Department of Applied Mathematics, University of Venice

WORKING PAPER SERIES

Giovanni Fasano

Notes on a 3-term Conjugacy Recurrence for the Iterative Solution of Symmetric Linear Systems

Working Paper n. 179/2008
November 2008

ISSN: 1828-6887

Notes on a 3-term Conjugacy Recurrence for the Iterative Solution of Symmetric Linear Systems[*]

Giovanni Fasano
<fasano@unive.it>
Dept. of Applied Mathematics
University of Venice

(November 2008)

Abstract. We consider a 3-term recurrence, namely CG_2step, for the iterative solution of symmetric linear systems. The new algorithm generates conjugate directions and extends some standard theoretical properties of the Conjugate Gradient (CG) method [10]. We prove the finite convergence of CG_2step, and we provide some error analysis.
Then, we introduce preconditioning for CG_2step, and we prove that standard error bounds for the CG also hold for our proposal.

Keywords: Iterative methods, 3-term recurrences, Conjugate Gradient method, Error Analysis, Preconditioning.

JEL Classification Numbers: C61, C63.

MathSci Classification Numbers: 90C99, 65K05.

Correspondence to:

Giovanni Fasano	Dept. of Applied Mathematics, University of Venice
	Dorsoduro 3825/e
	30123 Venezia, Italy
Phone:	[++39] (041)-234-6922
Fax:	[++39] (041)-522-1756
E-mail:	fasano@unive.it

[*] The author thanks *Progetto di Ricerca VISIR* and *Programma PRIN 20079PLLN7 "Nonlinear Optimization, Variational Inequalities, and Equilibrium Problems"*, for the support received.

1 Introduction

Several real applications have required in the last decades the solution of both dense and sparse challenging large scale linear systems. In a parallel process we have seen relevant advances in the development of efficient and reliable iterative solvers for symmetric linear systems [4]. More recently, parallel computing has also provided effective new tools to tackle problems with specific structure [16, 15].
Iterative solvers for large linear systems often suffer from finite precision along with round-off in the computation. However, depending on the problem in hand, the linear systems in optimization frameworks may possibly not require large accuracy in the solution. In these cases the stability analysis provides key suggestions [11], for the practical implementation of the algorithms. In addition, optimization frameworks often require the computation of reliable search directions, whose effective computation may result from the use of appropriate linear systems solvers, based on conjugate directions. Indeed, Newton-type directions can be easily generated using conjugate vectors, both with positive definite and indefinite linear system [7, 5, 2].

Here we propose a 3-term conjugacy recurrence for solving symmetric positive definite linear systems. We analyze some theoretical properties of the new algorithm, aiming to possibly controlling the inaccuracy which arises in the CG. To a large extent our method encompasses the CG and has some similarities with the Lanczos process, which belongs to the class of *3-term recurrence* iterative methods, too. Now we briefly review the latter methods (see also [6]), in order to detail the differences with our proposal.

The CG algorithm is commonly used to iteratively solving the linear system

$$Ax = b, \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is *symmetric positive definite*, $b \in \mathbb{R}^n$. Observe that the CG is often applied to the *preconditioned* version of (1), i.e. $\mathcal{M}Ax = \mathcal{M}b$, where $\mathcal{M} \succ 0$ is the preconditioner. Though the theory for the CG requires A to be positive definite, in several practical applications the CG is successfully used when A is indefinite, too [9]. The CG method is reported in Table 1, and at Step k it iteratively generates the pair of vectors r_k and p_k. The sequences $\{r_k\}$ (*residuals*) and $\{p_k\}$ (*conjugate directions*) satisfy at Step k [6]

$$\begin{array}{lll} \text{orthogonality property}: & r_i^T r_j = 0, & 0 \le i \ne j \le k, \qquad (2) \\ \text{conjugacy property}: & p_i^T A p_j = 0, & 0 \le i \ne j \le k, \qquad (3) \end{array}$$

and the CG method solves (1) in at most n iterations (finite convergence), i.e. $Ax_h = b$ for some $h \le n$. Condition (2) states that at Step k, the CG method computes the residual r_k such that the so called Ritz-Galerkin condition

$$r_k \perp \mathcal{K}_{k-1}(r_0, A)$$

holds with

$$\mathcal{K}_{k-1}(r_0, A) = span\{b, Ab, A^2b, \dots, A^{k-1}b\} \equiv span\{r_0, \dots, r_{k-1}\},$$

where $\mathcal{K}_{k-1}(r_0, A)$ is a Krylov subspace of dimension k. Moreover, the direction p_k is computed at Step k by imposing the conjugacy condition $p_k^T A p_{k-1} = 0$. It can be proved that the latter condition implicitly satisfies relations (3). Moreover, the vectors $\{p_0, \dots, p_k\}$ are linearly independent.
We remark that on practical problems, due to *finite precision* and *roundoff* in the iterative computation of the sequences $\{p_k\}$ and $\{r_k\}$, if $|i-j|$ is large relations (2)-(3) may fail. Thus, in the practical implementation of the CG, some theoretical properties may not be satisfied and in particular the conjugacy properties (3) may progressively fail.

The Lanczos process (see Table 2) is another iterative Krylov-based method, widely used to solve (1) or its preconditioned version. Unlike the CG, here the matrix A is possibly indefinite (though non singular). Similarly to the CG, at Step k the Lanczos process generates the sequence $\{u_k\}$ (*Lanczos vectors*) which satisfies at Step k

$$\text{orthogonality property}: \qquad u_i^T u_j = 0, \qquad 0 \le i \ne j \le k.$$

In addition, in at most n steps (finite convergence) the Lanczos process provides the solution of (1). Observe that by a simple inspection of Step k in Tables 1 and 2, the CG is slightly computationally cheaper than the Lanczos process. On the other hand, the CG is slightly less efficient than the Lanczos process if no preconditioning is used [1]. We recall that both the CG and the Lanczos process are 3-term recurrence methods, with respect to the residuals and the Lanczos vectors. In other words, the following relations hold for $k \ge 1$

$$r_{k+1} \in span\{Ar_k, r_k, r_{k-1}\}$$

$$u_{k+1} \in span\{Au_k, u_k, u_{k-1}\}.$$

Whenever A is positive definite and $x_0 = 0$ in Tables 1-2, a full theoretical correspondence between the sequences $\{r_k\}$ and $\{u_k\}$ is known [3], being

$$u_k = s_k \frac{r_k}{\|r_k\|}, \qquad s_k \in \{-1, +1\}.$$

Unfortunately, in practical computation the latter theoretical correspondence may be unattainable, due to the nature of matrix A.

