Solving Semidefinite Programs using Preconditioned Conjugate Gradients

Henry Wolkowicz *

April 21, 2003

University of Waterloo
Department of Combinatorics & Optimization
Waterloo, Ontario N2L 3G1, Canada
Research Report CORR 2001-49 [1]

Key words: Semidefinite Programming, Large Sparse Problems, Inexact Gauss-Newton Method, Preconditioned Conjugate Gradients, Crossover, Max-Cut Problem.

Abstract

The contribution of this paper is to describe a general technique to solve some classes of large but sparse semidefinite problems via a robust primal-dual interior-point technique which uses an inexact Gauss-Newton approach with a matrix free preconditioned conjugate gradient method. This approach avoids the ill-conditioning pitfalls that result from symmetrization and from forming the so-called normal equations, while maintaining the primal-dual framework.

First, we apply a preprocessing step that reduces the optimality conditions *before linearization* and results in a single, well-conditioned, overdetermined bilinear operator equation. We then use preconditioned conjugate-gradients to approximately solve the linearization at every step of a path-following approach. We do not form the matrix representation of the linearization. In addition, once close to the optimal solution, we apply a crossover technique after which the iterates are no longer forced to be positive definite.

In the experimental part of this paper, we use Max-Cut to illustrate the technique. We describe preconditioners that exploit the operator structure of the constraints and show how the crossover can be effective in practice and how it allows for warm starts. In all cases we obtain high accuracy solutions.

*Research supported by The Natural Sciences and Engineering Research Council of Canada. E-mail hwolkowicz@uwaterloo.ca

[1] A preliminary version of this paper, *Semidefinite Programming and Some Closest Matrix Approximation Problems*, was presented at the 1st Annual McMaster Optimization Conference (MOPTA 01), August 2-4, 2001 Hamilton, Ontario.

[1] URL for paper and MATLAB programs:
http://orion.math.uwaterloo.ca/~hwolkowi/henry/reports/ABSTRACTS.html

Contents

1 Introduction

The many applications, elegant theory, and practical algorithms for Semidefinite Programming (SDP) have, arguably, made SDP the hottest area of optimization. Its popularity, however, remains concentrated among specialists rather than mainstream nonlinear programming practitioners and users. Most of the current algorithmic approaches use symmetrizations and apply Newton's method. The next (complicated and costly) step is to use block elimination and construct the matrix for the *normal equations*. In general, this results in a dense ill-conditioned system that can be solved using Cholesky factorization. However, there is a lack of SDP solvers that can efficiently exploit sparsity. (For example, see the papers [13, 24] and the references therein.) In addition, it is difficult to avoid the ill-conditioning of the normal equations system near the optimum or near the semidefinite boundary. (See e.g. the improved factorization scheme in [29] and the references therein.) Therefore, it is hard to apply iterative methods such as preconditioned conjugate gradients (PCG). These difficulties mean that the SDP model will not replace the simpler Linear Programming (LP) model even in cases where SDP provides stronger relaxations.

The purpose of this paper is to illustrate a simple alternative algorithmic development for SDP that is completely based on standard principles from numerical analysis, i.e. on the (inexact) Gauss-Newton (GN) method with PCG. The only additional tool we use is the definition of a linear operator and its adjoint. There is no need to construct matrix representations of operators. No ill-conditioned system is formed. (Thus long steps to and beyond the semidefinite boundary can be used.) Instead, we use preprocessing (presolve) to

reduce the optimality conditions at the start, before any linearizations are performed. In addition, we use a crossover technique once the barrier parameter μ is small enough, i.e. after the crossover we use a step length of one and no longer backtrack to force positive semidefiniteness of the iterates. (This allows for warm starts.)

For illustration, we restrict ourselves in this paper to the semidefinite relaxation of the Max-Cut problem. We derive and use the *optimal diagonal* and the *incomplete Cholesky* preconditioners in the PCG method. Our approach is currently being applied to several other problems with more sophisticated preconditioners, see e.g. [1, 28, 16].

1.1 Related Work

The GN direction was introduced in [20]. A scaled variant was shown to converge in polynomial time in [7], see also [21].

Several recent papers have concentrated on exploiting the special structure of the SDP relaxation for the Max-Cut problem. A discussion of several of the methods is given in Burer-Montreiro [5]. (See also [32].) In particular, Benson et al [3] present an interior-point method based on potential-reduction and dual-scaling; while, Helmberg-Rendl [18] use a bundle-trust approach on a nondifferentiable function arising from the Lagrangian dual. Both of these methods exploit the small dimension n of the dual problem compared to the dimension $\begin{pmatrix} n+1 \\ 2 \end{pmatrix} = n(n+1)/2$ of the primal problem. Moreover, the dual feasibility equation is sparse if the matrix of the quadratic form Q is sparse. Therefore each iteration is inexpensive. However, these are not primal-dual methods and do not easily obtain high accuracy approximations of optimal solutions without many iterations.

The methods in [5, 32] use Cholesky type transformations $X = VV^T$ and discard the semidefinite constraint. For example, the method in [5] reduces to a first order gradient projection method. They argue for using a search direction based on first order information only (rather than second order information as used in interior-point methods) since this results in fast and inexpensive iterations. However, the cost is that they need many iterations, contrary to interior-point methods which need relatively few. Therefore, their approach is useful in obtaining optimal solutions to moderate accuracy. Their formulation has $\begin{pmatrix} n \\ 2 \end{pmatrix} = n(n-1)/2$ variables (unknowns).

Similarly, our algorithm has $\begin{pmatrix} n+1 \\ 2 \end{pmatrix}$ variables. However, it is based on the primal-dual framework. We make the opposite argument, (common in nonlinear programming), i.e. *it is important to start with a good search direction.* We find a GN search direction using a PCG approach. We use an *inexact Newton* framework, e.g. [9], to lower the cost of each iteration. In fact, restricting CG to one iteration results in a first order method with the same cost per iteration as that in [5]. Since we have a well-posed system, (full rank Jacobian), we can obtain q-quadratic convergence and highly accurate approximations of optimal solutions. The cost of each CG iteration is a sparse matrix multiplication (essentially $Z\Delta X$) and a matrix scaling (essentially Diag $(\Delta y)X$).

2 Motivation and Bilinear Optimality Conditions

2.1 General Case

The primal semidefinite problem we consider is

$$\begin{array}{rrll} & p^* := & \max & \operatorname{trace} CX \\ (\text{PSDP}) & & \text{s.t.} & \mathcal{A}X = b \\ & & & X \succeq 0. \end{array} \tag{2.1}$$

Its dual is

$$\begin{array}{rrll} & d^* := & \min & b^T y \\ (\text{DSDP}) & & \text{s.t.} & \mathcal{A}^* y - Z = C \\ & & & Z \succeq 0, \end{array} \tag{2.2}$$

where $X, Z \in \mathcal{S}^n$, $\mathcal{S}^n$ denotes the space of $n \times n$ real symmetric matrices, and $\succeq$ denotes positive semidefiniteness. $\mathcal{A} : \mathcal{S}^n \rightarrow \Re^m$ is a linear operator and $\mathcal{A}^*$ is the adjoint operator.

SDP looks just like a LP and many of the properties from LP follow through. *Weak duality* $p^* \leq d^*$ holds. However, as in general convex programming, *strong duality* can fail; there can exist a nonzero duality gap at optimality $p^* < d^*$ and/or the dual may not be attained. (Strong duality can be ensured using a suitable constraint qualification (CQ), e.g. Slater's CQ; strict feasibility.)

The primal-dual pair leads to the elegant primal-dual optimality conditions, i.e. primal feasibility, dual feasibility, and complementary slackness, e.g. [27].

