Eigenvalues versus singular values study in conjugate gradient algorithms for large-scale unconstrained optimization

Neculai Andrei

Research Institute for Informatics,
Center for Advanced Modeling and Optimization
8-10 Averescu Avenue, Bucharest 1, Romania
E-mail: nandrei@ici.ro

August 1, 2016

Abstract. Two different approaches based on eigenvalues and singular values of the matrix representing the search direction in conjugate gradient algorithms are considered. Using a special approximation of the inverse Hessian of the objective function, which depends by a positive parameter, we get the search direction which satisfies both the sufficient descent condition and the Dai-Liao's conjugacy condition. In the first approach the parameter in the search direction is determined by clustering the eigenvalues of the matrix defining it. The second approach uses the minimizing the condition number of the matrix representing the search direction. In this case the obtained conjugate gradient algorithm is exactly the three-term conjugate gradient algorithm proposed by Zhang, Zhou and Li. The global convergence of the algorithms is proved for uniformly convex functions. Intensive numerical experiments, using 800 unconstrained optimization test problems, prove that both these approaches have similar numerical performances. We prove that both algorithms are significantly more efficient and more robust than CG-DESCENT algorithm by Hager and Zhang. By solving five applications from the MINPACK-2 test problem collection, with 10^6 variables, we show that the suggested conjugate gradient algorithms are top performer versus CG-DESCENT.

Key words: Unconstrained optimization; conjugate gradient algorithms; eigenvalues; singular values; Wolfe conditions; convergence; sufficient descent condition; conjugacy condition

2000 Mathematics Subject Classification: 49M07; 49M10; 90C06; 65K

1. Introduction

For solving the large-scale unconstrained optimization problem

$$\min\{f(x): x \in R^n\}, \tag{1}$$

where $f: R^n \to R$ is a continuously differentiable function and bounded below, we consider the following very well known algorithm

$$x_{k+1} = x_k + \alpha_k d_k, \tag{2}$$

where the stepsize α_k is positive and the direction d_k is computed using the updating formula:

$$d_{k+1} = -g_{k+1} + u_{k+1}. \tag{3}$$

Here, $g_k = \nabla f(x_k)$, and $u_{k+1} \in R^n$ is a vector to be determined. Usually, in (2), the stepsize α_k is computed using the Wolfe line search conditions [1,2]:

$$f(x_k + \alpha_k d_k) \le f(x_k) + \rho \alpha_k g_k^T d_k, \tag{4}$$

$$g_{k+1}^T d_k \ge \sigma g_k^T d_k, \tag{5}$$

where $0 < \rho \leq \sigma < 1$. Also, the strong Wolfe line search conditions consisting of (4) and the following strengthened version of (5):

$$\left| g_{k+1}^T d_k \right| \leq -\sigma g_k^T d_k, \tag{6}$$

can be used.

Observe that (3) is a general updating formula for the search direction computation. The following particularizations of (3) can be presented. If $u_{k+1} = 0,$ then we get the steepest descent algorithm. If $u_{k+1} = (I - \nabla^2 f(x_{k+1})^{-1}) g_{k+1},$ then the Newton method is obtained. Besides, if $u_{k+1} = (I - B_{k+1}^{-1}) g_{k+1},$ where B_{k+1} is an approximation of the Hessian $\nabla^2 f(x_{k+1}),$ then we find the quasi-Newton methods. On the other hand, if $u_{k+1} = \beta_k d_k,$ where β_k is a scalar and $d_0 = -g_0,$ the family of conjugate gradient algorithms is generated.

In this paper we focus on conjugate gradient method. This method was introduced by Hestenes and Stiefel [3] and Stiefel [4], ($\beta_k^{HS} = g_{k+1}^T y_k / y_k^T d_k$), to minimize positive definite quadratic objective functions. (Here $y_k = g_{k+1} - g_k$.) This algorithm for solving positive definite linear algebraic systems of equations $Ax = b,$ $A \in R^{n \times n},$ $b \in R^n,$ is known as *linear conjugate gradient*. In exact arithmetic the linear conjugate gradient algorithm gives the correct solution in at most n steps (see [5]). In this case this is a direct method. However, in practice, this algorithm is regarded as an iterative method (see Reid [6]) because a sufficiently accurate approximation solution is often obtained in far fewer then n steps. In absence of rounding errors, the theoretical convergence rate has been studied by many authors. The conclusion is that the rate of convergence of linear conjugate gradient depends strongly on the distribution of eigenvalues of the matrix A. Further insights concerning this problem were studied by many researchers, see for example: Axelsson [7], Axelsson and Lindskog [8], Strakoš [9], Van der Sluis and Van der Vorst [10], Meurant [11], Winther [12]. Later on, the algorithm was generalized to *nonlinear conjugate gradient* in order to minimize arbitrary differentiable nonlinear functions, by Fletcher and Reeves [13], ($\beta_k^{FR} = \|g_{k+1}\|^2 / \|g_k\|^2$), Polak and Ribière [14] and Polyak [15], ($\beta_k^{PRP} = g_{k+1}^T y_k / \|g_k\|^2$), Dai and Yuan [16], ($\beta_k^{DY} = \|g_{k+1}\|^2 / y_k^T d_k$), Dai and Liao [17], ($\beta_k^{DL} = g_{k+1}^T (y_k - t s_k) / y_k^T d_k,$ $t > 0$), and many others. Here $\|.\|$ stands for the Euclidean norm. An impressive number of nonlinear conjugate gradient algorithms have been established, and a lot of papers have been published on this subject insisting both on theoretical and computational aspects. An excellent survey of conjugate gradient methods is that given by Hager and Zhang [18].

In the following we consider another approach to generate an efficient and robust conjugate gradient algorithm. We suggest a procedure for u_{k+1} computation by minimizing the quadratic approximation of the function f in x_{k+1} and by using a special representation of the inverse Hessian which depends on a positive parameter. The parameter in the matrix representing the search direction is determined in two different ways. The first one is based on the eigenvalues analysis of the matrix by trying to minimize the largest eigenvalue. This idea, taken from the linear conjugate gradient, is to cluster the eigenvalues of the matrix representing the search direction. The second way to determine the value of the parameter is based on the fact that if the matrix defining the search direction is ill-conditioned, then, even for small relative errors in the gradient, the relative errors in the search direction may be large. Therefore, the second way is to use the singular value analysis by minimizing the condition number of the matrix representing the search direction of the algorithm.

The algorithm and its properties are presented in section 2. We prove that the search direction used by these algorithms satisfies both the sufficient descent condition and the Dai and Liao conjugacy condition [17]. In section 3 we present an adaptive conjugate gradient algorithm

based on the clustering the eigenvalues of the matrix representing the search direction. Section 4 presents the corresponding conjugate gradient algorithm based on minimizing the condition number of the same matrix using the singular values analysis. Using standard assumptions, section 5 presents the global convergence of these algorithms for uniformly convex functions. In section 6 the numerical comparisons between these two conjugate gradient algorithms and the comparisons of these algorithms versus CG-DESCENT conjugate gradient algorithm [19] are presented. The purpose was to compare these two approaches of generating conjugate gradient algorithms. The computational results, for a set of 800 unconstrained optimization test problems, show that both these algorithms substantially outperform CG-DESCENT, being more efficient and more robust. On the other hand the algorithm based on the eigenvalues analysis (clustering the eigenvalues) is more efficient than the algorithm based on the minimizing the condition number. However, the algorithm using the idea of minimizing the condition number is more robust than the algorithm based on clustering the eigenvalues. Considering five applications from the MINPACK-2 test problem collection [20], with 10^6 variables, we show that both algorithms have similar performances and both are way more efficient and more robust than CG-DESCENT.

