On spectral properties of steepest descent methods †

ROBERTA DE ASMUNDIS ‡
Department of Statistical Sciences, University of Rome La Sapienza, Piazzale A. Moro 5, 00185 Rome, Italy

DANIELA DI SERAFINO §
Department of Mathematics, Second University of Naples, Viale A. Lincoln 5, 81100 Caserta, Italy, and Institute for High-Performance Computing and Networking, CNR, Via P. Castellino 111, 80131 Naples, Italy

FILIPPO RICCIO ¶
Department of Mathematics, Second University of Naples, Viale A. Lincoln 5, 81100 Caserta, Italy

GERARDO TORALDO ‖
Department of Agricultural Engineering and Agronomy, University of Naples Federico II, Via Università 100, 80055 Portici (Naples), Italy

[10 February 2012]

In recent years it has been made more and more clear that the critical issue in gradient methods is the choice of the step length, whereas using the gradient as search direction may lead to very effective algorithms, whose surprising behaviour has been only partially explained, mostly in terms of the spectrum of the Hessian matrix. On the other hand, the convergence of the classical Cauchy steepest descent (CSD) method has been extensively analysed and related to the spectral properties of the Hessian matrix, but the connection with the spectrum of the Hessian has been little exploited to modify the method in order to improve its behaviour. In this work we show how, for convex quadratic problems, moving from some theoretical properties of the CSD method, second-order information provided by the step length can be exploited to dramatically improve the usually poor practical behaviour of this method. This allows to achieve computational results comparable with those of the Barzilai and Borwein algorithm, with the further advantage of a monotonic behaviour.

Keywords: steepest descent methods, quadratic optimization, Hessian spectral properties.

1. Introduction

The general steepest descent (SD) algorithm for the unconstrained minimization problem

$$\begin{array}{l} \min f(x) \\ x \in \Re^n \end{array} \tag{1.1}$$

†This research was partially supported by the Italian Ministry of University and Research under PRIN 2008 Project *Optimization methods and software for inverse problems*, grant no. 2008T5KA4L, http://www.unife.it/prisma.

‡Email: roberta.deasmundis@uniroma1.it
§Email: daniela.diserafino@unina2.it
¶Email: filippo.riccio@unina2.it
‖Corresponding author. Email: toraldo@unina.it

generates a sequence $\{x_k\}$ by the following rule:

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

where $g_k = \nabla f(x_k)$ and the step length $\alpha_k > 0$ depends on the method used. In particular, in the classical (optimal) steepest descent method proposed by Cauchy (1847) for the solution of nonlinear systems of equations (henceforth named CSD), α_k is chosen as

$$\alpha_k^C = \operatorname*{argmin}_{\alpha} f(x_k - \alpha g_k). \tag{1.3}$$

Since the theoretical properties of the gradient methods derive from the minimization of a convex quadratic function, in this paper we focus our attention on the model problem

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} x^T A x - b^T x, \tag{1.4}$$

where $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite and $b \in \mathbb{R}^n$. This is a simple setting suitable to analyse the relevance of the eigenvalues of the Hessian of the objective function to the behaviour of the algorithms we consider; furthermore, it allows to highlight the ability of the classical CSD method to automatically reveal some second order information about the problem, which can be conveniently exploited to dramatically improve the usually poor behaviour of the algorithm. For Problem (1.4) the Cauchy step length α_k^C can be computed exactly as the reciprocal of the Rayleigh quotient of A at g_k, i.e.,

$$\alpha_k^C = \frac{g_k^T g_k}{g_k^T A g_k}. \tag{1.5}$$

The CSD method, despite of the minimal storage requirements and the very low computational cost per iteration, which is $O(n)$ floating-point operations besides a gradient evaluation, has long been considered very bad and ineffective because of its slow convergence rate and its oscillatory behaviour. However, in the last 20 years the interest for the steepest descent method has been renewed after the innovative approach of the Barzilai and Borwein (BB) method (Barzilai & Borwein (1988)), which stimulated novel choices for α_k in (1.2), proved to be largely superior to the Cauchy step length. In the BB approach the step length is computed through a secant condition by imposing either

$$\min_{\alpha} \|s_{k-1} - \alpha y_{k-1}\|, \tag{1.6}$$

or

$$\min_{\alpha} \left\| \frac{1}{\alpha} s_{k-1} - y_{k-1} \right\|, \tag{1.7}$$

where $\|\cdot\|$ is the L_2 vector norm, $s_{k-1} = x_k - x_{k-1}$, $y_{k-1} = g_k - g_{k-1}$, thus obtaining the following step lengths, respectively:

$$\alpha_k^{B1} = \frac{\|s_{k-1}\|^2}{s_{k-1}^T y_{k-1}^T}, \tag{1.8}$$

$$\alpha_k^{B2} = \frac{s_{k-1}^T y_{k-1}}{\|y_{k-1}\|^2}, \tag{1.9}$$

for which the inequality $\alpha_k^{B1} \geqslant \alpha_k^{B2}$ holds (see Lemma 2.1 in Raydan & Svaiter (2002)).

The BB step length (1.8) is equal to α^C_{k-1}, i.e., the Cauchy step length at the previous iteration, while (1.9) is α^g_{k-1}, where

$$\alpha^g_k = \frac{g_k A g_k}{g_k A^2 g_k} = \operatorname*{argmin}_{\alpha} \|\nabla f(x_k - \alpha g_k)\| = \operatorname*{argmin}_{\alpha} \|(I - \alpha A) g_k\|. \tag{1.10}$$

We note that (1.10) can be interpreted as the Cauchy step for the convex quadratic problem

$$\min_x \|Ax - b\| = \min_x \frac{1}{2} x^T A^2 x - A b^T x, \tag{1.11}$$

which is obviously equivalent to (1.4). Therefore, both the BB step lengths (1.8)-(1.9) can be seen as Cauchy step lengths with one delay. The use of larger delays has been investigated in Friedlander *et al.* (1998), extending the r-linear convergence results which hold for BB (Dai & Liao (2002)). A deeper analysis of the asymptotic behaviour of BB and related methods is proposed in Dai & Fletcher (2005). Fletcher (2005) makes some intuitive considerations about the relationship between the non-monotonicity of such methods and their surprising computational performance; he also discusses about the circumstances under which BB (and related) methods might be competitive with Conjugate Gradient (CG) methods, and he argues that the former represent an effective alternative to the latter when moving from (1.4) to constrained or non-quadratic problems (Birgin *et al.* (2000); Dai & Fletcher (2005, 2006); Hager & Zhang (2006); Andretta *et al.* (2010)). Moreover, as observed in Friedlander *et al.* (1998), gradient methods are very competitive with CG when low accuracy in the solution is required, for instance in the context of inexact Newton methods. All of these interesting observations, illustrated through several computational experiences in Friedlander *et al.* (1998); Raydan & Svaiter (2002); Fletcher (2005), justify the interest in designing effective gradient methods and the need of better understanding their behaviour. In recent years it has been made more and more clear that the critical issue in gradient methods is the choice of the step length, whereas using the gradient as search direction may lead to very effective algorithms. The surprising behaviour of these algorithms has been only partially explained (Raydan (1997); Fletcher (2005)), pointing out that the effectiveness of the approach is related to the eigenvalues of the Hessian rather than to the decrease of the function value.

For the CSD method the convergence has been extensively analysed and related to the spectral properties of the Hessian matrix A, for instance in the pioneeristic works of Akaike (1959) and Forsythe (1968). However, the connection with the spectrum of A has been little exploited to modify the CSD method in order to improve its behaviour. The recurrence

$$g_{k+1} = g_k - \alpha_k A g_k = \alpha_k \left(\frac{1}{\alpha_k} g_k - A g_k \right), \tag{1.12}$$

which holds for any gradient method, suggests that, in order to make them converge faster, a greedy approach like (1.5) might result unsatisfactory, whereas, for instance, fostering the search direction to align with an eigendirection of A could speed up the convergence of the algorithm (Frassoldati *et al.* (2008)).

