An adaptive conjugate gradient algorithm with clustering the singular values of the search direction matrix 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

July 18, 2016

Abstract. An adaptive nonlinear conjugate gradient algorithm based on clustering the singular values of the search direction matrix and on the inexact Wolfe line search is presented. The search direction is dependent to a positive parameter. The value of this parameter is selected in such a way that the singular values of the matrix defining the search direction are clustered around 1. We prove that for general nonlinear functions and independent of the line search procedure the search direction satisfies both the sufficient descent condition and the Dai and Liao conjugacy condition. According to the value of the parameter, the algorithm uses the suggested search direction, or it triggers to the Hestenes and Stiefel direction. Under classical assumptions, for uniformly convex functions, we prove that the algorithm is globally convergent. Using a set of 800 unconstrained optimization test problems we prove that our algorithm is significantly more efficient and more robust than CG-DESCENT algorithm and slightly more efficient and more robust than ADCG algorithm. By solving five applications from the MINPACK-2 test problem collection, with 10^6 variables, we show that the suggested adaptive conjugate gradient algorithm is top performer versus CG_DESCENT.

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

1. Introduction

Let us consider the unconstrained optimization problem

$$\min\{f(x), x \in R^n\}, \tag{1.1}$$

where $f: R^n \to R$ is continuously differentiable and bounded below. For solving this problem we suggest a nonlinear conjugate gradient algorithm, where the iterates x_k, $k = 0,1,\ldots$ are generated as

$$x_{k+1} = x_k + \alpha_k d_k, \tag{1.2}$$

the stepsize α_k is positive and the search directions d_k are computed as:

$$d_{k+1} = -g_{k+1} + \beta_k^N s_k, \quad d_0 = -g_0, \tag{1.3}$$

$$\beta_k^N = \frac{y_k^T g_{k+1}}{y_k^T s_k} - \omega_k \frac{\|y_k\|^2}{y_k^T s_k} \frac{s_k^T g_{k+1}}{y_k^T s_k}, \tag{1.4}$$

where $\|.\|$ is Euclidian norm, $g_k = \nabla f(x_k)$, $y_k = g_{k+1} - g_k$, $s_k = x_{k+1} - x_k$ and ω_k is a positive parameter which follows to be determined. Observe that (1.4) is very close to the conjugate gradient parameter of Hager and Zhang conjugate gradient algorithm [22], where $\omega_k = 2$. In this paper we are interested to determine a value for the parameter ω_k in such a way to get an

efficient and robust conjugate gradient algorithm able to solve large-scale unconstrained optimization problems.

Observe that, if f is a quadratic function and the step length α_k is selected to achieve the exact minimum of f in the direction d_k, then $s_k^T g_{k+1} = 0$, i.e., the formula (1.4) for β_k^N reduces to the Hestenes and Stiefel [24] (HS) scheme. However, in this paper we consider general nonlinear functions and inexact line search based on Wolfe conditions [33, 34]:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \delta \alpha_k g_k^T d_k, \tag{1.5}$$

$$g_{k+1}^T d_k \geq \sigma g_k^T d_k, \tag{1.6}$$

where $0 < \delta \leq \sigma < 1$.

We see that the parameter β_k^N defined in (1.4) can be viewed as a modification of the HS conjugate gradient algorithm. Observe that if the step length α_k is computed according to the Wolfe line search conditions (1.5) and (1.6), then $y_k^T s_k > 0$. Therefore, along the iterations, when the step s_k is small (in norm), the factor y_k in the numerator of $\beta_k^{HS} = y_k^T g_{k+1} / y_k^T s_k$ tends to zero. On the other hand, when the step s_k is small, again the factor y_k in the numerator the second term of β_k^N tends to zero. Hence, β_k^N becomes small and the new search direction d_{k+1} is essentially a small alteration of the steepest descent direction $-g_{k+1}$. In other words our method automatically adjust β_k^N to avoid or at least to attenuate jamming, which is the main defect of the steepest descent direction.

It is worth saying that a conjugate gradient method related to our computational scheme (1.3) and (1.4) is that given by Dai and Liao [12], in which the parameter β_k^N in (1.3) is replaced by:

$$\beta_k^{DL} = \frac{1}{y_k^T s_k}(y_k - t s_k)^T g_{k+1}, \tag{1.7}$$

where $t > 0$ is a constant parameter. For different choices of t, the computational scheme of Dai and Liao generates different results. An optimal value for t in this algorithm is not known (see [3]). Observe that the method (1.3) and (1.4) can be viewed as an adaptive version of (1.7) where t, at each iteration, is updated as $t = \omega_k \|y_k\|^2 / (y_k^T s_k)$.

The purpose of this paper is to find a value of the parameter ω_k, in such a way to get an efficient and robust conjugate gradient algorithm. For this we suggest using the structure of the singular values of the matrix associated to the search direction (1.3) and (1.4).

Using (1.4) in (1.3) the search direction in our algorithm is computed as:

$$d_{k+1} = -g_{k+1} + \frac{y_k^T g_{k+1}}{y_k^T s_k} s_k - \omega_k \frac{\|y_k\|^2}{y_k^T s_k} \frac{s_k^T g_{k+1}}{y_k^T s_k} s_k, \tag{1.8}$$

where ω_k is a positive parameter. Now, considering the Perry [28] idea the search direction (1.8) can be represented as:

$$d_{k+1} = -H_{k+1} g_{k+1}, \tag{1.9}$$

where

$$H_{k+1} = I - \frac{s_k y_k^T}{y_k^T s_k} + \omega_k \frac{\|y_k\|^2}{y_k^T s_k} \frac{s_k s_k^T}{y_k^T s_k}. \tag{1.10}$$

Observe that H_{k+1} is a sum of a symmetric matrix $I + \omega_k [\|y_k\|^2 / (y_k^T s_k)^2] s_k s_k^T$ and a non-symmetric one $s_k y_k^T / y_k^T s_k$, i.e., H_{k+1} is a non-symmetric matrix. Therefore, (1.9) represents an

ad hoc algebraic expression of the search direction d_{k+1} in which the non-symmetric matrix H_{k+1} is not an approximation to the inverse Hessian $\nabla^2 f(x_{k+1})^{-1}$. It is this algebraic form of the parameter β_k^N, given by (1.4), which leads us to this expression of H_{k+1}. In other words, strictly speaking, (1.9) is not a real, quasi-Newton direction.

