NUMERICAL EXPERIENCE WITH ITERATIVE METHODS FOR EQUALITY CONSTRAINED NONLINEAR PROGRAMMING PROBLEMS [1]

Ladislav Lukšan, Jan Vlček
Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic and Technical University of Liberec, Hálkova 6, 461 17 Liberec, Czech Republic

Abstract. We show that indefinitely preconditioned symmetric Krylov-subspace methods are very efficient for solving linearized KKT systems arising in equality constrained optimization. We give a numerical comparison of various Krylov subspace methods in three different forms (original system, null-space transformation, range-space transformation). Furthermore, we give a survey of our previous results concerning indefinite preconditioners and merit functions and prove new propositions.

Keywords. Nonlinear programming, sparse problems, equality constraints, inexact Newton method, line search methods, indefinite systems, indefinite preconditioners, symmetric Krylov subspace methods, residual smoothing, computational experiments.

1 Introduction

We are concerned with the problem of finding a point $x^* \in R^n$, such that

$$x^* = \arg\min_{x \in \mathcal{F}} F(x), \tag{1}$$

where $\mathcal{F} \subset R^n$ is a feasible set defined by the system of equations

$$\mathcal{F} = \{x \in R^n : c_k(x) = 0, 1 \le k \le m\}, \tag{2}$$

where $m \le n$ (in fact we allow local minima). Here $F : R^n \to R$ and $c_k : R^n \to R$, $1 \le k \le m$, are twice continuously differentiable functions whose gradients and Hessian matrices will be denoted by $\nabla F(x)$, $\nabla c_k(x)$, $1 \le k \le m$, and $\nabla^2 F(x)$, $\nabla^2 c_k(x)$, $1 \le k \le m$, respectively. Furthermore, we use the notation $c(x) = [c_1(x), \ldots, c_m(x)]^T$ and $A(x) = [a_1(x), \ldots, a_m(x)] = [\nabla c_1(x), \ldots, \nabla c_m(x)]$ and we suppose that the matrix $A(x)$ has a full column rank. Then the solution $x^* \in R^n$ of the problem (1)-(2) satisfies the Karush-Kuhn-Tucker (KKT) conditions, i.e. there exists a vector $u^* \in R^m$, such that

$$\begin{aligned} \nabla_x L(x^*, u^*) &= \nabla F(x^*) + A(x^*) u^* = 0, && (3) \\ \nabla_u L(x^*, u^*) &= c(x^*) = 0, && (4) \end{aligned}$$

[1] This work was supported by grant GA ČR 201/00/0080 and by MURST 1999 Programma di Cofinanziamento, Numerical Methods for Evolutionary Problems

where

$$L(x,u) = F(x) + u^T c(x) \tag{5}$$

is the Lagrangian function, whose gradient and Hessian matrix will be denoted by

$$\begin{aligned} g(x,u) &= \nabla_x L(x,u) = \nabla F(x) + \sum_{k=1}^{m} u_k \nabla c_k(x), \\ G(x,u) &= \nabla_x^2 L(x,u) = \nabla^2 F(x) + \sum_{k=1}^{m} u_k \nabla^2 c_k(x), \end{aligned}$$

(first order necessary conditions). Let $Z(x)$ be a matrix whose columns form an orthonormal basis in the null-space of $A^T(x)$ so that $A^T(x)Z(x) = 0$ and $Z^T(x)Z(x) = I$. If, in addition to (3)-(4), the matrix $Z^T(x^*)G(x^*,u^*)Z(x^*)$ is positive definite, then the point $x^* \in R^n$ is a solution of the problem (1)-(2) (second order sufficient conditions).

Basic methods for solving problem (1)-(2) are iterative and their iteration step has the form

$$\begin{aligned} x^+ &= x + \alpha_x d_x, & (6) \\ u^+ &= u + \alpha_u d_u, & (7) \end{aligned}$$

where $d_x \in R^n$, $d_u \in R^m$ are direction vectors and $\alpha_x > 0$, $\alpha_u > 0$ are stepsizes. In this contribution, we focus our attention on large sparse problems. Since a sparse Hessian matrix $G(x,u)$ can be efficiently computed either numerically [5] or automatically [16], we confine our considerations to methods derived from the Newton method applied to nonlinear KKT system (3)-(4). The iteration step of the Newton method has the form (6)-(7), where $\alpha_x = 1$, $\alpha_u = 1$ and

$$Kd = \begin{bmatrix} B & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} d_x \\ d_u \end{bmatrix} = \begin{bmatrix} b_x \\ b_u \end{bmatrix} = b, \tag{8}$$

where $B = G(x,u)$ (computed by finite differences in our case), $A = A(x)$ (we assume that A has a full column rank), $b_x = -g(x,u)$ and $b_u = -c(x)$. This is a system of $n+m$ linear equations with $n+m$ unknowns $(d_x, d_u) \in R^{n+m}$, whose matrix is always indefinite. Moreover, matrix B is not positive definite in general even if matrix $Z(x)^T G(x,u) Z(x)$ has this property.

Methods for solving linearized KKT systems were intensively studied in the past decades. When the system is dense of a moderate size, direct methods based on the null-space transformation are commonly used [11]. Furthermore, recent investigations have shown that direct ABS-type methods [1] are extremely suitable in this case, since they are stable and they outperform other direct methods measured by the number of algebraic operations. When the system is sparse of a large scale, then the sparse Bunch-Parlett decomposition of matrix K is often advantageous. However, rather large fill-in can occur if this matrix has an unsuitable sparsity pattern or it is ill-conditioned and the optimal ordering cannot be used. Therefore, iterative methods were looked for to overcome this difficulty. Iterative methods based on the null-space transformation which use a nonorthogonal matrix Z are still very popular (see [25]). However, this approach can lead to ill-conditioned reduced systems as it is mentioned in [25]. Recently, the use of matrix Z was replaced by the use of a nonorthogonal projection matrix, which is computed from matrix A directly [15]. This way lead to fully iterative methods, since the difficult direct computation of matrix Z is eliminated.

Independently of the null-space approach, preconditioned Krylov subspace methods applied to the original system were developed. This research was also stimulated by the solution of the Stokes and Navier-Stokes equations [28]. However, the later applications have a slightly different feature, since matrix B is always positive definite. Therefore, positive definite preconditioners are commonly used and the MINRES method is very popular in this case [30]. In the optimization applications, matrix B is frequently indefinite even singular (see Section 4) so positive definite preconditioners based on its decomposition or inversion are usually unsuitable. For this reason, indefinite preconditioners were studied and successfully applied either to nonsymmetric methods (GMRES, CGS) [19], [27] or to symmetric methods (CG, SCG) [22].

This paper has two purposes. Firstly, we give a comprehensive survey of our previous theoretical results on indefinitely preconditioned symmetric Krylov subspace methods and introduce new generalizations and explanations concerning equality constrained optimization. Secondly, we propose a large number of various algorithms (some of them are new) based on three different approaches (original system, null-space transformation, range-space transformation) and give results of extensive numerical experiments comparing individual algorithms and confirming their efficiency. These experiments primarily concerns preconditioner (9). Notice that we have tested additional preconditioners (some results are given in [22]) and also additional Krylov subspace methods (GMRES, CGS, MINRES), but algorithms presented in this paper has not been outperformed.

This paper is organized as follows. In Section 2, we describe various iterative methods for solving linearized KKT system. Section 3 reviews line search methods and gives new results concerning the termination of Krylov subspace algorithms. This section is not exhaustive, since our attention is primarily focused in preconditioned Krylov subspace methods for linearized KKT systems. Nevertheless, this review is necessary for understanding to Algorithm 3.1, which was used for obtaining our results. More details concerning SQP methods and merit functions can be found in [2]. In Section 4, results of extensive computational experiments are reported and certain conclusions are formulated. Symbol $\|.\|$ denotes the Euclidean norm in this text.

2 Solution of the linear KKT system

The most important part of a method for equality constrained nonlinear programming problems is the solution of the linear KKT system (8). There are three basic possibilities: the solution of the original system, the use of the null-space transformation and the use of the range-space transformation.

2.1 Solution of the original system

Let us write system (8) in simple form $Kd = b$, where K is a symmetric nonsingular matrix. The following theorem, proved in [14], characterizes the indefiniteness of the matrix K.

Theorem 1. *Let k_+, k_- and k_0 be the number of positive, negative and zero eigenvalues of the matrix K and let l_+, l_- and l_0 be the number of positive, negative and zero eigenvalues of the matrix Z^TBZ. Then $k_- = l_- + m$, $k_+ = l_+ + m$ and $k_0 = l_0$.*

If matrix K is sufficiently sparse, the system $Kd = b$ can be solved directly using a sparse elimination procedure. Since matrix K is indefinite, the sparse Choleski decomposition has to be replaced, e.g., by the sparse Bunch-Parlett decomposition [8] implemented in the AERE Harwell code MA27 [9]. Here, we focus our attention on suitably preconditioned iterative Krylov-subspace methods. Notice that system (8) need not be solved precisely, since we use the inexact Newton method [6], which is locally convergent if $\|Kd-b\| \le \omega\|b\|$, $0 \le \omega \le \overline{\omega} < 1$, holds in all iterations. The superlinear convergence is assured if $\omega \to 0$ as $\|b\| \to 0$. Unfortunately, condition $\|Kd - b\| \le \omega\|b\|$ is not sufficient for a successful use of standard merit functions (see Theorem 8). For this reason, this condition is replaced by two simultaneous conditions $\|r_x\| \le \omega\|b_x\|$ and $\|r_u\| \le \omega\|b_u\|$, where $\|r_x\|$ and $\|r_u\|$ are the first and the second parts of the residual vector $r = b - Kd$, respectively.