We will show how, moving from some theoretical properties of the CSD method, second order information provided by the step length (1.5) can be exploited in order to improve dramatically the usually poor practical behaviour of the Cauchy method, achieving computational results comparable with those of the BB algorithm, while preserving monotonicity.

This paper is organized as follows. In Section 2 some classical convergence results about the CSD method are briefly reviewed, which are the theoretical basis of the analysis carried out in the sequel of

the paper. In Section 3 we highlight that the sequence of Cauchy step lengths has the nice feature of providing an approximation to the sum of the extreme eigenvalues of the Hessian. Based on that, we propose a modification of the CSD method, called CSD1, aimed to align the search direction with the eigendirection corresponding to the smallest eigenvalue, and then to eventually force the algorithm in the one-dimensional subspace spanned by that eigenvector. In Section 4 we show that a gradient method where the step length is twice the Cauchy step length (1.5) eventually ends up in a one-dimensional subspace spanned by the eigenvector associated with the largest eigenvalue of A. This result gives a further motivation for the relaxed Cauchy steepest descent (RCSD) method by Raydan & Svaiter (2002), and actually suggests that it is worth fostering an over-relaxation. Finally, in Section 5 we provide some numerical evidence about the performance of the algorithmic approaches presented in Sections 3 and 4, compared to the standard BB algorithm.

In the rest of this paper we denote by $\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ the eigenvalues of the matrix A and by $\{d_1, d_2, \dots, d_n\}$ a set of associated orthonormal eigenvectors. We make the following assumptions:

ASSUMPTION 1 The eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_n$ are such that

$$\lambda_1 > \lambda_2 > \lambda_3, \dots, > \lambda_n > 0.$$

ASSUMPTION 2 For all the methods considered in this work, any starting point x_0 is such that

$$g_0^T d_1 \neq 0, \quad g_0^T d_n \neq 0.$$

Finally, we denote by x^* the solution of Problem (1.4) and by $\kappa(A)$ the spectral condition number of A.

2. The Steepest Descent Algorithm

The most general steepest descent (SD) method iterates according to the following algorithmic framework:

ALGORITHM 1 (SD)
choose $x_0 \in \Re^n$;
$g_0 \leftarrow Ax_0 - b$; $k \leftarrow 0$
while (not stop_condition)
  choose a suitable step length $\alpha_k > 0$
  $x_{k+1} \leftarrow x_k - \alpha_k g_k$; $g_{k+1} \leftarrow g_k - \alpha_k A g_k$
  $k \leftarrow k+1$
endwhile

For the optimal choice (1.5) of the step length it is well known that the algorithm has q-linear rate of convergence which depends on the spectral radius of the Hessian matrix; more precisely, the following result holds.

PROPOSITION 2.1 [Akaike (1959)] The sequence $\{x_k\}$ generated by the CSD algorithm converges q-linearly to x_* with rate of convergence

$$\rho = \frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n}. \tag{2.1}$$

The convergence of the SD algorithm holds for a choice of the step length much more general than (1.5). If we consider $2\alpha_k^C$ as step length, then

$$f(x_k - 2\alpha_k^C g_k) = f(x_k),$$

and the following decrease condition holds:

$$f(x_k - \alpha g_k) < f(x_k), \text{ for all } \alpha \in (0, 2\alpha_k^C). \tag{2.2}$$

The next proposition states the condition under which Algorithm 1, with a step condition inspired by (2.2), converges to the solution of (1.4).

PROPOSITION 2.2 [Raydan & Svaiter (2002)] The sequence $\{x_k\}$ generated by Algorithm 1, with $\alpha_k = \rho_k \alpha_k^C$, $\rho_k \in [0,2]$, converges to x^* provided $\{\rho_k\}$ has an accumulation point in $(0,2)$.

We now present some known formulas which hold for the gradients of the sequence generated by Algorithm 1, for any choice of α_k. First, we observe that

$$g_{k+1} = g_k - \alpha_k A g_k = \prod_{j=1}^{k} (I - \alpha_j A) g_0. \tag{2.3}$$

Furthermore, if

$$g_0 = \sum_{i=1}^{n} \mu_i d_i,$$

then, by (2.3), we have:

$$g_{k+1} = \sum_{i=1}^{n} \mu_i^k d_i, \tag{2.4}$$

where

$$\mu_i^k = \mu_i \prod_{j=1}^{k} (1 - \alpha_j \lambda_i). \tag{2.5}$$

Formulas (2.4)-(2.5) have a very high relevance in the analysis of SD methods, since they allow to study the convergence in terms of the spectrum of the matrix A. If at the k-th iteration $\mu_i^k = 0$ for some i, it follows from (2.4) that for $h > k$ it will be $\mu_i^h = 0$, and therefore the component of the gradient along d_i will be zero at all subsequent iterations. We notice that the condition $\mu_i^k = 0$ holds if at the k-th iteration $\alpha_k = 1/\lambda_i$. Furthermore, from (2.3) il follows that the CSD method has finite termination if and only if at some iteration the gradient is an eigenvector of A.

The next proposition gives the asymptotic rate of convergence of $\{\mu_1^k\}$ for a quite general choice of the step length in the SD method.

PROPOSITION 2.3 [Friedlander *et al.* (1998)] In Algorithm 1, if the step length α_k is chosen as the reciprocal of the Rayleigh quotient of A at any nonzero vector, then the sequence $\{\mu_1^k\}$ converges q-linearly to zero with convergence factor $1 - \lambda_n/\lambda_1$.

Friedlander *et al.* (1998) present a large collection of possible choices of α_k (including some well known methods) for which $\{\mu_i^k\}$ vanishes for all i. The next result extends and summarizes previous results of Akaike (1959).

PROPOSITION 2.4 [Nocedal *et al.* (2002)] Let us consider the sequence $\{x_k\}$ generated by the CSD method, and suppose that Assumptions 1-2 hold. Then

$$\lim_k \frac{(\mu_1^k)^2}{\sum_{j=1}^n (\mu_j^k)^2} = \begin{cases} \dfrac{c^2}{1+c^2} & \text{for } k \text{ odd}, \\ \dfrac{1}{1+c^2} & \text{for } k \text{ even}, \end{cases} \tag{2.6}$$

$$\lim_k \frac{(\mu_n^k)^2}{\sum_{j=1}^n (\mu_j^k)^2} = \begin{cases} \dfrac{1}{1+c^2} & \text{for } k \text{ odd}, \\ \dfrac{c^2}{1+c^2} & \text{for } k \text{ even}, \end{cases} \tag{2.7}$$

$$\lim_k \frac{(\mu_i^k)^2}{\sum_{j=1}^n (\mu_j^k)^2} = 0 \qquad \text{for } 1 < i < n, \tag{2.8}$$

where c is a constant satisfying

$$c = \lim_k \frac{\mu_n^{2k}}{\mu_1^{2k}} = -\lim_k \frac{\mu_1^{2k+1}}{\mu_n^{2k+1}}.$$

Proposition 2.4 shows that the Cauchy method eventually performs its search in the two-dimensional subspace generated by d_1 and d_n, zigzagging between two directions, very slowly if the ratio λ_1/λ_n is large, without being able to eliminate from the basis of the current search direction any of the two components d_1 and d_n, and hence to align the gradient with an eigendirection of the Hessian matrix. Conversely, the nice behaviour of the BB methods is often explained saying that the non-monotonicity of such algorithms produces an erratic path of α_k in the interior of the spectrum of A which fosters all sequences $\{\mu_i^k\}$ to go to zero (Fletcher (2005)) in spite of the non-monotonicity of the overall approach in regard to both the sequences $\{\|g_k\|\}$ and $\{f(x_k)\}$.

