Solving some large scale semidefinite programs via the conjugate residual method

Kim-Chuan Toh * and Masakazu Kojima †

Abstract

Most current implementations of interior-point methods for semidefinite programming use a direct method to solve the Schur complement equation (SCE) $M\Delta y = h$ in computing the search direction. When the number of constraints is large, the problem of having insufficient memory to store M can be avoided if an iterative method is used instead. Numerical experiments have shown that the conjugate residual (CR) method typically takes a huge number of steps to generate a high accuracy solution. On the other hand, it is difficult to incorporate traditional preconditioners into the SCE, except for block diagonal preconditioners. We decompose the SCE into a 2×2 block system by decomposing Δy (similarly for h) into two orthogonal components with one lying in a certain subspace that is determined from the structure of M. Numerical experiments on semidefinite programming problems arising from Lovász θ-function of graphs and MAXCUT problems show that high accuracy solutions can be obtained with moderate number of CR steps using the proposed equation.

Key words: Large scale semidefinite programming, interior-point methods, inexact search directions, preconditioned conjugate residual method, deflated conjugate gradient method

AMS Subject classification: 90C05.

1 Introduction

Let $\mathcal{S}^n$ be the vector space of $n \times n$ real symmetric matrices endowed with the inner product $A \bullet B = \mathrm{Trace}(AB)$. Given an integer n, we let $\bar{n} = n(n+1)/2$. Let **svec**

*Department of Mathematics, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119620, Singapore (`mattohkc@math.nus.edu.sg`). This research is supported in part by National University of Singapore Academic Research Grant RP972685.

†Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, 2-12-1 Oh-Okayama, Meguro-ku, Tokyo 152, Japan (`kojima@is.titech.ac.jp`).

be an isometry identifying $\mathcal{S}^n$ with $I\!R^{\bar{n}}$ such that $K \bullet L = \mathbf{svec}(K)^T \mathbf{svec}(L)$ and let **smat** be the inverse of **svec**. Given $k \times l$ matrices G, H, we define $G \circledast H : \mathcal{S}^k \longrightarrow \mathcal{S}^l$ by $G \circledast H(M) = (HMG^T + GMH^T)/2$, for $M \in \mathcal{S}^l$.

Consider the standard semidefinite program (SDP)

$$\begin{array}{rrcl} \min_X & C \bullet X & & \\ & A_k \bullet X & = & b_k, \quad k = 1, \ldots, m \\ & X & \succeq & 0, \end{array} \tag{1}$$

where $b \in I\!R^m$, $A_k, C \in \mathcal{S}^n$, and $X \succeq 0$ means that X is positive semidefinite. The dual of (1) is

$$\begin{array}{ll} \max_{y,Z} & b^T y \\ & \sum_{k=1}^m y_k A_k + Z \;=\; C \\ & Z \succeq 0. \end{array} \tag{2}$$

For later discussion, let us introduce the linear map $\mathcal{A} : \mathcal{S}^n \longrightarrow I\!R^m$, defined by

$$\mathcal{A}(X) \;=\; [A_1 \bullet X \quad \cdots \quad A_m \bullet X]^T .$$

The adjoint of $\mathcal{A}$ with respect to the standard inner product in $\mathcal{S}^n$ and $I\!R^m$ is the linear map $\mathcal{A}^T : I\!R^m \longrightarrow \mathcal{S}^n$ defined by

$$\mathcal{A}^T y \;=\; \sum_{k=1}^m y_k A_k .$$

We consider primal-dual path-following methods [21, 23] for SDP using the Nesterov-Todd direction in which the general framework in each iteration is as follows. Given a current iterate (X, y, Z) and a centering parameter $\sigma \in [0, 1)$, where X, Z are symmetric positive definite, the methods find a search direction $(\Delta X, \Delta y, \Delta Z)$ so as to generate the next iterate by solving the following linear system of equations:

$$\mathcal{A}\Delta X \;=\; R_p \;:=\; b - \mathcal{A}X \tag{3a}$$

$$\mathcal{A}^T \Delta y + \Delta Z \;=\; R_d \;:=\; C - Z - \mathcal{A}^T y \tag{3b}$$

$$\mathcal{E}\Delta X + \mathcal{F}\Delta Z \;=\; R_c \;:=\; \sigma\mu I - \Sigma^2, \tag{3c}$$

where

$$\mu = X \bullet Z/n, \qquad \mathcal{E} = G^{-T} \circledast GZ, \qquad \mathcal{F} = G^{-T}X \circledast G,$$

and G is a matrix such that $\Sigma := GZG^T = G^{-T}XG^{-1}$ is a positive definite diagonal matrix. Note that $W := G^T G$ is the Nesterov-Todd scaling matrix with $WZW = X$; see [21].

As in the case of linear programming, we can avoid solving the linear system of $n^2 + m$ equations (3a)–(3c) directly by first solving a Schur complement equation

(SCE) involving only Δy and then computing $\Delta X, dZ$ in terms of Δy as follows:

$$M\Delta y \;=\; h \;:=\; R_p + \mathcal{A}\mathcal{E}^{-1}\mathcal{F}R_d - \mathcal{A}\mathcal{E}^{-1}R_c \tag{4a}$$

$$\Delta Z \;=\; R_d - \mathcal{A}^T\Delta y \tag{4b}$$

$$\Delta X \;=\; \mathcal{E}^{-1}R_c - \mathcal{E}^{-1}\mathcal{F}(R_d - \mathcal{A}^T\Delta y), \tag{4c}$$

where $M = \mathcal{A}\mathcal{E}^{-1}\mathcal{F}\mathcal{A}^T$ is the Schur complement matrix whose (i,j) element is given by

$$M_{ij} \;=\; A_i \bullet WA_jW, \quad i,j = 1,\ldots,m. \tag{4d}$$

Generally, (4a) is solved by a direct method through the following steps:

(i) Compute the $m \times m$ matrix M and store it in the computer memory;

(ii) Compute the Cholesky factorization of M and Δy is obtained by solving two triangular systems of linear equations.

The work required for step (i) is $2mn^3 + m^2n^2/2$ flops if the SDP data is dense. (A flop is one addition or one multiplication.) But when the data is sparse, substantial reduction in the computational cost of M is possible by exploiting sparsity; see [7] for the details. However, M is generally fully dense even when the SDP data is sparse. Thus when m is large, say more than a few thousands, it is impossible to store M in the memory of most current workstations. Furthermore, the $m^3/3$ flops required to compute the Cholesky factorization of M also becomes prohibitively expensive.

In this paper, we will mainly focus on SDPs where m is large, but n is moderate (say, less than 1000). It is well known that one can overcome the memory problem just mentioned by using an iterative method to solve (4a). The reason is that an iterative method only needs to access matrix-vector products of the form Mv, and these can be easily computed based on the operator description of M for any given vector v. As a result, M need not be formed explicitly and hence the problem of having insufficient computer memory to store M does not arise in this case.

Of course, memory problem can also occur when n is large, since the primal variable X is typically dense even if the SDP data and the resulting dual variable Z are sparse. However, the root cause of this problem lies in the primal-dual framework used to solve the SDP and it cannot be easily overcome by simply using an iterative method to compute the search direction. For such a problem, it is more appropriate to use methods that avoid the need to form X explicitly. One such method is the dual scaling interior-point method proposed in [3], where the method is able to solve large SDPs (with $m, n \approx 2000$) arising from MAXCUT problems. Another method that avoids the explicit formation of X is the spectral bundle method proposed in [10]. (This is a first order method that is designed for SDPs where $\mathrm{Trace}(X)$ is a constant. The authors succeeded in solving some large scale SDPs arising from MAXCUT problems and Lovász θ-function on graphs, but only with low accuracies in the duality gaps.)

The idea of using an iterative method to solve (4a) so as to avoid excessive memory requirement is well known. In [16], a preconditioned conjugate gradient method was proposed to solve the SCE approximately in each iteration of a primal-dual interior-point method. Besides reporting some impressive computational results on SDPs arising from Lovász θ-function on graphs and graph partitioning problems, that paper also gave a detailed discussion on how to incorporate inexact search directions into a primal-dual interior-point method. Continuing on the earlier work, a preconditioned conjugate residual method was proposed to solve the SCE in [17]. Recently, Choi and Ye [5] also reported computational results for large SDPs arising from MAXCUT problems (with n up to 14000). They used the dual-scaling interior-point method described in [3], but solve the associated SCE in each iteration by a preconditioned conjugate gradient method. Earlier works on using preconditioned conjugate gradient methods to solve the SCE in interior-point methods for SDP include [14] and [24].

As far as we are aware, all the earlier works mentioned above use diagonal or block diagonal preconditioners. These preconditioners are ineffective when the Schur complement matrix becomes more and more ill-conditioned as the interior-point iterates approach an optimal solution. Though attempts had been made in [14] to construct more effective preconditioners based on incomplete Cholesky factors, none has really succeeded in overcoming the ill-conditioning problem of the SCE. As a result, in all these works, only low accuracies in the duality gaps can be obtained at reasonable costs.

In this paper, we propose a method to overcome the ill-conditioning problem of the Schur complement matrix as interior iterates approach an optimal solution. By analyzing the structure of the Schur complement matrix, we decompose the SCE into a 2×2 block system by decomposing Δy (similarly for h) into two orthogonal components with one lying in a certain subspace of $I\!R^m$ that is determined from the structure of M. We call resulting 2×2 block system the projected SCE. Using the combination of applying the CR method to the SCE when interior-point iterates are not close to an optimal solution and switching to the projected SCE when they are, we are able to solve some large SDPs arising from Lovász θ-function of graphs and MAXCUT problems to moderately high accuracies, but at reasonable costs.

Although we focus only on the NT direction in this paper, we should mention that the method presented here also works for the dual scaling direction considered in [3], where $\mathcal{E} = I \circledast I$, $\mathcal{F} = Z^{-1} \circledast Z^{-1}/\mu$. In fact, if the dual scaling method is used, our idea can in principle be used to solve sparse SDP where m and n are both large. On the negative side, it seems that our proposed method cannot be adapted for the HRVW/KSH/M direction [11, 12, 15].