2. The basic algorithm

In this section we describe the basic algorithm and its properties. Let us consider that at the kth iteration of the algorithm an inexact Wolfe line search is executed, that is the stepsize α_k satisfying (4) and (5) is computed. With these the following elements $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$ are computed. Now, let us take the quadratic approximation of function f in x_{k+1} as

$$\Phi_{k+1}(d) = f_{k+1} + g_{k+1}^T d + \frac{1}{2} d^T B_{k+1} d, \tag{7}$$

where B_{k+1} is an approximation of the Hessian $\nabla^2 f(x_{k+1})$ of function f and d is the direction to be determined. The search direction d_{k+1} is computed as in (3), where u_{k+1} is computed as solution of the following minimizing problem

$$\min_{u_{k+1} \in R^n} \Phi_{k+1}(d_{k+1}). \tag{8}$$

Introducing d_{k+1} from (3) in the minimizing problem (8), then u_{k+1} is obtained as

$$u_{k+1} = (I - B_{k+1}^{-1}) g_{k+1}. \tag{9}$$

Clearly, using different approximations B_{k+1} of the Hessian $\nabla^2 f(x_{k+1})$ different search directions d_{k+1} can be obtained. In this paper we consider the following expression of B_{k+1}^{-1}:

$$B_{k+1}^{-1} = I - \frac{s_k y_k^T - y_k s_k^T}{y_k^T s_k} + \omega_k \frac{s_k s_k^T}{y_k^T s_k}, \tag{10}$$

where ω_k is a positive parameter which follows to be determined. Observe that B_{k+1}^{-1} is the sum of a skew symmetric matrix with zero diagonal elements $(s_k y_k^T - y_k s_k^T)/y_k^T s_k$, and a symmetric and positive definite one: $I + \omega_k (s_k s_k^T)/(y_k^T s_k)$. Since B_{k+1}^{-1} is non-symmetric, regardless of its effectiveness in terms of numerical performance, it does not seem a natural approximation of the Hessian $\nabla^2 f(x_{k+1})$. However, in this paper we are interested to use this expression of the B_{k+1}^{-1} in the frame of conjugate gradient algorithms, in order to get search directions satisfying both the sufficient descent and conjugacy conditions.

Again, observe that (10) is a small modification of the memoryless BFGS updating formula used by Shanno [21]. Note that under exact line search the search direction $d_{k+1} = -B_{k+1}^{-1} g_{k+1}$, where B_{k+1}^{-1} is given by (10), reduces to the Hestenes and Stiefel method [3].

Now, from (9) we get:

$$u_{k+1}=\left[\frac{s_k y_k^T-y_k s_k^T}{y_k^T s_k}-\omega_k\frac{s_k s_k^T}{y_k^T s_k}\right]g_{k+1}. \tag{11}$$

Denote $H_{k+1}=B_{k+1}^{-1}$. Therefore, using (11) in (3) the search direction can be expressed as:

$$d_{k+1}=-H_{k+1}g_{k+1}, \tag{12}$$

where

$$H_{k+1}=I-\frac{s_k y_k^T-y_k s_k^T}{y_k^T s_k}+\omega_k\frac{s_k s_k^T}{y_k^T s_k}. \tag{13}$$

Observe that the search direction (12), where H_{k+1} is given by (13), obtained by using the expression (10) of the inverse Hessian B_{k+1}^{-1}, is given by:

$$d_{k+1}=-g_{k+1}+\left(\frac{y_k^T g_{k+1}}{y_k^T s_k}-\omega_k\frac{s_k^T g_{k+1}}{y_k^T s_k}\right)s_k-\frac{s_k^T g_{k+1}}{y_k^T s_k}y_k. \tag{14}$$

PROPOSITION 2.1. *Consider $\omega_k\geq 0$ and the stepsize α_k in (2) is determined by the Wolfe line search conditions (4) and (5). Then the search direction (14) satisfies the descent condition* $g_{k+1}^T d_{k+1}\leq 0$.

Proof By direct computation, since $\omega_k\geq 0$, we get:

$$g_{k+1}^T d_{k+1}=-\|g_{k+1}\|^2-\omega_k\frac{(g_{k+1}^T s_k)^2}{y_k^T s_k}\leq 0. \qquad \blacksquare$$

PROPOSITION 2.2. *Consider $\omega_k\geq 0$ and the stepsize α_k in (2) is determined by the Wolfe line search conditions (4) and (5). Then the search direction (14) satisfies the Dai and Liao conjugacy condition $y_k^T d_{k+1}=-v_k(s_k^T g_{k+1})$, where $v_k\geq 0$.*

Proof By direct computation we have

$$y_k^T d_{k+1}=-\left[\omega_k+\frac{\|y_k\|^2}{y_k^T s_k}\right](s_k^T g_{k+1})\equiv -v_k(s_k^T g_{k+1}),$$

where $v_k\equiv\omega_k+\dfrac{\|y_k\|^2}{y_k^T s_k}$. By Wolfe line search conditions (4) and (5) it follows that $y_k^T s_k>0$, therefore $v_k>0$. $\blacksquare$

Although we have considered the expression of the inverse Hessian as that given by (10), which is a non-symmetric matrix, the search direction (14), obtained in this manner, satisfies both the descent condition and the Dai and Liao conjugacy condition. Therefore, the search direction (14) is a genuine conjugate gradient algorithm. The expression (10) of the inverse Hessian is only a technical argument to get the search direction (14). It is remarkable to say that from (12) our method can be considered as a quasi-Newton method in which the inverse Hessian, at each iteration, is expressed by the non-symmetric matrix H_{k+1}. More than this, the algorithm based on the search direction given by (14) can be considered as a three-term conjugate gradient algorithm.

In this point, to define the algorithm the only problem we face is to specify a suitable value for the positive parameter ω_k. In the following two sections of the paper we present a variant of the algorithm based on the eigenvalues analysis and another variant based on the singular values, respectively.

3. The algorithm based on clustering the eigenvalues

As we know, generally, in a small neighborhood of the current point, the nonlinear objective function f in the unconstrained optimization problem (1) behaves like a quadratic one for which the results from linear conjugate gradient can be applied. But, for faster convergence of linear conjugate gradient algorithms some approaches can be considered like: the presence of isolated smallest and/or largest eigenvalues of the matrix H_{k+1}, as well as gaps inside the eigenvalues spectrum [8], clustering of the eigenvalues about one point [12] or about several points [22], or preconditioning [23]. If the matrix has a number of certain distinct eigenvalues contained in m disjoint intervals of very small length, then the linear conjugate gradient method will produce a very small residual after m iterations. This is an important property of linear conjugate gradient method and we try to use it in nonlinear case in order to get efficient and robust nonlinear conjugate gradient algorithms. Therefore, we consider the extension of the method of clustering the eigenvalues of the matrix defining the search direction from linear conjugate gradient algorithm to nonlinear case.

The idea of this variant of the algorithm is to determine ω_k by clustering the eigenvalues of H_{k+1}, given by (13), by minimizing the largest eigenvalue of the matrix H_{k+1} from the spectrum of this matrix. The structure of the eigenvalues of the matrix H_{k+1} is given by the following theorem.

THEOREM 3.1. *Suppose that the stepsize α_k is determined by the Wolfe line search conditions (4) and (5). Let H_{k+1} be defined by (13). Then H_{k+1} is a nonsingular matrix and its eigenvalues consist of 1 ($n-2$ multiplicity), λ_{k+1}^+ and λ_{k+1}^-, where*

$$\lambda_{k+1}^+ = \frac{1}{2}\left[(2+\omega_k b_k) + \sqrt{\omega_k^2 b_k^2 - 4a_k + 4}\right], \tag{15}$$

$$\lambda_{k+1}^- = \frac{1}{2}\left[(2+\omega_k b_k) - \sqrt{\omega_k^2 b_k^2 - 4a_k + 4}\right], \tag{16}$$

and

$$a_k = \frac{\|y_k\|^2 \|s_k\|^2}{(y_k^T s_k)^2} > 1, \qquad b_k = \frac{\|s_k\|^2}{y_k^T s_k} \geq 0. \tag{17}$$

Proof By the Wolfe line search conditions (4) and (5) we have that $y_k^T s_k > 0$. Therefore, the vectors y_k and s_k are nonzero vectors. Let V be the vector space spanned by $\{s_k, y_k\}$. Clearly, $\dim(V) \leq 2$ and $\dim(V^\perp) \geq n-2$. Thus, there exist a set of mutually unit orthogonal vectors $\{u_k^i\}_{i=1}^{n-2} \subset V^\perp$ such that

$$s_k^T u_k^i = y_k^T u_k^i = 0, \ \ i = 1, \ldots, n-2,$$

which from (13) leads to

$$H_{k+1} u_k^i = u_k^i, \ \ i = 1, \ldots, n-2.$$

Therefore, the matrix H_{k+1} has $n-2$ eigenvalues equal to 1, which correspond to $\{u_k^i\}_{i=1}^{n-2}$ as eigenvectors.