3. A modified CSD method

In this section we suggest a simple way to modify the CSD method to force the gradients in a one-dimensional subspace as the iterations progress, to avoid the classical zig-zag pattern of the CSD which is the main responsible for the slow convergence of the method.

We first show that the sequence of step lengths $\{\alpha_k^C\}$ in the CSD method gives asymptotically some meaningful information about the spectrum of the Hessian matrix.

PROPOSITION 3.1 Let us consider the sequence $\{x_k\}$ generated by the CSD method applied to Problem (1.4), and suppose that Assumptions 1-2 hold. Then, the two sequences $\{\alpha_{2k}^C\}$ and $\{\alpha_{2k+1}^C\}$ are converging and

$$\lim_k \left(\frac{1}{\alpha_{2k}^C} + \frac{1}{\alpha_{2k+1}^C} \right) = \lambda_1 + \lambda_n \,. \tag{3.1}$$

Proof. By Lemma 3.3 in Nocedal *et al.* (2002), it is

$$\begin{aligned} \lim_k \alpha_{2k}^C &= \frac{1+c^2}{\lambda_n(1+c^2\gamma)}, \\ \lim_k \alpha_{2k+1}^C &= \frac{1+c^2}{\lambda_n(\gamma+c^2)}, \end{aligned}$$

where c is the same constant as in Proposition 2.4 and $\gamma = \kappa(A)$; then (3.1) trivially follows. □

PROPOSITION 3.2 Under Assumptions 1-2, the sequence $\{x_k\}$ generated by Algorithm 1, with constant step length

$$\widehat{\alpha} = \frac{1}{\lambda_1 + \lambda_n}, \tag{3.2}$$

converges to x^*. Moreover, if

$$g_0 = \mu_1 d_1 + \mu_n d_n, \tag{3.3}$$

then

$$\lim_k \frac{\mu_1^k}{\mu_n^k} = \frac{\mu_1}{\mu_n} \lim_k \left(\frac{\lambda_n}{\lambda_1}\right)^k = 0, \tag{3.4}$$

where μ_1^k and μ_n^k are defined in (2.5).

Proof. Since $\alpha_k^C \geqslant 1/\lambda_1$ for any k, then $\alpha_k^C \geqslant \widehat{\alpha}$; therefore, Proposition 2.2 applies and $\lim_k x_k = x^*$. From (2.5) we have that

$$\mu_1^k = \mu_1 \left(\frac{\lambda_n}{\lambda_n + \lambda_1}\right)^k, \quad \mu_n^k = \mu_n \left(\frac{\lambda_1}{\lambda_n + \lambda_1}\right)^k$$

and (3.4) clearly holds. □

Relation (3.4) indicates that, if the hypotheses of Proposition 3.2 hold, then the sequence $\{\mu_1^k\}$ goes to zero faster than $\{\mu_n^k\}$, the more so if $\kappa(A)$ is large. Therefore, in this case, a gradient method with step length (3.2) tends to align the search direction with the eigendirection of A corresponding to the minimum eigenvalue λ_n. Note that the constant step length (3.2) is half of the theoretically "optimal" constant step (see Elman & Golub (1994); Yuan (2008))

$$\alpha^{OPT} = \frac{2}{\lambda_1 + \lambda_n}. \tag{3.5}$$

Propositons 3.1 and 3.2 suggest a way to force the search direction to align with an eigendirection of the Hessian matrix A, to speed up the convergence of the algorithm. Proposition 2.4 shows that the condition (3.3) is going to be asymptotically satisfied, and therefore the step length (3.2) might be worth to be eventually adopted. Of course, computing the exact value of (3.2) is unrealistic, but Proposition 3.1 suggests that, for k sufficiently large,

$$\widetilde{\alpha}_k = \left(\frac{1}{\alpha_k^C} + \frac{1}{\alpha_{k+1}^C}\right)^{-1} = \frac{\alpha_k^C \alpha_{k+1}^C}{\alpha_k^C + \alpha_{k+1}^C}, \tag{3.6}$$

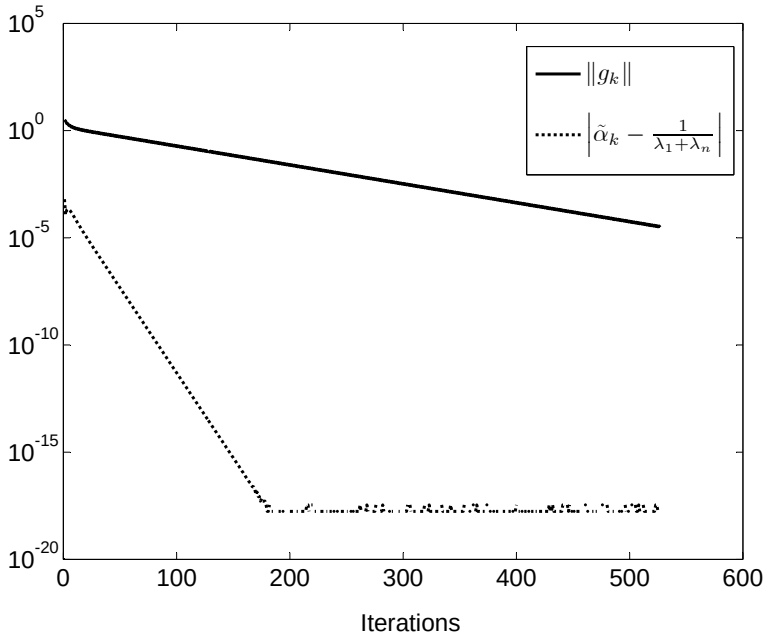


FIG. 1. Behaviour of the sequences $\left\{\left|\widetilde{\alpha}_k - \frac{1}{\lambda_1+\lambda_n}\right|\right\}$ and $\{\|g_k\|\}$ for the CSD method.

can be used as an approximate value for (3.2). We observe that the step length (3.6) is related to the step length

$$\alpha_k^Y = 2\left(\sqrt{\frac{1}{\alpha_k^C} - \frac{1}{\alpha_{k+1}^C} + 4\frac{\|g_k\|}{\|x_{k+1}-x_k\|}} + \frac{1}{\alpha_k^C} + \frac{1}{\alpha_{k+1}^C}\right)^{-1}, \tag{3.7}$$

determined by Yuan (2008) by imposing finite termination for two-dimensional quadratic problems, and that

$$\widetilde{\alpha}_k < \alpha_k^Y < \min\{\alpha_k^C, \alpha_{k+1}^C\}, \tag{3.8}$$

(see (2.22) in Yuan (2008)). By (3.1), eventually $\widetilde{\alpha}_k < \alpha^{OPT}$ and therefore, for k sufficiently large, the constant step length (3.2) ensures a reduction in the objective function and the convergence of the SD algorithm, because of Proposition 2.2. In Figure 1 we show the values of the sequence $\{|\widetilde{\alpha}_k - \alpha|\}$ computed by using the Cauchy step lengths resulting from the application of the CSD method to Problem (1.4), where $n = 10$, A is a randomly generated matrix with $\kappa(A) = 100$, $b = (1,\ldots,1)^T$ and $x_0 = (0,\ldots,0)^T$; the stop condition $\|g_k\| < 10^{-5}\|g_0\|$ has been considered. We notice that the sequence goes to zero very fast, although CSD performs very poorly and needs more than 500 iterations to find a solution with the required accuracy.

Motivated by the above results, we consider a modified version of the CSD method, named CSD1, where step lengths of the form (3.6) are chosen at some selected iterations.