Now we introduce some notations. The MATLAB notation $[x\,;\,y]$ is used to denote the column vector formed by appending a column vector y to x. We let $\mathcal{I}$ be the set of indices of nonzero elements of the matrix $\sum_{k=1}^m |A_k|$ (where $|A_k|$ is the matrix whose (i,j) element is the magnitude of the corresponding element of A_k), and

$$\begin{aligned}\rho_s &= (\text{number of nonzero elements of the matrix } \textstyle\sum_{k=1}^m |A_k|)/n^2,\\ \rho_t &= (\text{total number of nonzero elements of } A_1, A_2, \ldots, A_m)/(mn^2).\end{aligned}$$

We use $\|\cdot\|$ to denote the vector and matrix 2-norms, and $\|\cdot\|_F$ to denote the Frobenius norm.

An iterative method can only solve (3a)–(3c) approximately. In section 2, we discuss how to incorporate an approximately computed search direction (or inexact search direction) into a primal-dual interior-point method. In section 3, some basic results on the convergence of the CR method are given. The behavior of the CR method on SDPs arising from Lovász θ-function of graphs are given in section 4. The derivation of the projected Schur complement equation and discussions on related issues are given in section 5. This is followed by numerical results showing the effectiveness of the projected Schur complement approach on two classes of SDPs, namely, those arising from Lovász θ-function of graphs, and MAXCUT problems.

2 Inexact search directions

The use of iterative methods to solve the SCE (4a) requires less computer memory compared to using a direct method. It also has the added advantage that one can terminate the iterative solver whenever an approximate solution of (4a) is deemed sufficiently accurate. This can lead to a significant saving in the CPU time required in each interior-point iteration, especially during the initial phase where accurate computation of the search direction is not necessary. In [13], Kojima, Shida, and Shindoh proposed inexact search directions where (3a) and (3b) are satisfied exactly but (3c) is relaxed. Given a fixed parameter $\kappa \in [0, 1)$, we said that $(\Delta X, \Delta y, \Delta Z)$ is an admissible direction if (3a) and (3b) hold, and

$$\|(\mathcal{E}\Delta X + \mathcal{F}\Delta Z) - R_c\|_F \;\leq\; \kappa \, \|R_c\|_F. \tag{5}$$

Note that the Frobenius norm is used above out of convenience since it can usually be computed cheaply. However, the norms in (5) need not be computed exactly, and it is sufficient to have some reasonably accurate estimates. As a result, we should keep in mind that sometimes it is cheaper to use the 2-norm instead since it can be estimated via the Lanczos method [18, chapter 4] by calculating the dominant eigenvalue.

Under certain mild assumptions, it is shown in [13] that primal-dual interior-point methods using inexact admissible directions maintain the same polynomial complexity enjoyed by their counterparts that use exact search directions. In this paper, we follow the framework laid out in [13] for the computation of inexact directions. Suppose Δy only satisfies (4a) approximately, i.e.,

$$M\Delta y \;\approx\; h.$$

Let

$$r = h - M\Delta y.$$

Given such a Δy, we compute ΔZ and $\widehat{\Delta X}$ as in (4b) and (4c), respectively. Thus $(\widehat{\Delta X}, \Delta y, \Delta Z)$ satisfies (3b) and (3c) exactly, but not (3a), since

$$\mathcal{A}\widehat{\Delta X} \;=\; \mathcal{A}\mathcal{E}^{-1}R_c - \mathcal{A}\mathcal{E}^{-1}\mathcal{F}\left(R_d - \mathcal{A}^T\Delta y\right) \;=\; R_p - r.$$

However, (3a) can be made to hold exactly by replacing $\widehat{\Delta X}$ by the minimizer of the following linear least squares problem:

$$\begin{aligned} \min_{\Delta X} \|\Delta X - \widehat{\Delta X}\|_F \\ \text{s.t.} \quad \mathcal{A}\Delta X \;=\; R_p, \end{aligned}$$

whose solution is given by

$$\Delta X \;=\; \widehat{\Delta X} \;+\; \mathcal{A}^T(\mathcal{A}\mathcal{A}^T)^{-1}\, r. \tag{6}$$

By using $(\Delta X, \Delta y, \Delta Z)$ as the search direction, (3a) and (3b) are satisfied exactly, but not (3c), where

$$\|(\mathcal{E}\Delta X + \mathcal{F}\Delta Z) - R_c\|_F \;\;=\;\; \|\mathcal{E}[\mathcal{A}^T(\mathcal{A}\mathcal{A}^T)^{-1}\, r]\|_F \;=:\; \|\mathcal{R}\|_F. \tag{7}$$

Thus $(\Delta X, \Delta y, \Delta Z)$ is an admissible direction as soon as $\|\mathcal{R}\|_F \le \kappa \|R_c\|_F$.

Note that in (7), the quantity $\|\mathcal{R}\|_F$ can be computed in $3n^3$ flops, since for a given $U \in \mathcal{S}^n$,

$$\mathcal{E}(U) \;=\; \Big[(G^{-T}UG^{-1})\Sigma + \Sigma(G^{-T}UG^{-1})\Big]/2,$$

can be computed in $3n^3$ flops, to leading order.

3 The conjugate residual method

The conjugate residual (CR) method [18] is a Krylov subspace method that is analogous to the conjugate gradient (CG) method, except that the former minimizes the norm of the residual vector over the underlying Krylov subspace in each iteration while the latter orthogonalizes the residual vector against the underlying Krylov subspace. The advantage of using the CR method is that the residual norm decreases monotonically as the iteration progresses. In general, the CR and CG methods exhibit similar convergence behavior.

Given an $N \times N$ symmetric linear system of equations $Bx = d$. The standard implementation of the CR method is as follows [18].

CR method.

Compute $r_0 = d - Bx_0$, and Br_0. Set $p_0 = r_0$, and $Bp_0 = Br_0$.

For $j = 0, 1, 2, \ldots,$

$$\alpha_j \;=\; r_j^T B r_j / \|Bp_j\|^2$$

$$x_{j+1} = \; x_j + \alpha_j\, p_j$$

$$r_{j+1} = \; r_j - \alpha_j\, Bp_j$$

Compute Br_{j+1}

$$\beta_j \;=\; r_{j+1}^T B r_{j+1} / r_j^T B r_j$$

$$p_{j+1} = \; r_{j+1} + \beta_j\, p_j$$

Compute $Bp_{j+1} = Br_{j+1} + \beta_j \, Bp_j$

end

Each iteration of the algorithm requires $12N$ flops, plus the cost of computing Br_{j+1}.

It is well known that the CR method will suffer from slow convergence when the coefficient matrix B has an unfavorable eigenvalues distribution. For readers who are not familiar with the convergence theory of the CR method, we will now briefly review how its convergence is related to the spectrum for the case where B is positive definite. Suppose the eigenvalues of B are distributed almost continuously in an interval $[a, b]$ on the positive real line. Then, in exact arithmetic, the asymptotic convergence rate of the CR method is determined by the ratio $\tau := b/a$, specifically, the convergence rate is given by

$$\rho = \frac{\sqrt{\tau} - 1}{\sqrt{\tau} + 1}. \tag{8}$$

If in addition, B has t isolated eigenvalues lying on the right of $[a, b]$, then these outliers may cause a stagnation (that is, very little reduction in the residual norm) of at most t steps during the initial phase of the CR method, but they do not affect the asymptotic convergence rate. Similarly, if B has isolated eigenvalues that lie on the left of $[a, b]$, then these outliers may cause a stagnation during the initial phase of the CR method, but the stagnation can last for many steps, depending on how close the outliers are to the origin. However, we should note that stagnation due to outliers will not occur if the initial residual vector r_0 is orthogonal to the eigenspace associated with these isolated eigenvalues. For a more detailed account of the convergence behavior of the CR method and other Krylov subspace iterative methods, we refer the reader to [6].

From the above discussion, we see that when the eigenvalues of B are distributed over a large interval, or when there are eigenvalues close to the origin, the convergence of the CR method can be exceedingly slow. In such a situation, it is necessary to apply a preconditioner to B to improve the convergence rate. That is, instead of applying the CR method to the original linear system of equations, one applies it to a transformed linear system, say, $(L^{-1}BL^{-T})\,(L^T x) = L^{-1}d$, where L is a matrix chosen such that $Lz = f$ is easy to solve, and the spectrum of the preconditioned matrix $L^{-1}BL^{-T}$ is better behaved than that of the original matrix B in either having fewer outliers that are close to the origin or having eigenvalues that are distributed over a smaller interval.

So far, our discussion on the convergence behavior of the CR method is based on exact arithmetic. In the presence of rounding errors, there are further complications in the behavior of the CR method. Roughly speaking, there are two main effects caused by rounding errors.

1. The residual norm will stop decreasing beyond a certain accuracy level (known as the final attainable level of accuracy) even if one continues the CR iteration. This is caused by the fact that the residual vectors r_j generated by the CR method differ from the true residual vectors $d - Bx_j$ in the presence of rounding errors. In [9], it is shown that the CR method will stop making further progress when j is such that

$$\|d - Bx_j - r_j\| \;\approx\; u\,\|B\|\,\|x\|\,\mathcal{O}(j)\;\left(1 + \max\{\|x_i\|/\|x\| : i \le j\}\right).$$

Here u denotes the machine precision.

2. It deteriorates the convergence rate by causing a loss of orthogonality among the computed vectors (in exact arithmetic, they are orthogonal). For example, in exact arithmetic, the CR method will converge to the exact solution in no more than N steps. But due to rounding errors, it may take more than N steps to reduce the residual norm to the final attainable level of accuracy, and such a delay can be arbitrarily long; see [20].

4 Behavior of the CR method on the Schur complement equation

In this section, we will present some numerical experiments showing the convergence behavior of the CR method in solving the following preconditioned version of (4a):

$$\underbrace{L^{-1}ML^{-T}}_{\hat{M}} \quad (L^T \Delta y) \quad = \quad \underbrace{L^{-1}h}_{\hat{h}}, \tag{9}$$

where L is a non-singular lower triangular matrix chosen to precondition M. Our immediate task is to construct a suitable preconditioner for M. In the current literature, most preconditioning techniques are proposed for a sparse linear system of equations where its sparse coefficient matrix is known explicitly, and preconditioners such as incomplete Cholesky factors are generally quite effective [18]. However, as the reader may have recalled, our matrix M is dense and is not formed explicitly. Thus, a lot of the current preconditioning techniques [18, Chapter 10] are not applicable to our linear system. The only obvious and easily implementable choices for our system are block diagonal preconditioners. That is, L is chosen to be the Cholesky factor of a matrix of the form $\mathrm{diag}(M_1, M_2, \ldots, M_k)$, where the M_i's are diagonal blocks of M. Note that if each block is just a scalar, then the diagonal elements of L are simply the square roots of the diagonal elements of M.

