.

[LCB04] G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. Jordan. "Learning the kernel matrix with semidefinite programming." *The Journal of Machine Learning Research*, **5**:27–72, 2004.

[Liu92] J. W. H. Liu. "The multifrontal method for sparse matrix solution: theory and practice." *SIAM Review*, **34**:82–109, 1992.

[LL12] J. Lavaei and S. H. Low. "Zero duality gap in optimal power flow problem." *Power Systems, IEEE Transactions on*, **27**(1):92–107, 2012.

[LM79] P. L. Lions and B. Mercier. "Splitting algorithms for the sum of two nonlinear operators." *SIAM Journal on Numerical Analysis*, **16**(6):964–979, 1979.

[Lof04] J. Löfberg. *YALMIP : A Toolbox for Modeling and Optimization in MATLAB*, 2004.

[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, March 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, June 2014.

[LP66] E. Levitin and B. Polyak. "Constrained minimization methods." *USSR Computational Math. and Math. Physics*, **6**(5):1–50, 1966.

[LPP89] J. G. Lewis, B. W. Peyton, and A. Pothen. "A fast algorithm for reordering sparse matrices for parallel factorization." *SIAM Journal on Scientific and Statistical Computing*, **10**(6):1146–1173, 1989.

[LS91] L. Lovász and A. Schrijver. "Cones of matrices and set-functions and 0-1 optimization." *SIAM J. on Optimization*, **1**(2):166–190, 1991.

[LT93] Z.-Q. Luo and P. Tseng. "Error bounds and convergence analysis of feasible descent methods: a general approach." *Annals of Operations Research*, **46**(1):157–178, 1993.

[LV12] M. Laurent and A. Varvitsiotis. "The Gram dimension of a graph.", 2012. `arxiv.org/1112.5960`.

[Mal04] J. Malick. "A dual approach to semidefinite least-squares problems." *SIAM Journal on Matrix Analysis and Applications*, **26**(1):272–284, 2004.

[MAL14] R. Madani, M. Ashraphijuo, and J. Lavaei. "Promises of conic relaxation for contingency-constrained optimal power flow problem." In *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*, pp. 1064–1071. IEEE, 2014.

[Mar57] H. M. Markowitz. "The elimination form of the inverse and its application to linear programming." *Management Science*, **3**(3):255–269, 1957.

[Mar70] B. Martinet. "Régularisation d'inéquations variationnelles par approximations successives." *Revue Française d'Informatique et de Recherche Opérationnelle*, **4**(3):154–158, 1970.

[Mar14] B. Marr. "How Facebook is using Big Data: the good, the bad and the ugly." `https://www.linkedin.com/pulse/20140716060957-64875646-facebook-and-big-data-no-big-brother`, July 2014. Accessed: August 19, 2015.

[Meh92] S. Mehrotra. "On the implementation of a primal-dual interior point method." *SIAM Journal on Optimization*, **2**(4):575–601, November 1992.

[MLD14] D. K. Molzahn, B. C. Lesieutre, and C. L. DeMarco. "A sufficient condition for global optimality of solutions to the optimal power flow problem." *Power Systems, IEEE Transactions on*, **29**(2):978–979, 2014.

[MLL12] A. Mucherino, C. Lavor, L. Liberti, and N. Maculan. *Distance Geometry: Theory, Methods, and Applications*. Springer Science & Business Media, 2012.

[Mor65] J. J. Moreau. "Proximité et dualité dans un espace hilbertien." *Bull. Math. Soc. France*, **93**:273–299, 1965.

[MOS02] MOSEK ApS. *The MOSEK Optimization Tools Version 2.5. User's Manual and Reference*, 2002. Available from `www.mosek.com`.

[Nes83] Y. Nesterov. "A method of solving a convex programming problem with convergence rate $O(1/k^2)$." *Soviet Math. Dokl.*, **27**(2):372–376, 1983.

[Nes04] Yu. Nesterov. *Introductory Lectures on Convex Optimization*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2004.

[Nes12] Y. Nesterov. "Efficiency of coordinate descent methods on huge-scale optimization problems." *SIAM Journal on Optimization*, **22**(2):341–362, 2012.

[Neu50] J. v. Neumann. "Functional operators." In *The Geometry of Orthogonal Spaces, Ann. Math. Studies No. 22*, volume 2. Princeton Univ. Press Princeton, N. J, 1950.

[NFF03] K. Nakata, K. Fujitsawa, M. Fukuda, M. Kojima, and K. Murota. "Exploiting sparsity in semidefinite programming via matrix completion II: implementation and numerical details." *Mathematical Programming Series B*, **95**:303–327, 2003.

[NM53] J. Von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton Univ. Press, third edition, 1953.