The choice of a suitable preconditioner is slightly restricted, since matrix B in (8) is not positive definite in general. For this reason, indefinite preconditioners play an important role. On the other hand, the use of an indefinite preconditioner excludes some iterative methods. To explain this fact, we demonstrate a formal derivation of a preconditioned iterative method. Let C be a preconditioner (symmetric and nonsingular). We first assume that C is positive definite. In this case, we can replace system $Kd = b$ by system $\tilde{K}\tilde{d} = \tilde{b}$, where $\tilde{K} = C^{-1/2}KC^{-1/2}$, $\tilde{b} = C^{-1/2}b$, $\tilde{d} = C^{1/2}d$, and derive the iterative process for $\tilde{d}$ and $\tilde{r} = \tilde{b} - \tilde{K}\tilde{d}$. The iterative process for d and $r = b - Kd$ is then obtained by premultiplying corresponding recursions by $C^{-1/2}$ and $C^{1/2}$. (More specifically, the conjugate gradient method uses recursions $\tilde{d}^+ = \tilde{d} + \alpha\tilde{p}$, $\tilde{r}^+ = \tilde{r} - \alpha\tilde{q}$ and $\tilde{p}^+ = \tilde{r}^+ + \beta\tilde{p}$, where $\tilde{d} = C^{1/2}d$, $\tilde{r} = C^{-1/2}r$ and $\tilde{q} = C^{-1/2}KC^{-1/2}\tilde{p}$. After premultiplying the first recursion by $C^{-1/2}$, the second by $C^{1/2}$ and the third by $C^{-1/2}$, we can write $d^+ = d + \alpha p$, $r^+ = r + \alpha q$ and $p^+ = C^{-1}r^+ + \beta p$, where $p = C^{-1/2}\tilde{p}$ and $q = Kp$). The resulting process contains only integer powers of C so it can be advantageously used for an indefinite C. The last consideration is not quite obvious but is confirmed by Theorems 2-4.

If we apply the above procedure to the MINRES method, based on the symmetric Lanczos process, then expressions of the form $\sqrt{v^TC^{-1}v}$, $v \in R^n$, appear which are infeasible if $v^TC^{-1}v < 0$ for indefinite C. This fact excludes the MINRES method from our considerations implemented in both the classic form proposed in [26] and the modified form described in [30]. In the subsequent part of this section, we confine our attention to four iterative methods which can be indefinitely preconditioned: the conjugate gradient (CG) method, the smoothed conjugate gradient (SCG) method, the conjugate residual (CR) method and the symmetric form of the quasi-minimal residual (QMR) method.

The CG method, proposed in [18], is intended for solving symmetric positive definite systems. In this case, it has good numerical properties and it cannot break down. The following algorithm demonstrates the preconditioned CG method.

Algorithm PCG

$d = 0, \quad r = b,$
$\tilde{r} = C^{-1} r, \quad \gamma = \tilde{r}^T r,$
$p = \tilde{r},$

while $\|r_x\| > \omega \|b_x\|$ or $\|r_u\| > \omega \|b_u\|$ **do**

$q = Kp, \quad \alpha = \gamma / p^T q,$
$d^+ = d + \alpha p, \quad r^+ = r - \alpha q,$
$\tilde{r}^+ = C^{-1} r^+, \quad \gamma^+ = (\tilde{r}^+)^T r^+,$
$\beta = \gamma^+ / \gamma, \quad p^+ = \tilde{r}^+ + \beta p,$
delete index +

end while.

In the indefinite case, Algorithm PCG can break down due to a possible zero value of $p^T K p$. Moreover, if preconditioner C is indefinite, Algorithm PCG can break down due to a possible zero value of $r^T C^{-1} r$. These theoretical possibilities appear rarely either for special starting points or in quite pathological cases. Also division by near zeros may lead to very poor recurred quantities, but our computational experiments proposed in Section 4 confirm that this phenomenon appears exceptionally. Similar considerations hold for all preconditioned Krylov subspace methods described in this section.

The SCG method, proposed in [34], uses vectors v^+ and s^+ generated by the CG method (i.e., d^+ and r^+ are replaced by v^+ and s^+, respectively, in PCG). In the smoothing procedure, vectors $d^+ = d + \lambda(v^+ - d)$ and $r^+ = r + \lambda(s^+ - r)$ are determined, where λ is chosen to minimize $\|r^+\|$. This procedure guarantees that the residual vectors decrease monotonically. In [34], it is shown that the unpreconditioned SCG method is, in the unpreconditioned case, equivalent to the MINRES method. Nevertheless, the SCG method can be indefinitely preconditioned as is shown in the following algorithm.

Algorithm PSCG

$d = 0, \quad v = 0, \quad r = b, \quad s = b,$
$\tilde{s} = C^{-1} s, \quad \gamma = \tilde{s}^T s,$
$p = \tilde{s},$

while $\|r_x\| > \omega \|b_x\|$ or $\|r_u\| > \omega \|b_u\|$ **do**

$q = Kp, \quad \alpha = \gamma / p^T q,$
$v^+ = v + \alpha p, \quad s^+ = s - \alpha q,$
$\lambda = -r^T(s^+ - r)/\|s^+ - r\|^2,$
$d^+ = d + \lambda(v^+ - d), \quad r^+ = r + \lambda(s^+ - r),$
$\tilde{s}^+ = C^{-1}s^+, \quad \gamma^+ = (\tilde{s}^+)^T s^+,$
$\beta = \gamma^+/\gamma, \quad p^+ = \tilde{s}^+ + \beta p,$
delete index +

end while.

The CR method, proposed in [32], is based on the minimization of residual norms. The same requirement leads to the MINRES method, but the CR method uses different recurrences, which are closely related to the CG recurrences. The CR method can be indefinitely preconditioned as it is shown in the following algorithm.

Algorithm PCR

$d = 0, \quad r = b,$
$\tilde{r} = C^{-1}r, \quad \tilde{s} = K\tilde{r}, \quad \gamma = \tilde{s}^T\tilde{r},$
$p = \tilde{r}, \quad q = \tilde{s},$

while $\|r_x\| > \omega\|b_x\|$ or $\|r_u\| > \omega\|b_u\|$ **do**

$\tilde{q} = C^{-1}q, \quad \alpha = \gamma / q^T\tilde{q},$
$d^+ = d + \alpha p, \quad r^+ = r - \alpha q, \quad \tilde{r}^+ = \tilde{r} - \alpha\tilde{q},$
$\tilde{s}^+ = K\tilde{r}^+, \quad \gamma^+ = (\tilde{s}^+)^T\tilde{r}^+, \quad \beta = \gamma^+/\gamma,$
$p^+ = \tilde{r}^+ + \beta p, \quad q^+ = \tilde{s}^+ + \beta q,$
delete index +

end while.

The C^{-1}-symmetric QMR method, proposed in [12], was derived as a simplification of the more general nonsymmetric QMR method. The following algorithm demonstrates the preconditioned C^{-1}-symmetric QMR method.

Algorithm PQMR

$d = 0, \quad r = b, \quad s = b,$
$u = 0, \quad v = 0, \quad \vartheta = 0, \quad \tau = s^T s,$
$\tilde{s} = C^{-1}s, \quad \gamma = \tilde{s}^T s,$
$p = \tilde{s},$

while $\|r_x\| > \omega\|b_x\|$ or $\|r_u\| > \omega\|b_u\|$ **do**

$q = Kp, \quad \alpha = \gamma / p^T q,$
$s^+ = s - \alpha q, \quad \sigma = (s^+)^T s^+/\tau,$
$u^+ = (\vartheta u + \alpha p)/(1 + \sigma), \quad v^+ = (\vartheta v + \alpha q)/(1 + \sigma),$
$d^+ = d + u^+, \quad r^+ = r - v^+,$
$\vartheta^+ = \sigma, \quad \tau^+ = \tau\sigma/(1 + \sigma),$
$\tilde{s}^+ = C^{-1}s^+, \quad \gamma^+ = (\tilde{s}^+)^T s^+,$
$\beta = \gamma^+/\gamma, \quad p^+ = \tilde{s}^+ + \beta p,$
delete index +

end while.

Choice of a suitable preconditioner is a very important decision, since unpreconditioned iterative methods are not competitive with sparse direct elimination methods as will be shown below. Many efficient positive definite preconditioners exist when B is positive definite (see [28], [30]). In case B is not positive definite, these preconditioners are either undefined or inefficient. In such a situation, the indefinite preconditioner

$$C = \begin{bmatrix} D & A \\ A^T & 0 \end{bmatrix} \tag{9}$$

can be advantageously used, where D is a positive definite diagonal matrix derived usually from a diagonal of B. This preconditioner is investigated in [22], [23], [19], [27] and [29]. The following two theorems are proved in [22].

Theorem 2. *Consider the preconditioner given by (9). Then the matrix KC^{-1} has at least $2m$ unit eigenvalues but, if $B - D$ is nonsingular, only m linearly independent eigenvectors corresponding to these eigenvalues exist. Other eigenvalues of the matrix KC^{-1} are exactly eigenvalues of the matrix $Z^TBZ(Z^TDZ)^{-1}$. If Z^TBZ is positive definite, then all eigenvalues of the matrix KC^{-1} are positive.*

Theorem 3. *Consider the preconditioned CG method (Algorithm PCG) with preconditioner (9) applied to system (8). If a breakdown does not occur, then we obtain the solution $d^* \in R^{n+m}$ of (8) after of most $n - m + 2$ iterations.*

Another interesting result, proved in [22], concerns the special case when $b_u = 0$ in (8). This case occurs if an active set strategy is used for linearly constrained problems (see [10]) or if the horizontal step is computed for the Byrd-Omojokun trust region method (see [20]). This case also occurs after discretization of the Stokes problem (see [28]).