Theorem 2.1 *Suppose that suitable CQs hold for (PSDP), (DSDP) (for example Slater's). The primal-dual variables* (X, y, Z)*, with* $X, Z \succeq 0$*, are optimal for the primal-dual pair of SDPs if and only if*

$$F(X, y, Z) := \begin{pmatrix} \mathcal{A}^* y - Z - C \\ \mathcal{A}X - b \\ ZX \end{pmatrix} = 0. \tag{2.3}$$

■

Since the product ZX is not necessarily symmetric, we have $F : \mathcal{S}^n \times \Re^m \times \mathcal{S}^n \rightarrow \mathcal{S}^n \times \Re^m \times \mathcal{M}^n$, where $\mathcal{M}^n$ is the space of $n \times n$ matrices. (Z, X are diagonal matrices in the LP case. This is one of the differences between SDP and LP that, perhaps, has had the most influence on algorithmic development for SDP.) Currently, the successful primal-dual algorithms follow the LP approaches, i.e. they are path following interior-point algorithms that use Newton's method to solve (symmetrizations of!) the perturbed optimality conditions:

$$F_\mu(X, y, Z) := \begin{pmatrix} \mathcal{A}^* y - Z - C \\ \mathcal{A}X - b \\ ZX - \mu I \end{pmatrix} = 0. \tag{2.4}$$

These symmetrization schemes can be considered from the view of preconditioning of the optimality conditions (2.4), using some preconditioner. Suppose that we start with the

optimality conditions that arise from the dual log-barrier problem, i.e.

$$\begin{pmatrix} \mathcal{A}^*y - Z - C \\ \mathcal{A}X - b \\ X - \mu Z^{-1} \end{pmatrix} = 0. \tag{2.5}$$

Premultiplication by Z (a form of preconditioning) leads to a less nonlinear system, avoids the ill-conditioning, and results in the more familiar $(ZX - \mu I)$ form.

$$\begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & Z\cdot \end{pmatrix} \begin{pmatrix} \mathcal{A}^*y - Z - C \\ \mathcal{A}X - b \\ X - \mu Z^{-1} \end{pmatrix} = \begin{pmatrix} \mathcal{A}^*y - Z - C \\ \mathcal{A}X - b \\ ZX - \mu I \end{pmatrix} = 0. \tag{2.6}$$

Unfortunately, the system (2.6) is overdetermined, which prevents the next step used in LP, i.e. application of Newton's method. Therefore, preconditioning is done a second time using the linear operator $\mathcal{P}$ with symmetrization operator $\mathcal{S}$.

$$\mathcal{P}F_\mu(x,y,Z) = \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & \mathcal{S} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & Z\cdot \end{pmatrix} \begin{pmatrix} \mathcal{A}^*y - Z - C \\ \mathcal{A}X - b \\ X - \mu Z^{-1} \end{pmatrix} = 0. \tag{2.7}$$

However, the symmetrizations used in the literature, in general, reverse the previous process, i.e. after the symmetrization the ill-conditioning problem has returned. This framework encompasses many different symmetrized systems with various acronyms, similar to the area of quasi-Newton methods, e.g. AHO, HKM, NT, GT, MTW. The survey of search directions by Todd [30] presented several search directions and their theoretical properties, including: well-definedness, scale invariance, and primal-dual symmetry. For example, Section 4 in that paper studies twenty different primal-dual search directions.

The point of view and motivation taken in this paper is that after the change to the $ZX - \mu I$ system is made, the algorithm can be viewed as path following, i.e. the interiority condition is not essential. Moreover, the framework of symmetrization in itself (contrary to the premultiplication by Z) can be *counterproductive*. This can be seen from basic texts in Numerical Analysis and the preconditioning point of view in (2.7). Let us ignore that some of the variables are in matrix space, and look at the overdetermined nonlinear system (2.4), $F_\mu(x,y,Z) = 0$, with zero residual. Then the approach in any standard text (e.g. [10]) is to apply the GN method. This method has many desirable features. Preconditioning (from the left, since it is scale invariant from the right) is recommended if it results in *improved conditioning of the problem*, or in a problem that is *less nonlinear*. This view can be used to motivate our GN approach, i.e. the symmetrization schemes used in the literature are not motivated by conditioning considerations. But, on the other hand, they attempt to recreate the LP type framework. In instances studied in Kruk et al. [20], theoretical and empirical evidence shows that conditioning worsened for these symmetrization schemes, i.e. the opposite effect of preconditioning was observed. In particular, many of the symmetrizations have a catastrophic effect in that they result in an ill-posed problem, i.e. a singular Jacobian at $\mu = 0$, resulting in numerical difficulties in obtaining high accuracy approximations of optimal solutions.

The second motivation for GN arises from the need to exploit sparsity in many applications of SDP. The linearization of (2.7) yields

$$\begin{pmatrix} 0 & \mathcal{A}^* & -I \\ \mathcal{A} & 0 & 0 \\ \mathcal{S}(Z\cdot) & 0 & \mathcal{S}(\cdot X) \end{pmatrix} \begin{pmatrix} \Delta Z \\ \Delta y \\ \Delta X \end{pmatrix} = -F_\mu(X, y, Z). \tag{2.8}$$

As in LP, block elimination is used on (2.8) in order to exploit the four blocks of zeros. The first row is used to eliminate ΔZ in the third row. We can leave the symmetrization to later.

$$\begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -X & 0 & I \end{pmatrix} \begin{pmatrix} 0 & \mathcal{A}^* & -I \\ \mathcal{A} & 0 & 0 \\ Z\cdot & 0 & \cdot X \end{pmatrix} = \begin{pmatrix} 0 & \mathcal{A}^* & -I \\ \mathcal{A} & 0 & 0 \\ Z\cdot & -(\mathcal{A}^*\cdot)X & 0 \end{pmatrix} \tag{2.9}$$

Then the last row is used to eliminate ΔX; this leaves the normal equations of size m in Δy. Then backsolves are used to recover $\Delta Z, \Delta X$. (The backsolve for ΔX can also be shown to be ill-posed, i.e. it becomes ill-conditioned as the optimum is approached [16].) Therefore, though the literature views this as a true primal-dual method, it can also be viewed as a dual method, since it uses a modified/simplified system to find the dual variables Δy and then recovers the primal variables ΔX using an ill-conditioned backsolve.

The symmetrizations, the corresponding normal equations systems and the resulting ill-conditioning that arises from these two operations, combine to make it extremely difficult to exploit sparsity. Many attempts have been made. But in each case the matrix problem is changed to a vector problem and then the structure of the original problem is used in the vector problem. For the GN method, a standard approach to solving large sparse problems is to use a preconditioned conjugate gradient method. Again, as above, this does not need to take into account that some of the variables are in a matrix space. There is no need to change back and forth between matrices and vectors. One should consider the function as an operator between vector spaces, i.e. a black box for the GN method with preconditioned conjugate gradients (PCG).

In addition, we attempt to exploit the blocks of zeros in (2.8) in a different way, i.e. so as to avoid the ill-conditioning that results from the backsolve for ΔX. To fully exploit this approach, we first eliminate the linear equations from the optimality conditions in a preprocessing phase. For a linear constraint $\mathcal{A}(X) = b$, where the linear operator $\mathcal{A}$ is full-rank, m, let the linear operator $\mathcal{N} : \Re^{\binom{n+1}{2}-m} \to \mathcal{S}^n$ have range equal to the null space of $\mathcal{A}$. Then

$$\begin{aligned} \mathcal{A}(X) = b \quad &\text{iff} \quad X = \mathcal{A}^\dagger b + (I - \mathcal{A}^\dagger \mathcal{A})W, \quad \text{for some } W \in \mathcal{S}^n \\ &\text{iff} \quad X = \mathcal{A}^\dagger b + \mathcal{N}(w), \quad \text{for some } w \in \Re^{\binom{n+1}{2}-m}. \end{aligned} \tag{2.10}$$

We can now substitute for both Z, X in the complementarity equation and obtain a smaller but equivalent system describing the optimality conditions. (By abuse of notation, we keep the symbol F for the nonlinear operator. The meaning is clear from the context.) We now state the obvious corollary for future reference.

Corollary 2.2 *Suppose that suitable CQs hold for (PSDP), (DSDP) (for example Slater's). Suppose also that $\mathcal{A}$ is onto (full rank) and $\mathcal{N}$ defines the null space as in (2.10). Then variables (x, y) are optimal for the primal-dual pair of SDPs if and only if*

$$F(x, y) := (\mathcal{A}^*(y) - C)(\mathcal{A}^\dagger b + \mathcal{N}(x)) = 0, \tag{2.11}$$

$(\mathcal{A}^*(y) - C) \succeq 0$ *and* $(\mathcal{A}^\dagger b + \mathcal{N}(x)) \succeq 0$.

Proof. Follows from substitution of Z and X into the complementarity equation. ■

One interesting consequence of this re-writing of the optimality conditions is given in the following:

Theorem 2.3 *Consider the primal-dual SDP pair (PSDP),(DSDP). Suppose that $\mathcal{A}$ is onto (full rank), $\mathcal{N}$ defines the null space as in (2.10), and X, y, Z are unique primal-dual optimal solutions that satisfy strict complementary slackness, i.e. $X + Z \succ 0$. Then the matrix of the system*

$$\begin{aligned} -F(x, y) &= F'(x, y) \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \\ &= (\mathcal{A}^*(y) - C)\mathcal{N}(\Delta x) + \mathcal{A}^*(\Delta y)(\mathcal{N}(x) + \mathcal{A}^\dagger b) \end{aligned} \tag{2.12}$$

(F', Jacobian of $F(x, y)$) is full rank.