Now, we are interested to find the rest of the two remaining eigenvalues, denoted as λ_{k+1}^{+} and λ_{k+1}^{-}, respectively. From the formula of algebra (see for example ref. [5])

$$\det(I + pq^T + uv^T) = (1+q^T p)(1+v^T u) - (p^T v)(q^T u),$$

where $p = \dfrac{y_k + \omega_k s_k}{y_k^T s_k}$, $q = s_k$, $u = -\dfrac{s_k}{y_k^T s_k}$ and $v = y_k$, it follows that

$$\det(H_{k+1}) = \frac{\|s_k\|^2 \|y_k\|^2}{(y_k^T s_k)^2} + \omega_k \frac{\|s_k\|^2}{y_k^T s_k} \equiv a_k + \omega_k b_k. \tag{18}$$

But, $a_k > 1$ and $b_k \geq 0$, therefore, H_{k+1} is a nonsingular matrix.
On the other hand, by direct computation

$$tr(H_{k+1}) = n + \omega_k \frac{\|s_k\|^2}{y_k^T s_k} \equiv n + \omega_k b_k. \tag{19}$$

By the relationships between the determinant and the trace of a matrix and its eigenvalues, it follows that the other eigenvalues of H_{k+1} are the roots of the following quadratic polynomial:

$$\lambda^2 - (2 + \omega_k b_k)\lambda + (a_k + \omega_k b_k) = 0. \tag{20}$$

Clearly, the other two eigenvalues of the matrix H_{k+1} are determined from (20) as (15) and (16), respectively. Observe that $a_k > 1$ follows from Wolfe conditions and the inequality:

$$\frac{y_k^T s_k}{\|s_k\|^2} \leq \frac{\|y_k\|^2}{y_k^T s_k}. \qquad \blacksquare$$

In order to have both λ_{k+1}^{+} and λ_{k+1}^{-} as real eigenvalues, from (15) and (16) the following condition must be fulfilled $\omega_k^2 b_k^2 - 4a_k + 4 \geq 0$, out of which the following estimation of the parameter ω_k can be determined:

$$\omega_k \geq \frac{2\sqrt{a_k - 1}}{b_k}. \tag{21}$$

Since $a_k > 1$, if $\|s_k\| > 0$, it follows that the estimation of ω_k given in (21) is well defined.

From (20) we have

$$\lambda_{k+1}^{+} + \lambda_{k+1}^{-} = 2 + \omega_k b_k > 0, \tag{22}$$

$$\lambda_{k+1}^{+} \lambda_{k+1}^{-} = a_k + \omega_k b_k > 0. \tag{23}$$

Therefore, from (22) and (23) we have that both λ_{k+1}^{+} and λ_{k+1}^{-} are positive eigenvalues. Since $\omega_k^2 b_k^2 - 4a_k + 4 \geq 0$, from (15) and (16) we have that $\lambda_{k+1}^{+} \geq \lambda_{k+1}^{-}$. By direct computation, from (15), using (21) we get

$$\lambda_{k+1}^{+} \geq 1 + \sqrt{a_k - 1} > 1. \tag{24}$$

A simple analysis of equation (20) shows that $1 \leq \lambda_{k+1}^{-} \leq \lambda_{k+1}^{+}$. Therefore, the maximum eigenvalue of H_{k+1} is λ_{k+1}^{+} and its minimum eigenvalue is 1. Now, we see that if $n-2$ eigenvalues of H_{k+1} are equal to 1, and the remaining two are greater than 1 and they depend by ω_k like in (15) and (16), then clustering can affect only two eigenvalues. Since the smallest eigenvalue of H_{k+1} is equal to 1, the minimization of the largest eigenvalue of this matrix coincides with the clustering of the eigenvalues.

PROPOSITION 3.1. *The largest eigenvalue*

$$\lambda_{k+1}^{+}=\frac{1}{2}\left[(2+\omega_k b_k)+\sqrt{\omega_k^2 b_k^2-4a_k+4}\right] \tag{25}$$

gets its minimum $1+\sqrt{a_k-1}$, *when* $\omega_k=\dfrac{2\sqrt{a_k-1}}{b_k}$.

Proof Observe that $a_k>1$. By direct computation the minimum of (25) is obtained for $\omega_k=(2\sqrt{a_k-1})/b_k$, for which its minimum value is $1+\sqrt{a_k-1}$. ■

We see that according to proposition 3.1 when $\omega_k=(2\sqrt{a_k-1})/b_k$ the largest eigenvalue of H_{k+1} arrives at the minimum value, i.e. the spectrum of H_{k+1} is clustered. In fact for $\omega_k=(2\sqrt{a_k-1})/b_k$, $\lambda_{k+1}^{+}=\lambda_{k+1}^{-}=1+\sqrt{a_k-1}$. Therefore, from (17) the following estimation of ω_k can be obtained:

$$\omega_k=2\frac{y_k^T s_k}{\|s_k\|^2}\sqrt{a_k-1}. \tag{26}$$

From (17) $a_k>1$, hence if $\|s_k\|>0$ it follows that the estimation of ω_k given by (26) is well defined. However, we see that the minimum of λ_{k+1}^{+} obtained for $\omega_k=(2\sqrt{a_k-1})/b_k$ is given by $1+\sqrt{a_k-1}$. Therefore, if a_k is large, then the largest eigenvalue of the matrix H_{k+1} will be large. This motivates the parameter ω_k to be computed as:

$$\omega_k=\begin{cases}2\sqrt{\tau-1}\dfrac{y_k^T s_k}{\|s_k\|^2}, & \text{if } a_k\geq\tau,\\[2ex] 2\sqrt{a_k-1}\dfrac{y_k^T s_k}{\|s_k\|^2}, & \text{otherwise,}\end{cases} \tag{27}$$

where $\tau>1$ is a positive constant. Therefore, our algorithm is an adaptive conjugate gradient algorithm in which the value of the parameter ω_k in the search direction (14) is computed as in (27) trying to cluster all the eigenvalues of H_{k+1} defining the search direction of the algorithm.

Using the procedure of acceleration the conjugate gradient algorithms presented in [24], and taking into consideration the above developments, the following algorithm can be presented.

NADCG Algorithm (New Adaptive Conjugate Gradient Algorithm)

Step 1. Select a starting point $x_0\in R^n$ and compute: $f(x_0)$, $g_0=\nabla f(x_0)$. Select some positive values for ρ and σ used in Wolfe line search conditions. Consider a positive value for the parameter τ. ($\tau>1$) Set $d_0=-g_0$ and $k=0$.

Step 2. Test a criterion for stopping the iterations. If this test is satisfied, then stop; otherwise continue with step 3.

Step 3. Determine the stepsize α_k by using the Wolfe line search (4) and (5).

Step 4. Compute $z=x_k+\alpha_k d_k$, $g_z=\nabla f(z)$ and $y_k=g_k-g_z$.

Step 5. Compute: $\bar{a}_k=\alpha_k g_z^T d_k$ and $\bar{b}_k=-\alpha_k y_k^T d_k$.

Step 6. Acceleration scheme. If $\bar{b}_k>0$, then compute $\xi_k=-\bar{a}_k/\bar{b}_k$ and update the variables as $x_{k+1}=x_k+\xi_k\alpha_k d_k$, otherwise update the variables as $x_{k+1}=x_k+\alpha_k d_k$.

Step 7. Compute ω_k as in (27).

Step 8. Compute the search direction as in (14).

Step 9. Powell restart criterion. If $\left|g_{k+1}^T g_k\right| > 0.2\left\|g_{k+1}\right\|^2$, then set $d_{k+1} = -g_{k+1}$.

Step 10. Consider $k = k+1$ and go to step 2. ■

If function f is bounded along the direction d_k, then there exists a stepsize α_k satisfying the Wolfe line search (see for example ref. [25] or [26]). In our algorithm when the Beale-Powell restart condition is satisfied, then we restart the algorithm with the negative gradient $-g_{k+1}$. More sophisticated reasons for restarting the algorithms have been proposed in the literature [27], but we are interested in the performance of a conjugate gradient algorithm that uses this restart criterion associated to a direction satisfying both the descent and the conjugacy conditions. Under reasonable assumptions, the Wolfe conditions and the Powell restart criterion are sufficient to prove the global convergence of the algorithm. The first trial of the stepsize crucially affects the practical behavior of the algorithm. At every iteration $k \geq 1$ the starting guess for the step α_k in the line search is computed as $\alpha_{k-1}\left\|d_{k-1}\right\| / \left\|d_k\right\|$. For uniformly convex functions, we can prove the linear convergence of the acceleration scheme used in the algorithm [24].