Theorem 4. *Consider the preconditioned CG method (Algorithm PCG) with preconditioner (9) applied to system (8), where $b_u = 0$. Let matrix Z^TBZ be positive definite. Then:*
(a) The vector $d_x^* \in R^n$ (the first part of $d^* \in R^{n+m}$) is found after $n - m$ iterations at most.
(b) The algorithm cannot break down before d_x^* is found.
(c) The error $\|d_x - d_x^*\|$ converges to zero at least R - linearly with the quotient $(\sqrt{\kappa} - 1)/(\sqrt{\kappa} + 1)$ where $\kappa = \kappa(Z^TBZ(Z^TDZ)^{-1})$ is a spectral condition number of matrix $Z^TBZ(Z^TDZ)^{-1}$.

In the special case, when $b_u = 0$ in (8) (so $r_u = 0$ in all iterations), an interesting effect appears. If $d_x = d_x^*$ then $d_u = d_u^*$ usually does not hold, but Algorithm PCG with preconditioner (9) breaks down because $\gamma = 0$. Fortunately, d_u^* can be easily found as a solution of equation $A(d_u^* - d_u) = r_x$ in this case. Therefore, if $\|r_u\| = 0$ and $\gamma = 0$, we add $(A^TD^{-1}A)^{-1}A^TD^{-1}r_x$ to d_u and terminate the computation.

On the other hand, we can easily obtain the case when $r_u = 0$ in all iterations by the initial setting $d_x = D^{-1}A(A^TD^{-1}A)^{-1}b_u$ (the vertical step). Vector d_x can also be obtained as a solution of the following system

$$\begin{bmatrix} D & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} d_x \\ d_u \end{bmatrix} = \begin{bmatrix} 0 \\ b_u \end{bmatrix}. \tag{10}$$

This consideration leads to the preconditioned modified CG algorithm.

Algorithm PMCG

$d_x = D^{-1}A(A^TD^{-1}A)^{-1}b_u, \quad d_u = 0,$
$r_x = b_x - Bd_x, \quad r_u = b_u - A^Td_x,$
$\tilde{r} = C^{-1}r, \quad \gamma = \tilde{r}^Tr,$
$p = \tilde{r},$

while $\|r_x\| > \omega\|b_x\|$ **do**

$q = Kp, \quad \alpha = \gamma/p^Tq,$
$d^+ = d + \alpha p, \quad r^+ = r - \alpha q,$
$\tilde{r}^+ = C^{-1}r^+, \quad \gamma^+ = (\tilde{r}^+)^Tr^+,$

if $\gamma^+ \approx 0$ **then**
add $\tilde{r}_u$ to d_u and exit
else
$\beta = \gamma^+/\gamma, \quad p^+ = \tilde{r}^+ + \beta p,$
delete index +

end if

end while.

Notice that relation $\gamma^+ \approx 0$ is replaced by a more complicated test in the real algorithm.

The use of indefinite preconditioner (9) requires computation of the product $\tilde{v} = C^{-1}v$ for some vector v. In the sparse case, this computation can be carried out by two different ways. The first way is based on the explicit inversion

$$\begin{aligned} C^{-1}v &= \begin{bmatrix} D^{-1} - D^{-1}A(A^TD^{-1}A)^{-1}A^TD^{-1} & D^{-1}A(A^TD^{-1}A)^{-1} \\ (A^TD^{-1}A)^{-1}A^TD^{-1} & -(A^TD^{-1}A)^{-1} \end{bmatrix} \begin{bmatrix} v_x \\ v_u \end{bmatrix} \\ &= \begin{bmatrix} D^{-1}(v_x - A\tilde{v}_u) \\ \tilde{v}_u \end{bmatrix}, \end{aligned}$$

where $\tilde{v}_u = (A^TD^{-1}A)^{-1}(A^TD^{-1}v_x - v_u)$. The matrix $(A^TD^{-1}A)^{-1}$ is not computed, but the sparse Choleski decomposition is used instead. Unfortunately, the matrix $A^TD^{-1}A$ can be dense if A has dense rows. To eliminate this situation, we assume that $A^T = [A_s^T, A_d^T]$ and $D = \mathrm{diag}(D_s, D_d)$, where $A_s^TD_s^{-1}A_s$ is sparse and A_d consists of dense rows. Then

$$\begin{aligned} (A^TD^{-1}A)^{-1} &= (A_s^TD_s^{-1}A_s + A_d^TD_d^{-1}A_d)^{-1} \\ &= (A_s^TD_s^{-1}A_s)^{-1} - (A_s^TD_s^{-1}A_s)^{-1}A_d^TM_d^{-1}A_d(A_s^TD_s^{-1}A_s)^{-1}, \end{aligned}$$

where

$$M_d = D_d + A_d(A_s^TD_s^{-1}A_s)^{-1}A_d^T$$

is a (low-dimensional) dense matrix. Again the sparse Choleski decomposition is used instead of $(A_s^T D_s^{-1} A_s)^{-1}$. We will denote by PCG1, PSCG1, PCR1, PQMR1, PMCG1 the algorithms after applying this technique.

Instead of explicit inversion the sparse Bunch-Parlet decomposition $C = PLML^T P^T$ can be determined [8] with block diagonal matrix M, lower triangular matrix L and permutation matrix P. The decomposed system $PLML^T P^T \tilde{v} = v$ is then easily solved. The matrix (9) has the same form as matrices, which appear in interior point methods for linear programming, so the Bunch-Parlett decomposition can be carried out more economically than in the general case. We will denote by PCG2, PSCG2, PCR2, PQMR2, PMCG2 the algorithms after applying this technique.

2.2 Null-space transformation

Consider the unique representation $d_x = Zd_Z + D^{-1}Ad_A$, where

$$d_Z = (Z^T DZ)^{-1} Z^T D d_x \in R^{n-m}, \quad d_A = (A^T D^{-1} A)^{-1} A^T d_x \in R^m$$

(recall that $Z^T A = 0$). Using the second equation in (8), we get $d_A = (A^T D^{-1} A)^{-1} b_u$ and the first one gives $BZd_Z = b_x - BD^{-1}Ad_A - Ad_u$ which after premultiplying by Z^T gives

$$Z^T BZ d_Z = b_Z, \tag{11}$$

where $b_Z = Z^T(b_x - BD^{-1}Ad_A)$. Thus d_Z can be obtained by the conjugate gradient method with preconditioner $Z^T DZ$ applied to equation (11) (for this reason we introduce vectors $r_Z = b_Z - Z^T BZd_Z$, $\tilde{r}_Z = (Z^T DZ)^{-1} r_Z$ and p_Z, c.f. Algorithm PCG). One of the possibilities how the null-space transformation can be used is based on the computation of a nonorthogonal matrix Z. Consider matrices

$$A = \begin{bmatrix} A_B \\ A_N \end{bmatrix}, \quad Z = \begin{bmatrix} -(A_B^{-1})^T A_N^T \\ I \end{bmatrix},$$

where A_B is a regular square (so-called basic) submatrix of A and I is the unit matrix. Then $A^T Z = 0$ and $[A, Z]$ is a nonsingular square matrix. If we consider the decomposition $A_B = LU$, then matrix $Z^T BZ$ used in (11) can be expressed in the form

$$Z^T BZ = \left[-A_N U^{-1} L^{-1}, I\right] B \left[-A_N U^{-1} L^{-1}, I\right]^T.$$

The above approach requires a direct solver for sparse rectangular linear systems, which is not always available. Moreover, problems with stability can occur for nonorthogonal Z, since this matrix can be ill-conditioned (see [25]). Therefore, we use another possibility based on considerations given in [15]. Following [15], we can modify conjugate gradient iterations in such a way that they use vectors $d_z = Zd_Z$, $\tilde{r}_z = Z\tilde{r}_Z$, $p_z = Zp_Z$ and vectors r_z, q_z, such that $r_Z = Z^T r_z$, $q_Z = Z^T q_z$. After this transformation, multiplication $\tilde{r}_Z = (Z^T DZ)^{-1} r_Z$ can be replaced by the formula

$$\tilde{r}_z = Z(Z^T DZ)^{-1} Z^T r_z = (D^{-1} - D^{-1}A(A^T D^{-1} A)^{-1} A^T D^{-1}) r_z$$

so matrix Z need not be used explicitly. Since d_z does not influence formulas in conjugate gradient iterations directly, we can use $d_x = d_z + D^{-1}Ad_A$ instead of d_z in

CG recurrences. Since d_u cannot be obtained from conjugate gradient iterations, it has to be estimated in another way. The most natural choice is the weighted least-square minimization of the total residual $r_z - Ad_u$. This choice leads to formula $d_u = (A^T D^{-1} A)^{-1} A^T D^{-1} r_z$. The following algorithm summarizes our considerations.