Proof. That the original overdetermined system $F'(X, y, Z)$ is full rank is proved in [20]. We observe that the system is obtained by block Gaussian elimination, which is equivalent to premultiplication by invertible operators. ■

Remark 2.4 *If we choose $\mathcal{N}$ correctly, e.g. an isometry, then the condition number for the Jacobian F' is not increased by the reduction to the system in* (2.12)*, i.e. the condition number of the system in* (2.9) *is at least as large as the condition number after we restrict to a subspace*

$$\begin{pmatrix} 0 & \mathcal{A}^* & -I \\ \mathcal{A} & 0 & 0 \\ Z\cdot & -(\mathcal{A}^*\cdot)X & 0 \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & \mathcal{N} \end{pmatrix} = \begin{pmatrix} 0 & \mathcal{A}^* & -I \\ \mathcal{A}\mathcal{N} & 0 & 0 \\ Z\mathcal{N}\cdot & -(\mathcal{A}^*\cdot)X & 0 \end{pmatrix}. \tag{2.13}$$

The second row reduces to zero and the third row yields the system in (2.12).

2.2 Specializing to Max-Cut

The Max-Cut problem, e.g. [12], consists in finding a partition of the set of vertices of a given undirected graph with weights on the edges so that the sum of the weights of the edges cut by the partition is maximized. This NP-hard discrete optimization problem can

be formulated as the following (quadratic) program (e.g. Q is a multiple of the Laplacian matrix of the graph).

$$\text{(MC0)} \qquad \begin{array}{rll} \mu^* := & \max & v^T Q v \\ & \text{s.t.} & v_i^2 = 1, \quad i = 1, \ldots, n. \end{array} \tag{2.14}$$

Using Lagrangian relaxation, (see e.g. [2]), one can derive the following semidefinite relaxation of (MC0).

$$\text{(P)} \qquad \begin{array}{rll} \mu^* \leq \nu^* := & \max & \operatorname{trace} QX \\ & \text{s.t.} & \operatorname{diag}(X) = e \\ & & X \succeq 0, X \in \mathcal{S}^n, \end{array} \tag{2.15}$$

where diag denotes the vector formed from the diagonal elements and e denotes the (column) vector of ones. For our purposes we assume, without loss of generality, that $\operatorname{diag}(Q) = 0$. This relaxation has been extensively studied. It has been found to be surprisingly strong both in theory and in practice. (See e.g. the survey paper [14].)

We work with the trace inner product in matrix space

$$\langle M, N \rangle = \operatorname{trace} M^T N, \qquad M, N \in \mathcal{M}^n .$$

This definition follows through to $\mathcal{S}^n$, the space of $n \times n$ symmetric matrices. The induced norm is the *Frobenius norm*, $\|M\|_F = \sqrt{\operatorname{trace} M^T M}$. Upper case letters $M, X, \ldots$ are used to denote matrices; while lower case letters are used for vectors.

2.2.1 Adjoint Operators; Generalized Inverses; Projections

We define several linear operators on vectors and matrices. We also need the adjoints when finding the dual SDP and also when applying the conjugate gradient method. $v = \operatorname{vec}(M) \in \Re^{mn}$ takes the general rectangular $m \times n$ matrix M and forms a vector from its columns. The inverse mapping $\operatorname{Mat} := \operatorname{vec}^{-1}$; $x = \operatorname{u2svec}(X) \in \Re^{\binom{n}{2}}$ is $\sqrt{2}$ times the vector obtained columnwise from the strictly upper triangular part of the symmetric matrix X; $\binom{n}{2} = \frac{n(n-1)}{2}$. (The multiplication by $\sqrt{2}$ makes the mapping an isometry.) Let $\operatorname{u2sMat} := \operatorname{u2svec}^\dagger$ denote the *Moore-Penrose generalized inverse* mapping into $\mathcal{S}^n$. This is an inverse mapping if we restrict to the subspace of matrices with zero diagonal. The adjoint operator $\operatorname{u2sMat}^* = \operatorname{u2svec}$, since

$$\begin{aligned} \langle \operatorname{u2sMat}(v), S \rangle &= \operatorname{trace} \operatorname{u2sMat}(v) S \\ &= v^T \operatorname{u2svec}(S) = \langle v, \operatorname{u2svec}(S) \rangle . \end{aligned}$$

Define the orthogonal projection $\operatorname{offDiag}(S) := S - \operatorname{Diag}(\operatorname{diag}(S))$, where $\operatorname{diag}(S)$ denotes the diagonal of S and $\operatorname{diag}^*(v) = \operatorname{diag}^\dagger(v) = \operatorname{Diag}(v)$ is the adjoint operator, i.e. the diagonal matrix with diagonal elements from the vector v. Then: $\operatorname{diag}\operatorname{Diag} = I$ on $\Re^n$; $\operatorname{Diag}\operatorname{diag}^* = \operatorname{Diag}\operatorname{diag}^\dagger$ is the orthogonal projection on the subspace of diagonal matrices in $\mathcal{S}^n$, the range of Diag; and

$$\operatorname{u2sMat}\operatorname{u2sMat}^* = \operatorname{u2sMat}\operatorname{u2sMat}^\dagger = \operatorname{offDiag} = \operatorname{offDiag}^*,$$

is the orthogonal projection onto the subspace of matrices with 0 diagonal, the range of u2sMat .

When we apply (2.10) to (P), we obtain

$$\text{diag}\,(X) = e \quad \text{iff} \quad X := I + \text{u2sMat}\,(x), \quad \text{for some } x \in \Re^{\binom{n}{2}}. \tag{2.16}$$

Below, we equate $X := \text{u2sMat}\,(x) + I,\ Z := \text{Diag}\,(y) - Q$. We use $s = \begin{pmatrix} x \\ y \end{pmatrix}$. The following operators are used in the optimality conditions and in the GN method.

$$\boxed{\begin{array}{rl} \mathcal{X} : \Re^n \to \Re^{\binom{n}{2}}, & \mathcal{X}(\cdot) := \text{vec}\,(\text{Diag}\,(\cdot)X); \\ \mathcal{Z} : \Re^{\binom{n}{2}} \to \Re^n, & \mathcal{Z}(\cdot) := \text{vec}\,(Z\text{u2sMat}\,(\cdot)). \end{array}} \tag{2.17}$$

We evaluate the adjoint operators. Let $A \circ B$ denote the Hadamard (element-wise) product of the matrices A, B. Let $w = \text{vec}\,(W)$.

$$\begin{aligned} \langle w, \mathcal{X}(v)\rangle &= \text{trace}\, W^T \text{Diag}\,(v)X \\ &= \text{trace}\, \text{Diag}\,(v)XW^T \\ &= v^T \text{diag}\,(X\text{Mat}\,(w)^T) \\ &= (e^T(X \circ \text{Mat}\,(w)^T))v \\ &= ((X \circ \text{Mat}\,(w))e)^T v \\ &= \langle \mathcal{X}^*(w), v\rangle\,. \end{aligned}$$

$$\begin{aligned} \langle w, \mathcal{Z}(v)\rangle &= \text{trace}\, W^T Z\text{u2sMat}\,(v) \\ &= \text{trace}\, \text{u2sMat}\,(v)\frac{1}{2}\left\{ZW + W^T Z\right\} \\ &= v^T \frac{1}{2}\text{u2svec}\,\left\{Z\text{Mat}\,(w) + \text{Mat}\,(w)^T Z\right\} \\ &= \langle \mathcal{Z}^*(w), v\rangle\,. \end{aligned}$$

Summary:

$$\boxed{\begin{array}{rcl} \mathcal{X}^*(w) &=& \text{diag}\,\left(X\text{Mat}\,(w)^T\right) = (X \circ \text{Mat}\,(w))e; \\ \mathcal{X}^*\mathcal{X}(y) &=& \text{diag}\,\left(X^2\text{Diag}\,(y)\right) = (X \circ \text{Diag}\,(y)X)e; \\ \\ \mathcal{Z}^*(w) &=& \frac{1}{2}\text{u2svec}\,\left\{Z\text{Mat}\,(w) + (Z\text{Mat}\,(w))^T\right\}; \\ \mathcal{Z}^*\mathcal{Z}(x) &=& \frac{1}{2}\text{u2svec}\,\left\{Z^2\text{u2sMat}\,(x) + \text{u2sMat}\,(x)Z^2\right\}; \\ \\ \mathcal{Z}^*\mathcal{X}(y) &=& \frac{1}{2}\text{u2svec}\,\left\{Z\text{Diag}\,(y)X + X\text{Diag}\,(y)Z\right\}; \\ \mathcal{X}^*\mathcal{Z}(x) &=& \text{diag}\,\left(X\text{u2sMat}\,(x)Z\right). \end{array}} \tag{2.18}$$

