BFGS-like updates of constraint preconditioners for sequences of KKT linear systems

L. Bergamaschi[1], V. De Simone[2], D. di Serafino[2*], and A. Martínez[3]

[1]*Department of Civil, Environmental and Architectural Engineering, University of Padua, Padova, Italy*
[2]*Department of Mathematics and Physics, Second University of Naples, Caserta, Italy*
[3]*Department of Mathematics, University of Padua, Padova, Italy*

November 1, 2016

SUMMARY

We focus on efficient preconditioning techniques for sequences of KKT linear systems arising from the interior point solution of large convex quadratic programming problems. Constraint Preconditioners (CPs), though very effective in accelerating Krylov methods in the solution of KKT systems, have a very high computational cost in some instances, because their factorization may be the most time-consuming task at each interior point iteration. We overcome this problem by computing the CP from scratch only at selected interior point iterations and by updating the last computed CP at the remaining iterations, via suitable low-rank modifications based on a BFGS-like formula. This work extends the limited-memory preconditioners for symmetric positive definite matrices proposed by Gratton, Sartenaer and Tshimanga in [*SIAM J. Optim. 2011; 21(3):912–935*], by exploiting specific features of KKT systems and CPs. We prove that the updated preconditioners still belong to the class of exact CPs, thus allowing the use of the conjugate gradient method. Furthermore, they have the property of increasing the number of unit eigenvalues of the preconditioned matrix as compared to generally used CPs. Numerical experiments are reported, which show the effectiveness of our updating technique when the cost for the factorization of the CP is high.

KEY WORDS: KKT linear systems; constraint preconditioners; BFGS-like updates, interior point methods.

1. INTRODUCTION

We consider sequences of linear systems arising in the application of Interior Point (IP) methods to the following convex Quadratic Programming (QP) problem:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\, x^T Q x + c^T x, \\ \text{subject to} \quad & \tilde{A} x = b, \\ & x \geq 0, \end{aligned} \tag{1}$$

where $Q \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite, $\tilde{A} \in \mathbb{R}^{m\times n}$ is full rank and $m \leq n$. At each IP iteration, a search direction is computed by applying a Newton step to perturbed first-order optimality conditions; this leads, after some manipulation, to a so-called KKT (or saddle-point) linear system:

$$H_k u_k = d_k, \tag{2}$$

*Correspondence to: Daniela di Serafino, Department of Mathematics and Physics, Second University of Naples, viale A. Lincoln 5, 81100 Caserta, Italy. E-mail: daniela.diserafino@unina2.it

where

$$H_k = \begin{bmatrix} G_k & A^T \\ A & 0 \end{bmatrix}, \quad u_k = \begin{bmatrix} u_{1,k} \\ u_{2,k} \end{bmatrix}, \quad d_k = \begin{bmatrix} d_{1,k} \\ d_{2,k} \end{bmatrix}, \tag{3}$$

$$A = -\tilde{A}, \quad G_k = Q + \Theta_k, \quad \Theta_k = X_k^{-1} Z_k, \quad X_k = \text{diag}(x_k), \quad Z_k = \text{diag}(z_k), \tag{4}$$

k identifies the IP iteration, (x_k, z_k) is the pair of complementary variables at that iteration, u_k is the Newton step for updating the primal variable x_k and the vector of Lagrange multipliers associated with the equality constraints (see, e.g., [1, 2]), and $\text{diag}(v)$ denotes the diagonal matrix with diagonal entries equal to the components of v. Note that G_k is positive definite because x_k and z_k have positive components. We focus on large-scale problems, where the matrices Q and A are often sparse.

It is well known that the entries of Θ_k may tend either to zero or to infinity as the iterate approaches the optimal solution of problem (1), thus making H_k severely ill conditioned (see, e.g., [3]). If the KKT systems are solved by Krylov methods, as it is often the case for large-scale problems, the use of effective preconditioners is fundamental for the overall performance of the IP methods. On the other hand, building from scratch a high-quality preconditioner for each KKT system may still require a large computational effort, and the reuse of information coming from a preconditioner computed at a previous IP iteration appears to be a nice alternative.

Many preconditioners have been proposed for system (2)-(4). The most successful ones exploit the block structure of H_k, possibly together with information about the meaning and the structure of the blocks. In this work we focus on Constraint Preconditioners (CPs), which have largely demonstrated their effectiveness in optimization as well as in other contexts (see, e.g., [4, 5, 6, 7, 8, 9, 3]). A nice property of CPs is that they allow the use of the Conjugate Gradient (CG) algorithm, although the matrix of the KKT system and the preconditioner are indefinite (see Section 2). The application of CPs usually requires their factorization, which may result in a high computational cost when the problem is large scale. Therefore several approximations of CPs, known as inexact CPs, have been considered, with the aim of finding a good tradeoff between cost and effectiveness [4, 10, 11, 12, 13]. We note that CG generally cannot be used with inexact CPs; hence other Krylov solvers must be applied, such as GMRES and QMR, or the simplified version of QMR known as SQMR [14].

The idea of using a CP computed for a KKT system to obtain less expensive (inexact) CPs for subsequent KKT systems in a given sequence has been recently investigated in [15, 16, 17]. The procedure proposed in [15] builds an inexact CP for a KKT system at a certain IP iteration by performing a low-rank update of the factorized Schur complement of the (1,1) block of a "seed" CP, i.e., a CP computed at a previous IP iteration. The definition of the update is guided by theoretical results on the spectrum of the preconditioned matrix. For KKT systems with nonzero (2,2) block, in [16] the previous strategy is combined with a low-cost updating technique [18, 19], which is able to take into account information discarded by the low-rank correction and expressed as a diagonal modification of the preconditioner arising from the first update. In [17] the authors focus on sequences of KKT linear systems with varying off-diagonal blocks, where the computations with these blocks are much more expensive than the computations with the (1,1) block. Inexact CPs for the matrices of the sequence are built by applying generalizations of limited-memory quasi-Newton updates (see, e.g., [20, 21]) that act only on the off-diagonal blocks of a previously computed inexact CP of the type described in [12].

Limited-memory quasi-Newton updating techniques have been widely used to build preconditioners for sequences of linear systems, usually with slowly varying matrices (see, e.g., [22, 23, 24, 25, 26, 17]). However, to the best of our knowledge, the update of CPs via quasi-Newton techniques has been considered only in the aforementioned work [17]. In this article, we present a preconditioner updating technique based on multiple BFGS-like corrections that is tailored for CPs. The updated preconditioner still belongs to the class of exact CPs and hence allows the use of the CG method. Furthermore, it has the nice property of increasing the number of unit eigenvalues of the preconditioned matrix with respect to general CPs that are built from scratch. Our work extends the limited-memory preconditioners (LMPs) for symmetric positive definite (SPD) matrices discussed in [25], exploiting specific features of KKT systems and CPs. We note that, in [27], LMPs have

been also extended to general symmetric indefinite linear systems, without taking into account any special type of indefinite matrix (although they have been applied to KKT systems coupled with block-diagonal preconditioners).

This paper is organized as follows. In Section 2 we briefly describe CPs for the matrix in (3) and recall their main properties. In Section 3 we present our technique for updating CPs, providing theoretical results on the spectrum of the corresponding preconditioned matrix. In Section 4 we specialize the previous technique to obtain practical updating procedures. In Section 5 we provide some implementation details and in Section 6 we illustrate the behaviour of the updating procedures on sequences of KKT linear systems arising in the solution of convex QP problems by an IP method. Finally, some conclusions are given in Section 7.

Henceforth we use the following notations: $\|\cdot\|$ denotes either the vector or the matrix 2-norm; for any symmetric matrix M, $\lambda_{min}(M)$, $\lambda_{max}(M)$ and $\lambda(M)$ denote the minimum, the maximum and any eigenvalue of M, respectively; $\kappa(M)$ indicates the spectral condition number of M and $\mathrm{diag}(M)$ the diagonal matrix with the same diagonal entries as M; finally, I denotes the identity matrix of appropriate dimension.

2. CONSTRAINT PRECONDITIONERS

For simplicity of notation, we drop the subscript k from all the matrices and vectors in (2)–(4); hence the KKT system reads

$$Hu = d, \tag{5}$$

with

$$H = \begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix}, \quad u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \quad d = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}, \quad G = Q + \Theta. \tag{6}$$

CPs for matrix H have the following form:

$$B = \begin{bmatrix} E & A^T \\ A & 0 \end{bmatrix}, \tag{7}$$

where E is some symmetric approximation to G. Any preconditioner of this type can be regarded as the coefficient matrix of a KKT system associated with an optimization problem with the same constraints as the original problem, thus motivating the name of the preconditioner. We note that E should be chosen so that B is nonsingular and is "easier to invert" than H; furthermore, it must involve Θ in order to capture the key numerical properties of H. A common choice is

$$E = \mathrm{diag}(G); \tag{8}$$

a different approach consists in implicitly defining E by using a factorization of the form $B = MCM^T$, where M and C are specially chosen matrices [28]. Here we consider (8), which is SPD.

The spectral properties of the preconditioned matrix $B^{-1}H$ and the application of CG with preconditioner B to the KKT linear system have been deeply investigated. For the sake of completeness, in the next theorem we summarize some theoretical results about CPs, given in [4, 6, 7].