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 (1.8). The approach used here is to determine the value of the parameter ω_k in (1.8) in such a way to minimize the condition number of the iterate matrix H_{k+1} in (1.9). In other words, the idea of this paper is to determine the value of the parameter ω_k in order to achieve more numerical stability in computation of the search direction (1.9). The effect of ill-conditioning of H_{k+1} on the iterative algorithm (1.2) using the search direction (1.9) can be explained as follows (see also [8, 9]). For a vector $v \in R^n$, let us denote $fl(v) = [fl(v_1), \ldots, fl(v_n)]^T$ as a vector in R^n, where $fl(v_i)$, $i = 1, \ldots, n$, is the nearest floating point number to v_i. Using (9) we have

$$fl(d_{k+1}) = -H_{k+1} fl(g_{k+1}), \quad k = 0, 1, \ldots.$$

Therefore,

$$\begin{aligned} \|fl(d_{k+1}) - d_{k+1}\| &= \|-H_{k+1}(fl(g_{k+1}) - g_{k+1})\| \\ &\leq \|H_{k+1}\| \|fl(g_{k+1}) - g_{k+1}\|. \end{aligned}$$

Now, if the matrix H_{k+1} is nonsingular, it follows that

$$\|H_{k+1}\| \frac{\|fl(g_{k+1}) - g_{k+1}\|}{\|g_{k+1}\|} \geq \frac{\|fl(d_{k+1}) - d_{k+1}\|}{\|-H_{k+1}^{-1} d_{k+1}\|} \geq \frac{\|fl(d_{k+1}) - d_{k+1}\|}{\|H_{k+1}^{-1}\| \|d_{k+1}\|}.$$

Therefore, the following inequality between the relative errors of d_{k+1} and g_{k+1} can be established:

$$\frac{\|fl(d_{k+1}) - d_{k+1}\|}{\|d_{k+1}\|} \leq \kappa(H_{k+1}) \frac{\|fl(g_{k+1}) - g_{k+1}\|}{\|g_{k+1}\|}, \tag{1.11}$$

where $\kappa(H_{k+1}) = \|H_{k+1}\| \|H_{k+1}^{-1}\|$ is the condition number of H_{k+1}. Therefore, if the iteration matrix H_{k+1} is ill-conditioned, then even for small values of the relative error of g_{k+1}, the relative error of the search direction d_{k+1} may be large. In other words, if $\kappa(H_{k+1})$ is large, then the system (1.9) could be very sensitive to perturbations in g_{k+1}. The idea of this paper is to select a value for the parameter ω_k in (1.10) in such a way to minimize the condition number of the iteration matrix H_{k+1}. Minimizing the condition number of the iteration matrix in conjugate gradient algorithms have also been considered by Babaie-Kafaki and Ghanbari [8, 9] for Dai-Liao nonlinear conjugate gradient algorithm, or by Liu and Xu [25] for Perry descent conjugate gradient algorithm, leading them to efficient and robust conjugate gradient algorithms.

The structure of the paper is as follows. The algorithm and its properties are presented in Section 2. We prove that the search direction used by this algorithm satisfies both the sufficient descent condition and the Dai and Liao conjugacy condition, independent of the line search. The parameter ω_k in the search direction (1.8) is determined by minimizing the condition number of the iteration matrix H_{k+1}, i.e., by clustering the singular values of this matrix around to 1. Using standard assumptions, Section 3 presents the global convergence of the algorithm for uniformly convex functions. In Section 4 the numerical comparisons of our algorithm versus CG-

DESCENT [23] and ADCG [5] conjugate gradient algorithms are presented. The computational results, for a set of 800 unconstrained optimization test problems, show that this new algorithm substantially outperforms CG-DESCENT, and is slightly more efficient and more robust than ADCG. Considering five applications from the MINPACK-2 test problem collection [7], with 10^6 variables, we show that our algorithm is way more efficient and more robust than CG-DESCENT.

2. The algorithm

An important property of our conjugate gradient algorithm is that the search direction (1.8) always yields descent when $y_k^T s_k \neq 0$, a condition which is satisfied when f is strongly convex, or the step length α_k is computed according to the Wolfe conditions. The following properties of the search direction (1.8) can immediately be proved.

Theorem 2.1. *If the step length α_k in (1.2) is determined by the Wolfe line search conditions (1.5) and (1.6) and $\omega_k \geq 1/4$, then the search direction (1.8) satisfies the sufficient descent condition*

$$g_{k+1}^T d_{k+1} \leq -\left(1-\frac{1}{4\omega_k}\right)\|g_{k+1}\|^2 \leq 0. \tag{2.1}$$

Proof. Since $d_0 = -g_0$, it follows that $g_0^T d_0 = -\|g_0\|^2 \leq 0$. From (1.8) we get:

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + \frac{(y_k^T g_{k+1})(s_k^T g_{k+1})}{y_k^T s_k} - \omega_k \frac{\|y_k\|^2}{y_k^T s_k}\frac{(s_k^T g_{k+1})^2}{y_k^T s_k}. \tag{2.2}$$

Now, we apply the inequality $u^T v \leq \frac{1}{2}\left(\|u\|^2 + \|v\|^2\right)$ to the second term in (2.2) with

$$u = \frac{1}{\sqrt{2\omega_k}}(y_k^T s_k) g_{k+1} \quad \text{and} \quad v = \sqrt{2\omega_k}(s_k^T g_{k+1}) y_k$$

to obtain

$$\frac{(y_k^T g_{k+1})(s_k^T g_{k+1})}{y_k^T s_k} = \frac{(y_k^T g_{k+1})(y_k^T s_k)(s_k^T g_{k+1})}{(y_k^T s_k)^2} \leq \frac{1}{4\omega_k}\|g_{k+1}\|^2 + \omega_k \frac{\|y_k\|^2 (s_k^T g_{k+1})^2}{(y_k^T s_k)^2}.$$

Therefore, introducing this estimation in (2.2) we get (2.1) showing that the search direction (1.8) satisfies the sufficient descent condition when $\omega_k \geq 1/4$. ■

If f is strongly convex or the line search satisfies the Wolfe conditions (1.5) and (1.6), then $y_k^T s_k > 0$ and our computational scheme yield descent. Note that if $\omega_k \geq 1/4$, then $g_{k+1}^T d_{k+1}$ is bounded by $-(1-1/4\omega_k)\|g_{k+1}\|^2$, while in some other computational schemes, for example, of Dai and Yuan [14, 15] only the negativity of $g_{k+1}^T d_{k+1}$ is established. We note in passing that if $\omega_k = 2$, then from (2.1) $g_{k+1}^T d_{k+1} \leq -\frac{7}{8}\|g_{k+1}\|^2$, like in [22].