In [14], Lin and Saigal used incomplete Cholesky factors as preconditioners for the SCE arising from SDP relaxation of quadratic assignment problems. But their numerical results showed that these sophisticated preconditioners are not clearly better than the diagonal preconditioner.

We will first analyze the behavior of the CR method on the preconditioned SCE (9) associated with the SDP problem `theta2` taken from the SDPLIB collection [2]. For this problem, $n = 50$, $m = 498$, and we take LL^T to be the diagonal preconditioner of M. In each CR iteration, we compute $\hat{M}v$ for a given $v \in \mathbb{R}^m$ via the procedure described in Table 1, where the cost is also estimated.

All the numerical results presented in this paper are computed using MATLAB on a 400MHz Pentium II PC with 256M of memory. The parameter κ in (5) is set to $\kappa = 0.01$. The interior-point method we used is the primal-dual path-following method (without corrector) described in [22]. For easy reference, we will call the interior-point method in [22] as Algorithm PFchol, and the corresponding method that uses the CR

Computing	Number of flops required
$U := \mathcal{A}^T(L^{-T}v)$	$\rho_t\, mn^2$
$V := U\,W$	$2\rho_s\, n^3$
$\{T_{ij} \mid (i,j) \in \mathcal{I}\}$, where $T := W\,V$	$\rho_s\, n^3$
$u := \mathcal{A}(T)$	$\rho_t\, mn^2$
$\hat{M}v = L^{-1}u$	$3\rho_s\, n^3 + 2\rho_t\, mn^2$

Table 1: *computational cost of a matrix-vector product for (9).*

method to solve preconditioned SCE (9) as Algorithm PFCR. The default starting iterates described in [22] are used throughout.

Let

$$NCR(k) \;=\; \left\{ \begin{array}{l} \text{the number of CR steps required at the } k\text{th interior-point} \\ \text{iteration to solve (9) so that the admissible condition (5) is} \\ \text{satisfied.} \end{array} \right.$$

In Figure 1, we plot $NCR(k)$ against k. The corresponding duality gap $X^k \bullet Z^k$ is shown in the same figure. We use the superscript "k" to denote dependence on the k interior-point iterate (X^k, y^k, Z^k). Notice that the duality gap decreases at almost a linear rate. This implies that the norm $\|R_c^k\|_F$ also decreases at almost a linear rate as the interior-point iteration progresses. As a result, the admissible condition (5) becomes more and more stringent as k increases, and more CR steps are required for this condition to be fulfilled. But ideally, we would hope that $NCR(k)$ increase at most linearly with k. However, it is evident from Figure 1 that $NCR(k)$ increases far more rapidly than the reduction in $X^k \bullet Z^k$. As the latter becomes smaller, the convergence rate of the CR method deteriorates rapidly, resulting in a huge increase in $NCR(k)$. Furthermore, the effect of rounding errors also becomes more prominent, as can be seen from the fact that $NCR(22) > 5m$, where in fact this number should be at most m in exact arithmetic.

The deterioration in the convergence rate is due to the fact that the spectrum of $\hat{M}_k$ becomes "worse" as k increases. In Figures 2 and 3, the spectra of $\hat{M}_k$ are plotted for $k = 15$ and $k = 20$, respectively. Based on these spectra, the convergence rates are at best equal to 0.94 and 0.993, respectively. This phenomenon of worsening convergence rate as the duality gap decreases is typical for the SCE. It is to be expected since the eigenvalues of $\hat{M}_k$ are spread continuously in an interval of the form $[O(\mu^k), O(1)]$.

In Table 2, we show the cumulative CPU time (in the format hours: minutes: seconds) taken by Algorithms PFchol and PFCR to solve a number of SDP problems arising from Lovász θ-function of random graphs to an accuracy of 10^{-6} in the relative duality gaps. The graphs are generated using a program provided by K. Nakata, whose help we gratefully acknowledge. For the machine we used, the CPU time required to

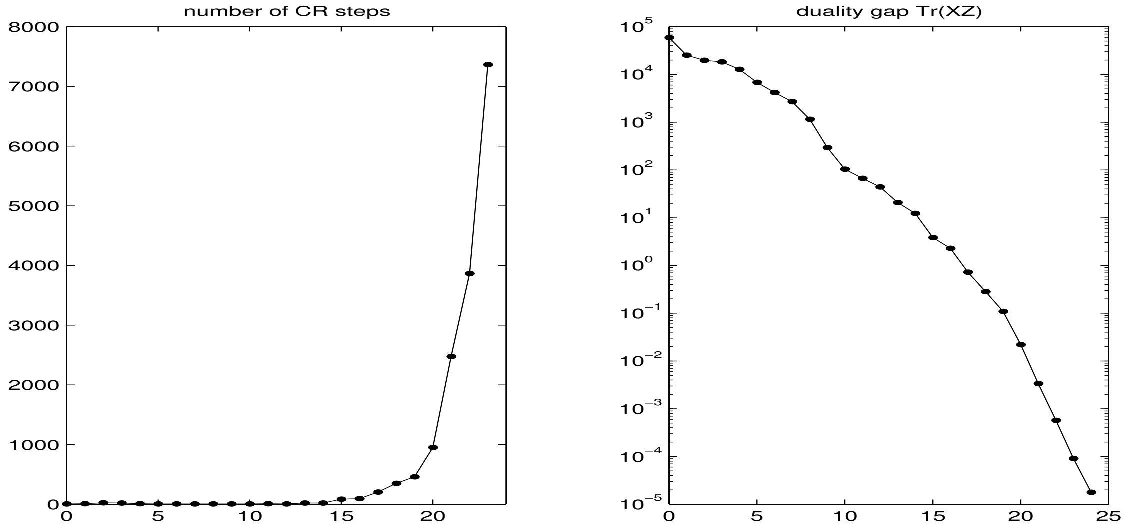

Figure 1: *Behavior of the CR method on the SDP problem* `theta2`. *The left-hand side plot is the number of CR steps taken in each interior-point iteration to solve the preconditioned Schur complement equation (9) approximately to satisfy the admissible condition (5). The right-hand side plot is the corresponding duality gap* $X \bullet Z$. *The residual in the infeasibilities,* $\max(\|r_p\|, \|R_d\|_F)$, *is less than* 10^{-12} *after the* 9*th interior-point iteration.*

compute the Cholesky factor of a given dense $m \times m$ positive definite matrix follows the empirical formula below:

$$chol\_time(m) \quad \approx \quad 4.446 \left(\frac{m}{100} \right)^3 \text{ secs.} \tag{10}$$

Those numbers appeared in typewriter style in Table 2 are the cumulative CPU times estimated from (10) to factorize M, and the memories needed to store M alone.

Observe that for the theta problems with large m, the CPU time taken by the CR method to solve these problems to an accuracy level of roughly 0.1 in the duality gaps can be one hundred times faster than that using the Cholesky factorization. This success can be attributed to the following facts when the duality gap is at a level above 0.1:

- the accuracy needed in computing an admissible direction is quite low;
- the matrix $\hat{M}$ is well-conditioned, leading to a fast convergence of the CR method where it takes only a relatively small number of steps to obtain an admissible direction.

But once the interior-point iteration progresses to the stage where the duality gaps are smaller than 0.01, the CR method becomes highly inefficient, resulting in a huge increase in the CPU time needed to compute an admissible direction. For example, for the problem with $m = 23872$, the CR method spent only about 1 hour in the first 15 interior-point iterations, but almost 99 hours in the last 6 iterations. On the other

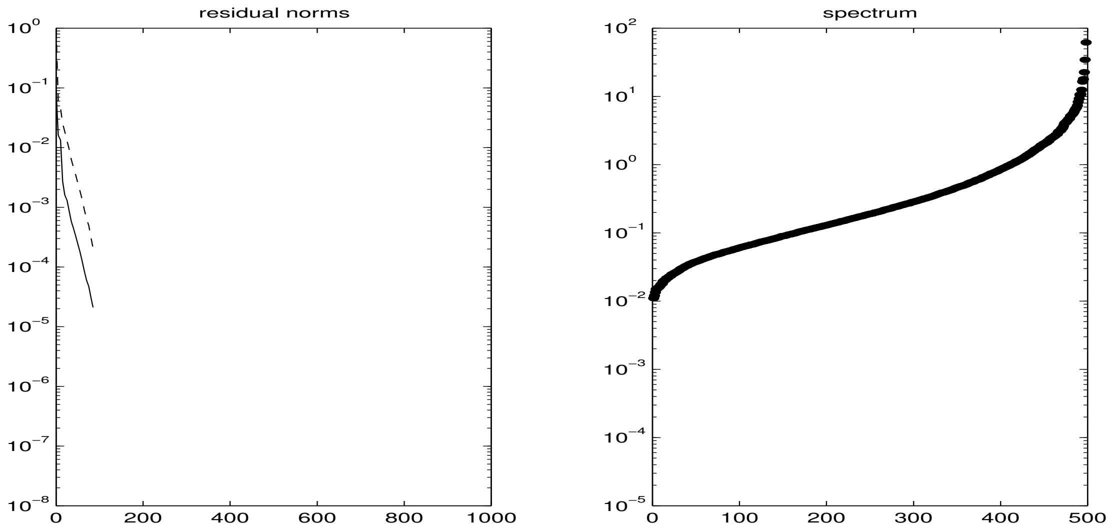

Figure 2: *The dashed line in the left-hand side plot is the residual norm* $\|h - Mv_j\|/\|h\|$ *generated by the CR method, where* M *and* h *correspond to the* $15th$ *interior-point iterate generated from the run in Fig. 1. The solid line corresponds to the quantity* $\|\mathcal{R}\|_F$ *defined in (7). The right-hand side plot is the spectrum of the preconditioned matrix* $\hat{M}$.