Theorem 2.1
Let H, G, A, B, and E be the matrices given in (6) and (7), and let $Z \in \mathbb{R}^{n\times(n-m)}$ be a matrix whose columns span the nullspace of A. Assume also that E is SPD. The following properties hold.

1. $B^{-1}H$ has an eigenvalue at 1 with multiplicity $2m$.
2. The remaining $n-m$ eigenvalues of $B^{-1}H$ are defined by the generalized eigenvalue problem
$$Z^T G Z w = \lambda Z^T E Z w. \tag{9}$$

3. The eigenvalues, λ, of (9) satisfy

$$\lambda_{min}(E^{-1}G) \leq \lambda \leq \lambda_{max}(E^{-1}G). \tag{10}$$

4. If CG is applied to system (5) with preconditioner B and starting guess $u^{(0)} = [u_1^{(0)}\ u_2^{(0)}]^T$ such that $Au_1^{(0)} = d_2$, then the corresponding iterates $u_1^{(j)}$ are the same as the ones generated by CG applied to

$$(Z^TGZ)u_1 = Z^T(d_1 - Gu_1^{(0)}), \tag{11}$$

with preconditioner Z^TEZ. Thus, u_1^* is obtained in at most $n-m$ iterations and the following inequality holds:

$$\|u_1^{(j)} - u_1^*\| \leq 2\sqrt{\kappa}\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^j \|u_1^{(0)} - u_1^*\|, \quad j = 1, \ldots, n-m,$$

where $\kappa = \kappa((Z^TEZ)^{-1}Z^TGZ)$.

5. The directions $p^{(j)}$ and the residuals $r^{(j)}$ generated by applying CG with preconditioner B to system (5), with the same starting guess as in item 4, take the following form:

$$p^{(j)} = \begin{bmatrix} Z\bar{p}_1^{(j)} \\ p_2^{(j)} \end{bmatrix}, \quad r^{(j)} = \begin{bmatrix} r_1^{(j)} \\ 0 \end{bmatrix}, \tag{12}$$

where $\bar{p}_1^{(j)}$ and $r_1^{(j)}$ are the direction and the residual, respectively, at the jth iteration of CG applied to (11) with preconditioner Z^TEZ, and

$$(p^{(j)})^T H p^{(i)} = (\bar{p}_1^{(j)})^T Z^T G Z \bar{p}_1^{(i)}. \tag{13}$$

From the previous theorem it follows that the preconditioned matrix has $2m$ unit eigenvalues independently of the particular choice of E; on the other hand, properties 2 and 3 show that the better E approximates G, the more the remaining $n-m$ eigenvalues of $B^{-1}H$ are clustered around 1. Furthermore, the application of CG to the KKT system (5) with preconditioner B is closely related to the application of CG to system (11) with preconditioner Z^TEZ. We note that property 4 does not guarantee that $u_2^{(j)} = u_2^*$ after at most $n-m$ iterations; actually, a breakdown may occur at the $(n-m+1)$-st iteration. However, this is a "lucky breakdown", in the sense that u_2^* can be easily obtained starting from the last computed approximation of it, as shown in [4]. More generally, since it may happen that the 2-norm of the preconditioned CG (PCG) residual may not decrease as fast as the H-norm of the PCG error[1], a suitable scaling of the KKT system matrix can be used to prevent this situation [29].

In order to apply the preconditioner B, we can compute the square-root free Cholesky factorization of the negative Schur complement of E in H,

$$AE^{-1}A^T = LDL^T, \tag{14}$$

and then consider the block factorization

$$B = \begin{bmatrix} E & A^T \\ A & 0 \end{bmatrix} = \begin{bmatrix} I & 0 \\ AE^{-1} & I \end{bmatrix} \begin{bmatrix} E & 0 \\ 0 & -LDL^T \end{bmatrix} \begin{bmatrix} I & E^{-1}A^T \\ 0 & I \end{bmatrix}. \tag{15}$$

[1]It is easy to show that $(e^{(j)})^T H e^{(j)} = (e_1^{(j)})^T G e_1^{(j)}$, where $e^{(j)} = \begin{bmatrix} e_1^{(j)} \\ e_2^{(j)} \end{bmatrix}$ is the PCG error. Then, we can consider $\|e^{(j)}\|_H = \sqrt{(e^{(j)})^T H e^{(j)}}$, which we call H-norm of $e^{(j)}$ although $\|e^{(j)}\|_H = 0$ when $e_1^{(j)} = 0$ and $e_2^{(j)} \neq 0$.

Furthermore, by exploiting (15) we can easily obtain the following factorization of the inverse of B:

$$P = B^{-1} = \begin{bmatrix} I & -E^{-1}A^T \\ 0 & I \end{bmatrix} \begin{bmatrix} E^{-1} & 0 \\ 0 & -L^{-T}D^{-1}L^{-1} \end{bmatrix} \begin{bmatrix} I & 0 \\ -AE^{-1} & I \end{bmatrix}. \tag{16}$$

Henceforth, the inverse of a CP will be called inverse CP.

We note that the effectiveness of CPs may be hidden by the computational cost for the factorization of their Schur complements, thus reducing the efficiency of the overall IP procedure. The updating strategy proposed in the next section has been motivated by this issue.

3. MULTIPLE BFGS-LIKE UPDATES OF THE CONSTRAINT PRECONDITIONER

In order to avoid the factorization (14) at a certain IP iteration k, we construct a preconditioner for the KKT system at that iteration by updating a CP computed at an iteration $i < k$. As already observed, we extend to KKT systems and CPs the preconditioner updating technique for SPD matrices presented in [25], which in turn exploits ideas from [20, 21].

In the following, we use H, B and P defined in (6), (7) and (16) to denote the matrix of the KKT system, the associated CP and its inverse at iteration k, respectively. Likewise we use

$$\widehat{H} = \begin{bmatrix} \widehat{G} & A^T \\ A & 0 \end{bmatrix}, \quad \widehat{B} = \begin{bmatrix} \widehat{E} & A^T \\ A & 0 \end{bmatrix}, \quad \widehat{P} = \widehat{B}^{-1} \tag{17}$$

to denote the KKT matrix, the CP and its inverse at iteration i (the "seed matrices").

Let us consider a matrix

$$S = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \in \mathbb{R}^{(n+m)\times q}, \quad q \leq n - m \tag{18}$$

with $S_1 \in \mathbb{R}^{n\times q}$ such that

$$\operatorname{rank}(S_1) = q, \quad AS_1 = 0. \tag{19}$$

We first define a preconditioner for H by applying a BFGS-like rank-$2q$ update to $\widehat{B}$:

$$B_{upd} = \widehat{B} + HS(S^THS)^{-1}S^TH - \widehat{B}S(S^T\widehat{B}S)^{-1}S^T\widehat{B}, \tag{20}$$

which is well defined because $S^T\widehat{B}S = S_1^T\widehat{E}S_1$ and $S^THS = S_1^TGS_1$. By using the Sherman-Morrison-Woodbury inversion formula, we get the inverse of B_{upd}:

$$P_{upd} = B_{upd}^{-1} = S(S^THS)^{-1}S^T + (I - S(S^THS)^{-1}S^TH)\widehat{P}(I - HS(S^THS)^{-1}S^T), \tag{21}$$

which is analogous to the BFGS update discussed in [25] for SPD matrices.

In the next section we will focus on the practical choice of the matrix S. Here, we prove some properties of the updated preconditioner that hold regardless of the specific choice of S. To this end, we will consider either B_{upd} in (20) or P_{upd} in (21), while in the implementation of the PCG iterations we will use only P_{upd}, since it can be applied through a recursive procedure requiring simple matrix-vector products (see Section 5).

We first prove that the previous rank-$2q$ update allows the preconditioned matrix to have at least q eigenvalues equal to 1. In order to simplify the notations, we define the matrix

$$\mathscr{S} = SL_S^{-T}, \tag{22}$$

where L_S is the lower triangular Cholesky factor of the matrix S^THS.

Theorem 3.1

Let us consider the KKT system (5) and a seed inverse CP, $\widehat{P}$. Let S be the matrix defined in (18)-(19) and P_{upd} the matrix in (21). Then the columns of S are eigenvectors of $P_{upd}H$ with corresponding eigenvalues equal to 1.

Proof
By using $\mathscr{S}$, the matrix P_{upd} in (21) can be written as

$$P_{upd} = \mathscr{S}\mathscr{S}^T + (I - \mathscr{S}\mathscr{S}^T H)\widehat{P}(I - H\mathscr{S}\mathscr{S}^T). \tag{23}$$

Moreover, it turns out that

$$\mathscr{S}\mathscr{S}^T HS = SL_S^{-T}L_S^{-1}S^T HS = S(S^T HS)^{-1}S^T HS = S,$$

from which we get

$$P_{upd}HS = \mathscr{S}\mathscr{S}^T HS + (I - \mathscr{S}\mathscr{S}^T H)\widehat{P}(HS - H\mathscr{S}\mathscr{S}^T HS) = S. \tag{24}$$

□

We also prove that the rank-$2q$ update (20) (or, equivalently, (21)) produces a CP.

Theorem 3.2
The matrix B_{upd} given in (20) is a CP for the matrix H in (5).