The algorithm CG_2step provides a framework which partially encompasses the CG and resembles the Lanczos process. In particular, the CG_2step generates both conjugate directions and orthogonal residuals, as the CG and the Lanczos process. However, the CG_2step yields a 3-term recurrence with respect to conjugate directions, and a 4-term recurrence with respect to the residuals. We remark that our proposal draws its inspiration from the necessity to delay the conjugacy loss of the CG, which occurs in (3) when $|i-j|$ is large and finite precision is adopted.

We complete this section observing that the CG in Table 1 simply stores at Step k the vectors r_{k-1} and p_{k-1}, in order to compute respectively r_k and p_k. Unlike the CG, at Step k our method requires the storage of one additional vector (namely p_{k-2}), which contains some information from iteration $k-2$. The idea of storing at Step k some information from iterations preceding Step $k-1$, is not new for Krylov-based methods. Some examples which differ from our approach are [15], for unsymmetric linear systems.

In this paper we only use standard notation and symbols. In Section 2 we describe our method and some related properties. Section 3 shows a numerical experience with our proposal, without introducing preconditioning. Then, in Section 4 we detail and comment the preconditioned version of our method. Finally, the section of Conclusions completes the paper and specifies the future work.

2 The CG_2step method

We sketch in Table 3 our new CG-based algorithm, namely algorithm CG_2step. We highlight that the computation of the direction p_k at Step k is the relevant difference between the CG and the CG_2step method. In particular, in Table 3 the pair of coefficients σ_{k-1} and ω_{k-1} is computed so that

$$\begin{aligned} p_k^T A p_{k-1} &= 0 \\ p_k^T A p_{k-2} &= 0, \end{aligned} \tag{4}$$

i.e. the conjugacy between the the direction p_k and both the directions p_{k-1} and p_{k-2} is imposed. On the other hand, the residual r_k is computed by imposing the orthogonality condition $r_k^T p_{k-1} = 0$, as in the standard CG. The resulting method is *evidently a bit more expensive than the CG*. Moreover, observe that after some computations we can prove that $r_{k+1} \in span\{Ar_k, r_k, r_{k-1}, r_{k-2}\}$. Thus, CG_2step provides a 4-term recurrence with respect to the residuals, and a 3-term recurrence with respect to the conjugate directions.

Assumption 2.1 *The matrix A in (1) is symmetric positive definite.*

Lemma 2.1 *Let Assumption 2.1 hold. At Step k of the* CG_2step *algorithm we have*

$$Ap_j \in span\{p_{j+1}, p_j, p_{j-1}\}, \qquad j \leq k-1. \tag{5}$$

<u>Proof</u>
From the Step 0 of CG_2step, relation (5) holds for $j = 0$. Then, for $j = 1, \dots, k-1$ the Step $j+1$ of CG_2step directly yields (5). □

Theorem 2.2 *Let Assumption 2.1 hold. At Step k of the* CG_2step *algorithm the directions $p_0, p_1, \dots, p_k$ are mutually conjugate, i.e. $p_i^T A p_j = 0$, with $0 \leq i \neq j \leq k$.*

<u>Proof</u>
The statement holds for $k = 1$, as a consequence of the choice of the coefficient σ_0 at Step 0. Suppose it holds for $k-1$; then, we have

$$\begin{aligned} p_k^T A p_j &= (Ap_{k-1} - \sigma_{k-1} p_{k-1} - \omega_{k-1} p_{k-2})^T A p_j \\ &= (Ap_{k-1})^T A p_j - \sigma_{k-1} p_{k-1}^T A p_j - \omega_{k-1} p_{k-2}^T A p_j = 0, \qquad j \leq k-1. \end{aligned}$$

In particular, for $j = k-1$ and $j = k-2$ the choice of the coefficients σ_{k-1} and ω_{k-1} yields directly $p_k^T A p_{k-1} = p_k^T A p_{k-2} = 0$. For $j < k-2$, the inductive hypothesis and Lemma 2.1 again yield the conjugacy. □

Lemma 2.3 *Let Assumption 2.1 hold. Then we have at Step k of the* CG_2step *algorithm*

$$(Ap_k)^T(Ap_i) = \begin{cases} \|Ap_k\|^2, & \text{if } i = k, \\ p_k^T Ap_k, & \text{if } i = k-1, \\ \emptyset, & \text{if } i \le k-2. \end{cases}$$

Proof
The statement is a trivial consequence of Step k, Lemma 2.1 and Theorem 2.2. □

Observe that from the previous lemma, a simplified expression for the coefficient σ_{k-2}, at Step k of CG_2step is available, inasmuch as

$$\sigma_{k-2} = \frac{p_{k-1}^T Ap_{k-1}}{p_{k-2}^T Ap_{k-2}}. \tag{6}$$

Relation (6) has a remarkable importance: it avoids the storage of the vector Ap_{k-2} at Step k, requiring only the storage of the quantity $p_{k-2}^T Ap_{k-2}$. Also observe that despite the CG, the sequence $\{r_k\}$ in CG_2step is computed independently from the sequence $\{p_k\}$. Moreover, the residual r_k is simply computed at Step k in order to check the stopping condition for the algorithm.

The following result proves that the CG_2step algorithm substantially recovers the main theoretical features of the standard CG.

Theorem 2.4 *Let Assumption 2.1 hold. Let $r_{k+1} \neq 0$ at Step $k+1$ of the* CG_2step *algorithm. Then, the directions $p_0, p_1, \ldots, p_k$ and the residuals $r_0, r_1, \ldots, r_{k+1}$ satisfy*

$$r_{k+1}^T p_j = 0, \qquad j \le k, \tag{7}$$
$$r_{k+1}^T r_j = 0, \qquad j \le k. \tag{8}$$

Proof
From Step $k+1$ of CG_2step we have $r_{k+1} = r_k - \alpha_k Ap_k = r_j - \sum_{i=j}^k \alpha_i Ap_i$, $j \le k$. Then, from Theorem 2.2 and the choice of coefficients α_j we obtain

$$r_{k+1}^T p_j = \left(r_j - \sum_{i=j}^k \alpha_i Ap_i \right)^T p_j = r_j^T p_j - \sum_{i=j}^k \alpha_i p_i^T Ap_j = 0, \qquad j \le k,$$

which proves (7). As regards relation (8), for $k = 0$ we obtain from the choice of α_0

$$r_1^T r_0 = r_1^T p_0 = 0.$$

Then, assuming by induction that (8) holds for $k-1$, we have

$$\begin{aligned} r_{k+1}^T r_j &= (r_k - \alpha_k Ap_k)^T r_j = (r_k - \alpha_k Ap_k)^T \left(r_0 - \sum_{i=0}^{j-1} \alpha_i Ap_i \right) \\ &= r_k^T r_0 - \sum_{i=0}^{j-1} \alpha_i r_k^T Ap_i - \alpha_k p_k^T Ar_0 + \sum_{i=0}^{j-1} \alpha_i \alpha_k (Ap_k)^T Ap_i, \qquad j \le k. \end{aligned}$$