[NN94] Yu. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Methods in Convex Programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, 1994.

[NW06] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006.

[NWV08] M. Nouralishahi, C. Wu, and L. Vandenberghe. "Model calibration for optical lithography via semidefinite programming." *Optimization and Engineering*, **9**:19–35, 2008.

[Par00] P. A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, 2000.

[Par03] P. A. Parrilo. "Semidefinite programming relaxations for semialgebraic problems." *Mathematical Programming Series B*, **96**:293–320, 2003.

[Pas79] G. H. Passty. "Ergodic convergence to a zero of the sum of monotone operators in Hilbert space." *Journal of Mathematical Analysis and Applications*, **72**:383–390, 1979.

[PB13] N. Parikh and S. Boyd. "Proximal algorithms." *Foundations and Trends in Optimization*, **1**(3):123–231, 2013.

[PL03] P. A. Parrilo and S. Lall. "Semidefinite programming relaxations and algebraic optimizations in control." *European Journal of Control*, **9**(2–3):307–321, 2003.

[Pol87] B. T. Polyak. *Introduction to Optimization*. Optimization Software, Inc., New York, 1987.

[Pow73] M. J. D. Powell. "On search directions for minimization algorithms." *Mathematical Programming*, **4**(1):193–201, 1973.

[Pri57] R. C. Prim. "Shortest connection networks and some generalizations." *Bell system technical journal*, **36**(6):1389–1401, 1957.

[PSB14] P. Patrinos, L. Stella, and A. Bemporad. "Douglas-Rachford splitting: Complexity estimates and accelerated variants." In *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pp. 4234–4239. IEEE, 2014.

[Qi13] H.-D. Qi. "A semismooth Newton method for the nearest Euclidean distance matrix problem." *SIAM Journal on Matrix Analysis and Applications*, **34**(1):67–93, 2013.

[QS06] H. Qi and D. Sun. "A quadratically convergent Newton method for computing the nearest correlation matrix." *SIAM journal on matrix analysis and applications*, **28**(2):360–385, 2006.

[QS10] H. Qi and D. Sun. "Correlation stress testing for value-at-risk: an unconstrained convex optimization approach." *Computational Optimization and Applications*, **45**(2):427–462, 2010.

[QS11] H. Qi and D. Sun. "An augmented Lagrangian dual approach for the H-weighted nearest correlation matrix problem." *IMA Journal of Numerical Analysis*, **31**(2):491–511, 2011.

[QXY13] H.-D. Qi, N. Xiu, and X. Yuan. "A Lagrangian dual approach to the single-source localization problem." *Signal Processing, IEEE Transactions on*, **61**(15):3815–3826, 2013.

[Ray05] E. G. Birgin and M. Raydan. "Robust stopping criteria for Dykstra's algorithm." *SIAM Journal on Scientific Computing*, **26**(4):1405–1414, 2005.

[RJ99] R. Rebonato and P. Jäckel. "The most general methodology to create a valid correlation matrix for risk management and option pricing

purposes." *Quantitative Research Centre of the NatWest Group*, **19**, 1999.

[Roc70] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, second edition, 1970.

[Roc76] R. T. Rockafellar. "Monotone operators and the proximal point algorithm." *SIAM J. Control and Opt.*, **14**(5):877–898, August 1976.

[RS09] J. K. Reid and J. A. Scott. "An out-of-core sparse Cholesky solver." *ACM Transactions on Mathematical Software (TOMS)*, **36**(2):9, 2009.

[RT14] P. Richtárik and M. Takáč. "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function." *Mathematical Programming*, **144**(1-2):1–38, 2014.

[RTL76] D. J. Rose, R. E. Tarjan, and G. S. Lueker. "Algorithmic aspects of vertex elimination on graphs." *SIAM Journal on Computing*, **5**(2):266–283, 1976.

[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.

[Sch35] I. J. Schoenberg. "Remarks to Maurice Fréchet's article "Sur la définition axiomatique d'une classe d'espaces vectoriels distanciés applicables vectoriellement sur l'espace de Hilbert"." *Annals of Mathematics*, **36**(3):724–732, 1935.

[Sch38] I. J. Schoenberg. "Metric spaces and positive definite functions." *Transactions of the American Mathematical Society*, **44**(3):522–536, 1938.

[Smi08] R. L. Smith. "The positive definite completion problem revisited." *Linear Algebra and Its Applications*, **429**:1442–1452, 2008.

[Spi83] J. E. Spingarn. "Partial inverse of a monotone operator." *Applied Mathematics and Optimization*, **10**:247–265, 1983.