Proof
We show that the update (20) involves only the $(1,1)$ block of $\widehat{B}$ and that B_{upd} is nonsingular; hence the thesis holds. Let us split $\mathscr{S}$ into two blocks:

$$\mathscr{S} = \begin{bmatrix} \mathscr{S}_1 \\ \mathscr{S}_2 \end{bmatrix}, \quad \mathscr{S}_1 \in \mathbb{R}^{n\times q}, \quad \mathscr{S}_2 \in \mathbb{R}^{m\times q}.$$

From (19) it follows that $A\mathscr{S}_1 = 0$. Then,

$$HSS^T H = \begin{bmatrix} G\mathscr{S}_1 + A^T\mathscr{S}_2 \\ 0 \end{bmatrix} \begin{bmatrix} \mathscr{S}_1^T G + \mathscr{S}_2^T A & 0 \end{bmatrix} = \begin{bmatrix} \Gamma & 0 \\ 0 & 0 \end{bmatrix},$$

where

$$\Gamma = (G\mathscr{S}_1 + A^T\mathscr{S}_2)(\mathscr{S}_1^T G + \mathscr{S}_2^T A).$$

Likewise, we have

$$\widehat{B}S_1(S^T\widehat{B}S)^{-1}S^T\widehat{B} = \begin{bmatrix} \Phi & 0 \\ 0 & 0 \end{bmatrix},$$

where

$$\Phi = (\widehat{E}S_1 + A^T S_2)(S_1^T\widehat{E}S_1)^{-1}(S_1^T\widehat{E} + S_2^T A).$$

It follows that

$$B_{upd} = \widehat{B} + H\mathscr{S}\mathscr{S}^T H - \widehat{B}S(S^T\widehat{B}S)^{-1}S^T\widehat{B} = \begin{bmatrix} \widehat{E} + \Gamma - \Phi & A^T \\ A & 0 \end{bmatrix}. \tag{25}$$

In order to prove that B_{upd} is nonsingular, we consider a matrix $Z \in \mathbb{R}^{n\times(n-m)}$ whose columns span the nullspace of A and prove that $Z^T(\widehat{E} + \Gamma - \Phi)Z$ is SPD (see, e.g., [3]). We observe that

$$\begin{aligned} Z^T(\widehat{E} + \Gamma - \Phi)Z &= Z^T\widehat{E}Z + Z^T(G\mathscr{S}_1 + A^T\mathscr{S}_2)(\mathscr{S}_1^T G + \mathscr{S}_2^T A)Z \\ &\quad - Z^T(\widehat{E}S_1 + A^T S_2)(S_1^T\widehat{E}S_1)^{-1}(S_1^T\widehat{E} + S_2^T A)Z \\ &= Z^T\widehat{E}Z + Z^T G\mathscr{S}_1\mathscr{S}_1^T GZ - Z^T\widehat{E}S_1(S_1^T\widehat{E}S_1)^{-1}S_1^T\widehat{E}Z \\ &= Z^T E_{upd}Z, \end{aligned} \tag{26}$$

where we set

$$E_{upd} = \widehat{E} + G\mathscr{S}_1\mathscr{S}_1^T G - \widehat{E}S_1(S_1^T\widehat{E}S_1)^{-1}S_1^T\widehat{E}. \tag{27}$$

Then, by using the Sherman-Morrison-Woodbury formula, we have

$$E_{upd}^{-1} = \mathscr{S}_1\mathscr{S}_1^T + (I - \mathscr{S}_1\mathscr{S}_1^T G)\widehat{E}^{-1}(I - G\mathscr{S}_1\mathscr{S}_1^T), \tag{28}$$

which implies that E_{upd} is SPD. This concludes the proof.

□

Now we are able to prove a result about the unit eigenvalues of the preconditioned matrix $P_{upd}H$, which, in view of Theorem 2.1, are those of

$$(Z^TGZ)w = \lambda(Z^T(\widehat{E} + \Gamma - \Phi)Z)w, \tag{29}$$

where $Z \in \mathbb{R}^{n\times(n-m)}$ spans the nullspace of A.

Theorem 3.3
Let H be the matrix in (5) and P_{upd} the matrix in (21). Then $P_{upd}H$ has an eigenvalue at 1 with multiplicity at least $2m + q$.

Proof
Since P_{upd} is an inverse CP, by Theorem 2.1 $P_{upd}H$ has $2m$ eigenvalues equal to 1 and its remaining $n - m$ eigenvalues are defined by the generalized eigenvalue problem (29). In order to conclude the proof, we show that this problem has at least q eigenvalues equal to 1.

From $AS_1 = 0$ it follows that $S_1 = ZW$, where $Z \in \mathbb{R}^{n\times(n-m)}$ spans the nullspace of A and $W \in \mathbb{R}^{(n-m)\times q}$. Furthermore, by (26) and (27), the generalized eigenvalue problem (29) is equivalent to

$$Z^TGZw = \lambda Z^TE_{upd}Zw. \tag{30}$$

We show that this problem has the solution pair $(1, w_i)$, where w_i is any column of W. The definition of $\mathscr{S}$ (see (22)) implies $\mathscr{S}_1\mathscr{S}_1^TGS_1 = S_1$, and hence we have

$$\begin{aligned} Z^TE_{upd}ZW = Z^TE_{upd}S_1 &= Z^T\widehat{E}S_1 + Z^TG\mathscr{S}_1\mathscr{S}_1^TGS_1 - Z^T\widehat{E}S_1(S_1^T\widehat{E}S_1)^{-1}S_1^T\widehat{E}S_1 \\ &= Z^T\widehat{E}S_1 + Z^TGS_1 - Z^T\widehat{E}S_1 \\ &= Z^TGS_1 = Z^TGZW, \end{aligned}$$

which proves that any column of W is an eigenvector of problem (30) corresponding to the eigenvalue 1. □

The next theorem shows that the nonunit extremal eigenvalues of $E_{upd}^{-1}G$ are bounded by the extremal eigenvalues of $\widehat{E}^{-1}G$. Thus, by Theorem 2.1, we expect the application of P_{upd} to H to yield better spectral properties than the application of $\widehat{P}$.

Theorem 3.4
Let E_{upd}, G and $\widehat{E}$ be the matrices in (27), (6) and (17), respectively. Then any eigenvalue of $E_{upd}^{-1}G$ satisfies

$$\min\left\{\lambda_{min}(\widehat{E}^{-1}G), 1\right\} \le \lambda(E_{upd}^{-1}G) \le \max\left\{\lambda_{max}(\widehat{E}^{-1}G), 1\right\}.$$

Proof
Since E_{upd}^{-1} has the form given in (28), where G and $\widehat{E}$ are SPD and $\mathscr{S}_1$ is full rank, we can use Theorem 3.4 in [25] to bound the eigenvalues of $E_{upd}^{-1}G$ in terms of the eigenvalues of $\widehat{E}^{-1}G$. Then, letting $\sigma_1, \sigma_2, \dots \sigma_n$ be the eigenvalues of $\widehat{E}^{-1}G$ sorted in nondecreasing order, we found that the eigenvalues $\lambda_1, \dots, \lambda_n$ of $E_{upd}^{-1}G$ can be divided into two groups as follows:

1. $\sigma_j \le \lambda_j \le \sigma_{j+q}$, for $j \in \{1, \dots, n-q\}$,
2. $\lambda_j = 1$, for $j \in \{n-q+1, \dots, n\}$.

This concludes the proof. □

In summary, thanks to the previous theorems, we expect the updating strategy to improve $\widehat{P}$; on the other hand, by Theorem 2.1, P_{upd} can be a reasonably good preconditioner as long as $\widehat{E} + \Gamma - \Phi$ in (25) is not too far from G. If this is not the case, the performance of CG may deteriorate significantly and a recomputation of the CP from scratch may be more appropriate. Furthermore, the number q of vectors forming the matrix S must be selected taking into account that larger values of q can provide preconditioners that are more effective, but also more expensive. Numerical experiments have shown that in practice $q \ll n$ must be considered to achieve a good tradeoff between effectiveness and computational cost.

4. CHOICE OF THE MATRIX S

Now we focus on the choice of the matrix S needed for building P_{upd} for the k-th KKT system (2)-(3) in the sequence under consideration. We first observe that by setting

$$u_k^{(0)} = Pd_k \tag{31}$$

as a starting guess for PCG, where P is any inverse CP used during PCG, we have $Au_{1,k}^{(0)} = d_{2,k}$, and thus the residual $r_k^{(0)}$ corresponding to $u_k^{(0)}$ takes the form specified in (12). Henceforth we assume that the starting guess is computed as in (31), using the preconditioner to be applied during PCG. Then, a natural choice is to set the columns of S equal to q directions produced by the PCG algorithm applied to the KKT system at the previous IP iteration (of course, we assume that q is smaller than the number of PCG iterations for solving that system). Thus, we build S as described next.

> **BFGS-P.** Let k and $k-1$ identify the current and previous IP iterations. During the solution of the $(k-1)$-st system $H_{k-1}u_{k-1} = d_{k-1}$ by PCG, we store the first q H_{k-1}-conjugate directions $\{p^{(0)}, p^{(1)}, \ldots, p^{(q-1)}\}$ after normalizing them as $p^{(j)}/\sqrt{(p^{(j)})^T H_{k-1} p^{(j)}}$ (we neglect the index $k-1$ in $p^{(j)}$ in order to simplify the notations). The matrix S used to build P_{upd} for the KKT system at the k-th iteration is obtained by setting its columns equal to the normalized directions.