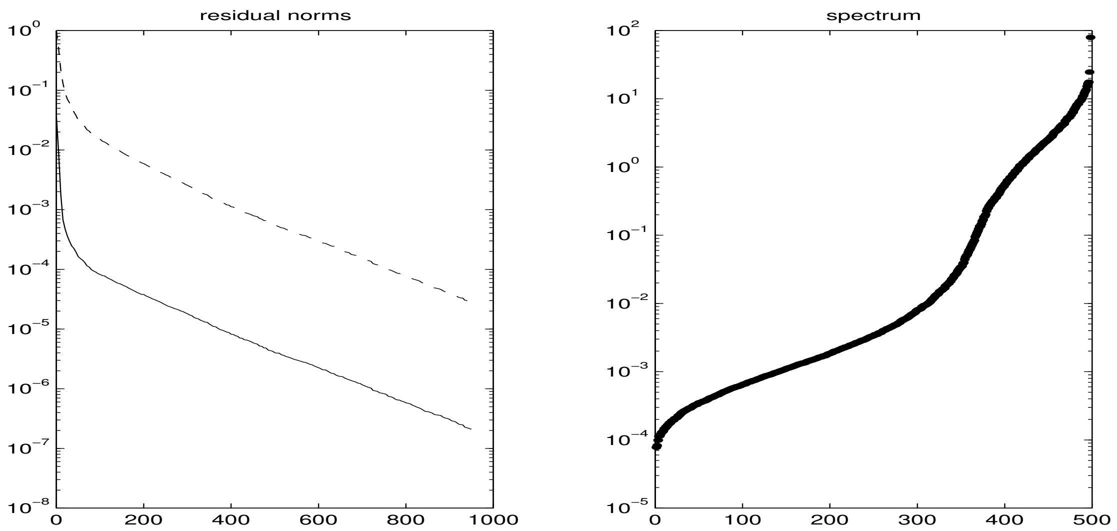

Figure 3: *Same as Fig. 2, but for the 20th interior-point iterate.*

hand, for the direct method using Cholesky factorization, the time taken is still the same as in all the previous iterations. The result is that, when the duality gaps are less than 0.01, an interior-point iteration using a direct method to solve (4a) is far more efficient than that using the CR method.

We note that [16] also reported the same observations on the behavior of the conjugate gradient method in solving a similar collection of SDPs.

Based on the above observations, one may want to combine the advantage of the CR and direct methods by using a hybrid method that employs the CR method to compute the search direction when the duality gap is above a certain level (say 0.01) and then switch to the direct method when the duality gap falls below that level. But as our main purpose in this paper is to address the memory problem when m is large, we shall not distract ourselves with such a hybrid method.

Finally, observe that from Figures 2 and 3, the ratio at step j,

$$\frac{\|\mathcal{R}_j\|}{\|r_j\|} \;=\; \frac{\|\mathcal{E}\mathcal{A}^T(\mathcal{A}\mathcal{A}^T)^{-1}r_j\|}{\|r_j\|},$$

stays almost at a constant value for all j. Thus we can first estimate this ratio, say c, at the beginning of the CR method, then stop the CR method when we have

$$\|\mathcal{R}_j\|_F \;\approx\; c\|r_j\| \;\leq\; \kappa\,\|R_c\|_F.$$

In this way, we avoid the need to compute the quantity $\|\mathcal{R}_j\|$ for each j. This can sometimes lead to a substantial saving in the computation time in each CR step, especially when computing $(\mathcal{A}\mathcal{A}^T)^{-1}r_j$ or $\mathcal{E}(U)$ for a given U is expensive. The empirical fact just mentioned can be very useful in cutting down computation time in practice.

		Solving (4a) via Cholesky factorization of M				Solving (4a) via the CR method on (9)				
n	m	iter. no.	cum. CPU time	$X \bullet Z$	mem. MB	iter. no.	cum. CPU time	$X \bullet Z$	mem. MB	$NCR(k)$
200	1949	18	12:47	$3.6e-1$	80	18	2:57	$3.6e-1$	18	335
		21	14:55	$3.9e-3$	80	21	11:11	$3.9e-3$	18	2500
		24	17:02	$3.6e-5$	80	24	57:48	$3.6e-5$	18	9745
200	5986		`3:58:30`		`287`	15	5:10	$1.7e-1$	20	245
			`4:46:12`		`287`	18	33:41	$1.3e-3$	20	3015
			`5:33:54`		`287`	21	6:03:50	$2.2e-5$	20	29930
300	4375	20	2:35:05	$2.1e-1$	219	20	14:14	$2.4e-1$	27	445
		23	2:57:18	$1.2e-3$	219	23	1:08:16	$1.3e-3$	27	6120
		26	3:19:23	$1.2e-5$	219	26	8:36:40	$1.3e-5$	27	31315
300	13390		`44:28:15`		`1434`	15	22:18	$2.3e-1$	44	290
			`53:21:54`		`1434`	18	2:33:16	$2.1e-3$	44	3450
			`62:15:33`		`1434`	21	35:02:47	$2.5e-5$	44	52320
400	7905		`12:12:00`		`500`	20	34:06	$2.9e-1$	39	475
			`14:01:48`		`500`	23	2:59:19	$1.9e-3$	39	5435
			`15:15:00`		`500`	25	15:19:15	$6.6e-5$	39	27420
400	23872		`251:56:15`		`4560`	15	1:02:39	$2.8e-1$	89	290
			`302:19:30`		`4560`	18	6:44:35	$3.2e-3$	89	2860
			`352:42:45`		`4560`	21	99:57:54	$1.2e-5$	89	57200

Table 2: Comparison of Algorithms PFchol and PFCR on a number of SDP problems arising from Lovász θ-function on graphs. The numbers in typewriter style are estimated figures needed to factorize and store M alone.

5 Projected Schur complement equation

The numerical results in the last section show that the CR method is very efficient in computing an admissible search direction when $X \bullet Z$ is greater than 0.1, but becomes exceedingly slow when $X \bullet Z$ is smaller than 0.01 for the theta problems. In this section, we propose a method to overcome this difficulty based on the structure of the Schur complement matrix when $X \bullet Z$ is small.

Given an interior-point iterate (X, y, Z), let $\mu := X \bullet Z/n$ and W be the associated Nesterov-Todd scaling matrix. Suppose (X, y, Z) is close to some optimal solution (X^*, y^*, Z^*) of the primal and dual SDP. If (X^*, Z^*) satisfies the strict complementarity, and the primal and dual nondegeneracy conditions defined in [1], then as (X, Z) approaches this optimal solution (i.e., when μ is sufficiently small), the eigenvalues of W will separate into two groups, one with large magnitudes of the order $\mathcal{O}(1/\sqrt{\mu})$ and the other with small magnitudes of the order $\mathcal{O}(\sqrt{\mu})$.

Now suppose that W has a group of p large eigenvalues, and a group of $q := n - p$ small eigenvalues. Let $W = QDQ^T$ be the eigenvalue decomposition of W. We can rewrite W as

$$W \;=\; Q_1 D_1 Q_1^T + Q_2 D_2 Q_2^T \;=:\; W_1 + W_2, \tag{11}$$

according to the partition $D = [D_1 \ 0\,;\, 0 \ D_2]$ and $Q = [Q_1 \ \ Q_2]$, where $D_1 \in I\!R^{p\times p}$, $Q_1 \in I\!R^{n\times p}$ correspond to the large eigenvalues and $D_2 \in I\!R^{q\times q}$, $Q_2 \in I\!R^{n\times q}$ correspond to the small eigenvalues.

Note that in practice, it is only necessary to compute a partial eigenvalue decomposition of W. All we need is D_1 and Q_1, and W_2 will be completely determined. If p is an integer that is much smaller than n, then D_1 and Q_1 can be computed cheaply by using variants of the Lanczos method for computing dominant eigenvalues. For the dual scaling direction used in [3], instead of W, it is the large eigenvalues of $Z^{-1}/\sqrt{\mu}$ that we should compute. When Z is sparse, computing the partial eigenvalue decomposition D_1 and Q_1 via Lanczos methods can sometimes be done efficiently even if n is large.

When μ is sufficiently small, the number of eigenvalues of W with magnitudes $\mathcal{O}(1/\sqrt{\mu})$ is equal to the rank of X^*. Thus p is usually equal to the rank of X^*. In actual computation, however, we can set p to be any integer such that $\bar{p} \leq m$, and it is not necessary to know the rank of X^*.

With the partition in (11), the Schur complement matrix M can be rewritten as

$$\begin{aligned} M \;&=\; \mathcal{A}\,(Q_1 \circledast Q_1)\ (D_1 \circledast D_1)\,(Q_1^T \circledast Q_1^T)\,\mathcal{A}^T + \mathcal{A}\,[(2W_1 + W_2) \circledast W_2]\,\mathcal{A}^T \\ &=\; \mathcal{A}_1 \mathcal{D}_1^2 \mathcal{A}_1^T + \mathcal{B}, \end{aligned} \tag{12}$$

where

$$\mathcal{A}_1 = \mathcal{A}(Q_1 \circledast Q_1), \qquad \mathcal{D}_1 = D_1^{1/2} \circledast D_1^{1/2}, \qquad \mathcal{B} = \mathcal{A}\,[(2W_1 + W_2) \circledast W_2]\,\mathcal{A}^T. \tag{13}$$

Note that $\mathcal{A}_1$ is a linear map from $\mathcal{S}^p$ into $I\!R^m$. But we will sometimes also view $\mathcal{A}_1$ as the matrix representation of the linear map with respect to the standard bases in

$\mathcal{S}^p$ and $\mathbb{R}^m$. Under the assumption that (X^*, Z^*) satisfies the strict complementarity, and the primal and dual nondegeneracy conditions defined in [1], the matrix $\mathcal{A}_1$ will have full column rank [1, Thm. 3] when μ is small and p is the rank of X^*.

We are now ready to describe our method to alleviate the problem of slow convergence when the CR method is applied to SCE (4a).