The inductive hypothesis and Theorem 2.2 yield

$$r_{k+1}^T r_j \;=\; -\sum_{i=0}^{j-1} \alpha_i r_k^T \left(p_{i+1} + \sigma_i p_i + \omega_i p_{i-1}\right) + \sum_{i=0}^{j-1} \alpha_i \alpha_k (Ap_k)^T Ap_i, \qquad j \leq k. \tag{9}$$

Therefore, if $j = k$ the relation (7) along with Lemma 2.3 and the choice of α_k yield

$$r_{k+1}^T r_k \;=\; -\alpha_{k-1} r_k^T p_k + \alpha_{k-1}\alpha_k p_k^T A p_k \;=\; 0.$$

On the other hand, if $j < k$ in (9), the inductive hypothesis, relation (7) and Lemma 2.3 yield (8). □

Finally, we prove that likewise the CG algorithm, in at most n iterations CG_2step determines the solution of the linear system (1), so that finite convergence holds.

Lemma 2.5 *Let Assumption 2.1 hold. Then, at Step k of the* CG_2step *algorithm the vectors $p_0, \ldots, p_k$ are linearly independent.*

Proof
Let by contradiction

$$\sum_{i=0}^{k} \gamma_i p_i = 0, \qquad \gamma \in \mathbb{R}^k, \; \gamma \neq 0. \tag{10}$$

Then, multiplying (10) by Ap_h, with $0 \leq h \leq k$, we obtain

$$\sum_{i=0}^{k} \gamma_i p_h^T A p_i = 0, \qquad \gamma \in \mathbb{R}^k, \; \gamma \neq 0,$$

and the conjugacy yields for $0 \leq h \leq k$

$$0 = \sum_{i=0}^{k} \gamma_i p_h^T A p_i = \gamma_h p_h^T A p_h.$$

Finally, since A is positive definite the last equality implies $\gamma_h = 0$, for any $0 \leq h \leq k$, which contradicts relation (10). □

Theorem 2.6 *Let Assumption 2.1 hold. Then, in at most n iterations the* CG_2step *algorithm computes the solution of the linear system (1), i.e. $Ax_h = b$, where $h \leq n$.*

Proof
From relation (8) at most n orthogonal residuals may be generated in the sequence $\{r_h\}$. Thus, suppose at Step h we have $r_h = 0$, with $h \leq n$. Then, by definition

$$x_h = x_0 + \sum_{j=0}^{h-1} \alpha_j p_j, \tag{11}$$

therefore, recalling that $r_h = r_0 - \sum_{j=0}^{h-1} \alpha_j A p_j$, multiplying (11) by $-A$ and adding b we obtain

$$b - Ax_h = r_0 - \sum_{j=0}^{h-1} \alpha_j A p_j = r_h = 0,$$

which proves that x_h is a solution of the linear system $Ax = b$. $\square$

We observe that the geometry of vectors $\{p_k\}$ and $\{r_k\}$ in CG_2step is substantially different with respect to the CG. Indeed, unlike the latter scheme where $r_k^T p_k = \|r_k\|^2 > 0$, for any k, for the CG_2step we have

$$\frac{p_k^T A p_k}{p_{k-1}^T A p_{k-1}} = \frac{(Ap_{k-1})^T A p_k}{p_{k-1}^T A p_{k-1}} = -\frac{\|r_k\|^2}{\alpha_k \alpha_{k-1} p_{k-1}^T A p_{k-1}} = -\frac{\|r_k\|^2 p_k^T A p_k}{r_k^T p_k r_{k-1}^T p_{k-1}},$$

and since $A \succ 0$ we obtain

$$(r_k^T p_k)(r_{k-1}^T p_{k-1}) < 0.$$

2.1 Issues on the conjugacy loss

Here we consider a simplified approach to describe the conjugacy loss for both the algorithms CG and CG_2step, under Assumption 2.1. Suppose that both the algorithms CG and CG_2step performed Step $k+1$, and in the practical implementation a nonzero *conjugacy error* $\varepsilon_{k,j}$ respectively occurs between directions p_k and p_j, $j < k$, i.e.

$$\varepsilon_{k,j} \;=\; p_k^T A p_j \;\neq\; 0, \qquad\qquad j < k.$$

Then, we calculate the conjugacy error

$$\varepsilon_{k+1,j} \;=\; p_{k+1}^T A p_j, \qquad\qquad j \le k,$$

for both the CG and the CG_2step. First observe that at Step $k+1$ of Table 1 we have

$$\begin{aligned} \varepsilon_{k+1,j} &= (r_{k+1} + \beta_k p_k)^T A p_j && (12) \\ &= (p_k - \beta_{k-1} p_{k-1} - \alpha_k A p_k)^T A p_j + \beta_k \varepsilon_{k,j} && (13) \\ &= (1 + \beta_k)\varepsilon_{k,j} - \beta_{k-1}\varepsilon_{k-1,j} - \alpha_k (Ap_k)^T A p_j. && (14) \end{aligned}$$

Then, from relation $Ap_j = (r_j - r_{j+1})/\alpha_j$ and relations (2)-(3) we have for the CG

$$(Ap_k)^T A p_j \;=\; \begin{cases} -\frac{p_k^T A p_k}{\alpha_{k-1}}, & j = k-1, \\ \\ \emptyset, & j \le k-2. \end{cases}$$

Thus, observing that for the standard CG $\varepsilon_{i,i-1} = 0$ and $\varepsilon_{i,i} = p_i^T A p_i$, $1 \le i \le k+1$, after some computation we obtain from (14)

$$\varepsilon_{k+1,j} = \begin{cases} \emptyset, & j = k, \\ \emptyset, & j = k-1, \\ (1+\beta_k)\varepsilon_{k,k-2}, & j = k-2, \\ (1+\beta_k)\varepsilon_{k,j} - \beta_{k-1}\varepsilon_{k-1,j} + \Sigma_{kj}, & j \le k-3, \end{cases} \tag{15}$$

where $\Sigma_{kj} \in \mathbb{R}$ summarizes the contribution of the term $\alpha_k (Ap_k)^T Ap_j$, due to a possible conjugacy loss.