2.2.2 Duality and Optimality Conditions

Recall the primal SDP given in (2.15). To obtain optimality conditions we use a dual problem. Slater's CQ (strict feasibility) holds for (P), which implies that we have strong duality with the Lagrangian dual (e.g. [26])

$$\begin{array}{rlll} & \nu^* = & \min & e^T y \\ (D) & & \text{subject to} & \text{Diag}\,(y) - Z = Q \\ & & & Z \succeq 0. \end{array} \tag{2.19}$$

Weak duality for feasible variables can be expressed as:

$$0 \leq e^T y - \text{trace}\, QX = e^T y - \text{trace}\,(\text{Diag}\,(y) - Z)X = (e - \text{diag}\,(X))^T y + \text{trace}\, ZX = \text{trace}\, ZX.$$

Therefore, a zero duality gap is equivalent to trace $ZX = 0$. Moreover, since $X, Z \succeq 0$, this is equivalent to $ZX = 0$. Since Slater's condition is also satisfied for the dual program, we have primal attainment and get the following well-known characterization of optimality for (P). (See Theorem 2.1.)

Theorem 2.5 *The primal-dual variables X, y, Z with $X \succeq 0, Z \succeq 0$ are optimal for (P),(D) if and only if*

$$\begin{array}{rl} \text{Diag}\,(y) - Z = Q & \textit{(dual feasibility)} \\ \text{diag}\,(X) = e & \textit{(primal feasibility)} \\ ZX = 0 & \textit{(complementary slackness)} \end{array}$$

Proof. *This is a specialization of Theorem 2.1 since Slater's CQ always holds for (P).* ■

Rather than linearize the optimality conditions and then reduce them to e.g. the normal equations, we apply elimination right from the start. First, since the diagonal of X is fixed, we can use the constant $K = e^T \text{diag}\,(Q)$ in the objective function. We could also set $\text{diag}\,(Q) = 0$ so that $K = 0$. Therefore we assume without loss of generality that $\text{diag}\,(Q) = 0$. The simplicity of the primal feasibility equation yields an equivalent problem to (P) with the representation $x = \text{u2svec}\,(X),\ X = \text{u2sMat}\,(x) + I$.

Remark 2.6 *In the general SDP notation, we found $\mathcal{A}X = b$ is equivalent to $X = \mathcal{N}(x) + \hat{X}$, where $\mathcal{N}$ is a linear operator with range equal to the null space of $\mathcal{A}$ and $\hat{X}$ is a particular solution that satisfies $X = \mathcal{N}(x) + \hat{X} \succ 0$. This approach can be directly applied to general problems with constraints of the form $\mathcal{A}X \preceq b,\ X \succeq 0$. Obtaining an initial feasible starting point can be done using the self-dual embedding, e.g. [6, 8].*

Theorem 2.7 *The primal-dual variables X, y, Z, with $X = \text{u2sMat}\,(x) + I \succeq 0,\ Z = (\text{Diag}\,(y) - Q) \succeq 0$, are optimal for (P),(D) if and only if they satisfy the single bilinear optimality equation*

$$F(x, y) := (\text{Diag}\,(y) - Q)(\text{u2sMat}\,(x) + I) = 0, \quad F(x, y) : \Re^{\binom{n}{2}} \times \Re^n \to \mathcal{M}^n\,. \tag{2.20}$$

Proof. *This is a specialization of Corollary 2.2.* ■

This leads to the single perturbed optimality conditions that we use for our primal-dual method.

$$F_\mu(x,y) := (\mathrm{Diag}\,(y)-Q)(\mathrm{u2sMat}\,(x)+I)-\mu I = 0, \quad F_\mu(x,y) : \Re^{\binom{n}{2}} \times \Re^n \to \mathcal{M}^n\,. \tag{2.21}$$

When we implement GN we use $\mathrm{vec}\,(F_\mu(x,y)) = 0$. This is a nonlinear (bilinear) overdetermined system. The linearization (or GN equation) for the search direction $\Delta x, \Delta y$ is (vec is understood)

$$\begin{aligned} -F_\mu(x,y) &= F'_\mu(x,y)\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix} \\ &= \mathcal{Z}(\Delta x) + \mathcal{X}(\Delta y) \\ &= (\mathrm{Diag}\,(y) - Q)\mathrm{u2sMat}\,(\Delta x) + \mathrm{Diag}\,(\Delta y)(\mathrm{u2sMat}\,(x) + I). \end{aligned} \tag{2.22}$$

This is a linear, full rank (in the nondegenerate case), overdetermined system and we find the least squares solution. The first part $\mathcal{Z}(\Delta x)$ is the large part of the system since it maps $\Re^{\binom{n}{2}} \to \mathcal{M}^n$. However, the operator $\mathcal{Z}$ is *sparse*, i.e. the cost of the operation $\mathcal{Z}(\Delta x)$ is not order n^4. Instead, it is equivalent to the matrix product of two symmetric matrices, where the first matrix Z is sparse, i.e. it is of order kn^2, where k is the average number of nonzeros per row in Z. The second part is the small part since it only has n variables, though the matrix X is usually dense. The latter is the size of the normal equations system that arises in the standard approaches for SDP. Sparse least squares techniques can be applied. In particular, the distinct division into two sets of columns can be exploited using projection and multifrontal methods, e.g. [4, 15, 22, 23].

2.3 Preconditioning

2.3.1 Preconditioning Techniques

The Jacobian $J := F' = [\mathcal{Z} \mid \mathcal{X}]$. Suppose that we have an *approximate* factorization $J^*J \cong R^*R$. Then we replace the linear system for the search direction $\Delta s = \begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}$, $J\Delta s = -F$, with the (better conditioned) system $JR^{-1}(R\Delta s) = -F$. (We can also precondition, or scale, from the left $MJR^{-1}(R\Delta s) = -MF$, for appropriate M. This will be explored in a future study.)

2.3.2 Diagonal Column Preconditioning

We begin with the simplest of the preconditioners. This has been identified as a successful preconditioner for least squares problems, see [17, Sect. 10.5], [31], and [11, Prop. 2.1(v)]. In the latter reference, it was shown to be the optimal diagonal scaling in the sense that, for $A, m \times n, m \geq n$ a full rank matrix, the solution of the optimization problem $\min \omega((AD)^T(AD))$, over all positive diagonal matrices D, with condition number $\omega(K) = \frac{\mathrm{trace}\,(K)/n}{\det(K)^{\frac{1}{n}}}$, is given by $D_{rr} = \frac{1}{\|A_{:r}\|}$. And, ω is a measure of the condition number,

in the sense that it is bounded above and below by a constant times the standard condition number (ratio of largest to smallest singular values).

We first find the columns of the operator $F'(x,y)(\cdot,\cdot) = \mathcal{Z}(\cdot) + \mathcal{X}(\cdot)$, which we write as $F' = [\mathcal{Z} \mid \mathcal{X}]$. The columns are ordered using $c = 1, 2, \ldots$ where c represents (k,j), $1 \le k < j \le n$ for the upper triangular part of X; and then represents $k = 1, \ldots, n$ for the elements of y.

1.

$$\begin{aligned}
\mathcal{Z}(e_c) &= \mathrm{vec}\,(\mathrm{Diag}\,(y) - Q)\mathrm{u2sMat}\,(e_c) \\
&= \tfrac{1}{\sqrt{2}}\mathrm{vec}\,(\mathrm{Diag}\,(y) - Q)\left(e_k e_l^T + e_l e_k^T\right) \\
&= \tfrac{1}{\sqrt{2}}\mathrm{vec}\,\left\{\left(y_k e_k e_l^T + y_l e_l e_k^T\right) - \left(Q_{:k} e_l^T + Q_{:l} e_k^T\right)\right\}.
\end{aligned}$$

Therefore

$$\begin{aligned}
\|\mathcal{Z}(e_c)\|_F^2 &= \tfrac{1}{2}\left\{(Z^2)_{kk} + (Z^2)_{ll}\right\} \\
&= \tfrac{1}{2}\left\{\|y_k e_k - Q_{:k}\|^2 + \|y_l e_l - Q_{:l}\|^2\right\} \\
&= \tfrac{1}{2}\left\{\|Q_{:k}\|^2 + \|Q_{:l}\|^2 + y_k^2 + y_l^2\right\},
\end{aligned} \tag{2.23}$$

since $\mathrm{diag}\,(Q) = 0$ by assumption. We see that this calculation is inexpensive since one need only find the norms and sums of the columns of the sparse matrix Q once, at the beginning of the algorithm.