Algorithm NPCG

$d_x = D^{-1}A(A^T D^{-1} A)^{-1} b_u$
$r_z = b_x - Bd_x,$
$d_u = (A^T D^{-1} A)^{-1} A^T D^{-1} r_z$
$\tilde{r}_z = D^{-1}(r_z - Ad_u),$
$\gamma = r_z^T \tilde{r}_z, \quad \overline{\gamma} = \gamma$
$p_z = \tilde{r}_z,$

while $\gamma > \omega\overline{\gamma}$ **do**

$q_z = Bp_z, \quad \alpha = \gamma / p_z^T q_z,$
$d_x^+ = d_x + \alpha p_z, \quad r_z^+ = r_z - \alpha q_z,$
$d_u^+ = (A^T D^{-1} A)^{-1} A^T D^{-1} r_z^+$
$\tilde{r}_z^+ = D^{-1}(r_z^+ - Ad_u^+),$
$\gamma^+ = (r_z^+)^T \tilde{r}_z^+, \quad \beta = \gamma^+ / \gamma,$
$p_z^+ = \tilde{r}_z^+ + \beta p_z,$
delete index +

end while

We denote this variant of NPCG by NPCG1. Vertical step $d_x = D^{-1}A(A^T D^{-1} A)^{-1} b_u$ can also be computed as a solution of (10) and vectors $\tilde{r}_z$, d_u as a solution of system

$$\begin{bmatrix} D & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} \tilde{r}_z \\ d_u \end{bmatrix} = \begin{bmatrix} r_z \\ 0 \end{bmatrix} \tag{12}$$

by using the Bunch-Parlett decomposition of matrix (9) (vector d_u is a by-product of the determination of $\tilde{r}_z$ so it need not be obtained by an extra computation). In such a way we obtain an additional variant of NPCG, which we denote by NPCG2.

The following theorem was proved in [23].

Theorem 5. *Algorithms NPCG and PMCG are equivalent in the sense that both generate the same vectors d_x and, if $\omega = 0$, they give the same final pair (d_x, d_u).*

2.3 Range-space transformation

The range-space transformation is based on the sparse Gill-Murray [13] decomposition $B + E = PLML^T P^T$, where M, E are posive definite diagonal matrices, L is a lower triangular matrix and P is a permutation matrix. If B is not positive definite, then $E \neq 0$ and $PLML^T P^T$ can differ from B considerably. To decrease this effect, we first add the second equation of (8), multiplied by ρA, to the first one which yields

$$\begin{bmatrix} \tilde{B} & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} d_x \\ d_u \end{bmatrix} = \begin{bmatrix} \tilde{b}_x \\ b_u \end{bmatrix}. \tag{13}$$

where $\tilde{B} = B + \rho AA^T$ and $\tilde{b}_x = b_x + \rho Ab_u$. The following theorem is valid (see [10]).

Theorem 6. *Let matrix Z^TBZ be positive definite. Then either B is positive definite or there exists a number $\overline{\rho} \geq 0$ such that $\tilde{B}$ is positive definite if and only if $\rho > \overline{\rho}$.*

Another useful theorem of this type is proved in [22].

Theorem 7. *Let matrix K be nonsingular. Then either $A^TB^{-1}A$ is positive definite or there exists a number $\overline{\rho} \geq 0$ such that $A^T\tilde{B}^{-1}A$ is positive definite if and only if $\rho > \overline{\rho}$. If matrix Z^TBZ is positive definite, then $\overline{\rho}$ has the same value as in Theorem 6. If matrix B is nonsingular and $A^TB^{-1}A$ is not positive definite, then*

$$1/\overline{\rho} = -\max_{\mu_i<0} \mu_i, \tag{14}$$

where μ_i, $1 \leq i \leq m$, are eigenvalues of matrix $A^TB^{-1}A$.

The last theorem shows us that, for sufficiently large ρ, matrix $A^T\tilde{B}^{-1}A$ is positive definite independently of the positive definiteness of matrices Z^TBZ and $\tilde{B}$ (when K is nonsingular). This result seems to be impractical, since Z^TBZ should be positive definite in the neighbourhood of the isolated minimum (second order sufficient conditions). The following example demonstrates that the situation where Z^TBZ is indefinite but the Newton method gives a good direction vector can occur when we are far from the solution, especially far from the feasible region (where the quadratic model does not approximate the original problem accurately).

Example 1. *Consider the minimization of function $F(x) = x_1^3 + 3x_1(x_2^2 - 1)$ subject to constraint $c(x) = (x_1-3)^2 + x_2^2 - 1 = 0$. This problem has the unique solution $x^* = [2,0]^T$ which satisfies the second order sufficient conditions (with the Lagrange multiplier $u^* = 4.5$). Consider the starting quantities $x = [-0.5, 0]^T$, $u = 0$. Then $c(x) = (x_1-3)^2 + x_2^2 - 1 = 11.25$,*

$$A(x) = \nabla c(x) = \begin{bmatrix} 2(x_1-3) \\ 2x_2 \end{bmatrix} = \begin{bmatrix} -7 \\ 0 \end{bmatrix}, \quad Z(x) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$g(x,u) = \nabla F(x) + A(x)u = \begin{bmatrix} 3x_1^2 + 3(x_2^2-1) + 2(x_1-3)u \\ 6x_1x_2 + 2x_2u \end{bmatrix} = \begin{bmatrix} -2.25 \\ 0 \end{bmatrix},$$

$$G(x,u) = \nabla^2 F(x) + \nabla^2 c(x)u = \begin{bmatrix} 6x_1 + 2u & 6x_2 \\ 6x_2 & 6x_1 + 2u \end{bmatrix} = \begin{bmatrix} -3 & 0 \\ 0 & -3 \end{bmatrix},$$

and, therefore, $Z(x)^TG(x,u)Z(x) = -3 < 0$. System (8) has the form

$$\begin{bmatrix} -3 & 0 & -7 \\ 0 & -3 & 0 \\ -7 & 0 & 0 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_u \end{bmatrix} = \begin{bmatrix} 2.25 \\ 0 \\ -11.25 \end{bmatrix},$$

which gives $d_1 \doteq 1.6071$, $d_2 = 0$ and $d_u \doteq -1.0102$. The direction vector $d_x = [d_1, d_2]^T$ is quite suitable. It is oriented towards the solution and for the new quantities $x^+ = x + d_x \doteq [1.1071, 0]^T$, $u^+ = u + d_u \doteq -1.0102$ the inequality $Z(x^+)^TG(x^+,u^+)Z(x^+) \doteq 4.6266 > 0$ holds. Notice that the starting point

$x = [-0.5, 0]^T$ *is not exceptional. Similar properties have points lying in the relatively large region containing this starting point.*

Number $\overline{\rho}$ appearing in Theorem 7 can be determined from (14). Its computation based on the computation of eigenvalues is difficult, especially if the number of constraints is large. Moreover, the value of ρ influences the choice of penalty coefficient σ (see [22]). For this reason, we prefer to keep ρ constant in all iterations and choose its value as a parameter of the method (see Algorithm 3.1). If this value is unsuitable, we can repeat computations with ρ increased.

Using partial elimination of variables, equation (13) can be transformed to the form

$$\tilde{B} d_x = \tilde{b}_x - A d_u, \tag{15}$$

$$A^T \tilde{B}^{-1} A d_u = A^T \tilde{B}^{-1} \tilde{b}_x - b_u. \tag{16}$$

Equation (15) can be solved directly by sparse Gill-Murray decomposition $\tilde{B} + E = LDL^T$, where E is a diagonal matrix and LDL^T is positive definite. If $\tilde{B}$ is not positive definite, then LDL^T can differ from $\tilde{B}$ considerably so that we obtain an incorrect solution of equation (15). Therefore, the sparse Bunch-Parlett decomposition $\tilde{B} = PLMfamL^T P^T$, where M is a block diagonal matrix, is more suitable in this case. Theorem 7 guarantees that matrix $A^T \tilde{B}^{-1} A$ is positive definite for sufficiently large ρ. We assume that matrix $A^T \tilde{B}^{-1} A$ is large and dense so that equation (16) is solved iteratively and matrix $A^T \tilde{B}^{-1} A$ is not built explicitly. Multiplication by this matrix is replaced by successive multiplication by matrices A, $\tilde{B}^{-1}$ after the sparse decomposition and A^T. We can use a variant of the conjugate gradient method. Our computational tests shown that the smoothed CG method is more efficient than the standard one. The following algorithm summarizes our considerations.

Algorithm RPSCG

$d_u = 0, \quad v_u = 0 \quad r_u = A^T \tilde{B}^{-1} \tilde{b}_x - b_u, \quad s_u = r_u$
$\tilde{s}_u = C_u^{-1} s_u, \quad \gamma = \tilde{s}_u{}^T s_u, \quad \overline{\gamma} = \gamma,$
$p_u = \tilde{s}_u,$

while $\gamma > \omega \overline{\gamma}$ **do**

$q_u = A^T \tilde{B}^{-1} A p_u, \quad \alpha = \gamma / p_u^T q_u,$
$v_u^+ = v_u + \alpha p_u, \quad s_u^+ = s_u - \alpha q_u,$
$\lambda = -(r_u - s_u^+)^T s_u^+ / \|r_u - s_u^+\|^2,$
$d_u^+ = v_u^+ + \lambda(d_u - v_u^+), \quad r_u^+ = s_u^+ + \lambda(r_u - s_u^+),$
$\tilde{s}_u^+ = C_u^{-1} s_u^+, \quad \gamma^+ = (\tilde{s}_u^+)^T s_u^+,$
$\beta = \gamma^+ / \gamma, \quad p_u^+ = \tilde{s}_u^+ + \beta p_u,$
delete index +

end while,

solve $\tilde{B} d_x = \tilde{b}_x - A d_u$.

Using the Gill-Murray and the Bunch-Parlett decompositions we obtain the RPSCG1 and RPSCG2 algorithms, respectively. Our computational tests showed that these algorithms usually work well without preconditioning, i.e., with $C_u = I$. In some ill-conditioned cases, it is advantageous to use preconditioner $C_u = A^T \tilde{D}^{-1} A$ with the positive definite diagonal matrix $\tilde{D}$ derived from the diagonal of $\tilde{B}$. The unpreconditioned algorithms will be denoted by RSCG1 and RSCG2, respectively.