Theorem 5.1 *Suppose $\mathcal{A}_1 \in \mathbb{R}^{m\times\bar{p}}$ has full column rank. Let $H = \mathcal{A}_1^T\mathcal{A}_1$. Then $H \in \mathbb{R}^{\bar{p}\times\bar{p}}$ is nonsingular, and solving the SCE, $Mv = h$, for v is equivalent to solving for v_1 and v_2 from the following linear system of equations:*

$$\underbrace{\begin{bmatrix} I + \mathcal{D}_1^{-1}H^{-1}\mathcal{A}_1^T\mathcal{B}\mathcal{A}_1H^{-1}\mathcal{D}_1^{-1} & \mathcal{D}_1^{-1}H^{-1}\mathcal{A}_1^T\mathcal{B}\mathcal{Q} \\ \mathcal{Q}\mathcal{B}\mathcal{A}_1H^{-1}\mathcal{D}_1^{-1} & \mathcal{Q}\mathcal{B}\mathcal{Q} \end{bmatrix}}_{K} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \underbrace{\begin{bmatrix} \mathcal{D}_1^{-1}H^{-1}\mathcal{A}_1^T h \\ \mathcal{Q}h \end{bmatrix}}_{g}, \tag{14}$$

where $\mathcal{Q} = I - \mathcal{A}_1H^{-1}\mathcal{A}_1^T$. The solution v of the SCE can be recovered from

$$v = \mathcal{A}_1H^{-1}\mathcal{D}_1^{-1}v_1 + v_2. \tag{15}$$

In addition, if we let $[r_1\,;\, r_2] = g - K[v_1\,;\, v_2]$, then

$$h - Mv = \mathcal{A}_1\mathcal{D}_1r_1 + r_2. \tag{16}$$

Proof. In this proof, we will view $\mathcal{A}_1$ as a matrix in $\mathbb{R}^{m\times\bar{p}}$ instead of a linear operator from $\mathcal{S}^p$ to $\mathbb{R}^m$. Consider the decomposition

$$\mathcal{A}_1 = PR, \tag{17}$$

where $P \in \mathbb{R}^{m\times\bar{p}}$ is a matrix whose columns form an orthonormal set, and $R \in \mathbb{R}^{\bar{p}\times\bar{p}}$ is an upper triangular matrix. Let $\mathcal{Q} = I - PP^T$ be the orthogonal projection of $\mathbb{R}^m$ onto the orthogonal complement of Range$(\mathcal{A}_1)$. Then $v \in \mathbb{R}^m$ can be decomposed as

$$v = P\tilde{v}_1 + v_2, \tag{18}$$

where $\tilde{v}_1 = P^Tv$ and $v_2 = \mathcal{Q}v$. Substituting (17) and (18) into the SCE, $Mv = h$, and using (12), we have

$$PR\mathcal{D}_1^2R^T\tilde{v}_1 + \mathcal{B}P\tilde{v}_1 + \mathcal{B}v_2 = h,$$

which implies that

$$\begin{aligned} (R\mathcal{D}_1^2R^T + P^T\mathcal{B}P)\tilde{v}_1 + P^T\mathcal{B}v_2 &= P^Th \\ \mathcal{Q}\mathcal{B}P\tilde{v}_1 + \mathcal{Q}\mathcal{B}v_2 &= \mathcal{Q}h. \end{aligned}$$

Thus

$$\left.\begin{aligned} \left(I + \mathcal{D}_1^{-1}R^{-1}P^T\mathcal{B}PR^{-T}\mathcal{D}_1^{-1}\right)v_1 + \mathcal{D}_1^{-1}R^{-1}P^T\mathcal{B}v_2 &= \mathcal{D}_1^{-1}R^{-1}P^Th \\ \mathcal{Q}\mathcal{B}PR^{-T}\mathcal{D}_1^{-1}v_1 + \mathcal{Q}\mathcal{B}v_2 &= \mathcal{Q}h, \end{aligned}\right\} \tag{19}$$

where $v_1 = \mathcal{D}_1 R^T \tilde{v}_1$. We can avoid the explicit formation of the $m \times \bar{p}$ matrix P by observing that $P = \mathcal{A}_1 R^{-1}$, and R is the Cholesky factor of the matrix $H = R^T R$. With this observation, it is readily verified that (19) can be rewritten as (14), and $\mathcal{Q} = I - \mathcal{A}_1 H^{-1} \mathcal{A}_1^T$. Finally, from (18), it is readily shown that (15) holds. □

We will call the linear system of equations in (14) the *projected Schur complement equation* since its derivation is based on the orthogonal projection of $I\!R^m$ onto $\mathrm{Range}(\mathcal{A}_1)$. From the above theorem, we see that instead of applying the CR method to compute Δy approximately from (4a), we can compute it from (14). We may view (14) as a preconditioned version of (4a), but not in the conventional sense since an explicit description of an approximate inverse of M is not available.

The matrix K in (14) is a singular positive semidefinite matrix, but the system (14) is consistent. In fact, K has $\bar{p}$ zero eigenvalues. But these zero eigenvalues does not impede the convergence of the CR method since the right-hand side vector is orthogonal to the eigen-space associated with the zero eigenvalues. By a result in [4], the convergence rate of the CR method for (14) is determined solely by the positive eigenvalues of K when the initial guess of the CR method is chosen to be the zero vector. To be more quantitative, the next theorem states a result on the positive eigenvalues of K.

Theorem 5.2 *Let Λ_+ be the set of positive eigenvalues of the matrix $\mathcal{QBQ}$. Assume that a partition in* (12) *is chosen such that $\|D_1\| = \mathcal{O}(1/\sqrt{\mu})$, $\|D_2\| = \mathcal{O}(\sqrt{\mu})$, and* $\mathrm{dist}(0, \Lambda_+) \gg \mathcal{O}(\sqrt{\mu})$. *Suppose λ is a positive eigenvalue of the matrix K in* (14). *Then*

$$\mathrm{dist}(\lambda, \Lambda_+ \cup \{1\}) \;=\; \|\mathcal{QBA}_1\|\, \|H^{-1}\|\, \mathcal{O}(\sqrt{\mu}).$$

Proof. Observe that K can be written as

$$K \;=\; \begin{bmatrix} I & \\ & \mathcal{QBQ} \end{bmatrix} + \begin{bmatrix} \mathcal{D}_1^{-1} H^{-1} \mathcal{A}_1^T \mathcal{B} \mathcal{A}_1 H^{-1} \mathcal{D}_1^{-1} & \mathcal{D}_1^{-1} H^{-1} \mathcal{A}_1^T \mathcal{BQ} \\ \mathcal{QBA}_1 H^{-1} \mathcal{D}_1^{-1} & \end{bmatrix},$$

where the second matrix in the right-hand side is considered as a perturbation to the first matrix. By the Bauer-Fike theorem, we have

$$\mathrm{dist}(\lambda, \Lambda_+ \cup \{1\}) \;\le\; \|\mathcal{QBA}_1\|\, \|H^{-1}\|\, \|D_1^{-1}\| \;+\; \|H^{-1} \mathcal{A}_1^T \mathcal{B} \mathcal{A}_1 H^{-1}\|\, \|\mathcal{D}_1^{-1}\|^2.$$

Now, the required result follows readily from the above inequality. □

The above theorem implies that the positive eigenvalues of K are roughly contained in the set $\Lambda_+ \cup \{1\}$ when μ is small. Therefore, the convergence rate of the CR method applied to (14) is governed by the positive eigenvalues of $\mathcal{QBQ}$, which are usually spread over a smaller interval compared to the eigenvalues of $\hat{M}$. Thus, we would expect the CR method to converge faster in this case than when it is applied to (9).

5.1 Connection to the deflated conjugate gradient method

The projection step used in the derivation of the projected SCE bears some resemblances to the deflated conjugate gradient (CG) method described in [19]. The basic process involved in the deflated CG method for solving a linear system of equation $Bx = d$ is as follows. Suppose a matrix U of appropriate dimension is given and $U^T BU$ is nonsingular. Given an initial guess x_0 such that $U^T r_0 = 0$, where $r_0 = d - Bx_0$. The deflated CG constructs successively for each j, an approximate solution x_{j+1} such that

$$r_{j+1} := d - Bx_{j+1} \perp \mathrm{Range}(U) + \langle r_0, r_1, \dots, r_j \rangle.$$

In doing so, the convergence rate of the deflated CG method is governed by the eigenvalues of the matrix $\tilde{B} := B - BU(U^T BU)^{-1} U^T B$ instead of the original matrix B.

Now, adapting the deflated CG to the Schur complement matrix M and taking $U = \mathcal{A}_1 H^{-1} D_1^{-1}$, we have

$$\tilde{M} := M - MU(U^T MU)^{-1} U^T M = \mathcal{Q}\mathcal{B}\mathcal{Q} - (I - \mathcal{Q})\mathcal{B}(I - \mathcal{Q}) + E,$$

where $\|E\| = \mathcal{O}(\|D_1^{-1}\|^2)$, and $\mathcal{B}, \mathcal{Q}$ are the matrices appeared in Thm. 5.1. Thus if we apply the deflated CG to the SCE, the convergence rate is basically governed by the matrix $\mathcal{Q}\mathcal{B}\mathcal{Q} - (I - \mathcal{Q})\mathcal{B}(I - \mathcal{Q})$ when $\|D_1^{-1}\|$ is small. This convergence rate is very similar to that for the projected SCE where it is governed by the matrix $\mathcal{Q}\mathcal{B}\mathcal{Q}$,

Computationally, each step j of the deflated CG method requires the solution of the linear system $U^T BU\xi = U^T Br_j$, which for the SCE is given by

$$(I + D_1^{-1} H^{-1} \mathcal{A}_1^T \mathcal{B} \mathcal{A}_1 H^{-1} D_1^{-1})\, \xi = (D_1 \mathcal{A}_1^T + D_1^{-1} H^{-1} \mathcal{A}_1^T \mathcal{B})\, r_j.$$

Notice that the coefficient matrix is the $(1,1)$ block of the matrix K in (14). To apply the CR method to (14), this matrix need not be formed explicitly, but for the deflated CG, we need to compute it explicitly. However, computing this coefficient matrix is very expensive. Due to this serious disadvantage, we will not explore the use of the deflated CG method to solve the SCE any further in this paper.

5.2 Cost per CR step

To minimize the computation time of the CR method when applied to (14), we need to compute a matrix-vector product for (14) efficiently. To begin with, we state the cost for computing some basic steps required in the matrix-vector product in Table 3.