ALGORITHM 2 (CSD1)
choose $x_0 \in \Re^n$, $\varepsilon > 0$, h integer
$g_0 \leftarrow Ax_0 - b$
$\alpha_0^C \leftarrow \frac{g_0^T g_0}{g_0^T A g_0}$; $x_1 \leftarrow x_0 - \alpha_0^C g_0$; $g_1 \leftarrow Ax_1 - b$
$\alpha_1^C \leftarrow \frac{g_1^T g_1}{g_1^T A g_1}$; $x_2 \leftarrow x_1 - \alpha_1^C g_1$; $g_2 \leftarrow Ax_2 - b$
$\widetilde{\alpha}_1 \leftarrow \frac{\alpha_1^C \alpha_0^C}{\alpha_1^C + \alpha_0^C}$
$k \leftarrow 1$; $s \leftarrow 1$
while (not stop_condition)
  repeat
    $p \leftarrow s$; $k \leftarrow k+1$
    $\alpha_k^C \leftarrow \frac{g_k^T g_k}{g_k^T A g_k}$; $x_{k+1} \leftarrow x_k - \alpha_k^C g_k$; $g_{k+1} \leftarrow g_k - \alpha_k^C A g_k$
    $\widetilde{\alpha}_k \leftarrow \frac{\alpha_k^C \alpha_p^C}{\alpha_k^C + \alpha_p^C}$
    $s \leftarrow k$
  until ($|\widetilde{\alpha}_k - \widetilde{\alpha}_p| < \varepsilon$) *switch condition*
  $\widetilde{\alpha} \leftarrow \widetilde{\alpha}_k$
  for $i = 1, h$
    $k \leftarrow k+1$
    $\bar{\alpha} \leftarrow \min\{\widetilde{\alpha}, 2\alpha_k^C\}$
    $x_k \leftarrow x_k - \bar{\alpha} g_k$ $g_k \leftarrow g_k - \bar{\alpha} A g_k$
  endfor
endwhile

More precisely, when the sequence $\{\widetilde{\alpha}_k\}$ settles down (see the switch condition in Algorithm 2), CSD1 performs h consecutive steps using as step length the last computed $\widetilde{\alpha}_k$, provided it produces a decrease in the objective function (otherwise, CSD1 adopts the double Cauchy step).

In Figure 2 we report the behaviour of the gradient norm in CSD1 for the above problem, with $\varepsilon = 10^{-4}$, for $h = 1$ and $h = 5$. We observe that CSD1 largely outperforms CSD; furthermore, the $\bar{\alpha}$ steps (big dots in the graph) have a rather negligible effect in terms of reduction of the gradient, but a very strong effect in reducing the overall number of iterations. This is because, as expected, such steps have an important role in aligning the search direction with the eigendirection d_n, as shown in Figure 3. We also note that a value of h larger than 1 tends to further speed up this alignment effect.

4. Relaxed Cauchy method

In this section we discuss a choice of the step length that fosters the SD algorithm to make its search in the one-dimensional space spanned by d_1. The approach was suggested by the recent work of Raydan & Svaiter (2002). They proposed a relaxation of the optimal step length (1.5) in the SD method, in order to accelerate the convergence of the classical Cauchy method. They suggest to adopt in Algorithm 1 a step length α_k chosen at random in $[0, 2\alpha_k^C]$, in order to escape from the zigzagging behaviour of the CSD method, as shown next.

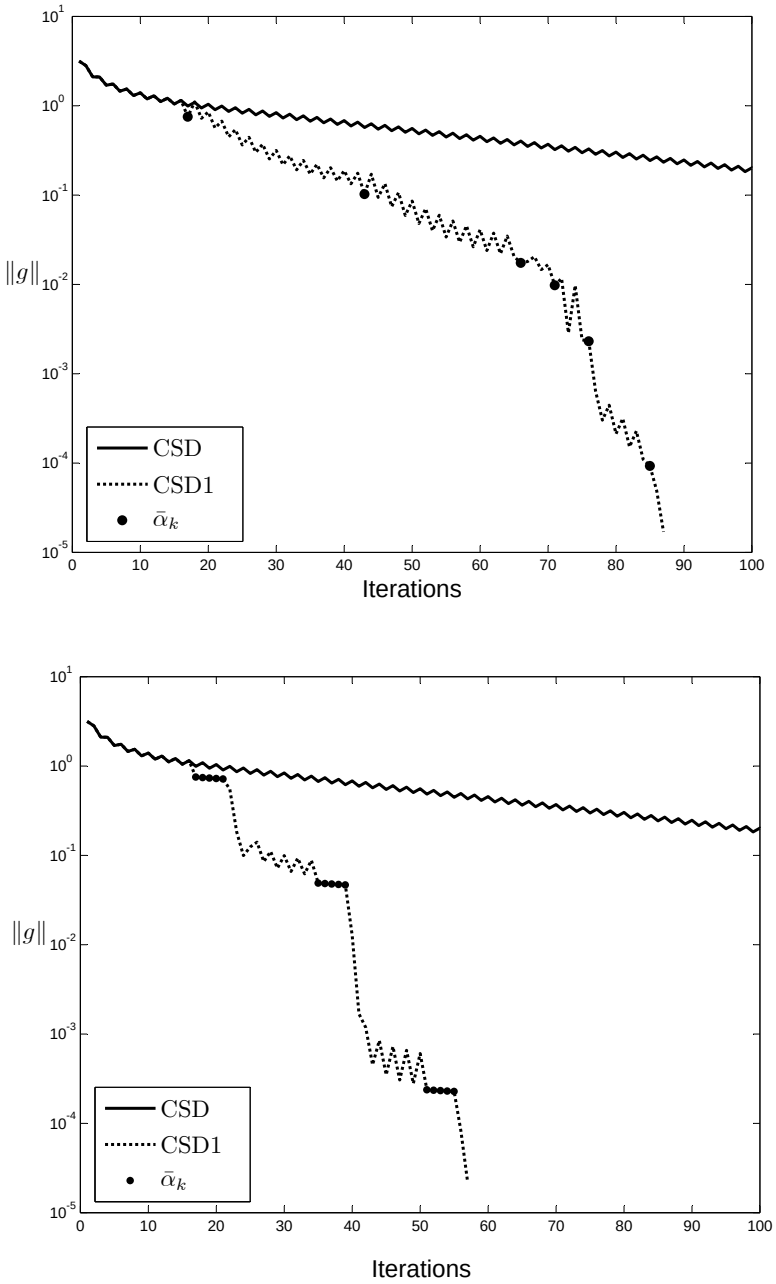


FIG. 2. Convergence of the CSD1 method, for $h = 1$ (top) and $h = 5$ (bottom).