4. The algorithm based on minimizing the condition number

As we know, the convergence rate of the nonlinear conjugate gradient algorithms depends on the structure of the eigenvalues of the Hessian and the condition number of this matrix [28] (see also ref. [29]). Therefore, in this context a possibility to generate conjugate gradient algorithms is based on minimizing the condition number of the matrix H_{k+1} (see for example refs. [30,31]). From (12) we see that the numerical performances and the efficiency of the quasi-Newton methods are based on the condition number of the successive approximations to the inverse Hessian. If the matrix H_{k+1} is ill-conditioned, then, even for small values of the relative error of g_{k+1}, the relative error of d_{k+1} may be large. Hence, when the condition number of H_{k+1} is large, the system (12) is potentially very sensitive to perturbations in g_{k+1}. In other words ill-conditioned matrices H_{k+1} may produce instability in iterative numerical computation with them. Therefore, the idea of this variant of the algorithm is to minimize the condition number of the matrix H_{k+1} using its singular values. For this, we briefly present the singular value analysis. The following definitions and theorems, taken from Watkins [32], clarify some aspects of this concept of condition number of a matrix.

THEOREM 4.1. [32] *Let $A \in R^{n\times m}$ be a nonzero matrix with rank r. Then, R^m has an orthonormal basis $v_1,\ldots,v_m$, R^n has an orthonormal basis $u_1,\ldots,u_n$, and there exist the scalars $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ such that*

$$Av_i = \begin{cases} \sigma_i u_i, & i = 1,\ldots,r, \\ 0, & i = r+1,\ldots,m, \end{cases} \quad \text{and} \quad A^T u_i = \begin{cases} \sigma_i v_i, & i = 1,\ldots,r, \\ 0, & i = r+1,\ldots,n. \end{cases} \tag{28}$$

DEFINITION 4.1. *The scalars $\sigma_1,\ldots,\sigma_r$ from the theorem 4.1 are called the singular values of the matrix A.*

Based on the Theorem 4.1, for any nonzero matrix $A \in R^{n\times m}$ with rank r it follows that

$$\|A\|_F^2 = \sigma_1^2 + \ldots + \sigma_r^2, \tag{29}$$

where $\|.\|_F$ represents the Frobenius norm. If $r = m = n,$ then

$$|\det(A)| = \sigma_1 \times \sigma_2 \times \cdots \times \sigma_n.$$

As we mentioned, a very important concept in the sensitivity analysis of numerical computations with matrices is the matrix condition number. A matrix with a large condition number is called an ill-conditioned matrix since the computations with this matrix are potentially very sensitive to changes in data of the problem involving this matrix.

DEFINITION 4.2. *For an arbitrary nonsingular matrix $A,$ the scalar $\kappa(A) = \|A\|\|A^{-1}\|$ is called the condition number of $A.$*

THEOREM 4.2. [32] *If $A \in R^{n\times n}$ is a nonsingular matrix with the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0,$ then $\kappa(A) = \sigma_1 / \sigma_n.$*

DEFINITION 4.3. *The condition number $\kappa(A)$ computed as above is called the spectral condition number.*

In our analysis we need to find the singular values of the matrix H_{k+1}. For this, in our developments we assume that $y_k^T s_k > 0,$ which is guaranteed by the Wolfe line search conditions (4) and (5).

THEOREM 4.3. *Suppose that the stepsize α_k is determined by the Wolfe line search conditions (4) and (5). Let H_{k+1} be defined by (13). Then H_{k+1} has $n-2$ singular values equal to 1 and the remaining singular values σ_{k+1}^+ and σ_{k+1}^- are given by*

$$\sigma_{k+1}^+ = \frac{1}{2}\left[\sqrt{(\omega_k b_k + 2)^2 + 4(a_k - 1)} + \omega_k b_k\right], \tag{30}$$

$$\sigma_{k+1}^- = \frac{1}{2}\left[\sqrt{(\omega_k b_k + 2)^2 + 4(a_k - 1)} - \omega_k b_k\right], \tag{31}$$

where a_k and b_k are given by (17).

Proof By the Wolfe line search conditions (4) and (5) we have that $y_k^T s_k > 0.$ Therefore, the vectors y_k and s_k are nonzero vectors. Since $y_k^T s_k \neq 0,$ there exists a set of mutually orthonormal vectors $\{u_k^i\}_{i=1}^{n-2}$ such that

$$s_k^T u_k^i = y_k^T u_k^i = 0, \ i = 1, \ldots, n-2,$$

which from (13) leads to

$$H_{k+1} u_k^i = H_{k+1}^T u_k^i = u_k^i, \ i = 1, \ldots, n-2.$$

Therefore, the matrix H_{k+1} has $n-2$ singular values being equal to 1. Next, we are interested to find the rest of the two remaining singular values, denoted as σ_{k+1}^+ and σ_{k+1}^-, respectively. But, by direct computation

$$tr(H_{k+1}^T H_{k+1}) = n - 2 + 2\omega_k b_k + \omega_k^2 b_k^2 + 2a_k.$$

Since $\|H_{k+1}\|_F^2 = tr(H_{k+1}^T H_{k+1}),$ from (29) we get

$$(\sigma_{k+1}^{+})^2 + (\sigma_{k+1}^{-})^2 = \omega_k^2 b_k^2 + 2\omega_k b_k + 2a_k. \tag{32}$$

As above in Theorem 3.1 (see (18)) the determinant of the iteration matrix H_{k+1} is the product of the singular values σ_{k+1}^{+} and σ_{k+1}^{-}, i.e.

$$\sigma_{k+1}^{+}\sigma_{k+1}^{-} = a_k + \omega_k b_k. \tag{33}$$

Now, from (32) and (33) the singular values σ_{k+1}^{+} and σ_{k+1}^{-} are the solution of the following quadratic equation

$$\sigma^2 - \left(\sqrt{\omega_k^2 b_k^2 + 4\omega_k b_k + 4a_k}\right)\sigma + (a_k + \omega_k b_k) = 0,$$

expressed as in (30) and (31), respectively. ■

REMARK 4.1. It is relatively easy to prove that, for the matrix H_{k+1} defined by (13), $\lambda_{k+1}^{+} < \sigma_{k+1}^{+}$, i.e. the largest eigenvalue of H_{k+1} given by (15) is strictly smaller than the largest singular value of H_{k+1} given by (30). Now, from (24) $1 < \lambda_{k+1}^{+}$, therefore, for H_{k+1} defined by (13), it follows that $1 < \lambda_{k+1}^{+} < \sigma_{k+1}^{+}$.

Obviously, $\sigma_{k+1}^{+} \geq \sigma_{k+1}^{-}$. But, $\sigma_{k+1}^{-} \geq 1$. Therefore $\kappa(H_{k+1}) = \sigma_{k+1}^{+}$. By direct computation, we see that $\kappa(H_{k+1})$ attains its minimum value $\sqrt{a_k}$ if and only if $\omega_k = 0$. Hence, minimizing the condition number of the matrix H_{k+1} given by (13) lead us to the following search direction

$$d_{k+1} = -g_{k+1} + \frac{y_k^T g_{k+1}}{y_k^T s_k} s_k - \frac{s_k^T g_{k+1}}{y_k^T s_k} y_k. \tag{34}$$

Observe that (34) is a simple modification of the Hestenes and Stiefel conjugate gradient algorithm [3]. In fact, this is exactly the search direction of the three-term conjugate gradient method proposed by Zhang et al. (see ref. [33]).

Using the procedure of acceleration of conjugate gradient algorithms presented in [24], and taking into consideration the above developments based on singular value study, the following algorithm can be presented.

SVCG Algorithm (Singular Value Conjugate Gradient Algorithm)

Step 1. Select a starting point $x_0 \in R^n$ and compute: $f(x_0)$, $g_0 = \nabla f(x_0)$. Select some positive values for ρ and σ used in Wolfe line search conditions. Set $d_0 = -g_0$ and $k = 0$.

Step 2. Test a criterion for stopping the iterations. If this test is satisfied, then stop; otherwise continue with step 3.

Step 3. Determine the stepsize α_k by using the Wolfe line search (4) and (5).

Step 4. Compute $z = x_k + \alpha_k d_k$, $g_z = \nabla f(z)$ and $y_k = g_k - g_z$.

Step 5. Compute: $\bar{a}_k = \alpha_k g_z^T d_k$ and $\bar{b}_k = -\alpha_k y_k^T d_k$.

Step 6. Acceleration scheme. If $\bar{b}_k > 0$, then compute $\xi_k = -\bar{a}_k / \bar{b}_k$ and update the variables as $x_{k+1} = x_k + \xi_k \alpha_k d_k$, otherwise update the variables as $x_{k+1} = x_k + \alpha_k d_k$.

Step 7. Compute the search direction as in (34).