Let us consider now for the CG_2step algorithm a result similar to (15). We obtain the following relations for $1 \le j \le k$

$$\begin{aligned} \varepsilon_{k+1,j} &= p_{k+1}^T A p_j = \left[Ap_k - \sigma_k p_k - \omega_k p_{k-1}\right]^T Ap_j \\ &= (Ap_k)^T Ap_j - \sigma_k \varepsilon_{k,j} - \omega_k \varepsilon_{k-1,j} \\ &= (Ap_k)^T \left[p_{j+1} + \sigma_j p_j + \omega_j p_{j-1}\right] - \sigma_k \varepsilon_{k,j} - \omega_k \varepsilon_{k-1,j} \\ &= \varepsilon_{k,j+1} + \left[\sigma_j - \sigma_k\right]\varepsilon_{k,j} + \omega_j \varepsilon_{k,j-1} - \omega_k \varepsilon_{k-1,j}, \end{aligned}$$

and considering (4), the conjugacy among directions $p_0, p_1, \ldots, p_k$ satisfies

$$\varepsilon_{h,l} = p_h^T A p_l = 0, \qquad \text{for any } \mid h - l \mid \in \{1, 2\}. \tag{16}$$

Thus, relation (6) and the expression of the coefficients in CG_2step yields

$$\varepsilon_{k+1,j} = \begin{cases} \emptyset, & j = k, \\ \emptyset, & j = k-1, \\ \omega_{k-2}\varepsilon_{k,k-3}, & j = k-2, \\ \left[\sigma_{k-3} - \sigma_k\right]\varepsilon_{k,k-3} + \omega_{k-3}\varepsilon_{k,k-4}, & j = k-3, \\ \varepsilon_{k,j+1} + \left[\sigma_j - \sigma_k\right]\varepsilon_{k,j} + \omega_j \varepsilon_{k,j-1} - \omega_k \varepsilon_{k-1,j}, & j \le k-4. \end{cases} \tag{17}$$

Finally, comparing relations (15) and (17) we have

- in case $j = k-2$ respectively the conjugacy error $\varepsilon_{k+1,k-2}$ is nonzero for both the algorithms. However, for the standard CG

$$\varepsilon_{k+1,k-2} > \varepsilon_{k,k-2}$$

since $(1+\beta_k) > 1$, which theoretically can lead to an harmful amplification of conjugacy errors. On the contrary, for the CG_2step the positive quantity ω_{k-2} in the expression of $\varepsilon_{k+1,k-2}$ can be possibly smaller than one.
Furthermore, we also have the following results which describe the sensitivity of the coefficients σ_k and ω_k to the condition number $\kappa(A)$ ($\lambda_m(A)$ and $\lambda_M(A)$ are the smallest/largest eigenvalues of A):

$$|\omega_k| = \frac{p_k^T A p_k}{p_{k-1}^T A p_{k-1}} \begin{cases} \geq \frac{\lambda_m(A)\|p_k\|^2}{\lambda_M(A)\|p_{k-1}\|^2} = \frac{1}{\kappa(A)}\frac{\|p_k\|^2}{\|p_{k-1}\|^2} \\ \\ \leq \frac{\lambda_M(A)\|p_k\|^2}{\lambda_m(A)\|p_{k-1}\|^2} = \kappa(A)\frac{\|p_k\|^2}{\|p_{k-1}\|^2}, \end{cases} \tag{18}$$

$$|\sigma_k| = \frac{\|Ap_k\|^2}{p_k^T A p_k} \begin{cases} \geq \frac{\lambda_m^2(A)\|p_k\|^2}{\lambda_M(A)\|p_k\|^2} = \frac{\lambda_m(A)}{\kappa(A)} \\ \\ \leq \frac{\lambda_M^2(A)\|p_k\|^2}{\lambda_m(A)\|p_k\|^2} = \lambda_M(A)\kappa(A). \end{cases} \tag{19}$$

This seems to indicate (as expected) that keeping $\kappa(A)$ sufficiently close to one (with $\lambda_m \approx \lambda_M \approx$ small) may be crucial for the success of CG_2step (see Section 4).

- for the CG_2step algorithm relation (16) holds, while for the CG algorithm we have $\varepsilon_{h,l} = 0$ for $\mid h - l \mid= 1$, which is a weaker condition.

3 Numerical results without preconditioning

We include some numerical results, where we compare the performance of the CG and the CG_2step (implemented as in Tables 1 and 3) to solve linear system (1). In the comparison we consider the positive definite matrix A with $n = 300$, then the condition number $\kappa = \lambda_M/\lambda_m$ of matrix A is suitably assigned. Here, $0 < \lambda_m \leq \lambda_M$ are respectively the smallest/largest eigenvalue of A. In particular, we consider in Tables 6-7 the values $\kappa = exp(2h)$, $h = 0, \ldots, 3$. Moreover, though λ_m and λ_M are assigned, the intermediate eigenvalues λ_i, $i = 2, \ldots, n-1$, of matrix A (i.e. $\lambda_m \leq \lambda_i \leq \lambda_M$) are randomly chosen.

The parameter it represents the number of iterations which are necessary to satisfy the stopping condition $\|r^*\|/\|r_1\| \leq 10^{-8}$, where $r^* = b - Ax^*$ is the last computed residual.

We checked both the conjugacy (among the directions $p_1, \ldots, p_k$) and the orthogonality (among the residuals $r_1, \ldots, r_k$) by reporting the quantities $p_1^T A p_k/(\|p_1\|\|p_k\|)$ and $r_1^T r_k/(\|r_1\|\|r_k\|)$, $k \in \{3, 5, 7, 9, 11, 13, 15\}$. Observe that from Table 7, in accordance with Section 2, the generation of the sequence $\{r_k\}$ substantially coincides for algorithms CG and CG_2step, i.e. the orthogonality loss is not a critical issue.
Finally, each row in Tables 6-7 summarizes the average results on 10 randomly generated instances, with the described parameters of matrix A. The starting point x_0 was randomly chosen, too.