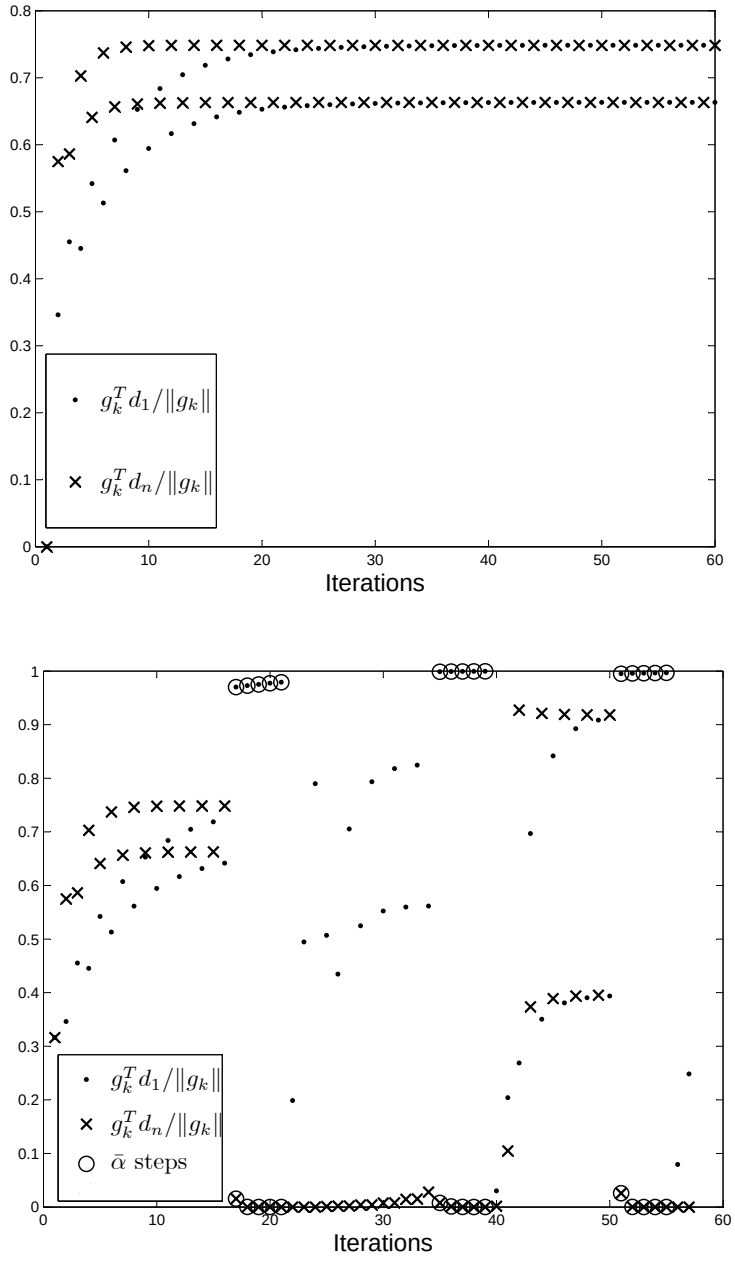


FIG. 3. Behaviour of the (normalized) components of the gradient along the eigendirections d_1 and d_n for CSD (top) and CSD1 with $h = 5$ (bottom).

ALGORITHM 3 (Relaxed CSD (RCSD))
choose $x_0 \in \Re^n$
$g_0 \leftarrow Ax_0 - b$; $k = 0$
while (not stop_condition)
  randomly choose $\alpha_k \in [0, 2\alpha_k^C]$
  $x_{k+1} \leftarrow x_k - \alpha_k g_k$; $g_{k+1} \leftarrow g_k - \alpha_k A g_k$
  $k \leftarrow k+1$
endwhile

Proposition 2.2 guarantees that the RCSD method converges monotonically to x^*. Numerical experiments in Raydan & Svaiter (2002) show that this method largely outperforms CSD, although the BB method and its Cauchy Barzilai Borwein (CBB) variant, which are non-monotone, are the fastest and most effective ones. From their numerical experiments the authors observe the tendency of the BB and CBB methods to force gradient directions to approximate eigenvectors of the Hessian matrix A; this explains, to some extent, the good behaviour of these methods.

The next proposition actually shows why and in which sense an over-relaxation of the Cauchy step fosters a similar tendency, and suggests a slightly different form of relaxation that produces better effects than a simple random choice of the step length in $[0, 2\alpha_k^C]$.

PROPOSITION 4.1 Let us consider the sequences $\{x_k\}$ and $\{g_k\}$ generated by the SD Algorithm 1, where

$$\alpha_k = 2\alpha_k^C; \tag{4.1}$$

then

$$\lim_k \frac{g_k}{\prod_{j=1}^k (1 - \alpha_j \lambda_1)} = \mu_1 d_1, \tag{4.2}$$

$$\lim_k \alpha_k = \frac{2}{\lambda_1}, \tag{4.3}$$

$$\lim_k \nabla f\left(x_k - \alpha_k^C g_k\right) = 0. \tag{4.4}$$

Proof. It is

$$g_k = \mu_1 \left(\prod_{j=1}^k (1 - \alpha_j \lambda_1) \right) d_1 + \sum_{i=2}^n \mu_i \left(\prod_{j=1}^k (1 - \alpha_j \lambda_i) \right) d_i$$

and hence

$$\frac{g_k}{\prod_{j=1}^k (1 - \alpha_j \lambda_1)} = \mu_1 d_1 + \sum_{i=2}^n \mu_i \prod_{j=1}^k \frac{(1 - \alpha_j \lambda_i)}{(1 - \alpha_j \lambda_1)} d_i. \tag{4.5}$$

Furthermore,

$$\frac{\lambda_n}{2} \leqslant \frac{1}{\alpha_j} \leqslant \frac{\lambda_1}{2} \tag{4.6}$$

and then

$$1 - \alpha_j \lambda_n \geqslant -1, \quad 1 - \alpha_j \lambda_1 \leqslant -1. \tag{4.7}$$

If we set $\theta = \lambda_1 - \lambda_2$, then $\lambda_1 \geqslant \lambda_i + \theta$, and it follows that

$$1 - \alpha_j \lambda_i \geqslant 1 - \alpha_j (\lambda_1 - \theta)$$

and hence, by (4.7),

$$\frac{1-\alpha_j\lambda_i}{1-\alpha_j\lambda_1} \leqslant 1+\frac{\theta\alpha_j}{1-\alpha_j\lambda_1}. \tag{4.8}$$

By using (4.6) we get

$$\frac{\theta\alpha_j}{1-\alpha_j\lambda_1} = \frac{\theta}{\frac{1}{\alpha_j}-\lambda_1} \leqslant \frac{\theta}{\frac{\lambda_n}{2}-\lambda_1}$$

and thus

$$\frac{1-\alpha_j\lambda_i}{1-\alpha_j\lambda_1} \leqslant 1-\rho, \tag{4.9}$$

with

$$\rho = \frac{2\theta}{2\lambda_1-\lambda_n}. \tag{4.10}$$

Since

$$\frac{1-\alpha_j\lambda_i}{1-\alpha_j\lambda_1} = -1+\frac{2-\alpha_j(\lambda_1+\lambda_i)}{1-\alpha_j\lambda_1} \tag{4.11}$$

and, by (4.6),

$$\frac{2-\alpha_j(\lambda_1+\lambda_i)}{1-\alpha_j\lambda_1} \geqslant \frac{2-\frac{2}{\lambda_n}(\lambda_1+\lambda_i)}{1-\alpha_j\lambda_1} =$$
$$\frac{2\lambda_n-2\lambda_1-2\lambda_i}{\lambda_n(1-\alpha_j\lambda_1)} = \frac{2\lambda_1-2\lambda_n+2\lambda_i}{\alpha_j\lambda_1\lambda_n-\lambda_n} \geqslant$$
$$\frac{2\theta+2\lambda_i}{2\lambda_1-\lambda_n} \geqslant \rho,$$

we get

$$-1+\rho \leqslant \frac{1-\alpha_j\lambda_i}{1-\alpha_j\lambda_1} \leqslant 1-\rho.$$

Therefore, by (4.5), we have (4.2).

Because of (4.2)

$$\lim_k \alpha_k = 2\frac{\mu_1^2 d_1^T d_1}{\mu_1^2 d_1^T A d_1},$$

and, since $Ad_1 = \lambda_1 d_1$, we have

$$\lim_k \alpha_k = 2\frac{\mu_1^2 d_1^T d_1}{\mu_1^2 \lambda_1 d_1^T d_1} = \frac{2}{\lambda_1}.$$

Thus (4.3) holds.