Step 8. Powell restart criterion. If $\left|g_{k+1}^T g_k\right| > 0.2\left\|g_{k+1}\right\|^2$, then set $d_{k+1} = -g_{k+1}$.

Step 9. Consider $k = k+1$ and go to step 2. ■

From (34) we see that $g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2$, i.e. the search direction (34) satisfies the sufficient descent condition. Besides, $y_k^T d_{k+1} = -(\|y_k\|^2 / y_k^T s_k)(s_k^T g_{k+1})$, i.e. the search direction (34) satisfies the Dai and Liao conjugacy condition.

5. Global convergence analysis

The global convergence analysis of the above algorithms follows the methodology based on bounding the norm of the search direction, presented by Gilbert and Nocedal [34], Nocedal [35] or by Dai et al. [36]. In this section we prove the global convergence of the above algorithms under the following assumptions:

(i) *The level set* $S = \{x \in R^n : f(x) \le f(x_0)\}$ *is bounded. (*x_0 *is the starting point of the iterative method (2).)*

(ii) *In a neighborhood* N *of* S *the function* f *is continuously differentiable and its gradient is Lipschitz continuous, i.e. there exists a constant* $L > 0$ *such that* $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|$, *for all* $x, y \in N$.

Since $\{f(x_k)\}$ is a decreasing sequence, it is clear that the sequence $\{x_k\}$ generated by the proposed algorithms NADCG and SVCG is contained in S. Besides, under the above assumptions on f there exists a constant $\Gamma \ge 0$ such that $\|\nabla f(x)\| \le \Gamma$ for all $x \in S$. For any conjugate gradient method with strong Wolfe line search the following general result holds [35].

PROPOSITION 5.1. *Suppose that the assumptions (i) and (ii) hold. Consider a conjugate gradient algorithm in which, for all* $k \ge 0$, *the search direction* d_k *is a descent direction and the stepsize* α_k *is determined by the Wolfe line search conditions. If*

$$\sum_{k \ge 0} \frac{1}{\|d_k\|^2} = \infty, \tag{35}$$

then the algorithm converges in the sense that

$$\liminf_{k \to \infty} \|g_k\| = 0. \tag{36}$$

For *uniformly convex functions* we can prove that the norm of the direction d_{k+1} computed as in (14) with (27) is bounded above. Therefore, by proposition 5.1 we can prove the following result.

THEOREM 5.1. *Suppose that the assumptions (i) and (ii) hold. Consider the algorithm NADCG where the search direction* d_k *is given by (14) and* ω_k *is computed as in (27). Suppose that* α_k *is computed by the strong Wolfe line search. Suppose that* f *is a uniformly convex function on* S, *i.e. there exists a constant* $\mu > 0$ *such that*

$$(\nabla f(x) - \nabla f(y))^T (x - y) \ge \mu \|x - y\|^2 \tag{37}$$

for all $x, y \in N$. *Then*

$$\lim_{k \to \infty} \|g_k\| = 0. \tag{38}$$

Proof From Lipschitz continuity we have $\|y_k\| \le L\|s_k\|$. On the other hand, from uniform convexity it follows that $y_k^T s_k \ge \mu \|s_k\|^2$. Now, from (27)

$$\omega_k = 2\sqrt{\tau - 1}\frac{\|y_k\|}{\|s_k\|} \le 2\sqrt{\tau - 1}\frac{L\|s_k\|}{\|s_k\|} = 2L\sqrt{\tau - 1}.$$

On the other hand, from (14) we have

$$\begin{aligned}\|d_{k+1}\| &\le \|g_{k+1}\| + \frac{\left|y_k^T g_{k+1}\right|}{y_k^T s_k}\|s_k\| + \omega_k \frac{\left|s_k^T g_{k+1}\right|}{y_k^T s_k}\|s_k\| + \frac{\left|s_k^T g_{k+1}\right|}{y_k^T s_k}\|y_k\| \\ &\le \Gamma + \frac{\|y_k\|\Gamma\|s_k\|}{\mu\|s_k\|^2} + 2L\sqrt{\tau - 1}\frac{\|s_k\|\Gamma\|s_k\|}{\mu\|s_k\|^2} + \frac{\|s_k\|\Gamma\|y_k\|}{\mu\|s_k\|^2} \le \Gamma + 2\frac{L\Gamma}{\mu} + 2L\sqrt{\tau - 1}\frac{\Gamma}{\mu},\end{aligned}$$

showing that (35) is true. By proposition 5.1 it follows that (36) is true, which for uniformly convex functions is equivalent to (38). ■

THEOREM 5.2. *Suppose that the assumptions (i) and (ii) hold. Consider the algorithm SVCG where the search direction* d_k *is given by (34). Suppose that* α_k *is computed by the strong Wolfe line search. Suppose that* f *is a uniformly convex function on* S*, i.e. there exists a constant* $\mu > 0$ *such that* $(\nabla f(x) - \nabla f(y))^T (x - y) \ge \mu\|x - y\|^2$ *for all* $x, y \in N$*. Then*

$$\lim_{k\to\infty}\|g_k\| = 0. \tag{39}$$

Proof From (34) we have

$$\|d_{k+1}\| \le \|g_{k+1}\| + \frac{\left|y_k^T g_{k+1}\right|}{y_k^T s_k}\|s_k\| + \frac{\left|s_k^T g_{k+1}\right|}{y_k^T s_k}\|y_k\| \le \Gamma + \frac{\|y_k\|\Gamma\|s_k\|}{\mu\|s_k\|^2} + \frac{\|s_k\|\Gamma\|y_k\|}{\mu\|s_k\|^2} \le \Gamma + 2\frac{L\Gamma}{\mu},$$

showing that (35) is true. Therefore (36) is true, which for uniformly convex functions is equivalent to (39). ■

REMARK 5.1. Suppose that the assumptions (*i*) and (*ii*) are satisfied. Consider the algorithm NADCG with parameter ω_k defined as in (27). It can be shown that there exists a positive constant Ω such that $0 \le \omega_k \le \Omega$. Hence, if the search directions, computed as in (14), are descent directions and the stepsizes are determined to satisfy the strong Wolfe conditions (4) and (6), then Theorem 3.6 of Dai and Liao [17] ensures the global convergence of the method for general objective functions. On the other hand, the convergence of the SVCG algorithm for general objective functions can be proved following the methodology given by Gilbert and Nocedal [34] and the Theorem 2.3 of Zhang et al. [33].

6. Numerical results and comparisons

The NADCG and SVCG algorithms were implemented in double precision Fortran using loop unrolling of depth 5 and compiled with f77 (default compiler settings) and run on a Workstation Intel Pentium 4 with 1.8 GHz. We selected a number of 80 large-scale unconstrained optimization test problems in generalized or extended form presented in [37]. For each test function we have considered 10 numerical experiments with the number of variables increasing as $n = 1000, 2000, \ldots, 10000$. The algorithms use the Wolfe line search conditions with cubic interpolation [5], $\rho = 0.0001$, $\sigma = 0.8$ and the same stopping criterion $\|g_k\|_\infty \le 10^{-6}$, where $\|.\|_\infty$ is the maximum absolute component of a vector.

The algorithms we compare in these numerical experiments find local solutions. Therefore, the comparisons of algorithms are given in the following context. Let f_i^{ALG1} and f_i^{ALG2} be the optimal values found by ALG1 and ALG2, for problem $i = 1, \ldots, 800$, respectively.

We say that, in the particular problem i, the performance of ALG1 was better than the performance of ALG2 if:

$$\left| f_i^{ALG1} - f_i^{ALG2} \right| < 10^{-3} \tag{40}$$

and the number of iterations (#iter), or the number of function-gradient evaluations (#fg), or the CPU time of ALG1 was less than the number of iterations, or the number of function-gradient evaluations, or the CPU time corresponding to ALG2, respectively. The test problems where the algorithms do not converge to the same function value, according to criterion (40), are discarded from comparisons.

In the first set of numerical experiments we compare NADCG versus SVCG for different values of the parameter τ. Figure 1 shows the Dolan-Moré's performance profiles subject to CPU time metric for different values of the parameter τ. That is, for each method, we plot the fraction of problems for which the method is within a factor of the best time. The left side of the figures gives the percentage of the test problems for which a method is the fastest; the right side gives the percentage of the test problems that are successfully solved by each of the methods. Clearly, the top curve corresponds to the method that solved the most problems in a time that was within a factor of the best time.