Evidently, considering the number of iterations it, the standard CG algorithm outperforms the CG_2step method. In addition, when $\kappa(A)$ is relatively small both conjugacy and orthogonality conditions of the CG and the CG_2step are comparable. Unfortunately,

```
Step 0: Set $k = 0$,  $x_0 \in \mathbb{R}^n$,  $r_0 = b - Ax_0$.
            If $r_0 = 0$, then STOP. Else, set $p_0 = r_0$, $k = k + 1$.
Step k: Compute $d_{k-1} = p_{k-1}^T A p_{k-1}$, $\alpha_{k-1} = r_{k-1}^T p_{k-1} / d_{k-1}$,
            $x_k = x_{k-1} + \alpha_{k-1} p_{k-1}$,
            $r_k = r_{k-1} - \alpha_{k-1} A p_{k-1}$.
            If $r_k = 0$, then STOP. Else, set  $\beta_{k-1} = \|r_k\|^2 / \|r_{k-1}\|^2$,
            $p_k = r_k + \beta_{k-1} p_{k-1}$,  $k \leftarrow k + 1$.
            Go to Step k.
```

Table 1: The CG algorithm for solving (1).

```
Step 0: Set $k = 0$, $x_0 \in \mathbb{R}^n$, $v_0 = b - Ax_0$.
            Set $u_0 = 0$, $\delta_0 = \|b\|$.
Step k: If  $\delta_k = 0$, then STOP. Else, set
            $u_{k+1} = v_k / \delta_k$,  $k \leftarrow k + 1$,  $\gamma_k = u_k^T A u_k$,
            $v_k = A u_k - \gamma_k u_k - \delta_{k-1} u_{k-1}$,
            $\delta_k = \|v_k\|$.
            Go to Step k.
```

Table 2: The Lanczos process for solving (1).

higher values for $\kappa(A)$ determine numerical instability for the latter method. This can be explained by recalling that from Table 3, at step k of CG_2step the coefficient σ_{k-1} *strongly depends* on the condition number of matrix A^2, indeed by (19) it depends on λ_M and $\kappa(A)$.

4 The Preconditioned CG_2step

In this section we propose the preconditioned version of algorithm CG_2step, in order to comply with the drawback highlighted at the end of the previous section.
Let $M \in \mathbb{R}^{n\times n}$ be *nonsingular* and consider the linear system (1). Since we have

$$\begin{aligned} Ax = b \quad &\Longleftrightarrow \quad \left(M^T M\right)^{-1} Ax = \left(M^T M\right)^{-1} b \qquad (20) \\ &\Longleftrightarrow \quad \left(M^{-T} A M^{-1}\right) Mx = M^{-T} b \\ &\Longleftrightarrow \quad \bar{A}\bar{x} = \bar{b}, \qquad (21) \end{aligned}$$

where

$$\begin{aligned} \bar{A} &= M^{-T} A M^{-1} \\ \bar{x} &= Mx \qquad (22) \\ \bar{b} &= M^{-T} b, \end{aligned}$$

```
Step 0: Set $k = 0$, $x_0 \in \mathbb{R}^n$, $r_0 = b - Ax_0$.
        If $r_0 = 0$, then STOP. Else, set $p_0 = r_0$, $k \leftarrow k + 1$.
        Set $\sigma_0 = \frac{\|Ap_0\|^2}{p_0^T Ap_0}$, $\alpha_0 = \frac{\|r_0\|^2}{p_0^T Ap_0}$,
        $x_1 = x_0 + \alpha_0 p_0$, $r_1 = r_0 - \alpha_0 Ap_0$.
        If $r_1 = 0$, then STOP. Else, set
        $p_1 = Ap_0 - \sigma_0 p_0$, $k \leftarrow k + 1$.
Step k: Set $\sigma_{k-1} = \frac{\|Ap_{k-1}\|^2}{p_{k-1}^T Ap_{k-1}}$, $\omega_{k-1} = \frac{(Ap_{k-1})^T Ap_{k-2}}{p_{k-2}^T Ap_{k-2}} = \frac{p_{k-1}^T Ap_{k-1}}{p_{k-2}^T Ap_{k-2}}$, $\alpha_{k-1} = \frac{r_{k-1}^T p_{k-1}}{p_{k-1}^T Ap_{k-1}}$,
        $x_k = x_{k-1} + \alpha_{k-1} p_{k-1}$, $r_k = r_{k-1} - \alpha_{k-1} Ap_{k-1}$.
        If $r_k = 0$, then STOP. Else, set
        $p_k = Ap_{k-1} - \sigma_{k-1} p_{k-1} - \omega_{k-1} p_{k-2}$, $k \leftarrow k + 1$.
        Go to Step $k$.
```

Table 3: The new CG_2step algorithm for solving (1).

```
Step 0: Set $k = 0$, $\bar{x}_0 \in \mathbb{R}^n$, $\bar{r}_0 = \bar{b} - \bar{A}\bar{x}_0$.
        If $\bar{r}_0 = 0$, then STOP. Else, set $\bar{p}_0 = \bar{r}_0$, $k \leftarrow k + 1$.
        Set $\bar{\sigma}_0 = \frac{\|\bar{A}\bar{p}_0\|^2}{\bar{p}_0^T \bar{A}\bar{p}_0}$, $\bar{\alpha}_0 = \frac{\|\bar{r}_0\|^2}{\bar{p}_0^T \bar{A}\bar{p}_0}$,
        $\bar{x}_1 = \bar{x}_0 + \bar{\alpha}_0 \bar{p}_0$, $\bar{r}_1 = \bar{r}_0 - \bar{\alpha}_0 \bar{A}\bar{p}_0$.
        If $\bar{r}_1 = 0$, then STOP. Else, set
        $\bar{p}_1 = \bar{A}\bar{p}_0 - \bar{\sigma}_0 \bar{p}_0$, $k \leftarrow k + 1$.
Step k: Set $\bar{\sigma}_{k-1} = \frac{\|\bar{A}\bar{p}_{k-1}\|^2}{\bar{p}_{k-1}^T \bar{A}\bar{p}_{k-1}}$, $\bar{\omega}_{k-1} = \frac{\bar{p}_{k-1}^T \bar{A}\bar{p}_{k-1}}{\bar{p}_{k-2}^T \bar{A}\bar{p}_{k-2}}$, $\bar{\alpha}_{k-1} = \frac{\bar{r}_{k-1}^T \bar{p}_{k-1}}{\bar{p}_{k-1}^T \bar{A}\bar{p}_{k-1}}$,
        $\bar{x}_k = \bar{x}_{k-1} + \bar{\alpha}_{k-1} \bar{p}_{k-1}$, $\bar{r}_k = \bar{r}_{k-1} - \bar{\alpha}_{k-1} \bar{A}\bar{p}_{k-1}$.
        If $\bar{r}_k = 0$, then STOP. Else, set
        $\bar{p}_k = \bar{A}\bar{p}_{k-1} - \bar{\sigma}_{k-1} \bar{p}_{k-1} - \bar{\omega}_{k-1} \bar{p}_{k-2}$, $k \leftarrow k + 1$.
        Go to Step $k$.
```