Note that the preconditioner used for the solution of the $(k-1)$-st KKT system can be any CP, e.g., a CP computed from scratch for the $(k-1)$-st system, a CP computed for a previous system, or a CP obtained by updating a given seed CP. Note also that the suffix -P in BFGS-P refers to the fact that S is built by using PCG directions coming from the previous KKT system.

Since we have experimentally verified that the PCG directions can rapidly lose the property of being mutually H_{k-1}-orthogonal when H_{k-1} is highly ill conditioned, we perform a selective reorthogonalization of the directions forming S, during their computation. More precisely, $p^{(i)}$ is H_{k-1}-reorthogonalized against $p^{(l)}$, with $l < i$, if

$$\left| (p^{(i)})^T H_{k-1} p^{(l)} \right| > \delta, \tag{32}$$

where δ is a small tolerance. Note that during the solution of $H_{k-1}u_{k-1} = d_{k-1}$ the PCG algorithm uses the orthogonalized directions as soon as they are computed.

We also considered other ways of selecting q directions among the ones generated by PCG at the $(k-1)$-st IP iteration, e.g., choosing the directions $p^{(jt)}$, where $t = \lfloor nit/(q-1) \rfloor$, nit is the number of PCG iterations performed on the $(k-1)$-st KKT system, and $j = 0, \ldots, q-1$ (normalization and reorthogonalization of the directions were applied too). However, we did not observe any improvement by performing numerical experiments with this choice.

We now present a different choice of S, which deserves a detailed explanation.

> **BFGS-C.** We first apply q iterations of PCG to $H_k u_k = d_k$, using a seed preconditioner $\widehat{P}$, and collect the corresponding normalized directions $p^{(j)}/\sqrt{(p^{(j)})^T H_k p^{(j)}}$ as columns of S. Then we restart PCG from the last computed iterate $u_k^{(q)}$, with the preconditioner P_{upd} built by updating $\widehat{P}$ through S. In this case
>
> $$S^T H_k S = I,$$
>
> which implies $\mathscr{S} = S$.

The suffix -C in BFGS-C refers to the fact that PCG directions from the current system are used to define S. As for BFGS-P, a selective H_k-reorthogonalization of the directions is performed in the first q PCG iterations.

It is easy to show (see Appendix A) that in exact arithmetic the BFGS-C procedure is equivalent to PCG with preconditioner $\widehat{P}$. Thus, BFGS-C may appear completely useless. Nevertheless, it is useful in finite-precision arithmetic. Freezing the preconditioner $\widehat{P}$ computed at a certain IP iteration and using it in subsequent IP iterations yields directions which rapidly and dramatically lose orthogonality. BFGS-C appears to mitigate this behaviour, improving the performance of PCG.

In order to illustrate this situation, we discuss some numerical results obtained with one of the 35 KKT systems arising in the solution, by an IP procedure, of the CVXQP3 convex QP problem from the CUTEst collection [30], with dimensions $n = 20000$ and $m = 15000$ (see Section 6 for the details). We considered PCG with the following preconditioning procedures:

- BFGS-C with $q = 50$ and seed CP recomputed every sixth IP iteration;
- CP recomputed from scratch every sixth IP iteration and frozen in the subsequent five iterations (henceforth this is referred to as **FIXED** preconditioning procedure).

In order to make a fair comparison, we performed a selective reorthogonalization of the first 50 directions during the execution of PCG with FIXED preconditioning. In this case, as well as in the execution of the BFGS-C procedure, we used $\delta = 10^{-12}$. We also run PCG with FIXED preconditioning without any reorthogonalization. We focus on the KKT system at the 24-th IP iteration, hence the seed preconditioner comes from the 19-th IP iteration.

Figure 1 shows the normalized scalar products

$$\frac{(p^{(j)})^T H p^{(l)}}{\|p^{(j)}\| \|H p^{(l)}\|}, \quad l = 0, 25, 50, \ j > l, \tag{33}$$

for BFGS-C and both versions of the FIXED procedure. In the latter case, we observe a quick loss of orthogonality with respect to the first q directions, even when the reorthogonalization procedure is applied. Conversely, BFGS-C appears to better preserve orthogonality. As a consequence, the number of PCG iterations corresponding to BFGS-C is smaller than in the other cases (122 iterations for BFGS-C vs 176 and 196 for FIXED with and without reorthogonalization, respectively).

We also applied the FIXED preconditioning approach with a selective reorthogonalization of each PCG direction with respect to the first 50 ones, and with respect to the directions of the 50 PCG iterations preceding the current one. In neither case we achieved a better PCG behaviour than the one obtained with the BFGS-C strategy. Furthermore, these variants of the FIXED procedure generally resulted less effective than the FIXED one with selective reorthogonalization of the first q directions. We believe that the behaviour of BFGS-C preconditioning vs the different versions of the FIXED one deserves deeper investigation, however it is beyond the scope of this paper.

Motivated by the previous observations, we consider a further preconditioning procedure.

> **DOUBLE.** We apply q PCG iterations to the current system $H_k u_k = d_k$ by using a CP, say $P^{(0)}_{upd}$, built with the BFGS-P procedure, i.e., by updating a seed preconditioner $\widehat{P}$ with the first q normalized PCG directions obtained at the $(k-1)$-st IP iteration. Then we restart PCG from the last computed iterate $u_k^{(q)}$, with the following preconditioner:
>
> $$P_{upd} = S^C (S^C)^T + (I - S^C (S^C)^T H_k) P^{(0)}_{upd} (I - H_k S^C (S^C)^T), \tag{34}$$
>
> where S^C contains the normalized directions computed in the first q PCG iterations.

As for the previous preconditioning procedures, a selective reorthogonalization is applied to the PCG directions used to build $P^{(0)}_{upd}$ and to those used for P_{upd}.

5. IMPLEMENTATION DETAILS

The preconditioning procedures described in the previous section are applied to the sequence of KKT systems as explained next. The CP preconditioner $\widehat{P}$ is computed from scratch every k-th IP iteration such that $k \bmod s = 0$, where s is a predefined positive integer. At IP iterations

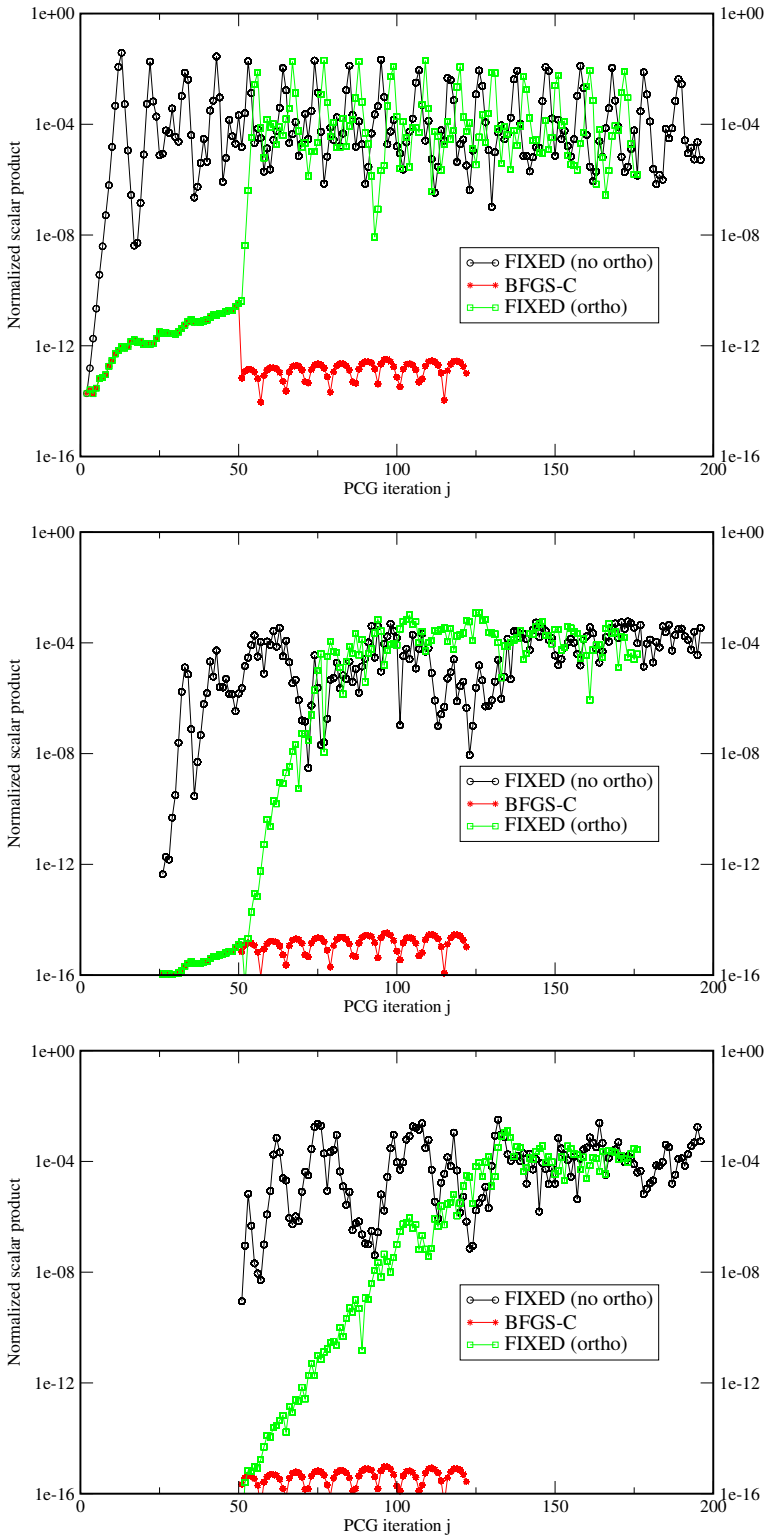

Figure 1. CVXQP3 test problem, KKT system at the 24-th IP iteration: normalized scalar products (33) for the BFGS-C and FIXED procedures (the latter with and without orthogonalization). Top: $l = 0$, middle: $l = 25$, bottom: $l = 50$.