From Figure 1, for example for $\tau = 2$, comparing NADCG versus SVCG subject to the number of iterations, we see that NADCG was better in 192 problems (i.e. it achieved the minimum number of iterations for solving 192 problems), SVCG was better in 182 problems and they achieved the same number of iterations in 416 problems, etc. Out of 800 problems, we considered in this numerical study, only for 790 problems does the criterion (40) hold. From figure 1 we see that for different values of the parameter $\tau > 1$ SVCG algorithm is more robust than NADCG. On the other hand, NADCG is way more efficient than SVCG. However, for large values of τ, both algorithms seem to have the same efficiency.

From Figure 1 we see that NADCG algorithm is very little sensitive to the values of the parameter τ. In fact, for $a_k \geq \tau$, from (14) we get:

$$\frac{\partial d_{k+1}}{\partial \tau} = -\frac{1}{\sqrt{\tau - 1}} \frac{s_k^T g_{k+1}}{\|s_k\|^2} s_k, \tag{41}$$

where $\tau > 1$. Therefore, since the gradient of the function f is Lipschitz continuous and the quantity $s_k^T g_{k+1}$ is going to zero it follows that along the iterations $\partial d_{k+1} / \partial \tau$ tends to zero, showing that along the iterations the search direction is less and less sensitive subject to the value of the parameter τ. For uniformly convex functions, using the assumptions from section 5 we get:

$$\left\| \frac{\partial d_{k+1}}{\partial \tau} \right\| \leq \frac{\Gamma}{\sqrt{\tau - 1}}. \tag{42}$$

Therefore, for example, for larger values of τ the variation of d_{k+1} subject to τ decreases showing that the NADCG algorithm is very little sensitive to the values of the parameter τ. This is illustrated in Figure 1 where the performance profiles have the same allure for different values of parameter $\tau > 1$.

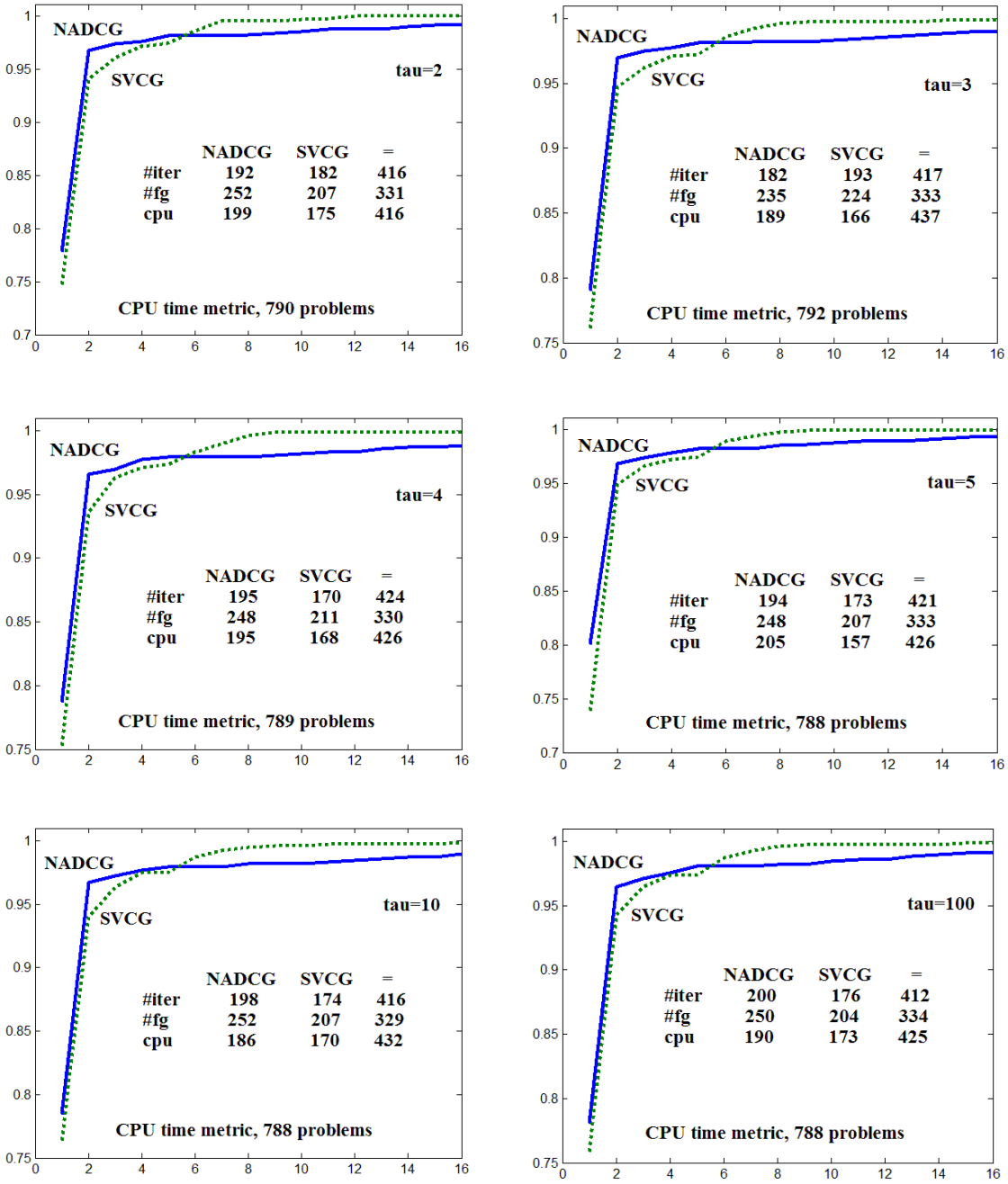

Figure 1. NADCG versus SVCG for different values of τ.

Since, CG-DESCENT [38] is among the best nonlinear conjugate gradient algorithms proposed in the literature, but not necessarily the best, in the second set of numerical experiments we compare our algorithm NADCG versus CG-DESCENT for different values of τ. Figure 2 presents the Dolan-Moré's performance profiles subject to CPU time metric for different values of τ.

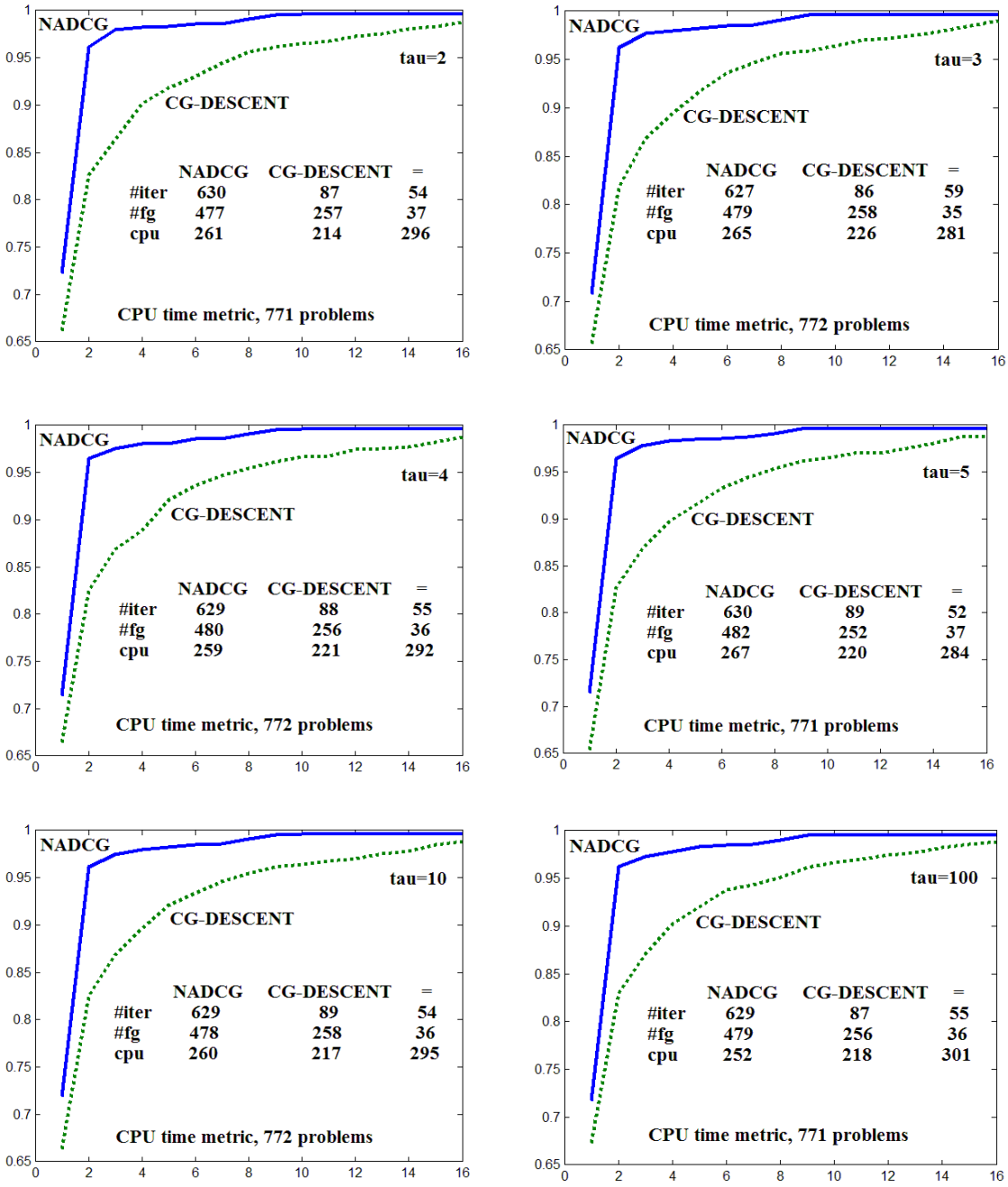

Figure 2. NADCG versus CG-DESCENT for different values of τ.