Finally, in order to prove (4.4) we first note that the sequence $\{\|g_k\|\}$, is bounded above, and so is $\{\prod_{j=1}^k(1-\alpha_j\lambda_1)\}$ because of (4.2). Then

$$\lim_k \nabla f\left(x_k-\frac{\alpha_k}{2}g_k\right) = \lim_k\left(g_{k-1}-\frac{\alpha_k}{2}Ag_{k-1}\right) =$$
$$\lim_k \prod_{j=1}^{k-1}(1-\alpha_j\lambda_1)\left(\frac{g_{k-1}}{\prod_{j=1}^{k-1}(1-\alpha_j\lambda_1)}-\frac{\alpha_k}{2}A\frac{g_{k-1}}{\prod_{j=1}^{k-1}(1-\alpha_j\lambda_1)}\right) =$$
$$\lim_k \prod_{j=1}^{k-1}(1-\alpha_j\lambda_1)\left(\mu_1 d_1-\frac{1}{\lambda_1}\mu_1 A d_1\right) = \lim_k \prod_{j=1}^{k-1}(1-\alpha_j\lambda_1)\left(\mu_1 d_1-\mu_1 d_1\right) = 0,$$

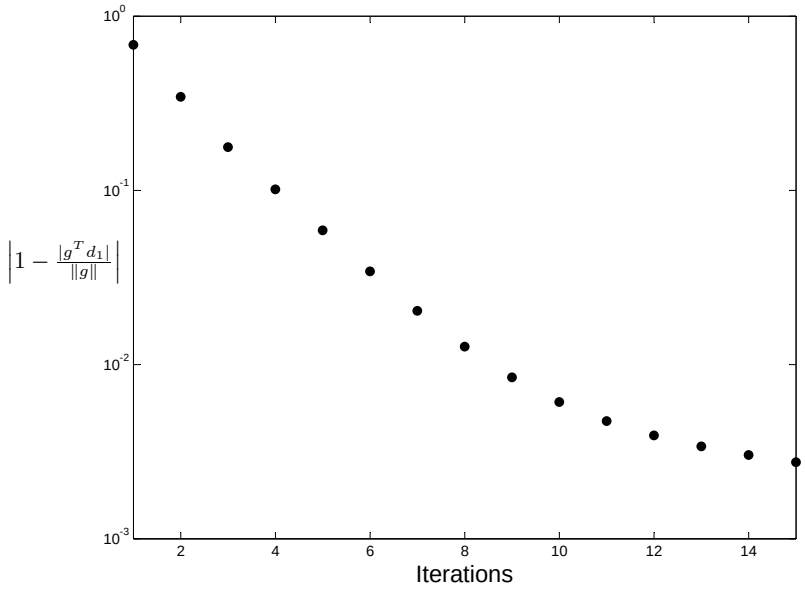


FIG. 4. Behaviour of the the (normalized) component of the gradient along the eigendirection d_1 in 15 consecutive double Cauchy steps.

hence (4.4) holds and the proof is complete. □

Proposition 4.1 suggests that the double Cauchy step, although meaningless in terms of function reduction, might have a significant impact in terms of alignment of the gradient with the eigenvector d_1, and this might be of some support in a general steepest descent framework.To verify such "alignment effect" we iteratively applied 15 consecutive double Cauchy steps to the problem described in Section 3. As predicted by Proposition 4.1, the component of the gradient along the eigendirection corresponding to the maximum eigenvalue of A becomes soon dominant, as shown in Figure 4.

For this problem, we also considered a modified version of the CSD method (CSDM), in which 5 consecutive double Cauchy steps are performed every 10 Cauchy steps; the results in Figure 5 shows that this simple modification of the Cauchy algorithm produces a rather meaningful speedup of the convergence.

About the RCSD method, Proposition 4.1 seems to suggest an over-relaxation rather than an under-relaxation of the Cauchy step, and therefore we considered a modified version of RCSD, called RCSD1, where

$$\alpha_k \in [0.8\alpha_k^C, 2\alpha_k^C]; \tag{4.12}$$

Figure 6 shows the results of RCSD1 applied to the same problem considered above; of course, because of the randomness in (4.12), a careful and deeper analysis is needed in order to evaluate the effectiveness of the method, especially to check the validity of our claim about the advantage in using RCSD1 rather than RCSD. Extensive numerical tests will be considered in the next section to get a clear picture of the numerical behaviour of the algorithmic approaches we proposed in the last two sections.

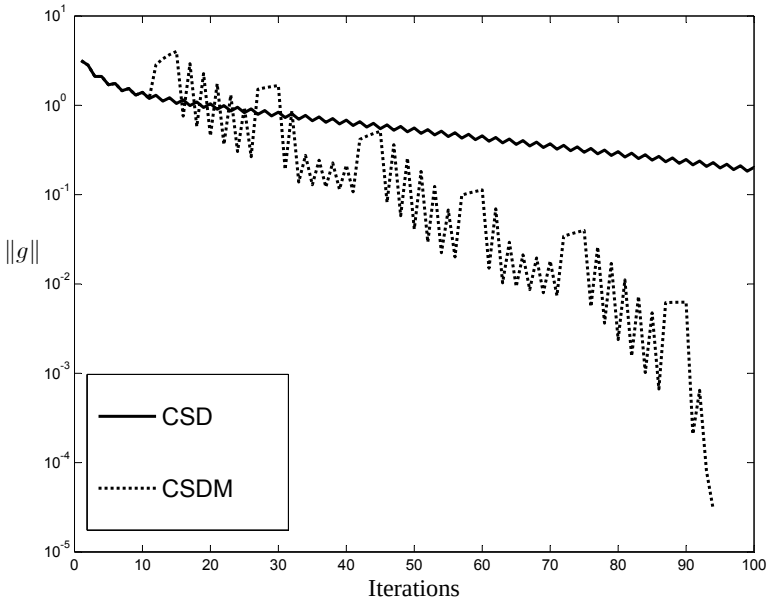


FIG. 5. Convergence of CSDM.

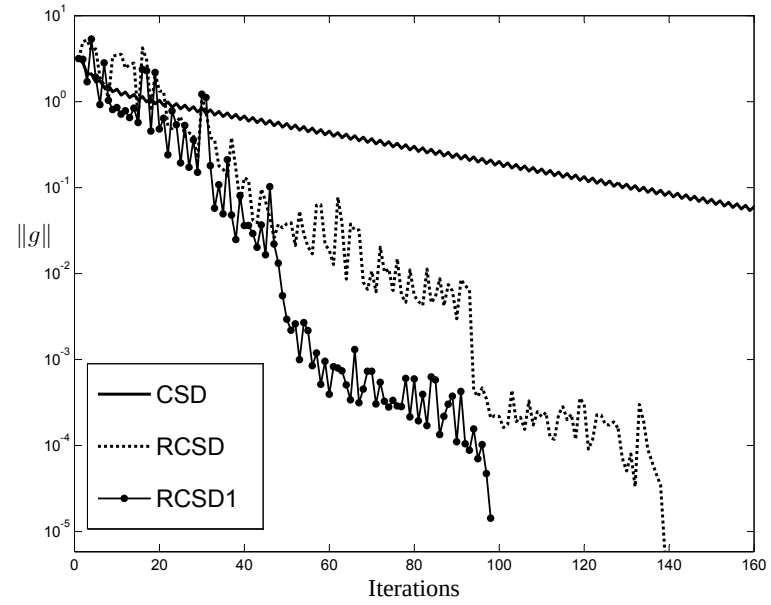


FIG. 6. Convergence of RCSD and RCSD1.