Algorithm 1 (Application of P_{upd} within the PCG solver: $w = P_{upd}\, r$)

1. $v_1 = S^T r$
2. a. Solve $L_S z_1 = v_1$
   b. Solve $L_S^T z_2 = z_1$
   c. $v_2 = r - S_H z_2$
3. $v_2 = \widehat{P} v_2$
4. $v_1 = v_1 - S_H^T v_2$
5. a. Solve $L_S z_1 = v_1$
   b. Solve $L_S^T z_2 = z_1$
   c. $w = v_2 + S z_2$

$k+1, k+2, \ldots, k+s-1$ the preconditioner P_{upd} is not built explicitly, but its application within PCG is performed as shown in Algorithm 1. Note that H refers to the matrix appearing in the definition of P_{upd} for the selected procedure, i.e., either H_{k-1} or H_k. Of course, with the DOUBLE procedure, Algorithm 1 is used to apply both $P_{upd}^{(0)}$ and P_{upd} (with $P_{upd}^{(0)}$ instead of $\widehat{P}$ when P_{upd} is considered). The algorithm assumes that

$$S_H = HS$$

and the Cholesky factorization

$$S^T S_H = L_S L_S^T \tag{35}$$

have been precomputed.

Apart from step 3, the main cost of the algorithm comes from the computation of the matrix-vector products involving S or S_H, which is roughly $2q(n+m) + 6qn$ floating-point operations, since the last m rows of S_H are zero, as well as the last m components of the residual r, and $q \ll n$ is used in practice. The matrix-vector products can be efficiently computed by using BLAS 2 routines.

We observe that Algorithm 1 is simplified when BFGS-C is considered, since $S^T S_H = I$ and the triangular solves are not required. Furhermore the computation of the columns of S_H is obtained as a byproduct of the application of PCG to the current KKT system.

When S comes from the solution of the previous KKT system, as for BFGS-P, the computation of S_H can be carried out using the following formula:

$$S_H = H_k S_{k-1} = (H_{k-1} + H_k - H_{k-1}) S_{k-1} = (S_H)_{k-1} + (\Theta_k - \Theta_{k-1})(S_1)_{k-1},$$

where S_{k-1} has as columns the normalized PCG directions computed at the $(k-1)$-st IP iteration, $(S_H)_{k-1} = H_{k-1} S_{k-1}$ (which is a byproduct of PCG applied to $H_{k-1} u_{k-1} = d_{k-1}$), $(S_1)_{k-1}$ consists of the first n rows of S_{k-1}, and Θ_k is defined in (4). Thus, the computation of S_H requires about $2qn$ floating-point operations. The cost for computing $S^T S_H$ is about $q^2 n$, by exploiting the symmetry of the matrix and the fact that the last m rows of S_H are zero, while the cost of the Cholesky factorization (35) is about $q^3/3$, which is small compared with $q^2 n$, if $q \ll n$.

In the DOUBLE procedure, the first q iterations have the same cost as q iterations performed with P_{upd} within BFGS-P, while the remaining iterations have the same cost as q iterations performed with P_{upd} within BFGS-C.

Finally, for all the updating procedures, the cost of the reorthogonalization is roughly $q^2 n/2$ floating-point operations in the worst case, i.e., if each of the q conjugate directions must be reorthogonalized against all the previous ones.

Table I. Characteristics of the test problems.

problem	n	m	$nz(A)$	$nz(AE^{-1}A^T)$	$nz(L)$
CVXQP3	20000	15000	44997	85472	869197
CVXQP3N	20000	15000	104983	278649	94015382
STCQP2	16385	8190	61425	61425	123046
STCQP2N	16385	8190	384907	19217794	141980157

6. COMPUTATIONAL RESULTS

In order to illustrate the behaviour of the preconditioning procedures described in the previous sections, we show the results obtained by applying them to the sequences of KKT systems arising in the solution of four convex QP problems. Two of these problems (CVXQP3 and STCQP2) come from the CUTEst collection, while the others have been obtained from the previous ones by adding nonzero entries to the matrix A in (6), in order to increase the number of nonzero entries of the Schur complement $AE^{-1}A^T$ of the CP preconditioner (7) (in our experience, $AE^{-1}A^T$ is inexpensive to factorize for the CUTEst convex QP problems). The new problems are denoted by CVXQP3N and STCQP2N, respectively. For each problem, the dimensions n and m, and the number of nonzero entries of A, of the CP Schur complement and of its Cholesky factor L (see 14) are given in Table I.

The sequences of KKT systems have been obtained by running the Fortran 95 PRQP code, which implements an infeasible inexact potential reduction IP method [8, 31, 3], and extracting the KKT matrices arising at each IP iteration and the corresponding right-hand sides. Afterwards, these sequences have been solved offline, by using PCG with the various preconditioning procedures, implemented as explained in Section 5. The starting point for the PRQP solver has been built with the STP2 algorithm described in [32], and the tolerances on the relative duality gap and the relative infeasibilities have been set to 10^{-6} and 10^{-7}, respectively. Within PRQP, each KKT system has been solved by the PCG algorithm with CP recomputed from scratch, with an adaptive stopping criterion that relates the accuracy in the solution of the KKT system to the quality of the current IP iterate. More precisely, the PCG iterations have been stopped as soon as

$$\|r^{(j)}\| \leq \tau, \quad \tau = \min\{\max\{\tau_1, 10^{-8}\}, 10^{-2}\|r^{(0)}\|\},$$

where τ_1 depends on the duality gap value at the current IP iteration (see [33] for the details). In the experiments reported in this section, the same stopping criterion has been applied, using for each system the value of τ computed at the corresponding IP iteration. A maximum number of 600 PCG iterations has been considered too, declaring a failure if the stopping criterion was not satified within this number. The orthogonalization threshold δ in (32) has been fixed to 10^{-12} for the FIXED and BFGS-C procedures, and to 0 for the BFGS-P and DOUBLE ones (these values of δ have been chosen by numerical experiments).

The preconditioning procedures BFGS-P, BFGS-C and DOUBLE have been applied with different values of s and q. The value of q has been dynamically defined by choosing a maximum value $q_{\max}$ for q and setting

$$q = \min\{q_{max}, nit_{prev}\},$$

where nit_{prev} is equal to the number of PCG iterations for solving the previous KKT system in the sequence. The experiments have been performed with $2 \leq s \leq 9$ and $q_{max} = 5, 10, 20, 50, 100$; however, we report only a selection of the results, which clearly shows the behaviour of the preconditioners. In the following, we use the notations BFGS-P(s, q_{max}), BFGS-C(s, q_{max}) and DOUBLE(s, q_{max}) to highlight the parameters s and q_{max} of the preconditioning procedures. E.g., BFGS-P(5,20) indicates that BFGS-P is used with $s = 5$ and $q_{max} = 20$. For comparison purposes, the FIXED procedure, consisting in recomputing the preconditioner from scratch every s-th IP iteration and in reusing it in the next $s - 1$ IP iterations, has been applied too. We write FIXED(s, q_{max}) to indicate the value of s used and to emphasize that a selective reorthogonalization of the first q PCG directions is carried out in the solution of each system, with q defined by using

Table II. Results for problem CVXQP3 (number of KKT systems in the sequence: 35).

Prec	s	q_{max}	PGC iters	Tf-Schur	Ta-seed	Tupd	Ttot
RECOM	–	—	489	2.34	1.60	–	4.89
FIXED	2	0	755	1.20	2.26	–	4.47
	2	5	752	1.19	2.24	0.03	4.50
	2	10	741	1.19	2.23	0.09	4.54
	2	20	733	1.19	2.23	0.17	4.63
BFGS-P	2	5	742	1.20	2.27	0.32	4.87
	2	10	696	1.21	2.16	0.60	5.02
	2	20	674	1.20	2.12	0.99	5.34
BFGS-C	2	5	702	1.21	2.11	0.16	4.52
	2	10	700	1.20	2.15	0.28	4.67
	2	20	700	1.18	2.15	0.40	4.76
DOUBLE	2	5	695	1.23	2.24	0.50	5.00
	2	10	665	1.26	2.16	0.90	5.29
	2	20	662	1.24	2.16	1.26	5.66
FIXED	3	0	1051	0.80	3.07	–	5.17
	3	5	1048	0.80	3.08	0.03	5.20
	3	10	1026	0.80	3.04	0.09	5.21
	3	20	1006	0.79	3.00	0.22	5.28
BFGS-P	3	5	1012	0.80	3.06	0.53	5.67
	3	10	951	0.80	2.93	1.03	6.00
	3	20	886	0.79	2.74	1.86	6.60
BFGS-C	3	5	951	0.81	2.84	0.28	5.15
	3	10	931	0.80	2.83	0.50	5.34
	3	20	926	0.79	2.84	0.76	5.61
DOUBLE	3	5	919	0.82	2.91	0.84	5.73
	3	10	876	0.81	2.72	1.51	6.05
	3	20	846	0.79	2.64	2.38	6.78

q_{max}. Note that FIXED$(s, 0)$ corresponds to no reorthogonalization. Finally, the PCG solver has been also applied with the CP recomputed from scratch for each KKT system (RECOM).