Another important property of the search direction (1.8) is that it satisfies the Dai and Liao conjugacy condition [12], which addresses to the inexact line search, but reduces to the old conjugacy condition $y_k^T d_k = 0$ when the line search is exact.

Theorem 2.2. *Consider* $\omega_k > 0$ *and the step length* α_k *in (1.2) is determined by the Wolfe line search conditions (1.5) and (1.6). Then the search direction (1.8) 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 (1.5) and (1.6) it follows that $y_k^T s_k > 0$, therefore $v_k > 0$. ■

In the following we are interested to specify a procedure for ω_k computation in (1.8) based on minimizing the condition number of the iterate matrix H_{k+1}, given by (1.10). For this, we briefly present the singular value analysis and the condition number of a matrix. The following definitions and theorems, taken from Watkins [32], clarify some aspects of this concept of condition number of a matrix.

Theorem 2.3. [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*

$$Av_i = \begin{cases} \sigma_i u_i, & i = 1, \ldots, r, \\ 0, & i = r+1, \ldots, m, \end{cases} \quad \text{and} \quad A^T u_i = \begin{cases} \sigma_i v_i, & i = 1, \ldots, r, \\ 0, & i = r+1, \ldots, n. \end{cases} \tag{2.3}$$

Definition 2.1. *The scalars* $\sigma_1, \ldots, \sigma_r$ *from the theorem 2.3 are called the singular values of the matrix* A.

Based on the Theorem 2.3, for any nonzero matrix $A \in R^{n \times m}$ with rank r it follows that

$$\|A\|_F^2 = \sigma_1^2 + \ldots + \sigma_r^2, \tag{2.4}$$

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

$$|\det(A)| = \sigma_1 \times \sigma_2 \times \cdots \times \sigma_n. \tag{2.5}$$

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

Theorem 2.4. [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 2.3. *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 (1.5) and (1.6). The structure of the singular values of the matrix H_{k+1} is given by the following theorem.

Theorem 2.5. *Suppose that the step length* α_k *is determined by the Wolfe line search conditions (1.5) and (1.6). Let* H_{k+1} *be defined by (1.10). Then* H_{k+1} *is a nonsingular matrix and its singular values consist of 1 (*$n-2$ *multiplicity),* σ_{k+1}^+ *and* σ_{k+1}^-*, where:*

$$\sigma_{k+1}^+ = \frac{1}{2}\left[\sqrt{(\omega_k a_k + 1)^2 + (a_k - 1)} + \sqrt{(\omega_k a_k - 1)^2 + (a_k - 1)}\right], \tag{2.6}$$

$$\sigma_{k+1}^- = \frac{1}{2}\left[\sqrt{(\omega_k a_k + 1)^2 + (a_k - 1)} - \sqrt{(\omega_k a_k - 1)^2 + (a_k - 1)}\right], \tag{2.7}$$

and

$$a_k = \frac{\|s_k\|^2 \|y_k\|^2}{(y_k^T s_k)^2} > 1. \tag{2.8}$$

Proof. By the Wolfe line search conditions (1.5) and (1.6) 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, $\dim(V) \le 2$ and $\dim(V^\perp) \ge n-2.$ 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 (1.10) leads to

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

Therefore, the matrix H_{k+1} has $n-2$ singular values equal to 1. Now, we are interested to find the rest of the two remaining singular values denoted as σ_{k+1}^+ and σ_{k+1}^-, respectively. From the formula of algebra (see for example [31])

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

where $p = -\dfrac{s_k}{y_k^T s_k}, \quad q = y_k, \quad u = \omega_k \dfrac{\|y_k\|^2}{(y_k^T s_k)^2} s_k$ and $v = s_k,$ it follows that

$$\det(H_{k+1}) = \omega_k \frac{\|s_k\|^2 \|y_k\|^2}{(y_k^T s_k)^2} = \omega_k a_k, \tag{2.9}$$

where a_k is given by (2.8). Since $\omega_k > 0$ and $a_k > 1,$ it follows that H_{k+1} is a nonsingular matrix. Now, by direct computation we get:

$$tr(H_{k+1}^T H_{k+1}) = n - 2 + a_k + \omega_k^2 a_k^2. \tag{2.10}$$

Since $\|H_{k+1}\|_F^2 = tr(H_{k+1}^T H_{k+1}),$ from (2.4) we get

$$(\sigma_{k+1}^+)^2 + (\sigma_{k+1}^-)^2 = a_k + \omega_k^2 a_k^2. \tag{2.11}$$

Also, from (2.5) and (2.9) we have

$$\sigma_{k+1}^{+}\sigma_{k+1}^{-} = \omega_k a_k. \tag{2.12}$$

Now, from (2.11) and (2.12), after some simple algebraic manipulations we obtain:

$$\sigma_{k+1}^{+} + \sigma_{k+1}^{-} = \sqrt{\omega_k^2 a_k^2 + 2\omega_k a_k + a_k}, \tag{2.13}$$

Therefore, from (2.12) and (2.13), the remaining singular values σ_{k+1}^{+} and σ_{k+1}^{-} of H_{k+1} are the roots of the following quadratic polynomial

$$\sigma^2 - \sqrt{\omega_k^2 a_k^2 + 2\omega_k a_k + a_k}\,\sigma + \omega_k a_k = 0. \tag{2.14}$$

Clearly, the other two singular values of the matrix H_{k+1} are determined from (2.14) as (2.6) and (2.7) respectively. Observe that $a_k > 1$ follows from Wolfe conditions and the inequality

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

Observe that since $a_k > 1$ it follows that the singular values σ_{k+1}^{+} and σ_{k+1}^{-} are well defined by (2.6) and (2.7), respectively. The following two proposition prove some important properties of the singular values σ_{k+1}^{+} and σ_{k+1}^{-}.