From (14), it is easy to check that if we let

$$z = \mathcal{B}\left(\mathcal{A}_1 H^{-1} \mathcal{D}_1^{-1} v_1 + \mathcal{Q} v_2\right) = \mathcal{B}\left(\mathcal{A}_1 H^{-1} \left[\mathcal{D}_1^{-1} v_1 - \mathcal{A}_1^T v_2\right] + v_2\right), \tag{20}$$

then

$$K \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} v_1 + \mathcal{D}_1^{-1}(H^{-1}\mathcal{A}_1^T z) \\ z - \mathcal{A}_1(H^{-1}\mathcal{A}_1^T z) \end{bmatrix}. \tag{21}$$

Computing		Number of flops required
$\mathcal{A}^T v$,	given $v \in I\!R^m$	$\rho_t\, mn^2$
$\mathcal{A}(T)$,	given $T \in \mathcal{S}^n$	$\rho_t\, mn^2$
$\mathcal{A}_1^T v$,	given $v \in I\!R^m$	$p^2 n + 2\rho_s\, pn^2 + \rho_t\, mn^2$
$\mathcal{A}_1(T)$,	given $T \in \mathcal{S}^p$	$2p^2 n + \rho_s\, pn^2 + \rho_t\, mn^2$

Table 3: *computational cost of some basic steps needed in a matrix-vector product for (14).*

Let

$$\omega_H \;=\; \text{the number of flops required to compute } H^{-1}f \text{ for a given } f. \tag{22}$$

If H is pre-computed with Cholesky factorization $H = R^T R$, then $\omega_H = 2\bar{p}^2$. For the numerical experiments in this paper, we compute H explicitly where the cost is estimated to be at most $p(4\rho_t mn^2 + 2p^2 n + (1+\rho_s)pn^2)$ flops.

In our implementation of the CR method for (14), we compute $K[v_1\,;\,v_2]$ via the procedure described in Table 4, where the cost is also estimated. Compared to the cost in computing $\hat{M}v$ in section 4, we see that computing $K[v_1\,;\,v_2]$ can be much more expensive. Exactly how much the latter is more expensive is difficult to estimate since it depends on the sparsity factors ρ_s, ρ_t, the cost ω_H in computing $H^{-1}f$, and the integers m, n, p. But hopefully, the reduction in the number of CR steps needed in computing an admissible direction will outweigh the higher cost required.

Computing	Number of flops required
$w := \mathcal{A}_1 H^{-1}(\mathcal{D}_1^{-1} v_1 - \mathcal{A}_1^T v_2) + v_2$	$3p^2 n + 3\rho_s\, pn^2 + 2\rho_t\, mn^2 + \omega_H$
$z := \mathcal{B} w$	$4\rho_s\, n^3 + 2\rho_t\, mn^2$
$H^{-1}\mathcal{A}_1^T z$	$p^2 n + 2\rho_s\, pn^2 + \rho_t\, mn^2 + \omega_H$
$\mathcal{A}_1(H^{-1}\mathcal{A}_1^T z)$	$2p^2 n + \rho_s\, pn^2 + \rho_t\, mn^2$
$K[v_1\,;\,v_2]$	$6p^2 n + 6\rho_s\, pn^2 + 4\rho_s\, n^3 + 6\rho_t\, mn^2 + 2\omega_H$

Table 4: *computational cost required in the matrix-vector product for (14).*

5.3 A preconditioner for K

The coefficient matrix K has an eigenvalues distribution that is much more favorable compared to that of the preconditioned Schur complement matrix $\hat{M}$. But we can further improve the eigenvalues distribution by applying a preconditioner to K.

The most obvious preconditioner for K is the diagonal preconditioner. However, it turns out that computing the diagonal elements of K is very expensive, where the cost of computing each diagonal element is at least $2p^2n+pn^2+3\rho_s n^3+2\rho_t mn^2+\omega_H$. Thus we are forced to look for a cheaper alternative. We find that the diagonal preconditioner of $\mathcal{B}$ is such an alternative. Suppose Φ is the diagonal preconditioner of $\mathcal{B}$, and $\Phi = LL^T$ is its Cholesky factorization.

We first rewrite (4a) using (12) as follows:

$$\mathcal{A}_1\mathcal{D}_1^2\mathcal{A}_1^T\,\Delta y + \mathcal{B}\,\Delta y \;=\; h. \tag{23}$$

Then we precondition the above system by L to get

$$\hat{\mathcal{A}}_1\mathcal{D}_1^2\hat{\mathcal{A}}_1^T\,\Delta\hat{y} + \hat{\mathcal{B}}\,\Delta\hat{y} \;=\; \hat{h}, \tag{24}$$

where

$$\hat{\mathcal{A}}_1 = L^{-1}\mathcal{A}_1, \quad \hat{\mathcal{B}} = L^{-1}\mathcal{B}L^{-T}, \quad \Delta\hat{y} = L^T\Delta y, \quad \hat{h} = L^{-1}h. \tag{25}$$

Thus instead of applying Theorem 5.1 to solve (23), we apply it to (24), and the associated equation has the form:

$$\underbrace{\begin{bmatrix} I+\mathcal{D}_1^{-1}\hat{H}^{-1}\hat{\mathcal{A}}_1^T\hat{\mathcal{B}}\hat{\mathcal{A}}_1\hat{H}^{-1}\mathcal{D}_1^{-1} & \mathcal{D}_1^{-1}\hat{H}^{-1}\hat{\mathcal{A}}_1^T\hat{\mathcal{B}}\hat{\mathcal{Q}} \\ \hat{\mathcal{Q}}\hat{\mathcal{B}}\hat{\mathcal{A}}_1\hat{H}^{-1}\mathcal{D}_1^{-1} & \hat{\mathcal{Q}}\hat{\mathcal{B}}\hat{\mathcal{Q}} \end{bmatrix}}_{\hat{K}} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \underbrace{\begin{bmatrix} \mathcal{D}_1^{-1}\hat{H}^{-1}\hat{\mathcal{A}}_1^T\hat{h} \\ \hat{\mathcal{Q}}\hat{h} \end{bmatrix}}_{\hat{g}}, \tag{26}$$

where $\hat{H} = \hat{\mathcal{A}}_1^T\hat{\mathcal{A}}_1$, and $\hat{\mathcal{Q}} = I - \hat{\mathcal{A}}_1\hat{H}^{-1}\hat{\mathcal{A}}_1^T$.

Let $\hat{K}$ be the coefficient matrix in (26). We observed that the eigenvalues distribution of $\hat{K}$ is better than that of K for the SDPs considered in this paper.

6 Behavior of the CR method on the projected Schur complement equation

In view of the efficiency of the CR method in computing an admissible search direction via (9) when the duality gap $X \bullet Z$ is not too small, we propose in this section a hybrid method that combines the advantage of applying the CR method to (9) and (26) for computing the search direction in each interior-point iteration. The details of the hybrid method are given in Algorithm PFCR1. For the numerical experiments in this section, when step 3(b) in Algorithm PFCR1 is invoked, the $\bar{p} \times \bar{p}$ matrix $\hat{H}$ is formed explicitly and its Cholesky factorization $\hat{H} = R^TR$ is computed before the CR method is applied to (26).

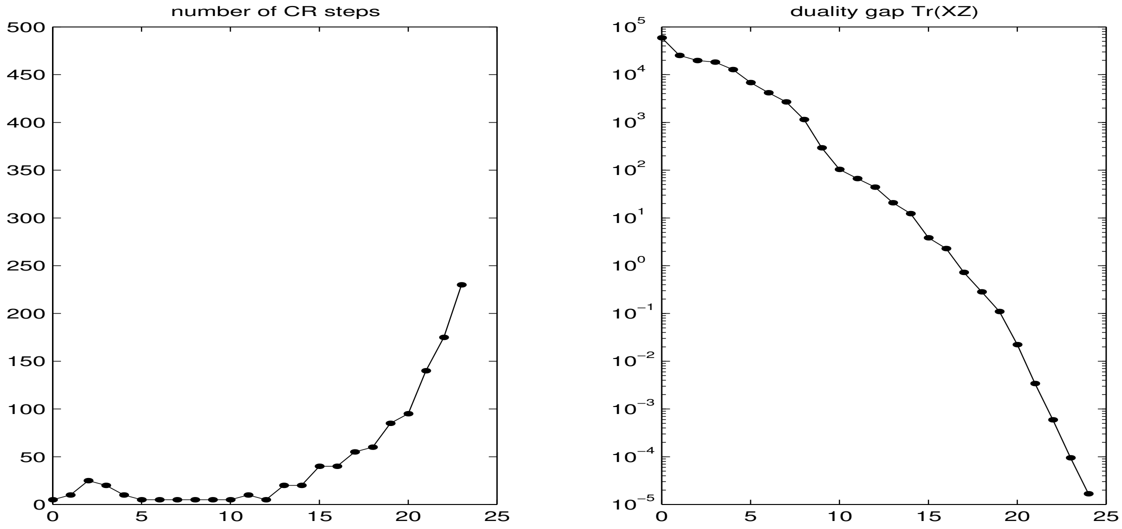

Figure 4: *Behavior of the CR method on the SDP problem theta2. The left-hand side plot is the number of CR steps taken in each iteration of Algorithm PFCR1 to solve either (9) or (26) approximately to satisfy the admissible condition (5). Note that from the* 15*th iteration onwards, (26) is used. The right-hand side plot is the corresponding duality gap* $X \bullet Z$*. Note the difference in scale between this figure and Fig. 1.*

Figures 4, 5, and 6 are the analogue of Figures 1, 2, and 3, respectively, for the SDP problem `theta2`, but using Algorithm PFCR1 instead of Algorithm PFCR. Comparing Figures 1 and 4, we see that the CR method method takes far fewer steps to solve (26) compared to that needed to solve (9) when $X \bullet Z$ is small. This indicates that the matrix $\hat{K}$ has a much more favorable eigenvalues distribution compared to $\hat{M}$. This is indeed confirmed in Figures 5 and 6, where the eigenvalues of $\hat{K}$ are plotted. From the spectra shown in these figures, the convergence rates of the CR method applied to (26) are roughly equal to 0.87 and 0.93 at the 15th and 20th interior-point iteration, respectively.