All the algorithms have been implemented in Fortran 95. The code has been run on an Intel Core i7 CPU (2.67 GHz) with 6 GB RAM and 8 MB cache, Linux O.S. and the gfortran compiler (GNU Fortran v. 4.8.4) with the -O4 option. The factorization of the Schur complement has been performed by the MA57 routine from HSL Mathematical Software Library (http://www.hsl.rl.ac.uk).

In Table II we report some results concerning the application of the preconditioning procedures, including the FIXED one, to CVXQP3: the cumulative number of PCG iterations (PGC iters), the execution times (in seconds) for the factorization of the Schur complement needed to construct the CP from scratch (Tf-Schur), the CPU time needed for the application of the seed preconditioner $\widehat{P}$ during the PCG iterations (Ta-seed), the times for the preconditioner updates and the reorthogonalization steps (Tupd), and the total times (Ttot). The same data are shown also for the case where the CP is recomputed from scratch for each KKT system (RECOM). We do not report the remaining PCG time, which is a small percentage of the total execution time. It can be inferred by difference between the total time and the sum of Tf-Schur, Ta-seed and Tupd.

In this case, because of the high sparsity of both the Schur complement and its Cholesky factor, the factorization of a CP from scratch and its application is relatively cheap. Therefore, only a very modest gain can be obtained in terms of execution time by using the updating procedures, choosing small values of q_{max} and s. The smallest times are obtained with FIXED(2,0), although the smallest number of iterations corresponds to DOUBLE(2,20); using higher values of s yields an increase of the number of iterations, which is not offset by the reduction of the number of Schur complement factorizations.

Table III. Results for problem CVXQP3N (number of KKT systems in the sequence: 36).

Prec	s	q_{max}	PGC iters	Tf-Schur	Ta-seed	Tupd	Ttot
RECOM	–	—	513	2421.51	46.57	–	2469.18
FIXED	6	0	3005	403.75	258.20	–	665.58
	6	20	2931	403.83	251.26	0.28	658.92
	6	50	2724	403.77	233.85	1.05	642.02
BFGS-P	6	20	2350	404.40	202.80	5.49	615.20
	6	50	2053	404.04	177.44	11.21	594.95
BFGS-C	6	20	2223	403.89	193.07	2.93	602.81
	6	50	2132	404.06	185.26	5.06	597.21
DOUBLE	6	20	1959	404.16	171.36	6.29	584.49
	6	50	1796	403.54	155.93	12.25	575.19
FIXED	8	0	3944	337.02	339.72	–	681.33
	8	20	3871	336.91	331.36	0.30	673.04
	8	50	3613	336.96	309.58	1.23	651.95
BFGS-P	8	20	3111	337.42	270.49	7.34	618.56
	8	50	failure on system 34				
BFGS-C	8	20	2848	336.52	246.34	4.05	590.42
	8	50	2663	336.55	230.47	7.22	577.60
DOUBLE	8	20	2489	336.85	216.70	8.20	564.42
	8	50	failure on system 34				

The situation changes for problem CVXQP3N, as shown in Table III (the reported values of s and q_{max} are among the best choices). Here the factorization of the Schur complement is rather expensive, and the recomputation of the CP from scratch produces by far the largest execution time, even if it yields a much smaller number of PCG iterations than the other preconditioning procedures. Furthermore, the updating procedures generally produce a significant reduction of the number of iterations with respect to the FIXED one, and hence smaller execution times. BFGS-C and DOUBLE turn out to be the most efficient procedures, although a failure occurs on one of the KKT systems by using BFGS-P(8,50) and DOUBLE(8,50) (more details on the failure are given later in this section).

In order to perform a deeper analysis, in Table IV we report, for the same problem, the number of PCG iterations obtained at selected IP iterations, with all the preconditioners. As expected, the effectiveness of the updating procedures increases with the distance of the current IP iteration from the one where the preconditioner $\widehat{P}$ has been computed. This behaviour is more noticeable as the IP method proceeds toward the solution of QP problem, because of the increasing ill conditioning of the KKT matrix and the increasing accuracy requirement determined by the adaptive tolerance τ.

The PCG convergence histories for the KKT systems at the 24-th and the 32-nd IP iteration, with the updating procedures and the FIXED ones for $s = 8$ and $q_{max} = 50$, clearly show how each procedure compares with the others in terms of PCG iterations (see Figures 2-3): the best preconditioning procedure is DOUBLE, followed by BFGS-P, BFGS-C and then FIXED. This is a general behaviour, although failures have been observed with BFGS-P and DOUBLE in cases where BFGS-C and FIXED work.

Figure 4 displays the history of the residual norm corresponding to the failure of BFGS-P(8,50) at the 34-th IP iteration. The residual norm decreases up to about 1.9×10^{-6} at the 38-th PCG iteration, and then it keeps increasing, without being able to reach the tolerance τ, which is 10^{-6} in this case (nevertheless, a reduction of the residual norm of about 12 orders of magnitude is obtained). Note that BFGS-P(8,20) is able to satify the stopping criterion, thus showing that using a large number of directions coming from the previous KKT system may not be beneficial in the last IP iterations. The behaviour of the residual norm is about the same with DOUBLE(8,50).

Table IV. Problem CVXQP3DN: number of PCG iterations at IP iterations 17 to 36, for all the preconditioning procedures (the data corresponding to the recomputation of the CP are in bold). BFGS-P and DOUBLE fail at the 34-th IP iteration and the updates for the next KKT systems are not computed.

IPit	RECOM	FIXED			BFGS-C		BFGS-P		DOUBLE	
		0	20	50	20	50	20	50	20	50
					.....					
17	**13**	**13**	**13**	**13**	**13**	**13**	**13**	**13**	**13**	**13**
18	13	30	28	28	25	25	22	22	22	22
19	14	44	40	38	36	36	34	30	31	30
20	14	70	66	61	52	52	58	46	50	46
21	14	103	102	86	75	73	76	59	67	58
22	14	145	144	124	98	92	118	86	92	76
23	14	255	254	237	179	156	208	166	161	131
24	14	391	390	375	257	216	300	259	230	178
25	**14**	**14**	**14**	**14**	**14**	**14**	**14**	**14**	**14**	**14**
26	14	30	29	29	30	30	27	27	27	27
27	13	48	44	40	38	38	41	36	37	36
28	15	75	73	68	61	61	61	48	52	48
29	15	122	115	96	81	78	103	77	77	67
30	28	216	211	195	147	138	163	137	127	111
31	28	250	248	233	172	161	195	184	155	146
32	28	366	364	352	273	239	299	267	240	209
33	**27**	**27**	**27**	**27**	**27**	**27**	**27**	**27**	**27**	**27**
34	28	57	53	52	46	46	47	—	41	—
35	28	92	88	76	67	67	77	—	61	—
36	28	184	179	159	114	110	158	—	110	—

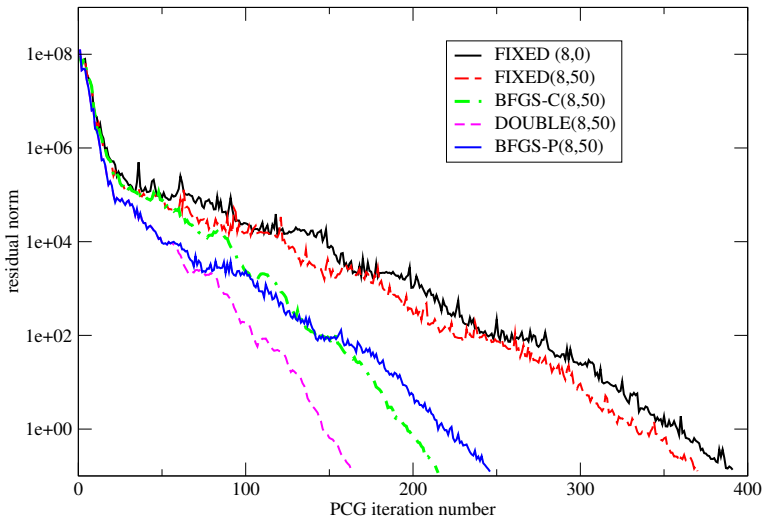

Figure 2. Problem CVXQP3N, KKT system at the 24-th IP iteration: convergence histories (residual norm vs PCG iteration) for the various preconditioning procedures, with $s = 8$ and $q_{\max} = 50$. The seed preconditioner $\widehat{P}$ comes from the 17-th IP iteration.