3 Line search methods

If $B = G(x,u)$, then iterative method (6)-(7) with $\alpha_x = 1$, $\alpha_u = 1$ and with $d_x \in R^n$, $d_u \in R^m$ derived as in the previous section is the inexact Newton method for solving nonlinear KKT system (3)-(4). This method is locally convergent if $\omega \le \overline{\omega} < 1$ (i.e., a number $\varepsilon > 0$ exists such that $x \to x^*$ and $u \to u^*$ if initial estimations x and u satisfy the inequalities $\|x - x^*\| \le \varepsilon$ and $\|u - u^*\| \le \varepsilon$) and the rate of convergence is superlinear if $\omega \to 0$ (see [7]). Unfortunately, if x and u is far from x^* and u^*, then this method can fail. Therefore, some globalization strategy is necessary. Usually, stepsizes $\alpha_x = \alpha$ and $\alpha_u = \alpha$ are chosen to decrease a value of some merit function which combines requirements of optimality and feasibility. We will investigate the following merit functions

$$
\begin{aligned}
P_1(\alpha) &= F(x + \alpha d_x) + \sigma \|\mathrm{diag}(u + d_u) c(x + \alpha d_x)\|_1, \\
P_2(\alpha) &= F(x + \alpha d_x) + (u + d_u)^T c(x + \alpha d_x) + \frac{\sigma}{2} \|c(x + \alpha d_x)\|^2, \\
P_3(\alpha) &= F(x + \alpha d_x) + (u + d_u)^T c(x + \alpha d_x) + \sigma \|c(x + \alpha d_x)\|_1, \\
P_4(\alpha) &= F(x + \alpha d_x) + \sigma \|c(x + \alpha d_x)\|_1, \\
P_5(\alpha) &= F(x + \alpha d_x) + (u + \alpha d_u)^T c(x + \alpha d_x) + \frac{\sigma}{2} \|c(x + \alpha d_x)\|^2,
\end{aligned}
$$

with penalty coefficient $\sigma > 0$. Functions P_4 and P_5 lead to globally convergent algorithms if penalty coefficient σ does not decrease during the iterative process (this is proved in [17] and [31], respectively). Unfortunately, this monotone strategy can sometimes lead to unsuitable large values of σ and, therefore, to slow down convergence as was indicated in our computational experiments. In this section, we describe two other strategies which give smaller values of σ. These strategies can easily be applied to merit functions P_1 - P_3. For this reason, merit functions P_1 - P_3 are very advantageous for real computations although the global convergence has not been proved for them in general.

The necessary condition for application of a line search is the inequality $P'(0) < 0$ where $P'(0)$ is the directional derivative. Denote by $\tilde{P}$ the function obtained from P by the linearization of F and c. Clearly $\tilde{P}'(0) = P'(0)$. Since linearized functions $\tilde{P}_1$, $\tilde{P}_3$, $\tilde{P}_4$ are convex, $\tilde{P}(1) - \tilde{P}(0) \ge \tilde{P}'(0)$ holds for them and we can use inequality $\tilde{P}(1) - \tilde{P}(0) < 0$ instead of $P'(0) < 0$ (this is advantageous, since functions P_1, P_3, P_4 are nondifferentiable and the computation of their directional derivatives is complicated). The following theorem demonstrates requirements concerning the size of the penalty coefficient and the precision of the direction vector.

Theorem 8. *Inequality $P'(0) < 0$ holds if $\varphi(r) < d_x^T B d_x + \alpha + \sigma\beta$, where values α, β and $\varphi(r)$ for merit functions P_1 - P_5 are given in Table 1. Moreover, if $\beta > 0$ and*

$\sigma > \overline{\sigma} = -(d_x^T B d_x + \alpha)/\beta$, then $\varphi(r) < d_x^T B d_x + \alpha + \sigma\beta$ holds for all $\|r\|$ sufficiently small.

	$\varphi(r)$	α	β
P_1	$-d_x^T r_x + (u+d_u)^T r_u$	$b_u^T(u+d_u)$	$\|\mathrm{diag}(u+d_u)b_u\|_1 - \|\mathrm{diag}(u+d_u)r_u\|_1$
P_2	$-d_x^T r_x$	0	$b_u^T(b_u - r_u)$
P_3	$-d_x^T r_x$	0	$\|b_u\|_1 - \|r_u\|_1$
P_4	$-d_x^T r_x + (u+d_u)^T r_u$	$b_u^T(u+d_u)$	$\|b_u\|_1 - \|r_u\|_1$
P_5	$-d_x^T r_x + d_u^T r_u$	$2b_u^T d_u$	$b_u^T(b_u - r_u)$

Table 1

Proof. We prove this theorem for merit functions P_1 and P_2, since the rest of proof can be carried out in the same way. Linearizing $F(x)$ and $c(x)$, we obtain

$$P_1'(0) \le \tilde{P}_1(1) - \tilde{P}_1(0) = d_x^T \nabla F(x) + \sigma(\|\mathrm{diag}(u+d_u)(c + A^T d_x)\|_1 - \|\mathrm{diag}(u+d_u)c\|_1)$$

and

$$P_2'(0) = d_x^T(\nabla F(x) + A(u+d_u)) + \sigma d_x^T A c,$$

where $c = c(x)$ and $A = A(x)$. Using (8) together with the definition of r and b ($b_x = -\nabla F(x) - Au$ and $b_u = -c(x)$), we obtain

$$B d_x + \nabla F(x) + A(u + d_u) = B d_x + A d_u - b_x = -r_x$$

and

$$A^T d_x + c = A^T d_x - b_u = -r_u.$$

Thus we can write

$$\begin{aligned} P_1'(0) \quad &\le \quad \tilde{P}_1(1) - \tilde{P}_1(0) = -d_x^T r_x - d_x^T B d_x - (b_u - r_u)^T(u + d_u) \\ &\quad -\sigma(\|\mathrm{diag}(u+d_u)b_u\|_1 - \|\mathrm{diag}(u+d_u)r_u\|_1) \end{aligned}$$

and

$$P_2'(0) = -d_x^T r_x - d_x^T B d_x - \sigma b_u^T(b_u - r_u).$$

Using values α, β and $\varphi(r)$ corresponding to P_1 and P_2 (Table 1), we obtain

$$P'(0) \le \varphi(r) - d_x^T B d_x - \alpha - \sigma\beta$$

for all merit functions considered. Now it is clear that for $\varphi(r) < d_x^T B d_x + \alpha + \sigma\beta$ we obtain $P'(0) < 0$. Moreover, if $\beta > 0$ and $\sigma > \overline{\sigma} = -(d_x^T B d_x + \alpha)/\beta$, then $d_x^T B d_x + \alpha + \sigma\beta > 0$ and since $\varphi(r) \to 0$ as $r \to 0$, $\varphi(r) < d_x^T B d_x + \alpha + \sigma\beta$ holds for all $\|r\|$ sufficiently small. $\square$

Theorem 8 demonstrates good properties of function P_2. Since the algorithms proposed in Section 2 guarantee that $\|r_u\| \le \omega\|b_u\| < \|b_u\|$, we have $\beta > 0$. If B is positive definite, then $\overline{\sigma} < 0$ and any value $\sigma \ge 0$ gives $d_x^T B d_x + \alpha + \sigma\beta > 0$. Therefore, function P_2 allows us to use small values of the penalty coefficient (this increases the probability that the unit stepsize, which is a necessary condition for proving the superlinear convergence, will be accepted). Function P_3 has similar properties, but the inequality $\beta > 0$ may not be satisfied in all iterations. This inequality is always satisfied if condition $\|r_u\| \le \omega \|b_u\|$ is replaced by condition $\|r_u\|_1 \le \omega \|b_u\|_1$ in the algorithms proposed in Section 2, or if the l_1 norm is replaced by the l_2 norm in P_3. Another possibility is to choose $\omega < 1/\sqrt{n}$. Fortunately, these modifications are not necessary since the restart procedure described below treats these exceptional cases. Function P_1 is also advantageous as is demonstrated in Theorem 9, but again the inequality $\beta > 0$ may not be satisfied in all iterations. Functions P_4 and P_5 are slightly worse, since terms $\alpha = b_u^T(u + d_u)$ and $\alpha = 2b_u^T d_u$ may be negative, which can lead to large values of σ.

We can design line search methods in such a way that the value $\sigma > \overline{\sigma}$ is chosen and the inequality $\varphi(r) < d_x^T B d_x + \alpha + \sigma\beta$ is checked in all iterations of a Krylov-subspace algorithm. The iterative process is terminated only if this inequality is satisfied (it comes after at most $n - m + 2$ iterations if perfect computations are used, see Theorem 3, but this number can be slightly greater in floating point arithmetic). Then always $P'(0) < 0$ and the line search method is well defined. Unfortunately, this procedure can give large values of σ if B is indefinite. Therefore, we prefer another procedure based on restarts (see [22]). We use a constant value $\sigma \ge 0$ without checking the inequality $\varphi(r) < d_x^T B d_x + \alpha + \sigma\beta$. If $P'(0) \ge 0$, we compute new vector d as a solution of (8) with matrix B replaced by a positive definite diagonal matrix D. The following theorem holds for functions P_1 - P_3.