Proposition 2.1. *For the singular value σ_{k+1}^{-} defined in (2.7), we have $\sigma_{k+1}^{-} \le 1$.*

Proof. Observe that if $\omega_k a_k < 1$, then since $\sigma_{k+1}^{-} \le \sigma_{k+1}^{+}$, from (2.12) we have that $\sigma_{k+1}^{-} \le 1$. On the other hand, if $\omega_k a_k \ge 1$, then from (2.6) we have:

$$\sigma_{k+1}^{+} \ge \frac{1}{2}(\omega_k a_k + 1) + \frac{1}{2}(\omega_k a_k - 1) = \omega_k a_k. \tag{2.15}$$

With this, from (2.12) it follows that $\sigma_{k+1}^{-} \le 1$. $\blacksquare$

Proposition 2.2. *For the singular value σ_{k+1}^{+} defined in (2.6), we have $\sigma_{k+1}^{+} \ge 1$.*

Proof. As in Proposition 2.1 above, if $\omega_k a_k \ge 1$, then from (2.15) we have that $\sigma_{k+1}^{+} \ge 1$. On the other hand, if $\omega_k a_k < 1$, then from (2.6) we have

$$\sigma_{k+1}^{+} \ge \frac{1}{2}(\omega_k a_k + 1) + \frac{1}{2}(1 - \omega_k a_k) = 1. \qquad \blacksquare$$

Now, since $a_k > 1$, from (2.12) and (2.13) it follows that both σ_{k+1}^{+} and σ_{k+1}^{-} are positive. Therefore, from the above propositions we have $0 < \sigma_{k+1}^{-} \le 1 \le \sigma_{k+1}^{+}$. From Theorem 2.4 we have that

$$\kappa(H_{k+1}) = \frac{\sigma_{k+1}^{+}}{\sigma_{k+1}^{-}}. \tag{2.16}$$

As we have mentioned in Section 1 in order to enhance the numerical stability in the search direction computation, it is reasonable to determine the value of the parameter ω_k in (1.8) by minimizing the condition number of H_{k+1}. In a simple computational scheme, from (2.16) we see

that minimizing $\kappa(H_{k+1})$ is to minimize the distance between σ_{k+1}^{-} and σ_{k+1}^{+}. Therefore, the optimal value of ω_k, denoted ω_k^*, is determined as:

$$\omega_k^* = \arg\min(\sigma_{k+1}^{+} - \sigma_{k+1}^{-}), \tag{2.17}$$

thus making σ_{k+1}^{+} as close as possible to σ_{k+1}^{-}. Since $0 < \sigma_{k+1}^{-} \leq 1 \leq \sigma_{k+1}^{+}$, it follows that ω_k^* solution of (2.17) makes $\kappa(H_{k+1}) \geq 1$ as close as possible to 1. From (2.6) and (2.7), a simple algebraic development shows that

$$\omega_k^* = \frac{1}{a_k}, \tag{2.18}$$

where $a_k > 1$ is given by (2.8). Therefore, for $\omega_k = 1/a_k$ the singular values of H_{k+1} are clustered around 1. Notice that for $\omega_k = 1/a_k$ the matrix H_{k+1} from (1.10) becomes:

$$H_{k+1} = I - \frac{s_k y_k^T}{y_k^T s_k} + \frac{s_k s_k^T}{\|s_k\|^2}.$$

In the following, from Theorem 2.1, we observe that the necessary condition for the sufficient descent condition of the search direction is $\omega_k \geq 1/4$. Therefore, the condition for minimizing $\kappa(H_{k+1})$ is $a_k \leq 4$. Now, we can define our algorithm as follows. If $a_k \leq 4$, then we select $\omega_k = 1/a_k$ in (1.8) in order to achieve both the sufficient descent condition and minimizing the condition number $\kappa(H_{k+1})$. Otherwise, the algorithm uses the Hestenes and Stiefel direction.

Since the search direction (1.8) has the property of sufficient descent for any value $\omega_k \geq 1/4$, it follows that for any value of $a_k \leq \tau$, where $1 < \tau \leq 4$ is a parameter, the singular values of the matrix H_{k+1} are clustered around 1. Therefore, the search direction of our algorithm is given by (1.3) where the parameter β^N is computed as:

$$\beta_k^N = \begin{cases} \dfrac{y_k^T g_{k+1}}{y_k^T s_k} - \dfrac{s_k^T g_{k+1}}{\|s_k\|^2}, & \text{if} \quad a_k \leq \tau, \\ \dfrac{y_k^T g_{k+1}}{y_k^T s_k}, & \text{if} \quad a_k > \tau. \end{cases} \tag{2.19}$$

Our algorithm (1.3) with (2.19) can be considered as an adaptive conjugate gradient algorithm subject to the parameter $1 < \tau \leq 4$. If $a_k > \tau$, then the search direction is triggered to the HS direction, otherwise the search direction is that specified in (1.8) with $\omega_k = 1/a_k$, where a_k is given by (2.8). We see that according to the value of the parameter τ the behavior of our algorithm is closer to that of the HS algorithm, or to the algorithm given by (1.8) where $\omega_k = 1/a_k$.

Observe that our algorithm is a modification of the HS conjugate gradient algorithm based on the idea of minimizing the condition number of the matrix defined by the search direction (1.3) and (1.4). The CG-DESCENT algorithm proposed by Hager and Zhang [22] also is a modification of the HS conjugate gradient algorithm by *ex abrupto* deleting a term from the search direction for the memoryless quasi-Newton scheme of Shanno [30]. Again, using this approach we get a value for the parameter t in the Dai and Liao conjugate gradient parameter (1.7) for which the condition number of the search matrix is minimized.

Taking into consideration the above developments and using the procedure of acceleration of conjugate gradient algorithms presented in [2], the following algorithm can be presented.

NCG Algorithm (New Conjugate Gradient Algorithm)

Step 1. Select a starting point $x_0 \in \mathbb{R}^n$ and compute: $f(x_0)$, $g_0 = \nabla f(x_0)$. Select some positive values for δ and σ used in Wolfe line search conditions. Consider a positive value for the parameter τ. $(1 < \tau \leq 4)$ 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 steplength α_k by using the Wolfe line search (1.5) and (1.6).

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 a_k as in (2.8).

Step 8. Compute the search direction as in (1.3) where β_k^N is computed as in (2.19).

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 [17] or [29]). 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 [13], but we are interested in the performance of a conjugate gradient algorithm that uses this restart criterion associated to a direction which satisfies 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 step length 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 the linear convergence of the acceleration scheme used in the algorithm NCG is proved in [2]. Clearly, the acceleration scheme improves the performances of the algorithm [2]. Numerical comparisons may drastically change by introducing acceleration. However, we are interested to see the performances of this algorithm equipped with an acceleration scheme.

3. Global convergence analysis

The global convergence analysis of the above algorithms is based on bounding the norm of the search direction, (see Gilbert and Nocedal, [19], Nocedal, [27] or Dai, et al [16]). In this section we prove the global convergence of the above algorithms under the following assumptions:
Assume that:

(i) *The level set* $S = \left\{x \in R^n : f(x) \leq f(x_0)\right\}$ *is bounded.*

(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* $\left\|\nabla f(x) - \nabla f(y)\right\| \leq L\left\|x - y\right\|$, *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 algorithm NCG is contained in S. Under these assumptions on f there exists a constant $\Gamma \geq 0$ such that $\|\nabla f(x)\| \leq \Gamma$ for all $x \in S$. Notice that the assumption that the function f is bounded below is weaker that the usual assumption that the level set is bounded.