Finally in Tables V and VI we report the results for problems STCQP2 and STCQP2N, with suitable values of s and q_{max}. As is the case with CVXQP3, the extreme sparsity of the Cholesky factor of the Schur complement explains why the updating procedures are not efficient on STCQP2. On the contrary, STCQP2N has a much denser Schur complement and the updating procedures

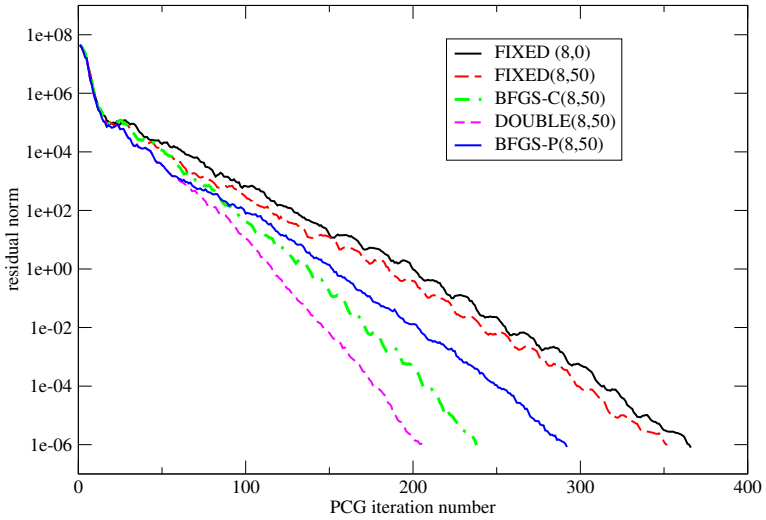

Figure 3. Problem CVXQP3N, KKT system at the 32-th IP iteration: convergence histories (residual norm vs PCG iteration) for the various preconditioning procedures with $s = 8$ and $q_{\max} = 50$. The seed preconditioner $\widehat{P}$ comes from the 25-th IP iteration.

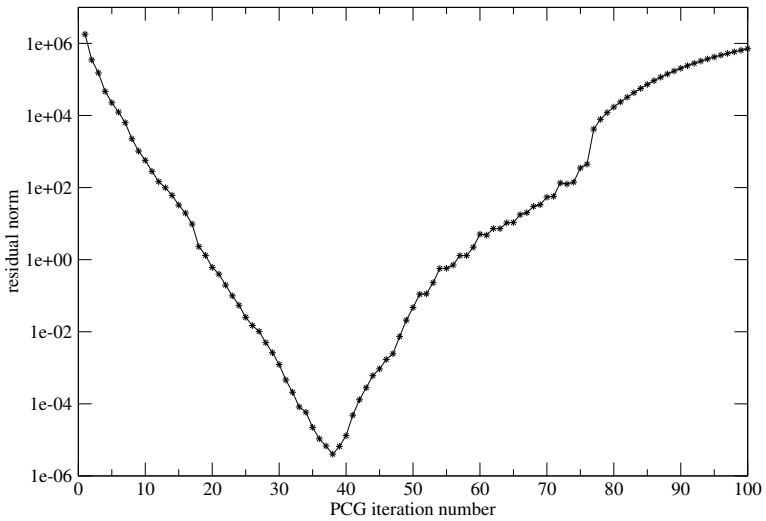

Figure 4. CVXQP3N, KKT system at the 34-th IP iteration: divergence of the residual norm with BFGS-P(8,50).

significantly reduce the execution time. Again BFGS-C and DOUBLE are the most effective procedures, irrespective of the value of $q_{\max}$. No failure has been reported for this test case.

7. CONCLUSIONS

We have analyzed a general technique for updating CPs in the solution of sequences of KKT systems arising in IP methods for convex QP problems. Our update extends the limited-memory preconditioners proposed in [25], by exploiting specific features of KKT systems and CPs. The updated preconditioners, computed through a rank-$2q$ BFGS-like correction of a seed preconditioner, still belong to the class of exact CPs and hence allow the use of the CG method. Theoretical results show that, compared with the CPs that are usually built from scratch, the updated

Table V. Results for problem STCQP2 (number of KKT systems in the sequence: 12).

Prec	s	q_{max}	PGC iters	Tf-Schur	Ta-seed	Tupd	Ttot
RECOM	–	—	209	0.18	0.19	–	0.73
FIXED	2	0	398	0.09	0.36	–	0.93
	2	5	397	0.08	0.37	0.01	0.96
	2	10	399	0.09	0.36	0.03	0.97
	2	20	397	0.09	0.36	0.07	1.03
BFGS-P	2	5	387	0.09	0.32	0.13	1.03
	2	10	389	0.09	0.32	0.28	1.19
	2	20	383	0.09	0.32	0.52	1.44
BFGS-C	2	5	394	0.09	0.32	0.07	0.97
	2	10	384	0.09	0.32	0.14	1.03
	2	20	371	0.08	0.31	0.24	1.11
DOUBLE	2	5	378	0.09	0.31	0.22	1.08
	2	10	374	0.09	0.32	0.46	1.29
	2	20	364	0.09	0.31	0.80	1.59

Table VI. Results for problem STCQP2N (number of KKT systems in the sequence: 12).

Prec	s	q_{max}	PGC iters	Tf-Schur	Ta-seed	Tupd	Ttot
RECOM	–	—	209	823.16	15.59	–	839.16
FIXED	7	0	1977	137.55	140.64	–	280.40
	7	20	1974	137.57	140.69	0.08	280.55
	7	50	1986	137.54	141.30	0.38	281.36
	7	100	1968	137.54	140.13	0.76	280.64
BFGS-P	7	20	1925	137.38	137.95	3.27	281.41
	7	50	1890	137.31	135.42	7.95	283.47
	7	100	1705	137.34	122.32	13.64	275.87
BFGS-C	7	20	1866	137.42	133.68	2.06	275.31
	7	50	1758	137.53	125.93	4.51	270.02
	7	100	1593	137.50	114.16	6.58	260.14
DOUBLE	7	20	1926	137.39	138.04	4.93	283.18
	7	50	1687	137.38	121.03	10.11	271.05
	7	100	1358	137.43	97.56	13.99	251.09

CPs have the property of clustering q more eigenvalues at 1. Furthermore, they provide better bounds on the nonunit eigenvalues of the preconditioned matrix than the corresponding seed preconditioner.

Different procedures that fit the general CP updating strategy have been considered. They differ by the choice of the matrix S used to apply the BFGS-like update to the seed preconditioner. Numerical experiments have shown that these procedures are able to reduce the time for the solution of the sequence of KKT systems when the cost for the factorization of the Schur complement is high.

Among the various updating procedures, the one called BFGS-C is equivalent, in exact arithmetic, to applying PCG with the seed preconditioner, but appears to produce better results in finite precision arithmetic. In particular, it reduces the loss of orthogonality of the PCG directions observed when PCG is used with the seed preconditioner. This behaviour deserves further investigation and will be studied in future work. The extension of our updating strategy to KKT systems with nonzero $(2, 2)$ block will be also considered.

ACKNOWLEDGEMENTS

The authors would like to thank Serge Gratton and Valeria Simoncini for interesting discussions on the

behaviour of PCG with the BFGS-C preconditioning procedure vs PCG with the FIXED one, in finite precision arithmetic.