5. Numerical experiments

In this section we report some numerical results comparing CSD1 and RCSD1 with the BB algorithm using the step length (1.8). A more extensive comparison with other gradient methods would be interesting, but out of the scope of this paper, that is mainly to show how powerful, and probably underestimated, although well known, is the Cauchy method in revealing the spectral properties of Problem (1.4). These properties can easily be plugged into the algorithm with rather surprising results. On the other hand, the BB method is considered a quite efficient strategy, well representative of the so-called gradient methods with retard, even competitive with CG methods when low accuracy is required (Friedlander *et al.* (1998), Fletcher (2005)); therefore, it is a valid benchmark for testing the effectiveness of the CSD1 and RCSD1 algorithms. We also report the performance of the algorithm RCSD, mainly to verify if the conjecture in Section 4 about the advisability of using (4.12) in the randomly relaxed Cauchy method, as suggested by Theorem 4.1. We considered two sets of test problems of type (1.4). The problems of the first set were randomly generated, by using Matlab functions, with dimensions ranging from 100 to 1000. The Hessian A was generated by `sprandsym`, using `density` = 0.8, `kind`=1, and condition number $\kappa(A) = 10^2, 10^3, 10^4, 10^5$. For each of the four instances, x^* was generated by `rand` with entries in $[-10, 10]$ and the linear term was built as $b = Ax^*$. Furthermore, for each instance, 5 starting points were generated by `rand` with entries in $[-10, 10]$. As stopping criterion we used

$$\|g_k\| \leqslant 10^{-6}\|g_0\|.$$

All algorithms were implemented in Matlab. In CSD1, h and ε were set to 5 and 10^{-2}, respectively.

In Figures 7-8 we report the number of iterations of the four algorithms, fixing the condition number and varying the matrix dimension. The number of iterations is the mean of the results obtained with the five different starting points. We first notice that the poorest results were obtained by the two random Cauchy algorithms RCSD and RCSD1. This is not surprising at all (Friedlander *et al.* (1998)); however, it is worth noting the clear superiority of RCSD1 over RCSD. The other two methods give the best results, and actually CSD1 improves a little with respect to BB (5% less iterations on the average). As expected, the performance of both the algorithms deteriorates on the most ill conditioned problems (Figures 8), while the problem size appears to be a much less critical issue.

In Figure 9 we compare the complete convergence history (norm of the gradient and function value) of the CSD1 and BB algorithms for a specific instance of the test problems (n=300, $\kappa(A) = 10^3$). The difference in the behaviour of the two algorithms clearly emerges. The CSD1 iterates with step length $\bar{\alpha}$ are highlighted in the picture, making clear their role in accelerating the decrease of the objective function; for instance, around iteration 110, a reduction of about two orders of magnitude can be observed after a sequence of $\bar{\alpha}$ steps. A noticeable feature of CSD1 is that it adopted the double Cauchy step only once in order to preserve the algorithm monotonicity, and actually, in the overall set of 400 random test problems, it took this step only 20 times.

Similar results were obtained with the second set of test problems, consisting of the Laplace1(a) and Laplace1(b) problems described in Fletcher (2005), which arise from a uniform 7-point finite-difference discretization of the 3D Poisson equation on a box, with homogeneous Dirichlet boundary conditions. These problems have 10^6 variables and a highly sparse Hessian matrix with condition number $10^{3.61}$. For each problem, the linear term b and the starting point x_0 were generated as in Fletcher (2005); the iteration was terminated when $\|g_k\| < \eta\|g_0\|$, with $\eta = 10^{-2}, 10^{-4}, 10^{-6}$, to check the effects of different accuracy requirements. The algorithms were also compared with the CG method implemented in the Matlab `pcg` function.

The results in Table 3 show that, when high accuracy is required, CG outperforms the other gradient

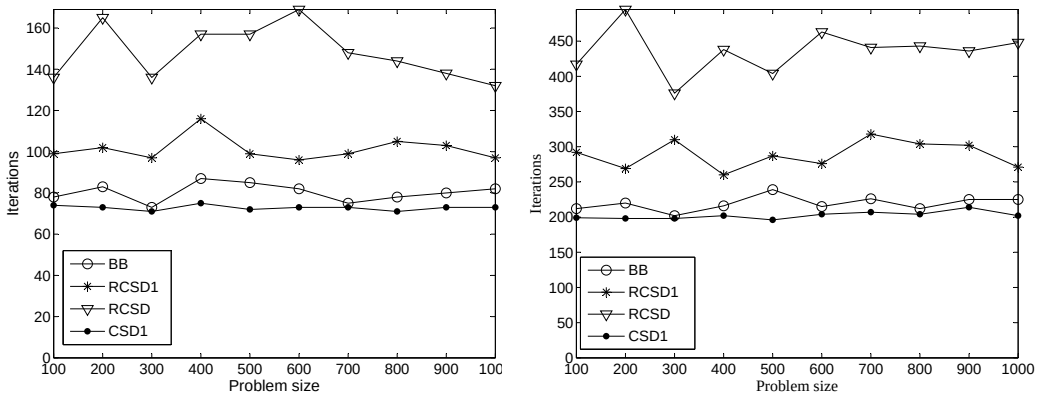


FIG. 7. Iterations for the randomly generated test problems, with $\kappa(A) = 10^2$ (left) and $\kappa(A) = 10^3$ (right).

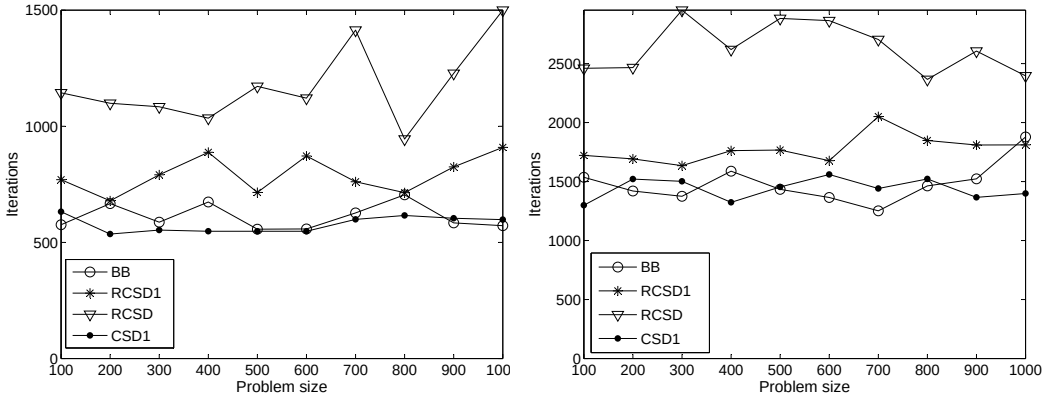
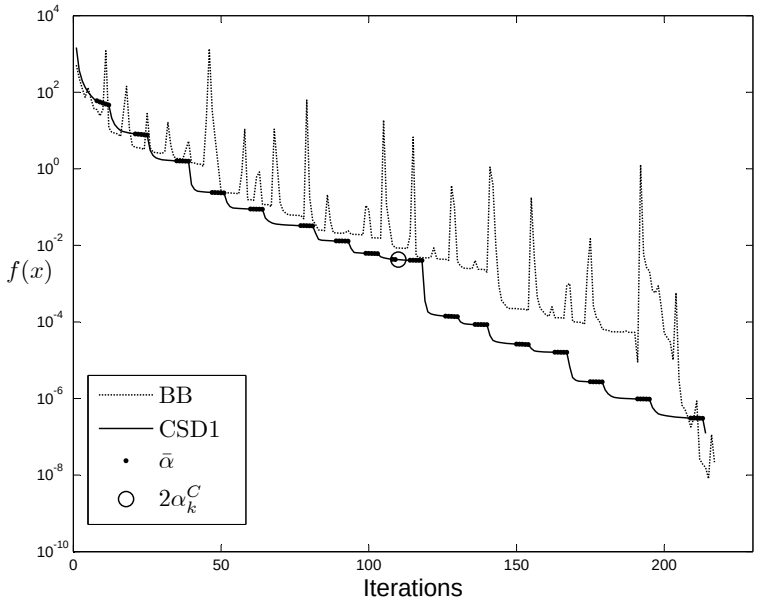


FIG. 8. Iterations for the randomly generated test problems, with $\kappa(A) = 10^4$ (left) and $\kappa(A) = 10^5$ (right).