Table 4: The new CG_2step algorithm for solving the linear system $\bar{A}\bar{x} = \bar{b}$ in (21).

```
Step 0: Set $k = 0$, $x_0 \in \mathbb{R}^n$, $r_0 = b - Ax_0$.
        If $r_0 = 0$, then STOP. Else, set $p_0 = \mathcal{M}r_0$,  $k \leftarrow k+1$.
        Set $\sigma_0 = \frac{\|Ap_0\|^2_{\mathcal{M}}}{p_0^T Ap_0}$,  $\alpha_0 = \frac{r_0^T p_0}{p_0^T Ap_0}$,
        $x_1 = x_0 + \alpha_0 p_0$,  $r_1 = r_0 - \alpha_0 Ap_0$.
        If $r_1 = 0$, then STOP. Else, set
        $p_1 = \mathcal{M}Ap_0 - \sigma_0 p_0$,  $k \leftarrow k+1$.
Step k: Set $\sigma_{k-1} = \frac{\|Ap_{k-1}\|^2_{\mathcal{M}}}{p_{k-1}^T Ap_{k-1}}$,  $\omega_{k-1} = \frac{p_{k-1}^T Ap_{k-1}}{p_{k-2}^T Ap_{k-2}}$,  $\alpha_{k-1} = \frac{r_{k-1}^T p_{k-1}}{p_{k-1}^T Ap_{k-1}}$,
        $x_k = x_{k-1} + \alpha_{k-1} p_{k-1}$,  $r_k = r_{k-1} - \alpha_{k-1} Ap_{k-1}$.
        If $r_k = 0$, then STOP. Else, set
        $p_k = \mathcal{M}Ap_{k-1} - \sigma_{k-1} p_{k-1} - \omega_{k-1} p_{k-2}$,  $k \leftarrow k+1$.
        Go to Step k.
```

Table 5: The new preconditioned CG_2step algorithm for solving (1).

solving (1) is equivalent to solve (20) or (21). Moreover, any eigenvalue λ_i, $i = 1, \ldots, n$, of $M^{-T}AM^{-1}$ is also an eigenvalue of $\left(M^T M\right)^{-1} A$. Indeed, if $(M^T M)^{-1}Az_i = \lambda_i z_i$, $i = 1, \ldots, n$, then

$$\left(M^{-1}M^{-T}\right) AM^{-1} (Mz_i) = \lambda_i z_i$$

and

$$M^{-T}AM^{-1} (Mz_i) = \lambda_i (Mz_i) .$$

Now, let us motivate the importance of selecting a promising matrix M in (21), in order to reduce $\kappa(\bar{A})$ (or equivalently to reduce $\kappa[(M^T M)^{-1}A]$). Observe that by using Chebyshev polynomials analysis we can prove that in exact algebra, for the CG and the CG_2step the following relation holds under the Assumption 2.1 (see [6] for details, and a similar analysis holds for CG_2step)

$$\frac{\|x_k - x^*\|_A}{\|x_0 - x^*\|_A} \leq 2 \left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \right)^k , \tag{23}$$

where $\|x_i - x^*\|_A = \left[(x_i - x^*)^T A(x_i - x^*)\right]^{1/2}$, $i \geq 1$, and $Ax^* = b$. Relation (23) reveals the strong dependency of the iterates generated by the CG and the CG_2step on $\kappa(A)$. In addition, if the CG and the CG_2step are used to solve (21) in place of (1), then the bound (23) becomes

$$\frac{\|x_k - x^*\|_A}{\|x_0 - x^*\|_A} \leq 2 \left(\frac{\sqrt{\kappa[(M^T M)^{-1}A]} - 1}{\sqrt{\kappa[(M^T M)^{-1}A]} + 1} \right)^k , \tag{24}$$

which encourages to use the *preconditioner* $(M^T M)^{-1}$ when $\kappa[(M^T M)^{-1}A] < \kappa(A)$.
On this guideline we want to introduce preconditioning in our scheme CG_2step, for solving the linear system (21), where M is non-singular. We do not expect that when $M = I$ (i.e. no preconditioning is considered) CG_2step outperforms the CG. Indeed, Table 6 confirms the opposite: $M = I$ implies that at Step k of Table 3 the numerator of the coefficient σ_{k-1}

κ	it	$\frac{p_1^T A p_3}{\|p_1\|\|p_3\|}$	$\frac{p_1^T A p_5}{\|p_1\|\|p_5\|}$	$\frac{p_1^T A p_7}{\|p_1\|\|p_7\|}$	$\frac{p_1^T A p_9}{\|p_1\|\|p_9\|}$	$\frac{p_1^T A p_{11}}{\|p_1\|\|p_{11}\|}$	$\frac{p_1^T A p_{13}}{\|p_1\|\|p_{13}\|}$	$\frac{p_1^T A p_{15}}{\|p_1\|\|p_{15}\|}$
$exp(0)$	1.0							
$exp(2)$	24.0	$0.0E+00$	$0.6E-15$	$0.1E-15$	$-0.6E-15$	$-0.3E-14$	$-0.3E-14$	$0.1E-14$
$exp(4)$	60.6	$0.0E+00$	$-0.8E-14$	$-0.1E-12$	$-0.2E-12$	$-0.2E-12$	$-0.3E-12$	$-0.5E-12$
$exp(6)$	137.2	$0.0E+00$	$-0.1E-12$	$-0.3E-11$	$-0.5E-11$	$0.3E-11$	$0.1E-10$	$0.4E-10$

κ	it	$\frac{p_1^T A p_3}{\|p_1\|\|p_3\|}$	$\frac{p_1^T A p_5}{\|p_1\|\|p_5\|}$	$\frac{p_1^T A p_7}{\|p_1\|\|p_7\|}$	$\frac{p_1^T A p_9}{\|p_1\|\|p_9\|}$	$\frac{p_1^T A p_{11}}{\|p_1\|\|p_{11}\|}$	$\frac{p_1^T A p_{13}}{\|p_1\|\|p_{13}\|}$	$\frac{p_1^T A p_{15}}{\|p_1\|\|p_{15}\|}$
$exp(0)$	1.0							
$exp(2)$	46.0	$0.0E+00$	$0.2E-14$	$-0.3E-14$	$-0.3E-14$	$0.3E-14$	$0.3E-14$	$-0.3E-14$
$exp(4)$	119.0	$0.0E+00$	$-0.3E-13$	$0.3E-13$	$0.5E-13$	$-0.4E-13$	$-0.7E-13$	$0.3E-13$
$exp(6)$	272.0	$0.0E+00$	$0.3E-13$	$0.2E-11$	$0.6E-12$	$0.2E-12$	$0.3E-12$	$0.2E-11$

Table 6: The *conjugacy loss* for the CG method (above) and the CG_2step method (below).

depends on A^2, which can be seriously harmful in case $\kappa(A)$ increases. On the contrary, suppose a suitable preconditioner $\mathcal{M} = (M^T M)^{-1}$ is selected when $\kappa(A)$ is large. Then, since the algorithm CG_2step imposes stronger conjugacy conditions with respect to the CG, it may possibly better recover the conjugacy loss.
We will soon see that $\mathcal{M}$ in CG_2step is just used to compute the product $\mathcal{M}p_{k-1}$, i.e. we do not need to store the possibly dense matrix $\mathcal{M}$.