Table 5 compares the cumulative CPU time taken by Algorithms PFchol and PFCR1 at various interior-point iterations so as to achieve an accuracy of 10^{-6} in the relative duality gaps. It shows that Algorithm PFCR1 performs very much better than Algorithm PFchol, especially for problems with large m. For example, consider the problem with $m = 13390$, Algorithm PFCR1 is at least ten times faster than Algorithm PFchol to achieve the required accuracy in the relative duality gap, even if there is enough memory to store M. Comparing Tables 2 and 5, we see that the number of CR steps needed to solve (26) is far less than that required by (9). But because computing a matrix-vector product for (26) is much more expensive, the saving in the CPU time is not as impressive in the reduction is the number of CR steps.

Observe that though we need to store the dense $\bar{p} \times \bar{p}$ matrix $\hat{H}$, we are able to solve SDPs with m that is much larger than that allowed by the direct method. This is especially the true when $\bar{p}$ is much smaller compared m. But we should note that

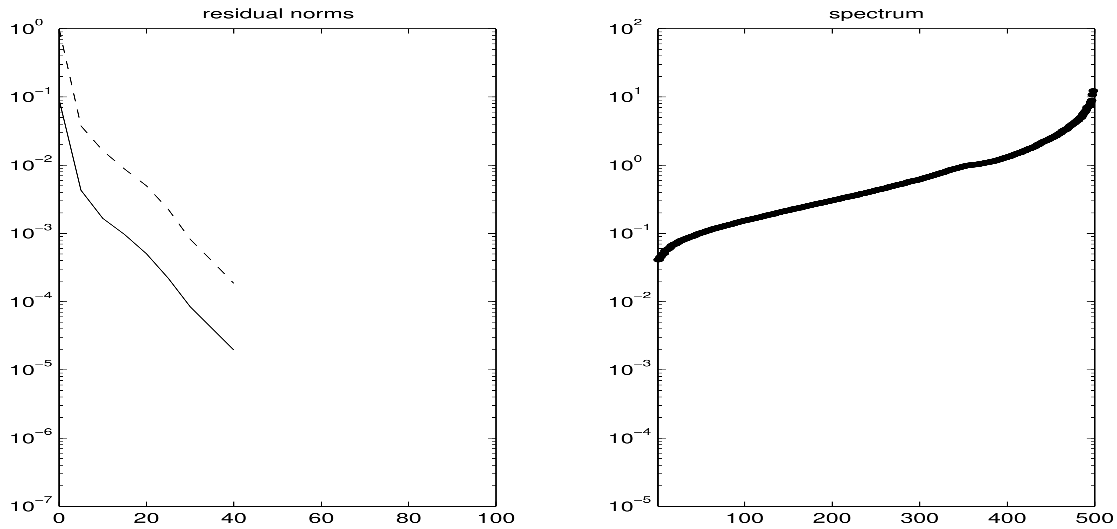

Figure 5: *The dashed line in left-hand side plot is the residual norm $\|h - Mv_j\|/\|h\|$ generated by the CR method, where M and h correspond to the 15th interior-point iterate generated from the run in Fig. 4. The solid line corresponds to the quantity $\|\mathcal{R}\|_F$ defined in (7). The right-hand side plot is the spectrum of $\hat{K}$. Note that there are 45 zeros eigenvalues not shown in the plot. Also note the difference in scale between this figure and Fig. 2.*

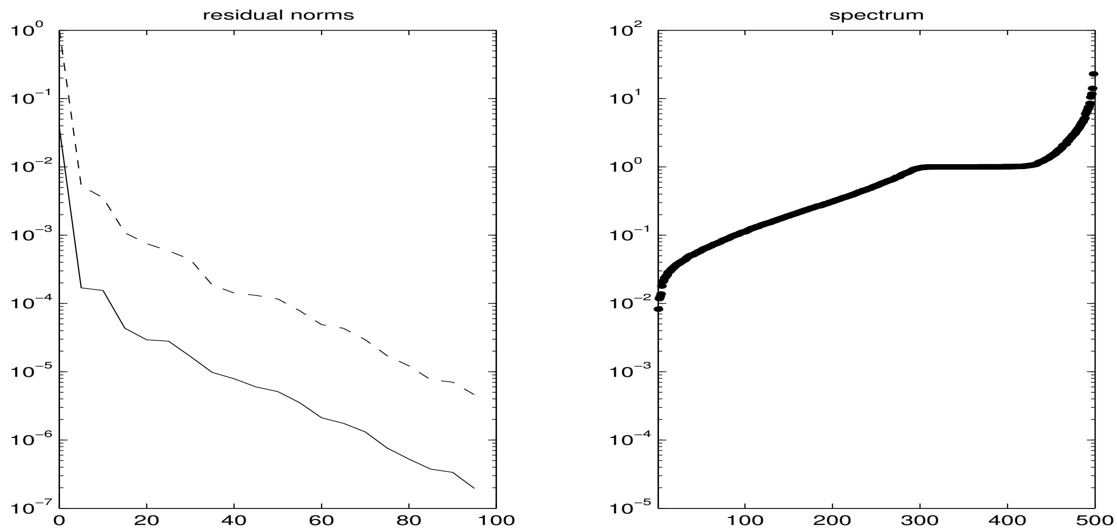

Figure 6: *Same as Fig. 5, but for the 20th interior-point iterate.*

although storing $\hat{H}$ requires significantly less memory than storing M for the SDPs we considered here, we may again face the problem of having insufficient memory when $\bar{p}$ is large, as can be seen from the last problem in Table 5. In a future paper, we will present a method where $\hat{H}$ is not formed explicitly to avoid possible memory problems. More specifically, we will solve the linear system $\hat{H}z = f$ using the CR method instead of using the Cholesky factorization.

Algorithm PFCR1. Suppose we are given an initial iterate (X^0, y^0, Z^0) with X^0, Z^0 positive definite. Select $\kappa = 0.01$. Set $\gamma^0 = 0.9$ and $\sigma^0 = 0.5$.

For $k = 0, 1, \dots$

Let the current and the next iterate be (X, y, Z) and (X^+, y^+, Z^+) respectively. Also, let the current and the next step-length (centering) parameter be denoted by γ and γ^+ (σ and σ^+) respectively.

1. Set $\mu = X \bullet Z/n$ and

$$\phi \;=\; \max\left(\frac{\|r_p\|}{\max(1,\|b\|)}, \frac{\|R_d\|_F}{\max(1,\|C\|_F)}\right). \tag{27}$$

   Stop the iteration if the infeasibility measure ϕ and the duality gap $X \bullet Z$ are sufficiently small.

2. Compute the Nesterov-Todd scaling matrix W and its eigenvalue decomposition $W = QDQ^T$. Let $d = \mathrm{diag}(D)$, where d is sorted in ascending order.

   If $\max(d)/\min(d) > 10^3$
   choose p to be the integer such that d_{p+1}/d_p is the maximum,
   else
   set $p = 0$,
   end

3. (a) If $p = 0$;

   Compute an admissible search direction $(\Delta X, \Delta y, \Delta Z)$ using the CR method via (9).

   (b) If $p > 0$;

   Compute an admissible search direction $(\Delta X, \Delta y, \Delta Z)$ using the CR method via the projected SCE (26).

4. Update (X, y, Z) to (X^+, y^+, Z^+) by

$$X^+ \;=\; X + \alpha\,\Delta X, \quad y^+ \;=\; y + \beta\,\Delta y, \quad Z^+ \;=\; Z + \beta\,\Delta Z, \tag{28}$$

   where

$$\alpha \;=\; \min\left(1, \frac{-\gamma}{\lambda_{\min}(X^{-1}\Delta X)}\right), \quad \beta \;=\; \min\left(1, \frac{-\gamma}{\lambda_{\min}(Z^{-1}\Delta Z)}\right). \tag{29}$$

   (Here $\lambda_{\min}(U)$ denotes the minimum eigenvalue of U; if the minimum eigenvalue in either expression is positive, we ignore the corresponding term.)

5. Update the step-length parameter by

$$\gamma^+ \;=\; 0.9 + 0.08\min(\alpha,\beta), \tag{30}$$

   and the centering parameter by $\sigma^+ \;=\; 1 - 0.9\min(\alpha,\beta)$.

		Solving (4a) via Cholesky factorization of M				Solving (4a) via the CR method on either (9) or (26)					
n	m	iter. no.	cum. CPU time	$X \bullet Z$	mem. MB	iter. no.	cum. CPU time	$X \bullet Z$	mem. MB	$NCR(k)$	p
200	1949	18	12:47	$3.6e-1$	80	18	2:43	$3.3e-1$	20	110	20
		21	14:55	$3.9e-3$	80	21	4:33	$3.8e-3$	22	125	32
		24	17:02	$3.6e-5$	80	24	7:31	$2.4e-5$	22	215	32
200	5986		3:58:30		287	15	4:48	$1.7e-1$	22	257	0
			4:46:12		287	18	20:13	$1.5e-3$	63	195	67
			5:33:54		287	21	56:50	$1.2e-5$	65	530	69
300	4375	20	2:35:05	$2.1e-1$	219	20	11:28	$2.3e-1$	38	75	41
		23	2:57:18	$1.2e-3$	219	23	21:05	$1.3e-3$	40	240	47
		26	3:19:23	$1.2e-5$	219	26	45:23	$1.3e-5$	41	735	48
300	13390		44:28:15		1434	15	20:32	$2.3e-1$	45	297	0
			53:21:54		1434	18	1:52:10	$2.9e-3$	242	165	100
			62:15:33		1434	21	4:18:52	$2.7e-5$	258	185	106
400	7905		12:12:00		500	20	32:58	$2.3e-1$	65	65	58
			14:01:48		500	23	1:03:25	$1.5e-3$	82	235	65
			15:15:00		500	25	1:38:52	$4.4e-5$	82	505	66
400	23872					insufficient memory to store the matrix $\hat{H}$ in (26)					

Table 5: Comparison between Algorithm PFchol and PFCR1. The latter uses the CR method to solve either (9) or (26), depending on the condition number of W in each iteration. A "0" in the last column means that (9) is used; otherwise (26) is used.