2.

$$\mathcal{X}(e_i) = \mathrm{vec}\,\left(\mathrm{Diag}\,(e_i)(\mathrm{u2sMat}\,(x) + I)\right).$$

Therefore

$$\|\mathcal{X}(e_i)\|_F^2 = \|X_{i,:}\|^2. \tag{2.24}$$

Therefore the diagonal of the preconditioning matrix D is found from the components defined in (2.23) and (2.24).

2.3.3 Partial (Block) Cholesky Preconditioning

We now look at finding a partial Cholesky factorization of

$$(F')^* F' = \begin{pmatrix} \mathcal{Z}^*\mathcal{Z} & \mathcal{Z}^*\mathcal{X} \\ \mathcal{X}^*\mathcal{Z} & \mathcal{X}^*\mathcal{X} \end{pmatrix}$$

by finding the Cholesky (or partial) factorizations of the two diagonal blocks.

From (2.18) and (2.24), we conclude that the bottom right block $\mathcal{X}^*\mathcal{X}$ is a positive semidefinite (positive definite if no columns of X are 0) diagonal matrix with diagonal elements $\|X_{i,:}\|^2$. Therefore, we cannot improve on the diagonal scaling derived in Section 2.3.2 by using a Cholesky factorization of this block. However, the top left block is not diagonal. We can use (2.18). As above, we use the transformation between indices

$$c \cong (k,l), \quad c = \frac{(l-1)(l-2)}{2} + k, \quad k \le c, 1 \le k < l \le n.$$

The columns of this top left block follow.

$$\begin{aligned}\mathcal{Z}^*Z(e_c) &= \tfrac{1}{2\sqrt{2}}\text{u2svec}\left\{Z^2\left(e_ke_l^T+e_le_k^T\right)+\left(e_ke_l^T+e_le_k^T\right)Z^2\right\}\\ &= \tfrac{1}{2\sqrt{2}}\text{u2svec}\begin{pmatrix}\text{in row } k & (Z^2)_{l:}\\ \text{in row } l & (Z^2)_{k:}\\ \begin{pmatrix}\text{in col } k\\ (Z^2)_{:l}\end{pmatrix} & \begin{pmatrix}\text{in col } l\\ (Z^2)_{:k}\end{pmatrix}\end{pmatrix}\end{aligned} \tag{2.25}$$

For $i\neq j$, we let $E_{ij}=\frac{1}{\sqrt{2}}\left(e_ie_j^T+e_je_i^T\right)$ denote the orthonormal basis for the symmetric matrix space. (When $i=j$, we use $e_ie_i^T$.) δ_{ij} denotes the *Kronecker delta*. Therefore, the element in row $r\cong(i,j)$ and column $c\cong(k,l)$, with $r\geq c$, of the matrix representation $M\cong\mathcal{Z}^*\mathcal{Z}$ is

$$\begin{aligned}M_{rc} &= \langle\text{u2svec}\,(E_{ij}),\mathcal{Z}^*\mathcal{Z}(\text{u2svec}\,(E_{kl}))\rangle\\ &= \tfrac{1}{2}\text{u2svec}\,(E_{ij})^T\text{u2svec}\,\{Z^2E_{kl}+E_{kl}Z^2\}\\ &= \tfrac{1}{2}\text{trace}\,(E_{ij})\{Z^2E_{kl}+E_{kl}Z^2\}\\ &= \text{trace}\,E_{ij}E_{kl}Z^2\\ &= \tfrac{1}{2}\text{trace}\,(e_ie_j^T+e_je_i^T)(e_ke_l^T+e_le_k^T)Z^2\\ &= \tfrac{1}{2}\text{trace}\left(e_ie_j^Te_ke_l^T+e_ie_j^Te_le_k^T+e_je_i^Te_ke_l^T+e_je_i^Te_le_k^T\right)Z^2\\ &= \tfrac{1}{2}\{\delta_{jk}(Z^2)_{li}+\delta_{jl}(Z^2)_{ki}+\delta_{ik}(Z^2)_{lj}+\delta_{il}(Z^2)_{kj}\}\\ &= \begin{cases}\frac{1}{2}\{(Z^2)_{ii}+(Z^2)_{jj}\}, & \text{if } r=c\cong(i,j)\\ \frac{1}{2}(Z^2)_{ik}, & \text{if } r>c, j=l\\ \frac{1}{2}(Z^2)_{jl}, & \text{if } r>c, i=k\\ \frac{1}{2}(Z^2)_{jk}, & \text{if } r>c, i=l\\ 0, & \text{otherwise.}\end{cases}\end{aligned} \tag{2.26}$$

Now let the column $c\cong(k,l)$ be fixed and look at the vector $w=(M_{rc})_{r\geq c}$, $r\cong(i,j)$. Then there are (at most) $2(n-2)+1$ nonzero elements in column c given by:

$$\begin{aligned}&r=c\cong(k,l): && M_{cc}=\tfrac{1}{2}\{(Z^2)_{kk}+(Z^2)_{ll}\}\\ &(k,l)\cong c<r\cong(i,j),\quad l\leq j: &&\\ &\quad k<i<j=l, i=k+1,\ldots,j-1: && M_{rc}=\tfrac{1}{2}(Z^2)_{ik}\\ &\quad i=k<l<j, j=l+1,\ldots,n: && M_{rc}=\tfrac{1}{2}(Z^2)_{jl}\\ &\quad k<i=l<j, j=l+1,\ldots,n: && M_{rc}=\tfrac{1}{2}(Z^2)_{kj}\end{aligned}$$

We can simplify as done for the diagonal preconditioning above, i.e.

$$\begin{aligned}M_{cc}&=\tfrac{1}{2}\{\|Q_{:k}\|^2+\|Q_{:l}\|^2+y_k^2+y_l^2\},\\ M_{rc}&=\tfrac{1}{2}(Z^2)_{st}=\tfrac{1}{2}\left(-(y_s+y_t)Q_{st}+(Q^2)_{st}\right).\end{aligned}$$

we see that we need only calculate the Q^2 once at the beginning of the algorithm during initialization. Updates to M_{rc} are done using the elements of y. Updates to $M_{rc}, c<r$, need to be done only if the corresponding $Q_{st}\neq 0$. Moreover, a reordering for sparsity in the Cholesky factorization of M need only be done once during the first iteration.

Remark 2.8 *If we first precondition on the left using the operator $\mathcal{M}(\cdot):=Z^{-1}(\cdot)$, then the large upper left block would now be diagonal and the work in the block preconditioning requires the partial Cholesky factorization of the smaller $n\times n$ lower right block, i.e. this reduces to the partial Cholesky of $Z^{-2}\circ X^2$.*

2.4 Long Steps and Crossover Criteria

Advantages of the GN approach include a zero residual and full rank of the Jacobian at each iteration, as well as at optimality *in the nondegenerate case*. Therefore, there is a local neighbourhood of quadratic convergence around each point on the central path for each $\mu \geq 0$; and, this neighbourhood is *not* restricted to $X, Z \succ 0$. Therefore, for each $\mu > 0$, we take long steps to the boundary and do not backtrack to maintain positive definiteness. Once we are in the neighbourhood for $\mu = 0$, we can safely set the centering parameter $\sigma = 0$ in the algorithm and use step lengths of one without backtracking to maintain positive definiteness.

This is in contrast to most interior-point algorithm where, at every step, care is taken to ensure that the iterates remain positive definite, i.e. do not get too close to the boundary. Gauss-Newton allows the iterates to go outside the cone, *because they will return*, as we now proceed to explain.

Standard convergence results, e.g. [10, 19] show that the GN method applied to $F(s) = 0$, $s = \begin{pmatrix} x \\ y \end{pmatrix}$, is locally q-quadratically convergent, since the Jacobian at the optimum is full column rank (one to one operator). We follow the notation in [10, Theorem 10.2.1] and discuss several constants used to determine the region of quadratic convergence. We use these constants to develop a heuristic for the crossover. (We include the theorem for completeness. Here $\text{Lip}_\gamma(D)$ denotes Lipschitz continuity with constant γ on the set D; and, $N(s_*, \epsilon)$ is a ball of radius ϵ centered at s_*.)