Although the search directions generated by the algorithm are always descent directions, to ensure convergence of the algorithm we need to constrain the choice of the step-length α_k. The following proposition shows that the Wolfe line search always gives a lower bound for the stepsize α_k.

Proposition 3.1. *Suppose that d_k is a descent direction and the gradient ∇f satisfies the Lipschitz condition*

$$\|\nabla f(x) - \nabla f(x_k)\| \leq L\|x - x_k\|$$

for all x on the line segment connecting x_k and x_{k+1}, where L is a positive constant. If the line search satisfies the Wolfe conditions (1.5) and (1.6), then

$$\alpha_k \geq \frac{(1-\sigma)\left|g_k^T d_k\right|}{L\|d_k\|^2}.$$

Proof. Subtracting $g_k^T d_k$ from both sides of (1.6) and using the Lipschitz continuity we get

$$(\sigma - 1)g_k^T d_k \leq (g_{k+1} - g_k)^T d_k = y_k^T d_k \leq \|y_k\|\|d_k\| \leq \alpha_k L\|d_k\|^2.$$

Since d_k is a descent direction and $\sigma < 1$, we get the conclusion of the proposition ■

For any conjugate gradient method with strong Wolfe line search the following general result holds [27].

Proposition 3.2. *Suppose that the above assumptions hold. Consider a conjugate gradient algorithm in which, for all $k \geq 0$, the search direction d_k is a descent direction and the stepsize α_k is determined by the Wolfe line search conditions. If*

$$\sum_{k \geq 0} \frac{1}{\|d_k\|^2} = \infty, \tag{3.1}$$

then the algorithm converges in the sense that

$$\liminf_{k \to \infty} \|g_k\| = 0. \tag{3.2}$$

For *uniformly convex functions* we can prove that the norm of the direction d_{k+1} computed as in (1.3) with (2.19) is bounded above. Therefore, by proposition 3.2 we can prove the following result.

Theorem 3.1. *Suppose that the assumptions (i) and (ii) hold. Consider the algorithm NCG where the search direction d_k is given by (1.3) and β_k^N is computed as in (2.19). Suppose that α_k is computed by the Wolfe line search. Suppose that f is a uniformly convex function on S, i.e. there exists a constant $\mu > 0$ such that*

$$(\nabla f(x) - \nabla f(y))^T (x - y) \geq \mu\|x - y\|^2 \tag{3.3}$$

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

$$\lim_{k\to\infty}\|g_k\| = 0. \tag{3.4}$$

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, using (2.19) in (1.3) for $a_k \le \tau$, we have

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

showing that (3.1) is true.
Again, using (2.19) in (1.3) for $a_k > \tau$ it follows that

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

showing that (3.1) is true. By proposition 3.2 it follows that (3.2) is true, which for uniformly convex functions is equivalent to (3.4) ■

The convergence analysis for general nonlinear functions follows the developments given by Hager and Zhang [22]. If the level set S is bounded, the Lipschitz condition $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|$ holds and the step length satisfies the Wolfe conditions (1.5) and (1.6), then for the algorithm (1.2), (1.3) and (2.19) either $g_k = 0$ for some k or $\liminf_{k\to\infty}\|g_k\| = 0$ (see theorem 3.2 in [22]).

4. Numerical results and comparisons

The NCG algorithm was 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 functions in generalized or extended form presented in [1]. For each test function we have considered 10 numerical experiments with the number of variables increasing as $n = 1000, 2000, \ldots, 10000$. The algorithms compared in this section use the Wolfe line search conditions with cubic interpolation [31], $\delta = 0.0001$, $\sigma = 0.8$ and the same stopping criterion $\|g_k\|_\infty \le 10^{-6}$, where $\|.\|_\infty$ is the maximum absolute component of a vector.

Since, CG-DESCENT [23] is among the best nonlinear conjugate gradient algorithms proposed in the literature, but not necessarily the best, in the following we compare our algorithm NCG versus CG-DESCENT. When the algorithms are compared we can consider at least two points of view: the first is based on the optimal point generated by the algorithm, and the second one is using the objective function value in this point. Since all the algorithms used and compared in this paper generate local solutions, we compare them by using the point of view based on the objective function value in the point determined by the algorithms. Therefore, the comparisons of algorithms are given in the following context. Let f_i^{ALG1} and f_i^{ALG2} be the optimal value 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:

$$\left| f_i^{ALG1} - f_i^{ALG2} \right| < 10^{-3} \tag{4.1}$$

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. Possibly, some other points of view for comparing the algorithms can be used, but in this paper we consider this one. Of

course, the test problems where the algorithms do not converge to the same function value, according to criterion (4.1), are discarded from comparisons.