The algorithm CG_2step for (21) is described in Table 4, where of course each 'bar' quantity has a corresponding quantity in Table 3. After substituting the positions

$$\begin{aligned} \bar{x}_k &= Mx_k \\ \bar{p}_k &= Mp_k \\ \bar{r}_k &= M^{-T} r_k \\ \mathcal{M} &= \left(M^T M\right)^{-1}, \end{aligned} \tag{25}$$

in Table 4, the vector $\bar{p}_k$ becomes

$$\bar{p}_k = Mp_k = M^{-T} A M^{-1} M p_{k-1} - \bar{\sigma}_{k-1} M p_{k-1} - \bar{\omega}_{k-1} M p_{k-2},$$

hence

$$p_k = \mathcal{M} A p_{k-1} - \bar{\sigma}_{k-1} p_{k-1} - \bar{\omega}_{k-1} p_{k-2}$$

with

$$\begin{aligned} \bar{\sigma}_{k-1} &= \frac{\|M^{-T} A p_{k-1}\|^2}{p_{k-1}^T A p_{k-1}} = \frac{(Ap_{k-1})^T \mathcal{M} A p_{k-1}}{p_{k-1}^T A p_{k-1}} \\ \bar{\omega}_{k-1} &= \frac{p_{k-1}^T M^T M^{-T} A M^{-1} M p_{k-1}}{p_{k-2}^T M^T M^{-T} A M^{-1} M p_{k-2}} = \frac{p_{k-1}^T A p_{k-1}}{p_{k-2}^T A p_{k-2}}. \end{aligned} \tag{26}$$

κ	it	$\frac{r_1^T r_3}{\|r_1\|\|r_3\|}$	$\frac{r_1^T r_5}{\|r_1\|\|r_5\|}$	$\frac{r_1^T r_7}{\|r_1\|\|r_7\|}$	$\frac{r_1^T r_9}{\|r_1\|\|r_9\|}$	$\frac{r_1^T r_{11}}{\|r_1\|\|r_{11}\|}$	$\frac{r_1^T r_{13}}{\|r_1\|\|r_{13}\|}$	$\frac{r_1^T r_{15}}{\|r_1\|\|r_{15}\|}$
$exp(0)$	1.0							
$exp(2)$	24.0	$0.0E+00$	$-0.7E-16$	$0.7E-15$	$-0.3E-14$	$-0.9E-15$	$-0.4E-14$	$0.2E-14$
$exp(4)$	60.6	$0.0E+00$	$-0.2E-14$	$-0.1E-13$	$-0.2E-13$	$-0.3E-13$	$-0.4E-13$	$-0.5E-13$
$exp(6)$	137.2	$0.0E+00$	$-0.1E-13$	$-0.1E-13$	$-0.1E-12$	$0.3E-13$	$0.2E-12$	$0.5E-12$

κ	it	$\frac{r_1^T r_3}{\|r_1\|\|r_3\|}$	$\frac{r_1^T r_5}{\|r_1\|\|r_5\|}$	$\frac{r_1^T r_7}{\|r_1\|\|r_7\|}$	$\frac{r_1^T r_9}{\|r_1\|\|r_9\|}$	$\frac{r_1^T r_{11}}{\|r_1\|\|r_{11}\|}$	$\frac{r_1^T r_{13}}{\|r_1\|\|r_{13}\|}$	$\frac{r_1^T r_{15}}{\|r_1\|\|r_{15}\|}$
$exp(0)$	1.0							
$exp(2)$	46.0	$0.0E+00$	$0.2E-13$	$0.3E-13$	$0.4E-13$	$0.1E-12$	$0.3E-12$	$0.6E-12$
$exp(4)$	119.0	$0.0E+00$	$-0.1E-13$	$0.8E-14$	$0.3E-13$	$-0.6E-14$	$-0.6E-13$	$0.1E-13$
$exp(6)$	272.0	$0.0E+00$	$-0.1E-12$	$-0.2E-12$	$-0.1E-12$	$-0.2E-12$	$-0.2E-12$	$-0.4E-12$

Table 7: The *orthogonality loss* for the CG method (above) and the CG_2step method (below).

Moreover, relation $\bar{r}_0 = \bar{b} - \bar{A}\bar{x}_0$ becomes

$$M^{-T} r_0 = M^{-T} b - M^{-T} A M^{-1} M x_0 \qquad \Longleftrightarrow \qquad r_0 = b - Ax_0,$$

and since $\bar{p}_0 = Mp_0 = \bar{r}_0 = M^{-T} r_0$ then $p_0 = \mathcal{M} r_0$, so that the coefficients $\bar{\sigma}_0$ and $\bar{\alpha}_0$ become

$$\begin{aligned} \bar{\sigma}_0 &= \frac{p_0^T M^T M^{-T} A M^{-1} M^{-T} A M^{-1} M p_0}{p_0^T A p_0} = \frac{(Ap_0)^T \mathcal{M} (Ap_0)}{p_0^T A p_0} = \frac{\|Ap_0\|_{\mathcal{M}}^2}{p_0^T A p_0} \qquad (27) \\ \bar{\alpha}_0 &= \frac{r_0^T M^{-1} M^{-T} r_0}{p_0^T M^T M^{-T} A M^{-1} M p_0} = \frac{r_0^T \mathcal{M} r_0}{p_0^T A p_0} = \frac{r_0^T p_0}{p_0^T A p_0}. \end{aligned}$$

As regards relation $\bar{p}_1 = \bar{A}\bar{p}_0 - \bar{\sigma}_0 \bar{p}_0$ we have

$$M p_1 = M^{-T} A M^{-1} M p_0 - \bar{\sigma}_0 M p_0,$$

hence

$$p_1 = \mathcal{M} A p_0 - \bar{\sigma}_0 p_0.$$

Finally, $\bar{r}_k = M^{-T} r_k$ implies

$$\bar{r}_k = M^{-T} r_k = M^{-T} r_{k-1} - \bar{\alpha}_{k-1} M^{-T} A M^{-1} M p_{k-1},$$

so that

$$r_k = r_{k-1} - \bar{\alpha}_{k-1} A p_{k-1}$$

and

$$\bar{\alpha}_{k-1} = \frac{r_{k-1}^T M^{-1} M p_{k-1}}{p_{k-1}^T M^T M^{-T} A M^{-1} M p_{k-1}} = \frac{r_{k-1}^T p_{k-1}}{p_{k-1}^T A p_{k-1}}.$$