Theorem 2.9 *([10, Theorem 10.2.1]) Let $F : \Re^n \to \Re^m$, and let $f(s) = \frac{1}{2}F(s)^T F(s)$ be twice continuously differentiable in an open convex set $D \subset \Re^n$. Assume that the Jacobian $J(s) := F'(s) \in \text{Lip}_\gamma(D)$ with $\|J(s)\|_2 \leq \alpha$ for all $s \in D$, and that there exists $s_* \in D$ and $\lambda, \bar{\sigma} \geq 0$, such that $J(s_*)^T F(s_*) = 0$, λ is the smallest eigenvalue of $J(s_*)^T J(s_*)$, and*

$$\|(J(s) - J(s_*))^T F(s_*)\|_2 \leq \bar{\sigma}\|s - s_*\|_2$$

for all $s \in D$. If $\bar{\sigma} < \lambda$, then for any $c \in (1, \lambda/\bar{\sigma})$, there exists $\epsilon > 0$ such that for all $s_0 \in N(s_*, \epsilon)$, the sequence generated by the GN method

$$s_{k+1} = s_k - (J(s_k)^T J(s_k))^{-1} J(s_k)^T F(s_k)$$

is well defined, converges to s_*, and obeys

$$\|s_{k+1} - s_*\| \leq \frac{c\bar{\sigma}}{\lambda}\|s_{k+1} - s_*\| + \frac{c\alpha\gamma}{2\lambda}\|s_{k+1} - s_*\|^2$$

and

$$\|s_{k+1} - s_*\| \leq \frac{c\bar{\sigma} + \lambda}{2\lambda}\|s_{k+1} - s_*\| < \|s_{k+1} - s_*\|^2.$$

■

In our case, we have a zero residual. This implies that the corresponding constant $\bar{\sigma} = 0$. Since

$$\begin{aligned}
\|F'(\Delta s)\|_F &= \|Z\text{u2sMat}\,(\Delta x) + \text{Diag}\,(\Delta y)X\|_F \\
&\leq \|Z\text{u2sMat}\,(\Delta x)\|_F + \|\text{Diag}\,(\Delta y)X\|_F \\
&\leq \|Z\|_F\|\text{u2sMat}\,(\Delta x)\|_F + \|\text{Diag}\,(\Delta y)\|_F\|X\|_F \\
&= \|Z\|_F\|\Delta x\|_2 + \|\Delta y\|_2\|X\|_F \\
&\leq \sqrt{\|Z\|_F^2 + \|X\|_F^2}\|\Delta s\|_2, \text{ (by Cauchy-Schwartz inequality)}
\end{aligned}$$

a bound on the norm of the Jacobian is $\alpha = \sqrt{\|Z\|_F^2 + \|X\|_F^2}$.

$$\begin{aligned}
\|F'(s-\bar{s})(\Delta s)\|_F &= \|(Z-\bar{Z})\text{u2sMat}\,(\Delta x) + \text{Diag}\,(\Delta y)(X-\bar{X})\|_F \\
&\leq \|(Z-\bar{Z})\|_F\|\text{u2sMat}\,(\Delta x)\|_F + \|\text{Diag}\,(\Delta y)\|_F\|(X-\bar{X})\|_F \\
&= \|(y-\bar{y})\|_2\|\Delta x\|_2 + \|\Delta y\|_2\|(x-\bar{x})\|_2.
\end{aligned}$$

Therefore the Lipschitz constant $\gamma = 1$.

We assume nondegeneracy of the optimum. Thus s^* is unique and the smallest singular value satisfies $\sigma_{\min}(F'(s)) \geq \sqrt{K}$, for all s in an ϵ_1 neighbourhood of s^*, for some constant $K > 0$. Following [10, Page 223], we define

$$\epsilon := \min\left\{\epsilon_1, \frac{K}{\alpha\gamma}\right\} = \min\left\{\epsilon_1, \frac{K}{\sqrt{\|Z^*\|_F^2 + \|X^*\|_F^2}}\right\}.$$

Then q-quadratic convergence is guaranteed once the current iterate is in this ϵ neighbourhood of the optimum. Suppose that $\overline{\Delta s}$ is the search direction found using (2.22) but with $\sigma = 0$. (This can be done efficiently using lsqr, i.e. two right hand sides can be used, or within a predictor corrector framework.) Then $\overline{\Delta s}$ is our current best estimate for the distance to the optimum s^*. One possible heuristic is to start the crossover if

$$\|\overline{\Delta s}\| \leq \rho\frac{\sigma_{\min}(F'(s_k))^2}{\sqrt{\|Z_k\|_F^2 + \|X_k\|_F^2}}, \quad \rho \in (0,1).$$

Note that this bound is overly restrictive since it does not take into account the direction of the step. Moreover, it is expensive to calculate if one wants an accurate estimate of the smallest singular value. But a cheaper estimate can be obtained by estimating the condition number κ (cheap condition number estimates exist), and the norm (hence the largest singular value) of F' and then get an estimate by

$$\|\overline{\Delta s}\| \leq \rho\frac{\kappa^2(F'(s_k))}{\|(F'(s_k))^2\|\sqrt{\|Z_k\|_F^2 + \|X_k\|_F^2}}, \quad \rho \in (0,1).$$

But even this is expensive and did not seem useful in practice. In our tests we started the crossover when the relative duality gap $\frac{\text{trace}\,ZX}{|\text{trace}\,QX|+1} < .1$. This simpler heuristic never failed.

Algorithm 3.1: Primal-Dual Gauss-Newton via PCG for Max-Cut

Input: $Q \in \mathcal{S}^n$; $\delta_1 > 0$; $\delta_2 > 0$ {Objective, gap and least-squares tolerances}
$q_j = \|Q_{:,j}\|_2$; $x = 0$; $y = 2\mathrm{Diag}\,(q) - Q$; {Initial iterate}
$c_f =$ `false`; {Crossover state}
repeat
  if $c_f =$ `true` **then**
    $\sigma = 0$;
  else
    Set σ adaptively, depending on last step size.
  end if
  Construct preconditioner P according to Section 2.3
  $[\Delta x, \Delta y] = \mathrm{LSQR}(A, b, Q, P, F_{\sigma\mu}, x, y, \delta_2)$; {Least squares solve of (2.22)}
  if $c_f =$ `true` **then**
    $\alpha = 1$; {Full Gauss-Newton step}
  else
    $\alpha = .9997$ of the distance to the semidefinite cone. {Damped Gauss-Newton step}
  end if
  $x = x + \alpha\Delta x$; $y = y + \alpha\Delta y$; $\mu = (\mathrm{trace}\,(\mathrm{u2sMat}\,(x) + I)(\mathrm{Diag}\,(y) - Q))/n$;
  if $\frac{\mu}{\mathrm{trace}\,Q(\mathrm{u2sMat}\,(x)+I)+1} < .1$ **then**
    $c_f =$ `true`; {Time to crossover}
  end if
until $\{\frac{\mu}{\mathrm{trace}\,Q(\mathrm{u2sMat}\,(x)+I)+1} < \delta_1\}$

3 Primal-Dual Interior-Point Algorithm

We use equation (2.21) and the linearization (2.22) to develop the primal-dual interior-point which we now sketch. We leave vague the update of σ and α because the complexity only obscures the presentation.

This modifies the standard approach in e.g. [33]. Our approach differs in that we have eliminated, in advance, the primal and dual linear feasibility equations. We work with an overdetermined nonlinear system rather than a square symmetrized system; thus we use a GN approach, [20]. We use a crossover step, i.e. we use $\sigma = 0$, and we do not backtrack to preserve positive definiteness of Z, X once we have estimated the region of quadratic convergence. The search direction is found using a preconditioned conjugate gradient method, `lsqr` [25]. The cost of each CG iteration is a (sparse) matrix multiplication and a diagonal matrix scaling, see e.g. (2.22). Before the crossover, the step length that preserves positive definiteness is the same for both the primal and dual variables.

There are many advantages for this algorithm:

- Primal and dual feasibility is exact during each iteration (assuming that the the basis for the null space was computed precisely);
- there is no (costly, dense) normal equations system to form;
- There is no need to find Z^{-1} (which becomes ill-conditioned as μ goes to 0);
- The sparsity of the data Q is exploited completely;

- By the robustness of the algorithm, there is no need to enforce positivity of Z, X once μ gets small enough; q-quadratic convergence is obtained;
- The entire work of the algorithm lies in finding the search direction at each iteration by solving a (sparse) least squares problem using a CG type algorithm. Each iteration of the CG algorithm involves a (sparse) matrix-matrix multiplication and a diagonal row scaling of a matrix. The more efficiently we can solve these least squares problems, the faster our algorithm will be. Better preconditioners, better solvers, and better parallelization in the future will improve the algorithm;
- The techniques can be extended directly to general SDPs, depending on efficient (sparse) representations of the primal (and/or dual) feasibility equation.