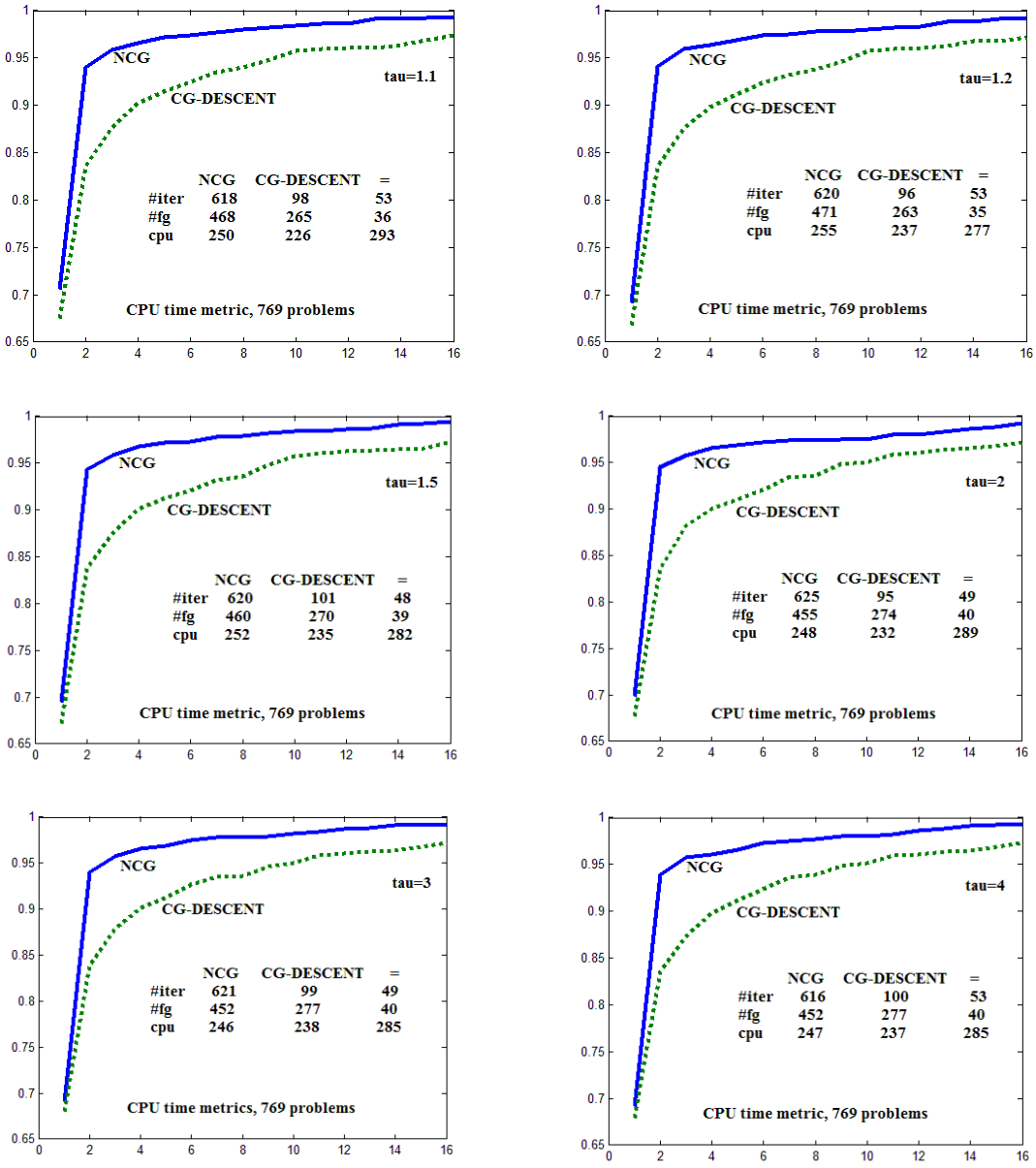

Fig. 1. NCG versus CG-DESCENT for different values of τ.

Figure 1 shows the performance profiles of Dolan-Moré [18] subject to CPU time metric for different values of 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.

Form figure 1, for example for $\tau = 1.1$, comparing NCG versus CG-DESCENT with Wolfe line search, subject to the number of iterations, we see that NCG was better in 618 problems (i.e. it achieved the minimum number of iterations for solving 618 problems), CG-DESCENT was better in 98 problems and they achieved the same number of iterations in 53 problems, etc. Out of 800 problems, we considered in this numerical study, only for 769 problems does the criterion (4.1) hold. From figure 1 we see that for different values of the parameter τ NCG algorithm is more efficient and more robust than CG-DESCENT. In comparison with CG-DESCENT, on average, NCG appears to generate better search direction. We see that this computational scheme based on clustering the singular values of the matrix representing the search direction (1.3) and (2.19) lead us to a conjugate gradient algorithm which substantially outperforms the CG-DESCENT, being way more efficient and more robust.

In the second set of numerical experiments we compare NCG versus ADCG algorithm [5]. The ADCG is an adaptive conjugate gradient algorithm where the search direction is computed as the sum of the negative gradient and a vector determined by minimizing the quadratic approximation of the function f at the current point. Using a special approximation to the inverse Hessian of the objective function, which depend by a positive parameter, a search direction is obtained which satisfies both the sufficient descent condition and the Dai-Liao's conjugacy condition. The parameter in the search direction is determined in an adaptive manner by minimizing the largest eigenvalue of the matrix defining it in order to cluster all the eigenvalues. The search direction in ADCG algorithm is computed as

$$d_{k+1}^{ADCG} = -Q_{k+1} g_{k+1}, \tag{4.2}$$

where

$$Q_{k+1} = I - \frac{s_k y_k^T - y_k s_k^T}{y_k^T s_k} + \omega_k \frac{\|y_k\|^2}{y_k^T s_k} \frac{s_k s_k^T}{y_k^T s_k}, \tag{4.3}$$

and the parameter ω_k is determined in such a way to cluster all its eigenvalues. In [5] the parameter ω_k is computed as:

$$\omega_k = t_k \frac{y_k^T s_k}{\|y_k\|^2}, \tag{4.4}$$

where

$$t_k = \begin{cases} 2\sqrt{\eta - 1}\,\|y_k\| / \|s_k\|, & \text{if } a_k \geq \eta, \\ 0 & \text{otherwise,} \end{cases} \tag{4.5}$$

a_k is defined by (2.8), and $\eta > 1$ is a positive constant. Therefore, the ADCG algorithm is based on clustering the eigenvalues of the search direction matrix (4.3). On the other hand, the NCD algorithm is using the clustering of the singular values of search direction matrix (1.10). Observe the differences between H_{k+1} given by (1.10) used in NCG algorithm and Q_{k+1} given by (4.3) used in ADCG algorithm. We see that $Q_{k+1} = H_{k+1} + y_k s_k^T / y_k^T s_k$. Both these matrices H_{k+1} and Q_{k+1} are not symmetric matrices, as usual in quasi-Newton methods. They are used in these algorithms in order to find the values of parameter ω_k to cluster the singular values of H_{k+1} or the eigenvalues of Q_{k+1}, respectively. In [5] we have the computational evidence that ADCG is not sensitive to the values of the parameter η, and is way more efficient and more robust than CG-DESCENT. In Figure 2 we present the performance profiles of Dolan-Moré subject to CPU time metric, of NCG versus ADCG, for different values of the parameters τ and η.