From Figure 2 we see that the NADCG is more efficient and more robust than CG-DESCENT for any values of the parameter $\tau > 1$ considered in this set of numerical experiments. The NADCG and CG-DESCENT algorithms (and codes) are different in many respects. Since both of them use the Wolfe line search (however, implemented in different manners), these algorithms mainly differ in their choice of the search direction. The search direction d_{k+1} given by (14) and (27) used in NADCG is more elaborate: it is adaptive and the eigenvalues of the matrix defined by it are clustered. In addition it satisfies both the descent condition and the conjugacy condition in a restart environment.

In the third set of numerical experiments we compare SVCG versus CG-DESCENT. Figure 3 shows the Dolan-Moré's performance profiles subject to CPU time metric.

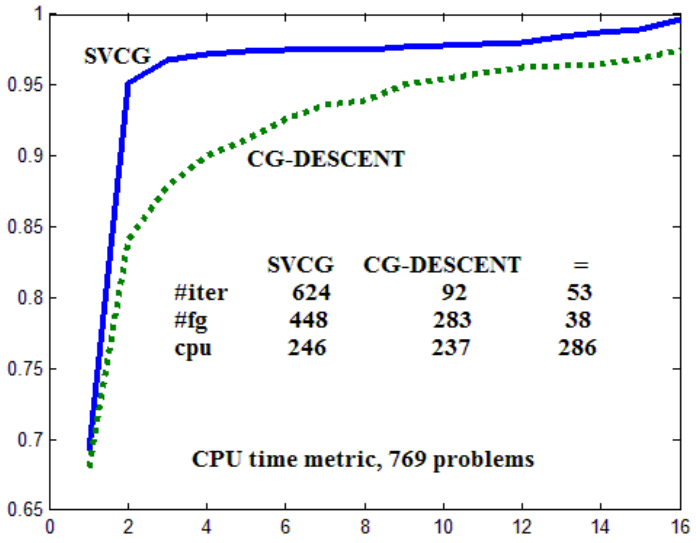

Figure 3. SVCG versus CG-DESCENT.

We see that, at least for this set of unconstrained optimization test problems, SVCG algorithm is more efficient and more robust than CG-DESCENT. From Propositions 2.1 and 2.2 we see that the search direction (34) used in SVCG algorithm satisfies both the sufficient descent condition and the Dai and Liao conjugacy condition. Even if the search direction (34) is very simple, it is more elaborate than the CG-DESCENT direction due to the presence of the term involving y_k.

In the last set of numerical experiments, we present comparisons between NADCG, SVCG and CG-DESCENT conjugate gradient algorithms for solving some applications from the MINPACK-2 test problem collection [20]. In Table 1 we present these applications, as well as the values of their parameters.

Table 1.
Applications from the MINPACK-2 collection.

A1	Elastic–plastic torsion [39, pp. 41–55], $c = 5$
A2	Pressure distribution in a journal bearing [40], $b = 10$, $\varepsilon = 0.1$
A3	Optimal design with composite materials [41], $\lambda = 0.008$
A4	Steady-state combustion [42, pp. 292–299], [43], $\lambda = 5$
A5	Minimal surfaces with Enneper conditions [44, pp. 80–85]

The infinite-dimensional version of these problems is transformed into a finite element approximation by triangulation. Thus a finite-dimensional minimization problem is obtained whose variables are the values of the piecewise linear function at the vertices of the triangulation. The discretization steps are $nx = 1{,}000$ and $ny = 1{,}000$, thus obtaining minimization problems with 1,000,000 variables. A comparison between NADCG (Powell restart criterion, $\|\nabla f(x_k)\|_\infty \le 10^{-6}$, $\rho = 0.0001$, $\sigma = 0.8$, $\tau = 2$), SVCG (Powell restart criterion,

$\|\nabla f(x_k)\|_\infty \leq 10^{-6}$, $\rho = 0.0001$, $\sigma = 0.8$) and CG-DESCENT (version 1.4, Wolfe line search, default settings, $\|\nabla f(x_k)\|_\infty \leq 10^{-6}$) for solving these applications is given in Table 2.

Table 2.
Performance of NADCG, SVCG and CG-DESCENT. 1,000,000 variables. CPU seconds.

	NADCG			**SVCG**			**CG-DESCENT**		
	#iter	#fg	cpu	#iter	#fg	cpu	#iter	#fg	cpu
A1	1113	2257	352.55	1111	2253	349.89	1145	2291	474.64
A2	2845	5718	1141.93	2845	5718	1137.32	3370	6741	1835.51
A3	4700	9437	2764.84	4372	8763	2570.41	4814	9630	3949.71
A4	1413	2864	2037.50	1413	2864	2023.98	1802	3605	3786.25
A5	1285	2606	581.92	1291	2607	580.91	1225	2451	753.75
Total	11356	22882	6878.74	11032	22205	**6662.51**	12356	24718	10799.86

From Table 2, we see that, subject to the CPU time metric, both NADCG and SVCG algorithms have similar performances, SVCG being slightly faster. Observe that subject to CPU time metric NADCG algorithm is top performer versus CG-DESCENT and the difference is significant, about **3921.12 seconds** for solving all these five applications. Similarly, SVCG algorithm is top performer versus CG-DESCENT with a difference of **4137.35 seconds** for solving the applications considered in this numerical study.

7. Conclusions

A theoretical development and a numerical study of two approaches based on eigenvalue analysis and singular values, respectively was presented in the context of conjugate gradient algorithms. The search direction is computed as the sum of the negative gradient and an arbitrary vector which was determined by minimizing the quadratic approximation of objective function at the current point. The solution of this quadratic minimization problem is a function of the inverse Hessian. In this paper we introduce a special expression of the inverse Hessian of the objective function which depends by a positive parameter ω_k. For any nonnegative values of this parameter the search direction satisfies both the sufficient descent condition and the Dai-Liao's conjugacy condition. Thus, the algorithm is a conjugate gradient one. The parameter in the search direction is determined by using two different approaches.

The first one is based on clustering the spectrum of the matrix defining the search direction. This idea is taken from the linear conjugate gradient, where clustering the eigenvalues of the matrix is very benefic with respect to the convergence. Mainly, in our nonlinear case, clustering the eigenvalues reduces to determine the value of the parameter ω_k to minimize the largest eigenvalue of the matrix. The adaptive computation of the parameter ω_k in the search direction is subject to a positive constant which has a very little impact on the performances of our algorithm.

The second approach is based on minimizing the condition number of the matrix defining the search direction. In this case the minimum value of the condition number is obtained for $\omega_k = 0$. The corresponding search direction is very simple, being a modification of the Hestenes and Stiefel's conjugate gradient algorithm [3] or being exactly the three-term conjugate gradient algorithm suggested by Zhang et al [33]. Both these approaches are dependent by the expression of the inverse Hessian approximation.

The stepsize is computed using the classical Wolfe line search conditions with a special initialization. In order to improve the reducing the values of the objective function to be minimized an acceleration scheme is used. Under classical assumptions, both algorithms are

globally convergent. Numerical experiments and intensive comparisons using 800 unconstrained optimization test problems, of different dimensions and complexity, proved that both algorithms have similar performances, the adaptive conjugate gradient algorithm based on eigenvalue clustering being slightly more efficient than the corresponding algorithm using the minimizing the condition number. On the other hand, the algorithm using the idea of minimizing the condition number is indeed more robust. In an effort to see the performances of these algorithms we found that both algorithms are more efficient and more robust than CG-DESCENT algorithm. By solving five large-scale nonlinear optimization applications from MINPACK-2 collection, up to 10^6 variables, we prove that both NADCG and SVCG algorithms are obvious more efficient than CG-DESCENT.