Theorem 9. *Let $B = D$, where D is a positive definite diagonal matrix. Then a Krylov-subspace algorithm (PCG, PSCG, PCR, PQMR, PMCG), preconditioned by (9) with the matrix D, terminates in its first iteration and $\tilde{P}(1) - \tilde{P}(0) < 0$ holds for any value $\sigma \ge 1$ (for P_1) or $\sigma \ge 0$ (for P_2 and P_3).*

Proof. Since $B = D$, we have $K = C$ and the exact solution of (8) is obtained in the first iteration of a Krylov-subspace algorithm. Moreover, $B = D$ is positive definite and $r = 0$ thus $\overline{\sigma} < 0$ and $\varphi(r) = 0$ for functions P_2 and P_3. For function P_1 we can write

$$\alpha + \sigma\beta = b_u^T(u + d_u) + \sigma \|\mathrm{diag}(u + d_u) b_u\|_1 = \sum_{i=1}^{m} b_u^i(u^i + d_u^i) + \sigma \sum_{i=1}^{m} |b_u^i(u^i + d_u^i)|$$

and $\alpha + \sigma\beta \ge 0$ for $\sigma \ge 1$. Therefore, assumptions of Theorem 9 imply assumptions of Theorem 8. □

Diagonal matrix D can be chosen in various ways. The simple choice $D = I$ has shown to be unsuitable in our experiments. We suggest the following choice

$$\begin{aligned} D_{ii} &= \underline{\Gamma}, \quad \text{if} \quad \frac{\|g\|}{10}|B_{ii}| < \underline{\Gamma}, \\ D_{ii} &= \frac{\|g\|}{10}|B_{ii}|, \quad \text{if} \quad \underline{\Gamma} \le \frac{\|g\|}{10}|B_{ii}| \le \overline{\Gamma}, \\ D_{ii} &= \overline{\Gamma}, \quad \text{if} \quad \overline{\Gamma} < \frac{\|g\|}{10}|B_{ii}| \end{aligned} \tag{17}$$

(here $0 < \underline{\Gamma} < \overline{\Gamma}$ are suitable constants). This procedure, which is quite heuristic, was found by using extensive experiments. The basic idea of this procedure is that

D_{ii} should be small when $|B_{ii}|$ or $\|g\|$ is small and large when $|B_{ii}|$ or $\|g\|$ is large. Nevertheless, D_{ii} lies in a reasonable interval $[\underline{\Gamma}, \overline{\Gamma}]$ even if $\|g\| \to 0$.

The following algorithm summarizes our considerations.

Algorithm 3.1.

Data: $\rho \geq 0$, $\sigma \geq 0$, $\varepsilon > 0$, $0 < \underline{\beta} < 1$, $0 < \underline{\gamma} < 1$, $0 < \underline{\Gamma} < \overline{\Gamma}$, $0 < \underline{\delta} < 1$,. $0 < \overline{\omega} < 1$, $\Delta > 0$

Input: Sparsity pattern of matrices $\nabla^2 F$ and A. Initial choice of vector x .

Step 1: *Initiation.* Determine sparsity pattern of matrix B and find dense rows of matrix A. Choose a suitable linear system (original system, null-space transformation, range-space transformation), a Krylov-subspace method and a preconditioning technique based on complete or incomplete sparse Choleski decomposition (in the latter case carry out the symbolic decomposition of the matrix $(A_s^T D_s^{-1} A_s)^{-1}$). Choose a suitable merit function. Compute value $F = F(x)$ and vector $c = c(x)$. Set $u := 0$ and $i := 0$.

Step 2: *Termination.* Compute matrix $A = A(x)$ and vector $g = g(x, u)$. If $\|c\| \leq \varepsilon$ and $\|g\| \leq \varepsilon$, then terminate the computation (the solution is found). Otherwise set $i := i + 1$.

Step 3: *Approximation of the Hessian matrix.* Compute approximation G of the Hessian matrix $G(x, u)$ by using the differences of gradient $g(x, u)$ as in [5].

Step 4: *Direction determination.* If the range-space transformation is used, compute matrix $B = G + \rho A A^T$, otherwise set $B = G$. Determine positive definite diagonal matrix D as an approximation of the diagonal of B and carry out the numerical decomposition of matrix $(A_s^T D_s^{-1} A_s)^{-1}$ to obtain a representation of inversion C^{-1} of preconditioner C. Set $\omega = \min(\|g\| + \|c\|, 1/i, \overline{\omega})$. Determine the direction pair (d_x, d_u) as an inexact solution (with precision ω) of the chosen linear system by using the preconditioned Krylov-subspace method. Compute directional derivative $P'(0)$ of the chosen merit function

Step 5: *Restart.* If $-P'(0) < \underline{\gamma}\|d_x\|\|g\|$, determine the diagonal matrix D by (17), set $B = D$ and go to Step 4. Otherwise set $\alpha := \min(1, \Delta/\|d_x\|)$ and compute the value of merit function $P(\alpha)$.

Step 6: *Termination of the stepsize selection.* If $P(\alpha) - P(0) \leq \underline{\delta}\alpha P'(0)$, then set $x := x + \alpha d_x$, $u := u + \alpha d_u$ and go to Step 2.

Step 7: *Continuation of the stepsize selection.* Set $\alpha := \underline{\beta}\alpha$, compute the value of the merit function $P(\alpha)$ and go to Step 6.

We have used default values $\rho = 0$, $\sigma = 1.0$, $\varepsilon = 10^{-6}$, $\underline{\beta} = 0.5$, $\underline{\gamma} = 10^{-5}$, $\underline{\Gamma} = 0.005$, $\overline{\Gamma} = 500.0$, $\underline{\delta} = 10^{-4}$, $\overline{\omega} = 0.9$, $\Delta = 10^3$ in our computations. Parameter Δ plays a similar role as the trust region radius (see [4]). This parameter is not important theoretically but it has a considerable practical significance. If the linear KKT system is ill-conditioned, the direction vector d_x can be rather large and point

$x + d_x$ may not lie in a reasonable region (e.g. overflows can appear). Therefore, a careful choice of parameter Δ is necessary for some difficult problems (see Table 5). Another sensitive parameter is the penalty coefficient σ. If merit functions $P_1 - P_3$ are used, the default value $\sigma = 1.0$ is suitable for many problems. If merit functions $P_4 - P_5$ are used or if the problem is difficult, a careful choice of parameter σ is necessary (see Table 3 and Table 5). Diagonal matrix D defining preconditioner (9) was chosen in such a way that $D_{ii} = |B_{ii}|$ if $10^{-3} \le |B_{ii}| \le 10^6$, but $D_{ii} = 10^{-3}$ or $D_{ii} = 10^6$ in the other cases.

4 Computational experiments

The algorithms described above were used for solving 18 equality constrained optimization problems with 1000 variables proposed in [22], where further details concerning sparsity are given. These problems can be downloaded from the web page `http://www.cs.cas.cz/~luksan/test.html`. Even if these problems are academic (Hessian matrices $G(x, u)$ have bandwidth at most seven), they are not simple at all. Only five of them have positive definite and diagonally dominant Hessian matrices at the solution. Hessian matrices of seven problems are indefinite, four are singular and the remaining are not diagonally dominant at the solution.

Computational efficiency of individual Krylov-subspace methods (with merit function P_2 and restart procedure (17)) is demonstrated in Table 2. In this table, NIT is the total number of nonlinear iterations (number of linear KKT systems solved), NFE is the total number of function evaluations, NGE is the total number of gradient evaluations (NGE is much greater than NIT, since second order derivatives are computed by using gradient differences), LIT is the total number of linear iterations (iterations required by the selected Krylov-subspace method), RS is the total number of restarts and NS is the number of problems for which the value $\sigma \neq 1$ was used (this number expresses sensitivity of algorithm to the choice of parameters). Notice that unpreconditioned methods reported on the bottom of Table 2 failed to solve 2 problems (results correspond to 16 problems). All computations were performed on a Pentium PC (300Mhz) computer by subroutines implemented in the interactive system for universal functional optimization UFO [21]. Results in Table 2 imply several conclusions.

- Unpreconditioned Krylov-subspace methods are rather inefficient and they are not competitive with the direct BP method. Furthermore, the unpreconditioned CG method is much worse, in the indefinite case, than unpreconditioned methods based on residual smoothing or residual minimizing strategies.

- Krylov-subspace methods preconditioned by (9) are very efficient and the difference between the preconditioned CG method and preconditioned methods based on residual smoothing or residual minimizing strategies is removed. Implementations based on the explicit inversion (denoted by 1) are faster than implementations based on the Bunch-Parlett decomposition (denoted by 2), usually faster than the direct BP method. Notice that we have tested efficiency of several other preconditioners, see [22], but the constraint preconditioner (9) was shown to be best. Notice that the efficiency of preconditioned methods could be influenced by the test problems used, but the additional tests given below confirm our conclusions.

- Methods based on the null-space transformation (NPCG) require the lowest number of CG iterations and the shortest computational time. On the other hand, CG methods applied to the original system (PCG, PSCG) are less sensitive to the choice of parameters. Methods based on the range-space transformation (RSCG) are very sensitive to the choice of parameters (they require a careful choice of parameter ρ which influences the choice of parameter σ). Notice that we have used unpreconditioned range-space transformation algorithms, since they were proved to be more efficient than the preconditioned ones for our set of test problems.