The overall resulting preconditioned algorithm CG_2step$_{\mathcal{M}}$ is detailed in Table 5. Observe that the coefficients α_{k-1} and ω_{k-1} in Tables 3 and 5 are invariant under the introduction of the preconditioner $\mathcal{M}$ (also observe that from (26) and (27) σ_{k-1} now depends on $A\mathcal{M}A$ and not on A^2). Moreover, we recall that in Table 5 the introduction of the preconditioner simply requires at each iteration k the additional cost of the product $\mathcal{M} \times (Ap_k)$ (similarly to the preconditioned CG, where at each iteration k the additional cost is given by $\mathcal{M} \times r_k$).

Finally, in Table 5 at iteration 0 the products $\mathcal{M}r_0$ and $\mathcal{M}(Ap_0)$ are both required, in order to compute σ_0 and α_0. This is just appearantly a drawback; indeed, considering that Step 0 of CG_2step is equivalent to two iterations of the CG, then the additional cost of preconditioning CG and CG_2step is the same.
Furthermore, there is an additional chance to replace the Step 0 in Table 3, with the following

```
Step 0: Set $k = 0$, $x_0 \in \mathbb{R}^n$, $r_0 = b - Ax_0$.
        If $r_0 = 0$, then STOP. Else, set $p_0 = r_0, \ k \leftarrow k + 1$.
        Set $\alpha_0 = \frac{\|r_0\|^2}{p_0^T A p_0}$,
        $x_1 = x_0 + \alpha_0 p_0, \ r_1 = r_0 - \alpha_0 A p_0$.
        If $r_1 = 0$, then STOP. Else, set $\sigma_0 = -\frac{\|r_1\|^2}{\|r_0\|^2}$,
        $p_1 = r_1 - \sigma_0 p_0, \ k \leftarrow k + 1$.
```

which substantially corresponds to perform two steps of the CG. All the main results still hold almost unchanged (some adjustments in Lemma 2.3), as well as the possibility of preconditioning the resulting new CG_2step.

5 Conclusions

We have proposed a CG-based method, namely the CG_2step algorithm, for the iterative solution of the symmetric positive definite linear system (1). Our method, which is based on the generation of conjugate directions, is slightly more expensive at Step k than the CG, and it requires the storage of one further vector. However, we proved for the CG_2step some theoretical properties, which are stronger than those provided for the CG method. Furthermore, we introduced preconditioning in our proposal, so that at Step k it may likely prevent from the conjugacy loss between the directions p_k and p_{k-2}.
We are considering a numerical experience, which includes convex optimization problems from CUTEr collection [8], where our preconditioned scheme is adopted to solve Newton's equation. In particular we want to investigate the choice $\mathcal{M} \approx A^{-1}$, where $\mathcal{M}$ is computed as a Quasi-Newton approximation of the inverse matrix A^{-1} (see also [3]).

References

[1] Axelsson, O, *Iterative Solution Methods*, Cambridge University Press, 1996.

[2] Fasano, G., *Conjugate Gradient (CG)-type Method for the Solution of Newton's Equation within Optimization Frameworks*, Optimization Methods and Software, Vol. 19, pp. 267–290, 2004.

[3] Fasano, G., *Lanczos Conjugate-Gradient Method and Pseudoinverse Computation on Indefinite and Singular Systems*, J. of Optimization Theory and Applications, Vol. 132, pp. 267–285, 2007.

[4] Freund, R.W., Golub, G.H., and Nachtigal, N.M., *Iterative Solution of Linear Systems*, Acta Numerica, pp. 1–44, 1992.

[6] Golub, G.H., and Van Loan, C.F., *Matrix computations - 3rd edition*, The John Hopkins Press, Baltimore, USA, 1989.

[5] Gould, N.I.M., Lucidi, S., Roma, M. and Toint, Ph.L., *Exploiting negative curvature directions in linesearch methods for unconstrained optimization*, Optim. Methods & Soft., Vol. 14, pp. 75–98, 2000.

[8] Gould, N.I.M., Orban, D. and Toint, Ph.L., *CUTE (and sifdec), a constrained and unconstrained testing environment, revised*, ACM Transaction on Mathematical Software, Vol. 29, pp. 373–394, 2003.

[7] Grippo, L., Lampariello, F. and Lucidi, S. *A truncated Newton method with non-monotone linesearch for unconstrained optimization*, J. of Optimization Theory and Applications, Vol. 60, pp. 401–419, 1989.

[9] Hestenes, M.R. *Conjugate Direction Methods in Optimization*, Springer Verlag, NewYork, Heidelberg, Berlin, 1980.

[10] Hestenes, M.R. and Stiefel, E., *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, Vol. 49, pp. 409–435, 1952.

[11] Higham, N.J., *Accuracy and Stability of Numerical Algorithms* , SIAM, Philadelphia, USA, 1996.

[12] Manvich, A.I., and Boudinov, E., *An efficient conjugate directions method without linear minimization*, Nuclear Instruments and Methods in Physics Research A 455, pp. 698–705, 2000.

[13] Manvich, A.I., and Boudinov, E., *A conjugate direction method with orthogonalization for large-scale problems*, European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS 2004, P. Neittaanmäki, T. Rossi, K. Majava and O. Pironneau (eds.), Jyväskylä, 24-28 July 2004.

[14] Saad, Y., and Van Der Vorst, H.A., *Iterative Solution of Linear Systems in the 20th Century*, Journal on Computational and Applied Mathematics, Vol. 123, pp. 1–33, 2000.

[15] Sleijpen, G.L.G. and Van DerVorst, H.A., *Krylov subspace methods for large linear systems of equations*, Preprint 803, Department of Mathematics, University of Utrecht, 1993.

[16] Van Der Vorst, H.A., and Chan, T.F., *Linear System Solvers: Sparse-Iterative Methods*, Parallel Numerical Algorithms, ICASE/LaRC Interdisciplinary Series in Science and Engineering, Edited by D.E.Keyes, A.Samed and V.Venkatakrishnan, Dordrecht, Kluwer Academic, Dodrecht, Holland, Vol. 4, pp. 91–118, 1997.