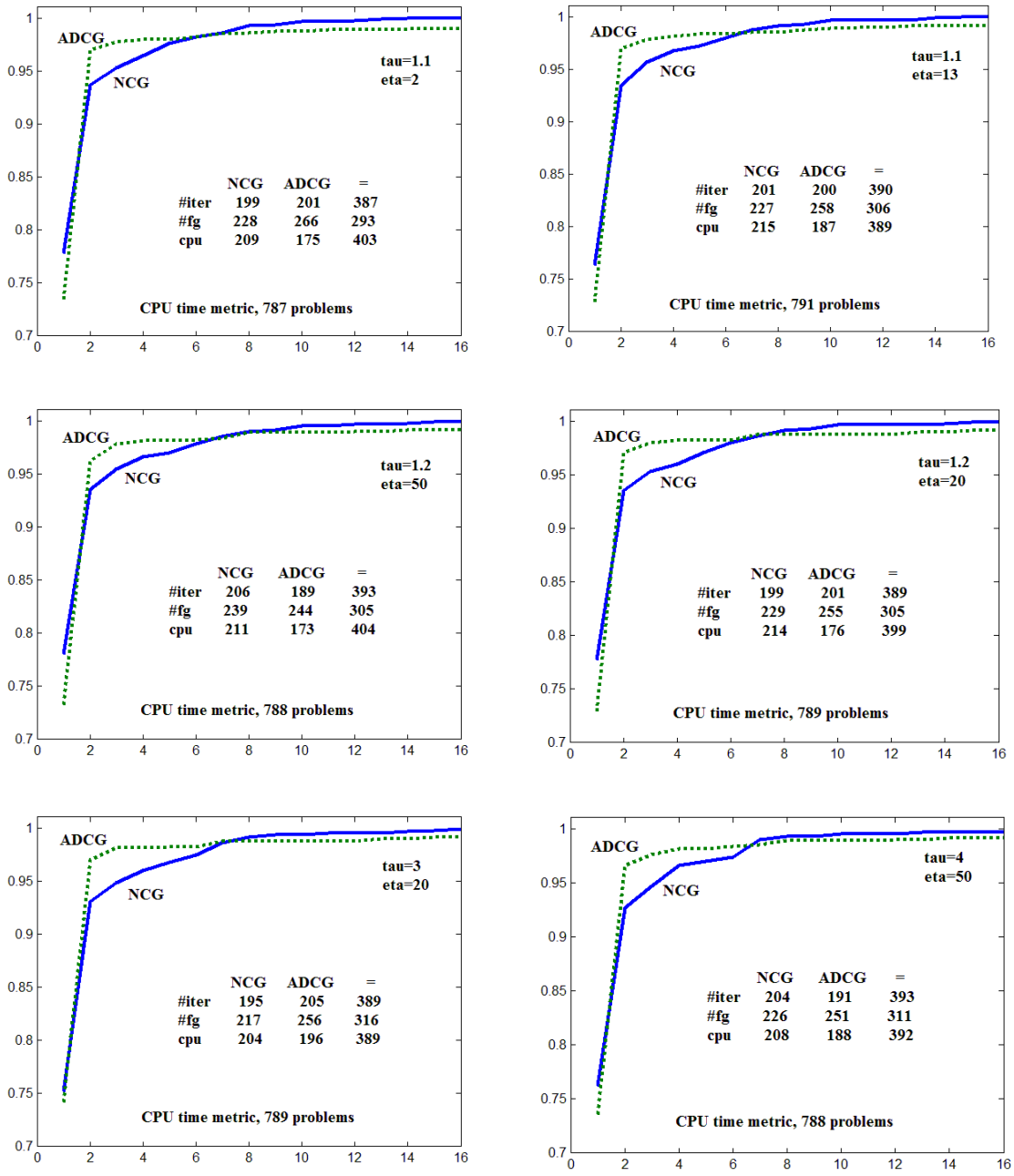

Fig. 2. NCG versus ADCG for different values of τ and η.

The NCG algorithm is based on minimizing the condition number of the matrix defining the search direction, i.e., on clustering the *singular values* around 1. On the other hand, the ADCG algorithm is based on clustering the *eigenvalues* of the same matrix. In Figure 2 we have the computational evidence that NCG algorithm is slightly more efficient and more robust than ADCG for any combination of parameters τ and η. From Figure 2 we see that both algorithms are not sensitive to the values of these parameters. Practically, all performance profiles have the same allure for any combination of τ and η. Singular values analysis in designing conjugate gradient algorithms is more profitable subject to efficiency and robustness, but this is not

overwhelming, both concepts (singular values and eigenvalues) leading to very similar results. (see also [4]).

In the following, in the third set of numerical experiments, we present comparisons between NCG and CG-DESCENT conjugate gradient algorithms for solving some real applications from the MINPACK-2 test problem collection [7]. 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 ([20], pp. 41–55), $c = 5$
A2	Pressure distribution in a journal bearing [11], $b = 10,\ \varepsilon = 0.1$
A3	Optimal design with composite materials [21], $\lambda = 0.008$
A4	Steady-state combustion ([6], pp. 292–299), [10], $\lambda = 5$
A5	Minimal surfaces with Enneper conditions ([26], 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 NCG (Powell restart criterion, $\|\nabla f(x_k)\|_\infty \le 10^{-6},\ \ \delta = 0.0001, \sigma = 0.8,\ \ \tau = 4$) and CG-DESCENT (version 1.4, Wolfe line search, default settings, $\|\nabla f(x_k)\|_\infty \le 10^{-6},$) for solving these applications is given in Table 2.

Table 2
Performance of NCG versus CG-DESCENT. 1,000,000 variables. $\tau = 4,$ CPU seconds.

	NCG			**CG-DESCENT**		
	#iter	#fg	cpu	#iter	#fg	cpu
A1	1113	2257	**351.62**	1145	2291	474.64
A2	2843	5714	**1143.97**	3370	6741	1835.51
A3	4725	9494	**2754.26**	4814	9630	3949.71
A4	1413	2864	**2014.17**	1802	3605	3786.25
A5	1270	2566	**571.45**	1225	2451	753.75
TOTAL	11364	22895	**6835.47**	12356	24718	10799.86

Form Table 2, we see that, subject to the CPU time metric, the NCG algorithm is top performer and the difference is significant, about **3964.39 seconds** for solving all these five applications. It is worth saying that intensive numerical experiments for solving the applications from MINPACK-2 collection with different values of the parameter $1 < \tau \le 4$ mainly yield similar results concerning the numerical performances of NCG algorithm. In all cases, for all these numerical experiments, NCG was top performer versus CG-DESCENT.

The NCG 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 (1.3) where the parameter β_k^N is computed as in (2.19) is more elaborate: it is adaptive and the singular values of the matrix defined by it are clustered around 1. In addition it satisfies both the descent condition and the conjugacy condition in a restart environment.