Method	NIT	NFE	NGE	LIT	RS	NS	time
BP	263	334	1969	-	14	4	9.78
PCG1	262	308	1995	838	15	1	8.63
PCG2	266	312	2023	851	19	1	10.00
PSCG1	270	300	2055	826	9	2	9.18
PSCG2	275	307	2090	835	13	2	10.49
PCR1	244	283	1879	704	14	6	8.96
PCR2	245	285	1886	707	14	6	9.28
PQMR1	284	318	2158	832	9	5	9.44
PQMR2	306	343	2305	889	13	5	10.27
PMCG1	271	330	2051	634	10	5	9.01
PMCG2	279	337	2107	664	11	5	9.73
NPCG1	245	282	1865	516	10	5	7.97
NPCG2	247	284	1880	540	11	6	8.79
RSCG1	265	358	1994	599	10	10	8.24
RSCG2	265	366	2012	1529	8	10	8.84
CG	342	407	2473	24666	14	9	32.95
SCG	252	336	1917	12469	20	6	21.81
CR	264	365	1985	12627	25	5	21.26
QMR	252	333	1909	12272	20	5	21.81

Table 2

To demonstrate properties of individual merit functions, we propose Table 3, which contains results for PCG1 algorithm (similar results were obtained for other Krylov-subspace algorithms). Rows denoted by P0 refer to computations with no merit function and rows denoted by P1 - P5 correspond to merit functions $P_1 - P_5$, respectively. Furthermore, the letter R denotes the use of restart procedure (17) and the letter P denotes the determination of penalty coefficients by Theorem 8.

Function	NIT	NFE	NGE	LIT	RS	NS	time
P0	336	336	2503	1299	10	2	9.45
P1 - R	266	325	2019	834	12	1	8.63
P1 - P	266	320	2013	829	2	3	8.85
P2 - R	262	308	1995	838	15	1	8.63
P2 - P	267	304	2031	829	8	2	8.68
P3 - R	257	296	1960	793	11	3	8.52
P3 - P	261	292	1991	810	7	5	8.73
P4 - R	261	309	1981	830	5	6	8.79
P4 - P	337	381	2513	1535	3	7	9.62
P5 - R	326	874	2444	1545	21	8	9.83
P5 - P	331	779	2480	1435	3	9	9.55

Table 3

The results contained in Table 3 show that merit functions P1 - P3 are more advantageous than P4 - P5 even if the global convergence can be proved only for the latter ones in general. Strategies based on restarts are more efficient than strategies based on determination of penalty coefficients (measured by the sensitivity to the choice of parameters and by the computational time). Surprisingly, computations with no merit function (but with a careful choice of the maximum stepsize Δ and a control of restarts based on the use of P_2) are relatively successful in these problems.

No.	Problem	n	m	m_A	m_B	n_D
1	DTOC1NA	1495	990	15840	26740	-
2	DTOC1NB	1495	990	15840	26740	-
3	DTOC1NC	1495	990	15840	26740	-
4	DTOC1ND	2998	1996	13972	22459	-
5	DTOC2	2998	1996	7984	16471	-
6	DTOC3	1499	998	3493	5491	-
7	DTOC4	1499	998	3493	5491	-
8	DTOC5	1999	999	2979	4996	-
9	DTOC6	1001	500	1500	2501	-
10	OPTCTRL3	1202	800	2800	4402	-
11	ORTHRDM2	1003	500	2500	4506	3
12	ORTHREGA	517	256	1792	3343	5
13	ORTHREGC	1005	500	3500	6515	5
14	ORTHREGD	1003	500	2500	4506	3
15	ORTHREGE	1506	1000	8000	9517	6
16	ORTHREGF	1205	400	3200	8415	5
17	ORTHRGDM	1003	500	2500	4506	3
18	ORTHRGDS	1003	500	2500	4506	3

Table 4

For a better comparison of individual algorithms, we performed additional tests with problems from the widely used CUTE collection [3]. Table 4 contains a list of these problems together with their dimensions. Here n is the number of variables, m is the number of constraints, m_A is the number of nonzero elements in the matrix A, m_B is the number of nonzero elements in the matrix B and n_D is the number of dense rows in the matrix A.

Results corresponding to the above test problems are given in Table 5. The numbers NIT, NFE, NGE, LIT and RS have the same meaning as in Table 2. In addition, we introduce values of penalty coefficient σ, maximum allowable stepsizes Δ and the final values $F(x^*)$ obtained by individual algorithms. Asterisk in the column correspondig to σ denotes that $\rho = \sigma$. Otherwise $\rho = 0$ (ρ is the parameter used in (13)). The values $\sigma = 1.0$ (or $\sigma = 1.5$ for NPCG and RPSCG) and $\Delta = 1000$ were chosen as default. Parameter Δ, which has similar meaning as the trust region bound, had to be often decreased. Results in Table 5 imply several conclusions.

- Methods applied to the original system (BP, PCG, PSCG) are very robust. Moreover, preconditioned iterative methods (PCG, PSCG) are always competitive with the direct BP method and they are significantly better than BP for ill-conditioned problems with more complicated sparsity patterns (ORTHREGA, ORTHREGE). However, it is necessary to note that implementations PCG1 and PSCG1 did not converge for problem ORTHREGA. They had to be replaced by implementations PCG2 and PSCG2, respectively, for a successful solution of this problem. This fact indicates that implementations based on the Bunch-Parlett decomposition (denoted by 2) are more stable for ill-conditioned problems than implementations based on the explicit inversion (denoted by 1) especially if matrix A contains dense rows.

- Methods based on the null-space transformation (NPCG) are very efficient as well and have similar properties as methods applied to the original system. However, matrix $D = I$ was used in preconditioner C for problem DTOC3, since the diagonal of B was unsuitable in this case.

- Methods based on the range-space transformation (RSCG) are usually less efficient and very sensitive to the choice of parameters. Moreover, individual methods sometimes fail to converge so we need to use algorithms RSCG1, RPSCG1 and RSCG2 alternatively. Nevertheless, these algorithms can be very efficient in some cases (ORTHREGA, ORTHREGC, ORTHREGF), which is explained in the next conclusion. This fact indicates that a software package for large sparse equality constrained optimization should contain several different methods.

- Notice that computational times for individual methods can differ considerably even if these methods use approximately the same number of iterations (for problem ORTHREGA the fastest method RSCG1 uses, surprisingly, the largest numbers of iterations NIT and LIT). This is given by the different numbers of operations per iteration, which can be rather large for the BP method and methods based on the Bunch-Parlett decomposition especially if matrix A contains dense rows.

No.	Method	NIT	NFE	NGE	LIT	RS	σ	Δ	$F(x^*)$	time
1	BP	7	7	252	-	0	1.0	1000	12.7020	17.63
	PCG1	7	7	252	10	0	1.0	1000	12.7020	18.68
	PSCG1	7	7	252	10	0	1.0	1000	12.7020	17.85
	NPCG1	7	7	252	7	0	1.5	1000	12.7020	17.96
	RPSCG1	8	8	288	13	0	1.5	1000	12.7020	21.14
2	BP	6	6	216	-	0	1.0	1000	15.9378	15.54
	PCG1	7	7	252	14	0	1.0	1000	15.9378	18.67
	PSCG1	7	7	252	13	0	1.0	1000	15.9378	17.90
	NPCG1	6	6	216	8	0	1.5	1000	15.9378	15.82
	RSCG2	13	20	468	450	0	1.5	1000	15.9378	32.74
3	BP	10	16	360	-	0	0.1	1000	24.9698	24.50
	PCG1	9	15	324	32	0	1.0	1000	24.9698	23.73
	PSCG1	8	8	288	23	0	1.0	1000	24.9698	20.15
	NPCG1	9	11	324	37	0	1.5	1000	24.9698	22.68
	RSCG2	13	23	468	968	0	100.0	1000	24.9698	35.92
4	BP	6	6	90	-	0	1.0	1000	23.7703	20.76
	PCG1	8	8	120	17	0	1.0	1000	23.7703	27.57
	PSCG1	7	7	105	17	0	1.0	1000	23.7703	24.55
	NPCG1	7	7	105	14	0	1.5	1000	23.7703	24.50
	RSCG2	16	33	240	720	0	1000.0	1000	23.7703	58.44
5	BP	7	8	84	-	0	1.0	1000	0.501840	14.01
	PCG1	7	8	84	36	0	1.0	1000	0.497223	14.33
	PSCG1	7	8	84	38	0	1.5	1000	0.497223	14.50
	NPCG1	8	12	96	20	0	1.5	1000	0.501840	15.87
	RPSCG1	13	17	156	333	0	100.0	1000	0.497223	28.61
6	BP	2	2	16	-	0	1.0	1000	235.084	1.04
	PCG1	2	2	16	3	0	1.0	1000	235.084	1.15
	PSCG1	2	2	16	3	0	1.0	1000	235.084	1.15
	NPCG1	3	3	24	3	0	1.5	1000	235.084	1.32
	RPSCG1	2	2	16	1	0	1.5	1000	235.084	1.15
7	BP	4	4	32	-	0	1.0	1000	2.88285	1.54
	PCG1	4	4	32	5	0	1.0	1000	2.88285	1.54
	PSCG1	4	4	32	5	0	1.0	1000	2.88285	1.65
	NPCG1	5	5	40	8	0	1.5	1000	2.88285	1.92
	RPSCG1	81	111	648	5317	0	10.0	10	2.88285	46.58
8	BP	4	4	20	-	0	1.0	1000	1.53495	1.92
	PCG1	4	4	20	7	0	1.0	1000	1.53495	2.14
	PSCG1	4	4	20	7	0	1.0	1000	1.53495	2.14
	NPCG1	5	6	25	10	0	1.5	1000	1.53495	2.53
	RPSCG1	5	5	25	7	0	1.5	1000	1.53495	2.59
9	BP	10	10	50	-	0	1.0	1000	6846.61	1.43
	PCG1	11	11	55	17	0	1.0	1000	6846.61	1.48
	PSCG1	11	11	55	14	0	1.0	1000	6846.61	1.54
	NPCG1	10	10	50	10	0	1.5	1000	6846.61	1.49
	RPSCG1	11	11	55	23	0	1.5	1000	6846.61	1.82