4 Numerical Tests

We present here only some tests to highlight three aspects of the algorithm, the handling of sparsity, the effect of preconditioning and the effect of the crossover.

First some general remarks. The tests were done using MATLAB 6.0.0.88 (R12) on a SUNW, Ultra $5-10$ with one GB of RAM using SunOS 5.8 (denoted by SUN), as well as on a Toshiba Pentium II, 300 MHZ, with 128 MB of RAM (denoted by PII) and finally an SGI computer. Diagonal preconditioning was used. (Some preliminary test results with the partial Cholesky preconditioning are included at the end of this section.) 99% of the cputime was spent on finding the search direction, i.e. in the PCG part of the code in finding the (approximate) least squares solution of the GN equation. The cost for the early iterations was very low, e.g. $21, 50$ CG iterations, $24, 58$ cpu seconds for the first two iterations for $n = 365$ on the SUN. (This reduced to $10, 11$ iterations for the partial Cholesky preconditioner.) This cost increased as the relative duality gap (and the requested tolerance for the least squares solution) decreased. Low accuracy solutions were obtained quickly, e.g. one decimal accuracy after 4 to 5 iterations. The cost per iteration increased steadily and then leveled off near the optimum.

The first conclusion from our tests is that the crossover to a steplength of 1 has a significant effect on both the number of iterations and the conditioning of the linear system. The number of iterations are approximately halved (See Table 4.1). The best results were obtained by staying well-centered before the crossover. This is in line with what we know from the theory. This appeared to help with the conditioning of the Jacobian and lowered the number of PCG iterations. For simplicity, the crossover was done when the relative duality gap was less than .1 and we used a steplength of 1 after the crossover, rather than a line search to guarantee sufficient decrease in $\|ZX\|_F$. In all our tests on random problems or on the SDPLIB problems, we converged to a high accuracy solution.

In Tables 4.2 and 4.3 we see the effect of preconditioning and dimension on the cputime of the algorithm. The effect of the dimension is predictable and trivial but the effect of preconditioning is surprising, especially since only the simple diagonal preconditioning is used in these tests. The number of major iterations does not change much,of course, but the work required by the least-squares solver is reduced considerably.

Crossover toler. in relative gap	nmbr of major iterations	norm(ZX) at end	min. eig. pos. violation in Z, X
1.0e-1	11	3.6281e-12	-4.6478e-12
1.0e-3	14	1.9319e-13	-2.1580e-13
1.0e-5	16	2.0917e-11	-2.2839e-11
1.0e-7	18	1.8293e-13	-1.5546e-13
1.0e-9	20	7.2534e-10	6.7006e-12
1.0e-11	20	7.2544e-10	6.7004e-12

Table 4.1: Size of Q, $n = 25$; Requested Accuracy in relative duality gap: 1.0e-10; Density is $\frac{1}{25}$; Optimal Value is 5.7041e+01.

To consider the effect of sparsity, Figure 4.1 illustrates the number of nonzeros in Q versus the increase in total cputime (when dimension runs from 15 to 105 with increments of 10).

Dimension of Q n	total cpu seconds	nmbr of major iterations	norm(ZX) at end	violation of eig. pos. in Z, X
15	7.75	9	1.7535e-014	-3.0863e-015
25	17.36	10	1.9291e-013	-1.5217e-013
35	30.43	10	1.1044e-012	-8.4977e-013
45	59.87	11	2.7926e-011	-2.763e-011
55	86.89	12	4.0912e-011	-3.7568e-011
65	131.11	11	4.9924e-012	-5.0519e-012
75	275.89	12	1.0448e-008	-1.4504e-008
85	468.46	14	8.6208e-011	-3.4404e-010
95	414.03	11	7.4253e-011	-3.0935e-010
105	878.64	15	7.5993e-010	-1.4946e-008

Table 4.2: Requested Accuracy in relative duality gap: 1.0e-10; Density is $\frac{3}{n}$; crossover tolerance 1.0e-01; optimal values of order 150; PII computer.

We have applied the algorithm to all the problems in SDPLIB. The results are shown in Table 4. The reader will keep in mind that the implementation is entirely in MATLAB and yet sizeable problems are solved. An equivalent implementation in C or FORTRAN would increase the attainable sizes by an order of magnitude.

Finally, we discuss some preliminary tests using the partial (block) Cholesky preconditioner discussed in Section 2.3.3. We used a SUN-FIRE-280R with 2 GIG RAM and with MATLAB 6.5, Release 13. We solved problems with $n = 400, 500, 600$. The number of major iterations and LSQR iterations were similar. For example, with $n = 600$ and 1,192 nonzeros in Q, (We note that the linear system of equations that we are solving for the search direction has 360,000 rows and 180,300 columns, or 3.250809×10^{10} entries in the matrix representation.) The early iterations took (approx.) 20 cpu seconds to find the preconditioner which contained (approx.) 2.5 million nonzeros. It took (approx.) 15 cpu seconds to find the search direction using LSQR (approx. 15 LSQR iterations). Later iterations increased

Dimension of Q n	total cpu seconds	nmbr of major iterations	norm(ZX) at end	violation of eig. pos. in Z, X
15	30.1	11	1.9039e-011	-3.5098e-012
25	134.13	15	1.9742e-009	-2.7465e-010
35	117.27	12	2.5534e-010	-8.8532e-011
45	174.34	11	1.3799e-008	-1.9154e-008
55	373.6	14	7.3741e-009	-3.1854e-009
65	1388.1	19	9.6109e-009	-2.7634e-009
75	507.46	13	3.5887e-009	-3.3886e-009
85	1856.1	16	6.4806e-011	-4.7964e-011

Table 4.3: **Without preconditioning**; Requested Accuracy in relative duality gap: 1.0e-10; Density is $\frac{3}{n}$; crossover tolerance 1.0e-01; optimal values of order 150; PII computer.

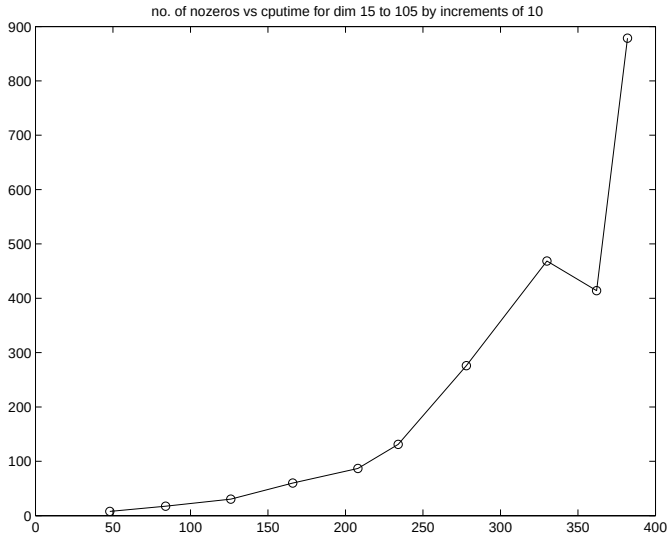

Figure 4.1: Nonzeros vs cputime. $n = 15 : 10 : 105$.

in cost to (approx.) double the time for finding the preconditioner and 10 times the time for LSQR to find the search direction. The increase in cost for the preconditioner appears to be due to a doubling in the number of nonzeros in the Cholesky factor. This and the increase in accuracy increases the cost for LSQR. The total number of major iterations were in line with those for the diagonal preconditioner.

Dim Q n	cpu sec	major iter.	rel. opt. at end	rel. norm(ZX) at end	rel. viol. eig. pos. Z, X	nnz in Q
100	96.34	12	226.16	1.2399e-13	-1.2062e-12	638
124	252.57	13	141.99	1.2238e-14	-2.3655e-13	410
124	171.99	13	269.88	2.0918e-11	-1.1324e-11	760
124	292.48	13	467.75	7.0013e-13	-5.9152e-13	1364
124	289.21	13	864.41	5.7229e-11	-1.035e-11	2666
250	1134.9	14	317.26	2.9025e-11	-5.5891e-11	892
250	1058.3	14	531.93	2.8552e-16	-5.5638e-16	1472
250	1633.6	12	981.17	8.9829e-12	-5.8956e-12	2816
250	3036	14	1682	3.481e-11	-2.1537e-11	5092
500	14669	19	598.15	4.12e-13	-2.9424e-12	1701
500	21489	17	1070.1	1.0229e-11	-2.7013e-11	2939
500	20691	15	1848	3.6924e-11	-1.364e-11	5210
500	46752	16	3566.7	3.3634e-11	-1.5e-11	10740

Table 4.4: Requested Accuracy in relative duality gap: 1.0e-10; SDPLIB test problems; crossover tolerance 1.0e-01; SGI (Irix6.5_64-Mips) computer.