The conclusion is that both these techniques based on eigenvalues clustering or on minimizing the condition number of the iteration matrix using the singular values are suitable to get efficient and robust conjugate gradient algorithms. Having in view the Remark 4.1, observe that by minimizing σ_{k+1}^{+} we minimize λ_{k+1}^{+}. Therefore, these two approaches considered in this paper represent two different ways to basically pursue similar ideas based on eigenvalues or on singular values of the iteration matrix, respectively.

It is worth saying that by using some other expressions to the inverse Hessian approximation, other conjugate gradient algorithms can be obtained. On the other hand, another conjugate gradient algorithm can also be generated if, instead of clustering the eigenvalues of H_{k+1} by minimizing the largest eigenvalue with respect to ω_k, a different standpoint for eigenvalues clustering is used, for example by minimizing the trace of H_{k+1}, subject to some constraints on ω_k.

References

[1] Wolfe, P., 1969, Convergence conditions for ascent methods. *SIAM Review*, **11**, 226-235.

[2] Wolfe, P., 1971, Convergence conditions for ascent methods. II: Some corrections. *SIAM Review*, **13**, 185-188.

[3] Hestenes, M.R. and Stiefel, E., 1952, Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards Sec. B.* **48**, 409-436.

[4] Stiefel, E., 1952, Über einige Methoden der Relaxationsrechnung. *Z. Angew. Math. Phys.*, **3**, 1-33.

[5] Sun, W. and Yuan, Y.X., 2006, *Optimization Theory and Methods. Nonlinear Programming.* (New York: Springer Science + Business Media).

[6] Reid, J.K., 1971, On the method of conjugate gradients for solution of large sparse systems of linear equations. In: J.K. Reid, (Ed) *Large Sparse Sets of Linear Equations* (London: Academic Press), pp. 231-254.

[7] Axelsson, O., 1976, A class of iterative methods for finite element equations. *Com. Meth. Appl. Mech. Eng.*, **9**, 123-137.

[8] Axelsson, O. and Lindskog, G., 1986, On the rate of convergence of the preconditioned conjugate gradient methods. *Numer. Math.*, **48**, 499-523.

[9] Strakoš, Z., 1991, On the real convergence rate of the conjugate gradient method. *Linear Algebra and its Applications*, **154-156**, 535-549.

[10] Van der Sluis, A. and Van der Vorst, H.A., 1986, The rate of convergence of conjugate gradients. *Numer. Math.*, **48**, 543-560.

[11] Meurant, G., 1999, *Computer Solution of Large Linear Systems.* Studies in Mathematics and its Applications, volume 28, (Amsterdam: North Holland).

[12] Winther, R., 1980, Some superlinear convergence results for the conjugate gradient method. *SIAM J. Numer. Anal.*, **17**, 14-17.

[13] Fletcher, R. and Reeves, C.M., 1964, Function minimization by conjugate gradients. *Computer Journal*, **7**, 149-154.
[14] Polak, E. and Ribière, G., 1969, Note sur la convergence de directions conjuguée, *Rev. Francaise Informat. Recherche Operationelle*, 3e Année, **16**, 35-43.
[15] Polyak, B.T., 1969, The conjugate gradient method in extreme problems. *USSR Comp. Math. Math. Phys.* **9**, 94-112.
[16] Dai, Y.H. and Yuan, Y., 1999, A nonlinear conjugate gradient method with a strong global convergence property. *SIAM J. Optim.* **10**, 177-182.
[17] Dai, Y.H. and Liao, L.Z., 2001, New conjugacy conditions and related nonlinear conjugate gradient methods. *Appl. Math. Optim*. 43, 87-101.
[18] Hager W.W. and Zhang, H., 2006, A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization*, **2**(1), 35-58.
[19] Hager, W.W. and Zhang, H., 2005, A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization*, **16**, 170-192.
[20] Averick, B.M., Carter, R.G., Moré, J.J. and Xue, G.L., 1992, The MINPACK-2 test problem collection, Mathematics and Computer Science Division, Argonne National Laboratory, Preprint MCS-P153-0692.
[21] Shanno, D.F., 1978, Conjugate gradient methods with inexact searches. *Mathematics of Operations Research*, **3**, 244-256.
[22] Kratzer, D., Parter, S.V. and Steuerwalt, M., 1983, Block splittings for the conjugate gradient method. *Comp. Fluid*, **11**, 255-279.
[23] Kaporin, I.E., 1994, New convergence results and preconditioning strategies for the conjugate gradient methods. *Numerical Linear Algebra with Applications*, **1**(2), 179-210.
[24] Andrei, N., 2009, Acceleration of conjugate gradient algorithms for unconstrained optimization. *Applied Mathematics and Computation*, **213**, 361-369.
[25] Dennis, J.E. and Schnabel, R.B., 1983, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* (Englewood Cliffs, NJ: Prentice-Hall, Inc.).
[26] Polyak, B.T., 1987, *Introduction to Optimization*. (New York: Optimization Software, Inc., Publications Division).
[27] Dai, Y.H., Liao, L.Z. and Duan, Li, 2004, On restart procedures for the conjugate gradient method. *Numerical Algorithms*, **35**, 249-260.
[28] Nocedal, J. and Wright, S.J., 2006, *Numerical optimization*. (2nd ed.). Springer Series in Optimizations Research. (New York: Springer Science+Business Media).
[29] Luenberger, D.G. and Ye, Y., 2008, *Linear and Nonlinear Programming*. International Series in Operations Research & Management Science 116. (New York: Springer Science+Business Media).
[30] Babaie-Kafaki, S. and Ghanbari, R., 2014, A modified scaled conjugate gradient method with global convergence for nonconvex functions. *Bulletin of the Belgian Mathematical Society-Simon Stevin*, **21**(3), 465-47.
[31] Babaie-Kafaki, S. and Ghanbari, R., 2014, The Dai-Liao nonlinear conjugate gradient method with optimal parameter choices. *European Journal of Operational Research*, **234**, 625-630.
[32] Watkins, D.S., 2002, *Fundamentals of Matrix Computations.* (2nd ed.). (New York: John Wiley and Sons, Inc.).
[33] Zhang, L., Zhou, W. and Li, D.H., 2007, Some descent three-term conjugate gradient methods and their global convergence. *Optimization Methods and Software*, **22**(4), 697-711.
[34] Gilbert, J.C. and Nocedal, J., 1992, Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal on Optimization*, **2**(1), 21-42.
[35] Nocedal, J., 1996, Conjugate gradient methods and nonlinear optimization. In: Adams, L. and Nazareth, J.L., (Eds) *Linear and Nonlinear Conjugate Gradient Related Methods*, SIAM, 9-23.

[36] Dai, Y., Han, J., Liu, G., Sun, D., Yin, H. and Yuan, Y-X., 1999, Convergence properties of nonlinear conjugate gradient methods. *SIAM Journal on Optimization,* **10**(2), 345-358.
[37] Andrei, N., 2013, Another collection of large-scale unconstrained optimization test functions. Technical Report January 30, Center for Advanced Modeling and Optimization, Bucharest.
[38] Hager, W.W. and Zhang, H., 2006, Algorithm 851: CG-DESCENT, a conjugate gradient method with guaranteed descent. *ACM Trans. Math. Softw*. **32**, 113-137.
[39] Glowinski, R., 1984, *Numerical Methods for Nonlinear Variational Problems* (Berlin: Springer-Verlag).
[40] Cimatti, G., 1977, On a problem of the theory of lubrication governed by a variational inequality, *Applied Mathematics and Optimization* **3**, 227–242.
[41] Goodman, J., Kohn, R. and Reyna, L., 1986, Numerical study of a relaxed variational problem from optimal design, *Computer Methods in Applied Mechanics and Engineering* **57**, 107–127.
[42] Aris, R., 1975, *The Mathematical Theory of Diffusion and Reaction in Permeable Catalysts* (Oxford).
[43] Bebernes, J. and Eberly, D., 1989, Mathematical Problems from Combustion Theory, In: *Applied Mathematical Sciences*, vol. 83, (Berlin: Springer-Verlag).
[44] Nitsche, J.C.C., 1989, *Lectures on Minimal Surfaces,* vol. 1, (Cambridge University Press).

---oooOooo---