REFERENCES

1. Wright SJ. *Primal-Dual Interior-Point Methods*. SIAM: Philadelphia, PA, 1997.
2. Gondzio J. Interior point methods 25 years later. *European J. Oper. Res.* 2012; **218**(3):587–601.
3. D'Apuzzo M, De Simone V, di Serafino D. On mutual impact of numerical linear algebra and large-scale optimization with focus on interior point methods. *Comput. Optim. Appl.* 2010; **45**(2):283–310.
4. Lukšan L, Vlček J. Indefinitely preconditioned inexact Newton method for large sparse equality constrained nonlinear programming problems. *Numer. Linear Algebra Appl.* 1998; **5**:219–247.
5. Golub GH, Wathen AJ. An iteration for indefinite systems and its application to the Navier-Stokes equations. *SIAM J. Sci. Comput.* 1998; **19**:530–539.
6. Keller C, Gould NIM, Wathen AJ. Constraint preconditioning for indefinite linear systems. *SIAM J. Matrix Anal. Appl.* 2000; **21**:1300–1317.
7. Bergamaschi L, Gondzio J, Zilli G. Preconditioning indefinite systems in interior point methods for optimization. *Comput. Optim. Appl.* 2004; **28**(2):149–171.
8. Cafieri S, D'Apuzzo M, De Simone V, di Serafino D. On the iterative solution of KKT systems in potential reduction software for large-scale quadratic problems. *Comput. Optim. Appl.* 2007; **38**(1):27–45.
9. Benzi M, Golub GH, Liesen J. Numerical solution of saddle point problems. *Acta Numer.* 2005; **14**:1–137.
10. Perugia I, Simoncini V. Block-diagonal and indefinite symmetric preconditioners for mixed finite element formulations. *Numer. Linear Algebra Appl.* 2000; **7**(7-8):585–616.
11. Durazzi C, Ruggiero V. Indefinitely preconditioned conjugate gradient method for large sparse equality and inequality constrained quadratic problems. *Numer. Linear Algebra Appl.* 2003; **10**(8):673–688.
12. Bergamaschi L, Gondzio J, Venturin M, Zilli G. Inexact constraint preconditioners for linear systems arising in interior point methods. *Comput. Optim. Appl.* 2007; **36**(2-3):137–147.
13. Sesana D, Simoncini V. Spectral analysis of inexact constraint preconditioning for symmetric saddle point matrices. *Linear Algebra Appl.* 2013; **438**(6):2683–2700.
14. Freund RW, Nachtigal NM. Software for simplified Lanczos and QMR algorithms. *Appl. Numer. Math.* 1995; **19**:319–341.
15. Bellavia S, De Simone V, di Serafino D, Morini B. Updating constraint preconditioners for KKT systems in quadratic programming via low-rank corrections. *SIAM J. Optim.* 2015; **25**(3):1787–1808.
16. Bellavia S, De Simone V, di Serafino D, Morini B. On the update of constraint preconditioners for regularized KKT systems. *Comput. Optim. Appl.* 2016; **65**(2):339–360.
17. Fisher M, Gratton S, Gürol S, Trémolet Y, Vasseur X. Low rank updates in preconditioning the saddle point systems arising from data assimilation problems. *Technical Report TR/PA/16/135*, CERFACS 2016.
18. Bellavia S, De Simone V, di Serafino D, Morini B. Efficient preconditioner updates for shifted linear systems. *SIAM J. Sci. Comput.* 2011; **33**(4):1785–1809.
19. Bellavia S, De Simone V, di Serafino D, Morini B. A preconditioning framework for sequences of diagonally modified linear systems arising in optimization. *SIAM J. Numer. Anal.* 2012; **50**(6):3280–3302.
20. Schnabel RB. Quasi-Newton methods using multiple secant equations. *Technical Report CU-CS-247-83*, Department of Computer Sciences, University of Colorado, Boulder, CO 1983.
21. Nash SG, Nocedal J. A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization. *SIAM J. Optim.* 1991; **1**(3):358–372.
22. Morales JL, Nocedal J. Automatic preconditioning by limited memory quasi-Newton updating. *SIAM J. Optim.* 2000; **10**(4):1079–1096.
23. Bergamaschi L, Bru R, Martínez A, Putti M. Quasi-Newton preconditioners for the inexact Newton method. *Electron. Trans. Numer. Anal.* 2006; **23**:76–87.
24. Bergamaschi L, Bru R, Martínez A. Low-rank update of preconditioners for the inexact Newton method with SPD Jacobian. *Math. Comput. Model.* 2011; **54**(7–8):1863–1873.
25. Gratton S, Sartenaer A, Tshimanga J. On a class of limited memory preconditioners for large scale linear systems with multiple right-hand sides. *SIAM J. Optim.* 2011; **21**(3):912–935.
26. Gower RM, Gondzio J. Action constrained quasi-Newton methods 2014. ArXiv:1412.8045v1 [math.OC].
27. Gratton S, Mercier S, Tardieu N, Vasseur X. Limited memory preconditioners for symmetric indefinite problems with application to structural mechanics. *Numer. Linear Algebra Appl.* 2016; doi:10.1002/nla.2058.
28. Dollar H, Wathen A. Approximate factorization constraint preconditioners for saddle-point matrices. *SIAM J. Sci. Comput.* 2006; **27**(5):1555–1572.
29. Rozložník M, Simoncini V. Krylov subspace methods for saddle point problems with indefinite preconditioning. *SIAM J. Matrix Anal. Appl.* 2002; **24**(2):368–391.
30. Gould NIM, Orban D, Toint PL. Cutest: a constrained and unconstrained testing environment with safe threads for mathematical optimization. *Comput. Optim. Appl.* 2015; **60**(3):545—557.
31. Cafieri S, D'Apuzzo M, De Simone V, di Serafino D, Toraldo G. Convergence analysis of an inexact potential reduction method for convex quadratic programming. *J. Optim. Theory Appl.* 2007; **135**(3):355–366.
32. D'Apuzzo M, De Simone V, di Serafino D. Starting-point strategies for an infeasible potential reduction method. *Optim. Lett.* 2010; **4**(1):131–146.
33. Cafieri S, D'Apuzzo M, De Simone V, di Serafino D. Stopping criteria for inner iterations in inexact potential reduction methods: a computational study. *Comput. Optim. Appl.* 2007; **36**(2–3):165–193.
34. Nazareth L. A relationship between the BFGS and conjugate gradient algorithms and its implications for new algorithm. *sinum* 1979; **16**(5):794–800.

A. THEORETICAL EQUIVALENCE BETWEEN BFGS-C AND SEED PRECONDITIONING

We show that (in exact arithmetic) PCG with the BFGS-C procedure is equivalent to PCG using the seed preconditioner of BFGS-C. For the sake of clarity, we first report the computations performed at each iteration of the PCG algorithm applied to system (5) with any CP, say P, that approximates H^{-1}. Letting $u^{(0)} = Pd$ (see the beginning of Section 4), $r^{(0)} = d - Hu^{(0)}$, $\beta^{(0)} = 0$ and $p^{(-1)} = 0$, the j-th PCG iteration, with $j \geq 0$, can be described as follows:

$$\begin{aligned} p^{(j)} &= Pr^{(j)} + \beta^{(j)} p^{(j-1)}, \\ \alpha^{(j)} &= \frac{(r^{(j)})^T Pr^{(j)}}{(p^{(j)})^T Hp^{(j)}}, \\ u^{(j+1)} &= u^{(j)} + \alpha^{(j)} p^{(j)}, \\ r^{(j+1)} &= r^{(j)} - \alpha^{(j)} Hp^{(j)}, \\ \beta^{(j+1)} &= \frac{(r^{(j+1)})^T Pr^{(j+1)}}{(r^{(j)})^T Pr^{(j)}}. \end{aligned}$$

We also recall that in the BFGS-C procedure the PCG algorithm is restarted after the first q iterations (here numbered from 0 to $q-1$), using the last computed iterate as starting guess and the corresponding preconditioned residual as starting direction.

The following result holds.

Theorem A.1
The directions, residuals, and iterates generated by the BFGS-C procedure described in Section 4 are the same as the directions, residuals, and iterates generated by the PCG algorithm with preconditioner $\widehat{P}$.

Proof
Let $\bar{p}^{(j)}$, $\bar{r}^{(j+1)}$, $\bar{u}^{(j+1)}$, $\bar{\alpha}^{(j)}$ and $\bar{\beta}^{(j+1)}$ be the direction, residual, iterate and related scalars obtained at the j-th iteration of the BFGS-C procedure, and let $\tilde{p}^{(j)}$, $\tilde{r}^{(j+1)}$, $\tilde{u}^{(j+1)}$, $\tilde{\alpha}^{(j)}$ and $\tilde{\beta}^{(j+1)}$ be the corresponding vectors and scalars at the j-th iteration of the PCG algorithm with preconditioner $\widehat{P}$. The thesis obviously holds for $j = 0, \ldots, q-1$.

By the properties of the PCG algorithm applied to system (5) with preconditioner $\widehat{P}$, we have

$$(\tilde{p}^{(j)})^T H \widehat{P} \tilde{r}^{(q)} = 0, \quad j = 0, \ldots, q-2, \tag{36}$$

$$(\tilde{p}^{(q-1)})^T H \widehat{P} \tilde{r}^{(q)} = \frac{(\tilde{r}^{(q-1)} - \tilde{r}^{(q)})^T \widehat{P} \tilde{r}^{(q)}}{\tilde{\alpha}^{(q-1)}} = -\frac{(\tilde{r}^{(q)})^T \widehat{P} \tilde{r}^{(q)}}{\tilde{\alpha}^{(q-1)}}; \tag{37}$$

furthermore, since the columns of the matrix S associated with the BFGS-C procedure are the normalized directions $\tilde{p}^{(j)}/\sqrt{(\tilde{p}^{(j)})^T H \tilde{p}^{(j)}}$ with $j = 0, \ldots, q-1$, we get

$$P_{upd} \tilde{r}^{(q)} = (I - SS^T H) \widehat{P} \tilde{r}^{(q)}. \tag{38}$$

From (36) and (37) it follows that

$$SS^T H \widehat{P} r^{(q)} = -\frac{(\tilde{r}^{(q)})^T \widehat{P} \tilde{r}^{(q)}}{(\tilde{p}^{(q-1)})^T H \tilde{p}^{(q-1)} \tilde{\alpha}^{(q-1)}} \tilde{p}^{(q-1)} = -\tilde{\beta}^{(q)} \tilde{p}^{(q-1)},$$

and hence, by (38), we have

$$P_{upd} \tilde{r}^{(q)} = \widehat{P} \tilde{r}^{(q)} + \tilde{\beta}^{(q)} \tilde{p}^{(q-1)} = \tilde{p}^{(q)}, \quad (\tilde{r}^{(q)})^T P_{upd} \tilde{r}^{(q)} = (\tilde{r}^{(q)})^T \widehat{P} \tilde{r}^{(q)}.$$

This yields

$$\bar{p}^{(q)} = \tilde{p}^{(q)}, \quad \bar{\alpha}^{(q)} = \tilde{\alpha}^{(q)}, \quad \bar{u}^{(q+1)} = \tilde{u}^{(q+1)}, \quad \bar{r}^{(q+1)} = \tilde{r}^{(q+1)}, \quad \bar{\beta}^{(q)} = \tilde{\beta}^{(q)}, \tag{39}$$

and

$$P_{upd} \bar{r}^{(q+1)} = (I - SS^T H) \widehat{P} \tilde{r}^{(q+1)} = \widehat{P} \tilde{r}^{(q+1)}. \tag{40}$$

By induction, it is straightforward to prove that (39) and (40) hold also when q is replaced by j, with $j > q$. □

Finally, we note that the previous theorem provides also an extension to KKT systems of the equivalence between the PCG and the BFGS methods stated in [34] for convex quadratic problems.