5. Conclusions

A new adaptive conjugate gradient algorithm based on singular values study of the search direction matrix has been presented. The idea of this paper is to generalize the search direction of CG-DESCENT conjugate gradient algorithm of Hager and Zhang [22] by introducing a positive parameter ω_k instead of constant 2 used in conjugate gradient parameter β_k^{HZ}. At the same time, the paper contains a development for a value of the positive parameter t used in conjugate gradient parameter β_k^{DL} from the Dai-Liao's conjugate gradient algorithm [12]. The value of this parameter is computed in such a way that the condition number of the matrix defining the search direction is minimized. Mainly, in our algorithm, minimizing the condition number of the iteration matrix defining the search direction reduces to determine the value of the parameter to minimize the distance between the singular values of the corresponding matrix, i.e., to cluster the singular values around 1. It is proved that the search direction satisfies both the sufficient descent condition and the Dai-Liao's conjugacy condition. Thus, the algorithm is a conjugate gradient one. To satisfy both the clustering of the singular values and the sufficient descent condition an adaptive scheme is used which depend by a positive parameter. The algorithm is not sensitive to the value of this parameter. 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. For uniformly convex functions, under classical assumptions, the algorithm is globally convergent. Numerical experiments and intensive comparisons using 800 unconstrained optimization problems of different dimensions and complexity proved that this conjugate gradient algorithm is way more efficient and more robust than CG-DESCENT algorithm [23], and slightly more efficient and more robust than ADCG algorithm [5]. In an effort to see the performances of this conjugate gradient algorithm we solved five large-scale real nonlinear optimization applications from MINPACK-2 collection, up to 10^6 variables, showing that NCG is clearly more efficient and more robust than CG-DESCENT.

References

[1] N. Andrei, An unconstrained optimization test functions collection. Advanced Modeling and Optimization, 10 (2008) 147-161.

[2] N. Andrei, Acceleration of conjugate gradient algorithms for unconstrained optimization. Applied Mathematics and Computation, 213 (2009) 361-369.

[3] N. Andrei, Open problems in conjugate gradient algorithms for unconstrained optimization. Bulletin of the Malaysian Mathematical Sciences Society, 34 (2011) 319-330.

[4] N. Andrei, Eigenvalues versus singular values study in conjugate gradient algorithms for large-scale unconstrained optimization. ICI Technical Report, December 3, 2015.

[5] N. Andrei, An adaptive conjugate gradient algorithm for large-scale unconstrained optimization. Journal of Computational and Applied Mathematics 292 (2016) 83-91.

[6] R. Aris, The Mathematical Theory of Diffusion and Reaction in Permeable Catalysts, Oxford, 1975.

[7] B.M. Averick, R.G., Carter, J.J., Moré, G.L., Xue, The MINPACK-2 test problem collection, Mathematics and Computer Science Division, Argonne National Laboratory, Preprint MCS-P153-0692, June 1992.

[8] S. Babaie-Kafaki, R. Ghanbari, The Dai-Liao nonlinear conjugate gradient method with optimal parameter choices. European Journal of Operational Research, 234 (2014) 625-630.

[9] S. Babaie-Kafaki, R. Ghanbari, Two optimal Dai-Liao conjugate gradient methods. Optimization, 64 (2014) 2277-2287.

[10] J. Bebernes, D. Eberly, Mathematical Problems from Combustion Theory. In: Applied Mathematical Sciences, 83, Springer-Verlag, 1989.

[11] G. Cimatti, On a problem of the theory of lubrication governed by a variational inequality, Applied Mathematics and Optimization 3 (1977) 227–242.
[12] Y.H. Dai, L.Z. Liao, New conjugate conditions and related nonlinear conjugate gradient methods. Appl. Math. Optim., 43 (2001) 87-101.
[13] Y.H. Dai, L.Z. Liao, Duan, Li, On restart procedures for the conjugate gradient method. Numerical Algorithms 35 (2004) 249-260.
[14] Y.H. Dai, Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property. SIAM J. Optim., 10 (1999) 177-182.
[15] Y.H. Dai, Y. Yuan, An efficient hybrid conjugate gradient method for unconstrained optimization. Annals of Operations Research, 103 (2001) 33-47.
[16] Y.H. Dai, J. Han, G. Liu, D. Sun, H. Yin, Y. Yuan, Convergence properties of nonlinear conjugate gradient methods. SIAM Journal on Optimization, 10 (1999) 345-358.
[17] J.E. Dennis, R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice-Hall, Englewood Cliffs, New Jeresy, 1983.
[18] E. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles. Math. Programming, 91 (2002) 201-213.
[19] J.C. Gilbert, J. Nocedal, Global convergence properties of conjugate gradient methods for optimization. SIAM Journal on Optimization, 2 (1992), 21-42.
[20] R. Glowinski, Numerical Methods for Nonlinear Variational Problems, Springer-Verlag, Berlin, 1984.
[21] J. Goodman, R. Kohn, L. Reyna, Numerical study of a relaxed variational problem from optimal design, Computer Methods in Applied Mechanics and Engineering 57 (1986) 107–127.
[22] W.W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM Journal on Optimization, 16 (2005) 170-192.
[23] W.W. Hager, H. Zhang, Algorithm 851: CG-DESCENT, a conjugate gradient method with guaranteed descent. ACM Trans. Math. Softw. 32 (2006) 113-137.
[24] M.R. Hestenes, E. Stiefel, Metods of conjugate gradients for solving linear systems. J. Research Nat. Bur. Standards Sec. B. 48 (1952) 409-436.
[25] D. Liu, G. Xu, Symmetric Perry conjugate gradient method. Computational Optimization and Applications, 56 (2013) 317-341.
[26] J.C.C. Nitsche, Lectures on Minimal Surfaces, Vol. 1, Cambridge University Press, 1989.
[27] J. Nocedal, Conjugate gradient methods and nonlinear optimization. In: Adams, L., Nazareth, J.L., (Eds.) Linear and Nonlinear Conjugate Gradient Related Methods, SIAM, (1996) 9-23.
[28] J.M. Perry, A class of conjugate gradient algorithms with a two step variable metric memory. Discussion paper 269, Center for Mathematical Studies in Economics and Management Science. Northwestern University, 1977.
[29] B.T. Polyak, Introduction to Optimization. Optimization Software, Inc., Publications Division, New York, 1987.
[30] D.F. Shanno, On the convergence of a new conjugate gradient algorithm. SIAM J. Numer. Anal., 15 (1978) 1247-1257.
[31] W. Sun, Y. Yuan, Optimization Theory and Methods. Nonlinear Programming. Springer Science + Business Media, New York, 2006.
[32] D.S. Watkins, Fundamentals of Matrix Computations. (2nd ed.). New York: John Wiley and Sons, Inc., 2002.
[33] P. Wolfe, Convergence conditions for ascent methods. SIAM Review, 11 (1969) 226-235.
[34] P. Wolfe, Convergence conditions for ascent methods. II: Some corrections. SIAM Review, 13 (1971) 185-188.