[Spi85] J. E. Spingarn. "Applications of the method of partial inverses to convex programming: decomposition." *Mathematical Programming*, **32**:199–223, 1985.

[SS73] R.W.H. Sargent and D.J. Sebastian. "On the convergence of sequential minimization algorithms." *Journal of Optimization Theory and Applications*, **12**(6):567–575, 1973.

[Stu99] J. F. Sturm. "Using SEDUMI 1.02, a Matlab Toolbox for Optimization Over Symmetric Cones." *Optimization Methods and Software*, **11-12**:625–653, 1999.

[SV04] G. Srijuntongsiri and S. Vavasis. "A fully sparse implementation of a primal-dual interior-point potential reduction method for semidefinite programming." 2004. `arXiv:cs/0412009`.

[SV15] Y. Sun and L. Vandenberghe. "Decomposition methods for sparse matrix nearness problems." 2015. Under review.

[Tar83] R. E. Tarjan. *Data Structures and Network Algorithms.* Society for Industrial and Applied Mathematics, 1983.

[Tro97] M. W. Trosset. "Applications of multidimensional scaling to molecular conformation." Technical report, Rice University, 1997.

[Tse88] P. Tseng. "A very simple polynomial-time algorithm for linear programming." Report No. LIDS-P-1818, September 1988.

[Tse90] P. Tseng. "Further applications of a splitting algorithm to decomposition in variational inequalities and convex programming." *Mathematical Programming*, **48**:249–263, 1990.

[Tse91] P. Tseng. "Applications of a splitting algorithm to decomposition in convex programming and variational inequalities." *SIAM Journal on Control and Optimization*, **29**(1):119–138, 1991.

[Tse93] P. Tseng. "Dual coordinate ascent methods for non-strictly convex minimization." *Mathematical Programming*, **59**:231–247, 1993.

[Tse01] P. Tseng. "Convergence of a block coordinate descent method for nondifferentiable minimization." *Journal of Optimization Theory and Applications*, **109**:475–494, 2001.

[Tse08] P. Tseng. "On accelerated proximal gradient methods for convex-concave optimization." 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 from `www.math.nus.edu.sg/~mattohkc/sdpt3.html`.

[TTT07] K. C. Toh, R. H. Tütüncü, and M. J. Todd. "Inexact primal-dual path-following algorithms for a special class of convex quadratic SDP and related problems." *Pacific Journal of Optimization*, **3**, 2007.

[TW67] W. F. Tinney and J. W. Walker. "Direct solutions of sparse network equations by optimally ordered triangular factorization." *Proceedings of the IEEE*, **55**(11):1801–1809, 1967.

[TY84] R. E. Tarjan and M. Yannakakis. "Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs." *SIAM Journal on Computing*, **13**(3):566–579, 1984.

[VA15] L. Vandenberghe and M. S. Andersen. "Chordal graphs and semidefinite optimization." *Foundations and Trends® in Optimization*, **1**(4):241433, 2015.

[VB95] L. Vandenberghe and S. Boyd. "Semidefinite programming." *SIAM Review*, pp. 49–95, 1995.

[VB99] L. Vandenberghe and S. Boyd. "Applications of semidefinite programming." *Applied Numerical Mathematics*, **29**:283–299, 1999.

[WL01] S. L. Wang and L. Z. Liao. "Decomposition method with a variable parameter for a class of monotone variational inequality problems." *Journal of Optimization Theory and Applications*, **109**(2):415–429, 2001.

[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.

[WS06] K. Q. Weinberger and L. K. Saul. "An introduction to nonlinear dimensionality reduction by maximum variance unfolding." In *AAAI*, volume 6, pp. 1683–1686, 2006.

[WSV00] H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors. *Handbook of Semidefinite Programming*, volume 27 of *International Series in Operations Research and Management Science*. Kluwer Academic Publishers, Boston, MA, 2000.

[Wut89] K. Wüthrich. "Protein structure determination in solution by nuclear magnetic resonance spectroscopy." *Science*, **243**(4887):45–50, 1989.

[Yan81] M. Yannakakis. "Computing the minimum fill-in is NP-complete." *SIAM Journal on Algebraic and Discrete Methods*, **2**(1):77–79, 1981.

[YFK03] M. Yamashita, K. Fujisawa, and M. Kojima. "Implementation and evaluation of SDPA 6.0 (SemiDefinite Programming Algorithm 6.0)." *Optimization Methods and Software*, **18**(4):491–505, 2003.

[YH38] G. Young and A. S. Householder. "Discussion of a set of points in terms of their mutual distances." *Psychometrika*, **3**(1):19–22, 1938.

[You13] F. W. Young. *Multidimensional Scaling: History, Theory, and Applications*. Psychology Press, 2013.