5 Conclusion

We have presented an alternative approach to solving SDPs and applied it to the SDP relaxation of the Max-Cut problem. This approach avoids both the ill-conditioning that arises from symmetrization of the optimality conditions as well as that which arises from the formation of the normal equations. The approach is based on the strong/robust primal-dual path-following interior-point framework. But, it can still exploit sparsity in the data. The method uses basic tools that are successful for solving an overdetermined system of nonlinear equations with zero residual, i.e. matrix free PCG applied to the GN method. We have shown how to derive preconditioners for this approach and, in particular, derived the optimal diagonal column scaling and block Cholesky preconditioners. The total cost of an iteration lies in the solution of a linear least squares problem. This least squares problem is solved using the (preconditioned) conjugate gradient type method of Paige and Saunders [25]. The cost of each CG iteration is a sparse matrix multiplication (essentially $Z\Delta X$) and a matrix scaling (essentially Diag $(\Delta y)X$).

Numerical tests are ongoing as are extensions to more general SDPs. This includes a comparison between using first order methods and an inexact GN framework, i.e. solving the least squares problem to a desired accuracy so that the work is of the same order as one iteration of first order methods.

Ongoing research involves: further testing on the SDPLIB problem set; developing improved block diagonal and partial Cholesky preconditioners including preconditioning from the left; parallelization of the PCG approach; purification techniques to recover positive semi-definiteness; and proving convergence for the crossover technique.

Acknowledgments
The author is indebted to Michael Saunders (Department of Management Science and Engineering, Stanford University) for providing the *LSQR* MATLAB code for the PCG-like method. The author would also like to thank Serge Kruk (Department of Mathematics and Statistics Oakland University) and Levent Tunçel (Department of Combinatorics and Optimization, University of Waterloo) for many helpful discussions and comments.

References

[1] M.F. ANJOS, N. HIGHAM, M. TAKOUDA, and H. WOLKOWICZ. A semidefinite programming approach for the closest correlation matrix problem. Technical Report CORR 2001-in progress, University of Waterloo, Waterloo, Ontario, 2001.

[2] M.F. ANJOS and H. WOLKOWICZ. A strengthened SDP relaxation via a second lifting for the Max-Cut problem. *Discrete Appl. Math.*, 119:79–106, 2002.

[3] S. J. BENSON, Y. YE, and X. ZHANG. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM J. Optim.*, 10(2):443–461 (electronic), 2000.

[4] Å. BJÖRCK. Methods for sparse least squares problems. In J.R. Bunch and D. J. Rose, editors, *Sparse Matrix Computations*, pages 177–199. Academic Press, New York, 1976.

[5] S. BURER and R.D.C. MONTEIRO. A projected gradient algorithm for solving the maxcut SDP relaxation. *Optimization Methods and Software*, 15(3-4):175–200, 2001.

[6] E. de KLERK. *Interior point methods for semidefinite programming.* PhD thesis, Delft University, 1997.

[7] E. de KLERK, J. PENG, C. ROOS, and T. TERLAKY. A scaled Gauss-Newton primal-dual search direction for semidefinite optimization. *SIAM J. Optim.*, 11(4):870–888 (electronic), 2001.

[8] E. de KLERK, C. ROOS, and T. TERLAKY. Initialization in semidefinite programming via a self-dual skew-symmetric embedding. *Oper. Res. Lett.*, 20(5):213–221, 1997.

[9] R.S DEMBO, S.C. EISENSTAT, and T. STEIHAUG. Inexact Newton methods. *SIAM J. Numer. Anal.*, 19:400–408, 1982.

[10] J.E. DENNIS Jr. and R.B. SCHNABEL. *Numerical methods for unconstrained optimization and nonlinear equations*, volume 16 of *Classics in Applied Mathematics.* Society

for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1996. Corrected reprint of the 1983 original.

[11] J.E. DENNIS Jr. and H. WOLKOWICZ. Sizing and least-change secant methods. *SIAM J. Numer. Anal.*, 30(5):1291–1314, 1993.

[12] M.M. DEZA and M. LAURENT. *Geometry of cuts and metrics*. Springer-Verlag, Berlin, 1997.

[13] K. FUJISAWA, M. KOJIMA, and K. NAKATA. Exploiting sparsity in primal-dual interior-point methods for semidefinite programming. *Math. Programming*, 79:235–253, 1997.

[14] M.X. GOEMANS and F. RENDL. Combinatorial optimization. In H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors, *HANDBOOK OF SEMIDEFINITE PROGRAMMING: Theory, Algorithms, and Applications*. Kluwer Academic Publishers, Boston, MA, 2000.

[15] G. H. GOLUB and V. PEREYRA. The differentiation of pseudoinverses and nonlinear least squares problems whose variables separate. *SIAM J. Numer. Anal.*, 10:413–432, 1973.

[16] M. GONZALEZ-LIMA, H. WEI, and H. WOLKOWICZ. A simple iterative method for linear programming. Technical Report CORR 2001-in progress, University of Waterloo, Waterloo, Ontario, 2001.

[17] A. GREENBAUM. *Iterative methods for solving linear systems*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[18] C. HELMBERG and F. RENDL. A spectral bundle method for semidefinite programming. *SIAM J. Optim.*, 10(3):673 – 696, 2000.

[19] C. T. KELLEY. *Iterative methods for linear and nonlinear equations*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995. With separately available software.

[20] S. KRUK, M. MURAMATSU, F. RENDL, R.J. VANDERBEI, and H. WOLKOWICZ. The Gauss-Newton direction in linear and semidefinite programming. *Optimization Methods and Software*, 15(1):1–27, 2001.

[21] S. KRUK and H. WOLKOWICZ. Convergence of an infeasible short-step path-following algorithm based on the Gauss-Newton direction. *J. Appl. Math.*, to appear(CORR 2000-23), 2003.

[22] P. MATSTOMS. *The Multifrontal Solution of Sparse Linear Least Squares Problems*. Licentiat thesis, Department of Mathematics, Linköping University, Sweden, 1991.

[23] P. MATSTOMS. Sparse QR factorization in MATLAB. *ACM Trans. Math. Software*, 20:136–159, 1994.

[24] K. NAKATA, K. FUJISAWA, M. FUKUDA, M. KOJIMA, and K. MUROTA. Exploiting sparsity in semidefinite programming via matrix completion II: implementation and numerical results. Technical Report B-368, Dept. of Information Sciences, Tokyo Institute of Technology, Tokyo, Japan, 2001.

[25] C.C. PAIGE and M.A. SAUNDERS. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8(1):43–71, 1982.

[26] M.V. RAMANA, L. TUNÇEL, and H. WOLKOWICZ. Strong duality for semidefinite programming. *SIAM J. Optim.*, 7(3):641–662, 1997.

[27] A. SHAPIRO. Duality and optimality conditions. In *HANDBOOK OF SEMIDEFINITE PROGRAMMING: Theory, Algorithms, and Applications.* Kluwer Academic Publishers, Boston, MA, 2000.

[28] R. SOTIROV and H. WOLKOWICZ. The simple method for the SDP relaxation of the QAP. Technical Report in progress, University of Waterloo, Waterloo, Canada, 2001.

[29] J.F. STURM. Avoiding numerical cancellation in the interior point method for solving semidefinite programs. Technical Report 2001-27, Tilburg University, The Netherlands, 2001.

[30] M.J. TODD. A study of search directions in primal-dual interior-point methods for semidefinite programming. *Optim. Methods Softw.*, 11&12:1–46, 1999.

[31] A. VAN der SLUIS. Condition numbers and equilibration of matrices. *Numer. Math.*, 14:14–23, 1969/1970.

[32] R.J. VANDERBEI and H.Y. BENSON. On formulating semidefinite programming problems as smooth convex nonlinear optimization problems. Technical report, Statistics and Operations Research, Princeton University, Princeton, NJ, 1999.

[33] S. WRIGHT. *Primal-Dual Interior-Point Methods.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pa, 1996.