Table 5

No.	Method	NIT	NFE	NGE	LIT	RS	σ	Δ	$F(x^*)$	time
10	BP	19	32	160	-	0	1.0	50	25140.2	3.79
	PCG1	19	19	152	62	0	1.0	50	25140.2	3.46
	PSCG1	13	15	104	60	0	1.0	100	25140.2	2.53
	NPCG1	13	30	112	60	0	1.5	100	188382.0	2.80
	RPSCG1	25	30	200	61	0	1.5	1000	25140.2	4.61
11	BP	6	7	36	-	0	1.0	5	38.5877	1.32
	PCG1	7	8	42	16	0	1.0	5	38.5877	1.43
	PSCG1	8	9	48	23	0	1.0	5	38.5877	1.59
	NPCG1	7	8	42	14	0	1.5	5	38.5877	1.54
	RSCG2	14	20	84	65	0	1.5	5	38.5877	2.58
12	BP	17	40	136	-	3	1.0	50	1664.80	3:57.94
	PCG2	12	12	96	56	0	1.0	10	1664.80	11.10
	PSCG2	11	11	88	69	0	1.0	10	1664.80	9.61
	NPCG2	12	12	96	40	0	1.5	10	1664.80	10.60
	RSCG1	58	58	464	153	0	*100.0	50	1414.06	4.34
13	BP	18	36	144	-	2	1.0	50	18.7906	4.34
	PCG1	14	33	112	69	2	1.0	5	18.7906	3.63
	PSCG1	12	13	96	73	0	1.0	5	18.8037	3.13
	NPCG1	12	12	96	48	0	1.5	5	18.7906	3.02
	RSCG1	11	15	88	17	0	*100.0	5	18.7906	2.96
14	BP	7	8	42	-	0	1.0	50	151.235	1.48
	PCG1	8	9	48	24	0	1.0	50	151.235	1.59
	PSCG1	9	10	54	30	0	1.0	50	151.235	1.81
	NPCG1	8	8	48	35	0	1.5	50	151.235	1.65
	RSCG2	17	23	102	770	0	1.5	50	151.235	5.27
15	BP	23	26	207	-	0	0.01	1	247.182	5:06.76
	PCG1	23	26	207	207	0	0.01	1	247.182	15.93
	PSCG1	26	29	234	248	0	0.01	1	247.700	18.18
	NPCG1	24	28	216	94	0	0.01	1	244.258	16.20
	RSCG2	25	25	225	452	0	0.01	1	299.346	31.91
16	BP	71	116	639	-	13	1.0	1	18.2567	18.57
	PCG1	92	226	828	1282	17	1.0	1	16.6200	27.47
	PSCG1	45	55	405	722	5	1.0	1	29.3773	13.84
	NPCG1	61	94	549	795	12	1.5	1	21.7654	17.91
	RSCG1	27	47	252	366	0	1.5	1	16.2006	7.96
17	BP	7	7	42	-	0	1.0	5	150.549	1.70
	PCG1	10	10	60	35	0	1.0	5	150.549	1.92
	PSCG1	10	10	60	37	0	1.0	5	150.549	1.93
	NPCG1	8	8	48	19	0	1.5	5	150.549	1.64
	RSCG2	23	23	138	715	0	1.5	1	150.549	7.96
18	BP	27	84	162	-	2	0.001	1000	151.235	4.88
	PCG1	15	28	90	65	1	0.01	1000	151.235	2.80
	PSCG1	24	61	144	153	3	0.001	1000	151.235	4.58
	NPCG1	16	26	96	62	0	0.001	1000	151.235	3.02
	RSCG2	18	21	108	765	0	0.001	1000	151.235	5.65

Table 5 - continued

References

[1] E.Bodon, A.Del Popolo, L.Lukšan, E.Spedicato: Computational experiments with ABS algorithms for KKT linear systems. To appear in Optimization Methods and Software.

[2] P.T.Boggs, J.W.Tolle: Sequential quadratic programming. Acta Numerica (1995) 1-51.

[3] I.Bongartz, A.R.Conn, N.Gould, P.L.Toint: CUTE: constrained and unconstrained testing environment. ACM Transactions on Mathematical Software 21 (1995) 123-160.

[4] A.R.Conn, N.Gould, P.L.Toint: Trust-Region Methods. SIAM Philadelphia, 2000.

[5] A.R.Curtis, M.J.D.Powell, J.K.Reid: On the estimation of sparse Jacobian matrices. IMA Journal of Aplied Mathematics 13 (1974) 117-119.

[6] R.S.Dembo, S.C.Eisenstat, T.Steihaug: Inexact Newton methods. SIAM J. Numerical Analysis 19 (1982) 400-408.

[7] J.E.Dennis, R.B.Schnabel: Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[8] I.S.Duff, M.Munksgaard, H.B.Nielsen, J.K.Reid: Direct solution of sets of linear equations whose matrix is sparse and indefinite. J. Inst. Maths. Applics. 23 (1979) 235-250.

[9] I.S.Duff, J.K.Reid: MA27 - A set of Fortran subroutines for solving sparse symmetric sets of linear equations. Tech. Report AERE R10533, HMSO, London, 1982.

[10] R.Fletcher: Practical Methods of Optimization. Wiley, Chichester, 1987.

[11] R.Fletcher, T.Johnson: On the stability of null-space methods for KKT systems. SIAM J. Matrix Analysis and Applications 18 (1997) 938-958.

[12] R.W.Freund, N.M.Nachtigal: A new Krylov-subspace method for symmetric indefinite linear systems. Report No. ORNL/TM-12754, Oak Ridge National Laboratory, Oak ridge 1994.

[13] P.E.Gill, W.Murray: Newton type methods for unconstrained and linearly constrained optimization. Mathematical Programming 7 (1974) 311-350.

[14] N.I.M.Gould: On practical conditions for the existence and uniqueness of solutions to the general equality quadratic programming problem. Mathematical Programming 32 (1985) 90-99.

[15] N.I.M.Gould, M.E.Hribar, J.Nocedal: On the solution of equality constrained quadratic programming problems arising in optimization. Technical Report RAL-TR-1998-069, Rutherford Appleton Laboratory, 1998.

[16] A.Griewank: Evaluating derivatives. Principles and Techniques of Algorithmic Differentiation. SIAM Philadelphia, 2000.

[17] S.P.Han: A globally convergent method for nonlinear programming. J. of Optimization Theory and Applications 22 (1977) 297-309.

[18] M.R.Hestenes, C.M.Stiefel: Methods of conjugate gradient for solving linear systems. J. Res. NBS 49 (1964) 409-436.

[19] Keller C., Gould N.I.M., Wathen A.J.: Constraint preconditioning for indefinite linear systems. SIAM J. on Matrix Analysis and Applications 21 (2000) 1300-1317.

[20] M.Lalee, J.Nocedal, T Plantenga: On the implementation of an algorithm for large-scale equality constrained optimization. SIAM J. Optimization 8 (1998) 682-706.

[21] L.Lukšan, M.Šiška, M.Tůma, J.Vlček, N.Ramešová: UFO 2000. Interactive system for universal functional optimization. Research Report No. V-826, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, 2000.

[22] L.Lukšan, J.Vlček: Indefinitely preconditioned inexact Newton method for large sparse equality constrained nonlinear programming problems. Numerical Linear Algebra with Applications 5 (1998) 219-247.

[23] L.Lukšan, J.Vlček: Conjugate gradient methods for saddle point linear systems. In: Proceedings of the SANM conference, Nečtiny 1999.

[24] L.Lukšan, J.Vlček: Sparse and partially separable test problems for unconstrained and equality constrained optimization. Report No. 767, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague 1999.

[25] S.G.Nash, A.Sofer: Preconditioning reduced matrices. SIAM J. Matrix Analysis and Applications. 17 (1996) 47-68.

[26] C.C.Paige, M.A.Saunders: Solution of sparse indefinite systems of linear equations. SIAM J. Numerical Analysis 12 (1975) 617-629.

[27] I.Perugia, V.Simoncini: Block-diagonal and indefinite symmetric preconditioners for mixed finite element formulations. Numerical Linear Algebra with Applications 7 (2000) 1-32.

[28] A.Ramage, A.J.Wathen: Iterative solution techniques for the Stokes and Navier-Stokes equations. International J. for Numerical Methods in Fluids 19 (1994) 67-83.

[29] M.Rozložník, V.Simoncini: Short-term recurrences for indefinite preconditioning of saddle point problems. Report No. SAM 2000-08, SAM, ETH Zurich, 2000

[30] A.Rusten, A.J.Winther: A preconditioned iterative method for saddle-point problems. SIAM J. on Matrix Analysis and Applications 13 (1992) 887-904.

[31] K.Schittkowski: On the convergence of a sequential quadratic programmming method with an augmented Lagrangian type line search function. Math. Operationsforsch. U. Statist., Ser. Optimization 14 (1983) 197-216. 887-904.

[32] E.Stiefel: Relaxationsmethoden bester Strategie zur Lösung linearer Gleichungssysteme. Comm. Math. Helv. 29 (1955) 157-179.

[33] R.J.Vanderbei: Symmetric quasi-definite matrices. SIAM J. Optimization, 5 (1995) 100-113.

[34] R.Weiss: Parameter-free Iterative Linear Solvers. Akademie Verlag, Berlin, 1996.