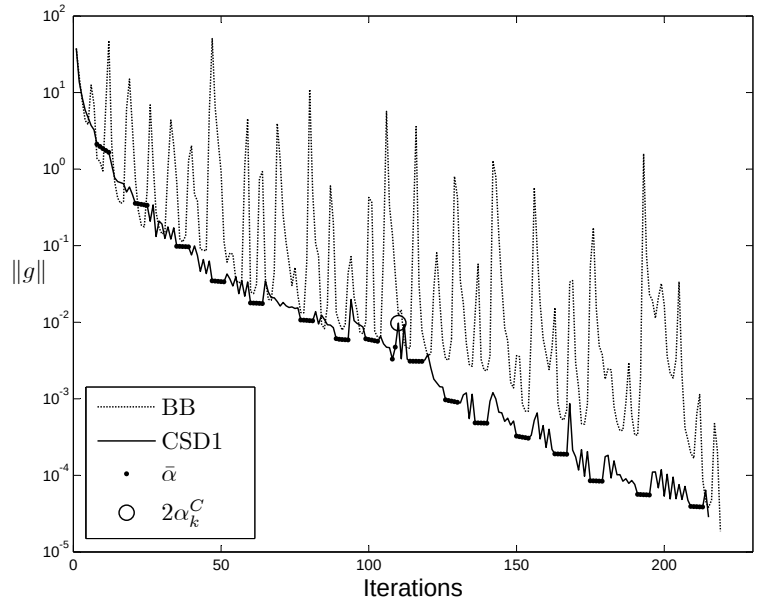


FIG. 9. Convergence history for algorithms BB and CSD1.

Problem	CG	BB	CSD1	RCSD	RCSD1
Laplace1 (a)	99	112	68	151	112
Laplace1 (b)	32	38	32	50	47

Table 1. Iterations for the Laplace problems, with stop condition $\|g_k\| < 10^{-2}\|g_0\|$.

Problem	CG	BB	CSD1	RCSD	RCSD1
Laplace1(a)	152	392	300	745	459
Laplace1(b)	179	188	174	366	251

Table 2. Iterations for the Laplace problems, with stop condition $\|g_k\| < 10^{-4}\|g_0\|$.

Problem	CG	BB	CSD1	RCSD	RCSD1
Laplace1(a)	189	561	562	1251	777
Laplace1(b)	274	383	386	791	561

Table 3. Iterations for the Laplace problems, with stop condition $\|g_k\| < 10^{-6}\|g_0\|$.

methods. RCSD1 and RCSD achieve the poorest results, with RCSD1 showing a significant improvement over RCSD; BB and CSD1 take a smaller number of iterations than the previous methods, but are still much slower than CG. Very interesting are the results in Tables 1 and 2, which suggest that, for low accuracy requirements, gradient methods, and especially CSD1, are reasonable alternatives to CG, for instance in the computational contexts outlined in Fletcher (2005). A possible explanation of the nice behaviour of CSD1 shown in Tables 1 and 2 is that a rather inaccurate alignment of the search direction with an eigendirection of A (which, in our experience, is usually achieved in very few iterations) can be sufficient to get a low-accuracy solution. When high accuracy is required, we guess that the effectiveness of the approach deteriorates because of the roundoff errors, that limit the extent to which $\widehat{\alpha}$ can actually be estimated through $\widetilde{\alpha}_k$.

In conclusion, about CSD1 and BB, we do not feel fair to state the superiority of one method with respect to the other. We just believe that our numerical experiences support the approach motivated by the theoretical results in Sections 3 and 4, which highlight some potentialities of the CSD algorithm, related to the spectral properties of A revealed by the method. The main drawback of CSD1 lies in the arbitrariness in the choice of the parameter h; however, numerical tests were carried out with different values of h, showing that, unless very small values are taken (say 1 or 2), the performance of CSD1 depends very little on h (varying its values between 3 and 10 was almost uninfluential on the algorithm performance).

Motivated by the encouraging numerical results, we hope the analysis in this paper can be further refined in order to design effective gradient methods for non-quadratic functions, for which the monotonicity property of CSD1 might represent a remarkable advantage over BB-like algorithms. Finally, we believe that using step lengths able to force the algorithm search in low-dimensional subspaces should keep its benefits also in the more general framework of constrained optimization; therefore, a possible further development of this research might be to incorporate the ideas outlined here in a projected gradient framework, to deal with bound constrained problems.

REFERENCES

AKAIKE, H. (1959) On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method. *Ann. Inst. Stat. Math. Tokyo*, **11**, 1–16.

ANDRETTA, M., BIRGIN, E. G. & MARTÍNEZ, J. M. (2010) Partial spectral projected gradient method with active-set strategy for linearly constrained optimization. *Numer. Algorithms*, **53**, 23–52.

BARZILAI, J. & BORWEIN, J. M. (1988) Two-point step size gradient methods. *IMA J. Numer. Anal.*, **8**, 141–148.

BIRGIN, E. G., MARTNEZ, J. M. & RAYDAN, M. (2000) Nonmonotone spectral projected gradient methods on convex sets. *SIAM J. Optimiz.*, 1196–1211.

CAUCHY, A. (1847) Méthodes générales pour la résolution des systèmes d'équations simultanées. *CR. Acad. Sci. Par.*, **25**, 536–538.

DAI, Y.-H. & FLETCHER, R. (2005) Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. *Numerische Mathematik*, **100**, 21–47.

DAI, Y.-H. & FLETCHER, R. (2006) New algorithms for singly linearly constrained quadratic programs subject to lower and upper bounds. *Math. Program.*, **106**, 403–421.

DAI, Y.-H. & LIAO, L.-Z. (2002) R-linear convergence of the Barzilai and Borwein gradient method. *IMA J Numer Anal*, **22**, 1–10.

ELMAN, H. C. & GOLUB, H. G. (1994) Inexact and preconditioned Uzawa algorithms for saddle point problems. *SIAM J. Numer. Anal.*, **31**.

FLETCHER, R. (2005) On the Barzilai-Borwein method. *Optimization and Control with Applications* (L. Qi, K. Teo, X. Yang, P. M. Pardalos & D. Hearn eds). Applied Optimization, vol. 96. Springer US, pp. 235–256.

FORSYTHE, G. E. (1968) On the asymptotic directions of the s-dimensional optimum gradient method. *Numer. Math.*, **11**, 57–76.

FRASSOLDATI, G., ZANNI, L. & ZANGHIRATI, G. (May 2008) New adaptive stepsize selections in gradient methods. *J. Ind. Manag. Optim.*, **4**, 299–312.

FRIEDLANDER, A., MARTÍNEZ, J. M., MOLINA, B. & RAYDAN, M. (1998) Gradient method with retards and generalizations. *SIAM J. Numer. Anal.*, **36**, 275–289.

HAGER, W. W. & ZHANG, H. (2006) A new active set algorithm for box constrained optimization. *SIAM Journal on Optimization*, **17**, 526–557.

NOCEDAL, J., SARTENAER, A. & ZHU, C. (2002) On the behavior of the gradient norm in the steepest descent method. *Comp. Optim. Appl.*, **22**, 5–35.

RAYDAN, M. (1997) The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM J. Optimiz.*, **7**, 26–33.

RAYDAN, M. & SVAITER, B. F. (2002) Relaxed steepest descent and Cauchy-Barzilai-Borwein method. *Comput. Optim. Appl.*, **21**, 155–167.

YUAN, Y. (2008) Step-sizes for the gradient method. *AMS/IP Studies in Advanced Mathematics*, **42**, 785.