In the context of this paper, we are primarily concerned with SDPs where $m \gg n$. However, for the sake of gaining more computational experiences in solving the SCE by the CR method and the method proposed in this section, we will also consider problems where $m \approx n$ are both large. As we have noted before, primal-dual interior-point methods are not the most appropriate methods to be used when $m \approx n$ are both large due to the fact that the memory required by the primal variable is as acute as the Schur complement matrix. (The appropriate method for this class of SDPs is the dual scaling method described in [3].) Thus our main concern now for these problems is not on alleviating the memory problem but on the speed at which the SCE can be solved by the CR method.

Table 6 compares the performance of PFchol, PFCR and PFCR1 on a number of SDPs arising from MAXCUT problems for random graphs. In the table, we report the cumulative CPU time taken to solve the SCE alone (including the time to form M if it is needed explicitly) via the Cholesky factorization, the CR method on (9), and the hybrid combination of the CR method on (9) and (26). We stop the interior point iterations when the relative duality gaps are less than 10^{-6}.

The most important observation we want to make from Table 6 is that the number of CR steps needed to solve (26) is far less than that needed to solve (9), though each step of the former is more expensive.

Notice that for PFchol, when there is enough memory to store the Schur complement matrix M, solving (4a) by the direct method takes far less CPU time compared to using an iterative method for the SDPs we tested. This can be attributed to two facts. Firstly, the SDPs arising from the MAXCUT problems are highly sparse and the formation of M can be done efficiently. Secondly, the Cholesky factorization routine we used is a highly optimized routine (about twice as fast as MATLAB's `chol`) for computer execution compared to the iterative routines we implemented. However, we would expect these advantages of the direct method to diminish when the dimensions m, n of the SDPs increase since computing the Cholesky factorization of M then becomes prohibitively expensive. But due to the huge memory requirement of the primal variable, we are unable to experiment with problems where n is much larger than 1000 in a primal-dual interior point framework.

As mentioned earlier, the projected SCE proposed in this paper can be adapted to a solve the SCE associated with a dual-scaling interior point method. Based on the computational experiences that have been reported in [3] and [5] for SDPs arising from MAXCUT problems, our guess is that it is very likely that the method we proposed here can efficiently solve the SCE associated with the dual scaling interior-point method for this class of problems.

			Solving (4a) via Cholesky factorization of M		Solving (4a) via the CR method on (9)			Solving (4a) via the CR method on (9) or (26)			
n	m	iter. no.	CPU time to solve (4a)	$X \bullet Z$	CPU time to solve (9)	$X \bullet Z$	$NCR(k)$	CPU time to solve (9) or (26)	$X \bullet Z$	$NCR(k)$	p
300	300	11	9	$4.0e+1$	9	$2.5e+1$	27	9	$2.5e+1$	27	0
		14	12	$1.0e+0$	26	$3.6e-1$	88	25	$3.7e-1$	16	8
		17	15	$1.0e-3$	73	$4.6e-4$	296	46	$4.3e-4$	26	8
500	500	14	74	$5.1e+1$	56	$1.3e+1$	75	63	$1.3e+1$	75	0
		17	91	$1.2e+0$	194	$2.3e-1$	380	217	$2.3e-1$	380	0
		20	108	$1.3e-2$	718	$4.4e-3$	1726	355	$4.3e-3$	37	11
1000	1000	12	65	$3.4e+2$	94	$2.9e+2$	26	93	$2.9e+2$	26	0
		15	81	$7.0e+0$	281	$8.5e+0$	114	278	$8.5e+0$	114	0
		18	98	$1.4e-2$	1176	$1.1e-2$	569	740	$9.2e-3$	41	14

Table 6: Comparison of Algorithms PFchol, PFCR, and PFCR1 on a number of SDPs arising from MAXCUT problems.

7 Conclusion and discussion

We introduced a decomposition of the SCE and convert it into what we called the projected SCE. Numerical experiments on SDPs arising theta and MAXCUT problems were conducted to show that when the CR method is applied to this equation, moderately accurate solutions can be obtained at reasonable cost. The proposed method is well suited for a primal and dual nondegenerate problem satisfying the strict complementarity condition, and that has an optimal primal solution whose rank is small.

The research work in this paper has generated a number of problems that deserve further investigations in the future.

The applicability of the projected Schur complement approach remains to be tested for wider classes of SDPs. For example, in a recent work [8], it was shown that a large sparse SDP can sometimes be converted into one with a block diagonal structure by increasing the number of linear constraints. It will be interesting to adapt the method we developed in this paper to solve this type of SDPs.

Another area that we need to improve on is a careful implementation of the projected SCE to fully exploit any sparsity and special structures present in the SDP data. This is especially important for large SDPs such as those arising from graph partition problems where the data involves dense matrices of small ranks.

We would also want to investigate the performance of the projected SCE on large SDPs arising from MAXCUT problems in a dual scaling interior point method. Based on extensive computational experiences reported in [3], [5] and [10] for SDPs arising from MAXCUT problems, our guess is that for this class of problems, the method we proposed in this paper can solve them efficiently. But this remains to be verified.

As the reader may have recalled, in each step of the CR method for the projected SCE, we need to solve a dense $\bar{p} \times \bar{p}$ linear system of equations $\hat{H}z = f$. Though $\bar{p}$ is generally much smaller than m, it can still be large, and memory problem may again arise. One possible way to overcome this problem is to solve the system by an iterative method. Our preliminary work has shown that this linear system can be solved without much difficulties by the CR method.

Lastly, the most difficult problem we have to face is perhaps to extend the idea in this paper to solve degenerate SDPs.

Acknowledgements

Part of this research was done while the first author was visiting the Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, Japan. He thanks his colleagues there for their hospitality and for providing an excellent research environment during his stint there.

References

[1] F. Alizadeh, J. A. Haeberly, and M. Overton, *Complementarity and Nondegeneracy in semidefinite programming*, Math. Prog., 8 (1998), 77 (1997), pp. 111–128.

[2] B. Borchers, *SDPLIB, A Library of Semidefinite Programming Test Problems*, available from `http://www.nmt.edu/~borchers/sdplib.html`.

[3] S. J. Benson, Y. Ye, and X. Zhang, *Solving large-scale sparse semidefinite programs for combinatorial optimization*, SIAM J. Optimization, 10 (2000), pp. 443-461.

[4] P.N. Brown, and H.F. Walker, *GMRES on (nearly) singular systems*, SIAM J. Matrix Analysis and Applications, 18 (1997), pp. 37–51.

[5] C. Choi, and Y. Ye, *Solving sparse semidefinite programs using the dual scaling algorithm with an iterative solver*, working paper, Computational Optimization Laboratory, University of Iowa, March, 2000.

[6] T.A. Driscoll, K.C. Toh and L.N. Trefethen, *From potential theory to matrix iterations in six steps*, SIAM Review, 40 (1998), pp. 547-578.

[7] K. Fujisawa, M. Kojima, and K. Nakata, *Exploiting sparsity in primal-dual interior-point methods for semidefinite programming*, Mathematical Programming, 79 (1997), pp. 235–253.

[8] M. Fukuda, M. Kojima, K. Murota, and K. Nakata, *Exploiting sparsity in semidefinite programming via matrix completion I: general framework*, Research Report B-358, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Tokyo, December, 1999.

[9] A. Greenbaum, *Estimating the attainable accuracy of recursively computed residual methods*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 535–551.

[10] C. Helmberg, and K.C. Kiwiel, *A spectral bundle method with bounds*, Preprint SC 99-37, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Germany.

[11] C. Helmberg, F. Rendl, R. Vanderbei, and H. Wolkowicz, *An interior-point method for semidefinite programming*, SIAM J. Optimization, 6 (1996), pp. 342–361.

[12] M. Kojima, S. Shindoh, and S. Hara, *Interior-point methods for the monotone semidefinite linear complementarity problem in symmetric matrices*, SIAM J. Optimization, 7 (1997), pp. 86–125.

[13] M. Kojima, M. Shida, and S. Shindoh, *Search directions in the SDP and the monotone SDLCP: generalization and inexact computation*, Mathematical Programming, 85 (1999), pp. 51–80.

[14] C.-J. Lin, and R. Saigal, *On solving large-scale semidefinite programming problems – a case study of quadratic assignment problem*, manuscript, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI.

[15] R. D. C. Monteiro, *Primal-dual path following algorithms for semidefinite programming*, SIAM J. Optimization, 7 (1997), pp. 663–678.

[16] K. Nakata, K. Fujisawa, and M. Kojima, *Using the conjugate gradient method in interior-points methods for semidefinite programs* (in Japanese), Proceedings of the Institute of Statistical Mathematics, 46 (1998), pp. 297–316.

[17] K. Nakata, S.-L. Zhang and M. Kojima, *Precondioned conjugate gradient methods for large scale and dense linear systems in semidefinite programming*, abstract based on talks delivered at INFORMS Meeting, Philadelphia, November 1999.

[18] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston.

[19] Y. Saad, M. Yeung, J. Erhel, and F. Guvomarc'h, *A deflated version of the conjugate gradient algorithm*, SIAM J. Sci. Comput., to appear.

[20] Z. Strakos, *On the real convergence rate of the conjugate gradient method*, Linear Algebra and its Applications, 154–156 (1991), pp. 535–549.

[21] M. J. Todd, K. C. Toh, and R. H. Tütüncü, *On the Nesterov-Todd direction in semidefinite programming*, SIAM J. Optimization, 8 (1998), pp. 769–796.

[22] K. C. Toh, M. J. Todd, and R. H. Tütüncü, *SDPT3 — a* MATLAB *software package for semidefinite programming, version 1.3*, Optimization Methods and Software, 11 (1999), pp. 545–581. A copy of the software can be obtained through the worldwide web from `http://www.math.nus.sg/~mattohkc`.

[23] Y. Zhang, *On extending some primal-dual interior-point algorithms from linear programming to semidefinite programming*, SIAM J. Optimization, 8 (1998), pp. 365–386.

[24] Q. Zhao, S.E. Karisch, F. Rendl, and H. Wolkowicz, *Semidefinite programming relaxations for the quadratic assignment problem*, Technical Report, Department of Combinatorics and Optimization, University of Waterloo, Ontario, Canada, 1996.