An efficient gradient method using the Yuan steplength

Roberta De Asmundis · Daniela di Serafino · William W. Hager · Gerardo Toraldo · Hongchao Zhang

May 17, 2014 - revised version

Abstract We propose a new gradient method for quadratic programming, named SDC, which alternates some SD iterates with some gradient iterates that use a constant steplength computed through the Yuan formula. The SDC method exploits the asymptotic spectral behaviour of the Yuan steplength to foster a selective elimination of the components of the gradient along the eigenvectors of the Hessian matrix, i.e., to push the search in subspaces of smaller and smaller dimensions. The new method has global and R-linear convergence. Furthermore, numerical experiments show that it tends to outperform the Dai-Yuan method, which is one of the fastest methods among the gradient ones. In particular, SDC appears superior as the Hessian condition number and the accuracy requirement increase. Finally, if the number of consecutive SD iterates is not too small, the SDC method shows a monotonic behaviour.

Keywords gradient methods · Yuan steplength · quadratic programming.

1 Introduction

We are interested in designing efficient gradient methods for the solution of the convex quadratic problem

$$\underset{x\in\Re^n}{\text{minimize}}\, f(x) \equiv \frac{1}{2}x^T Ax - b^T x, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric positive definite and $b \in \mathbb{R}^n$. This problem, possibly with the addition of bound constraints, is of practical importance, since it arises in many applications, e.g., in compressed sensing and image processing [16, 23], and in machine learning and data mining [25];

Work partially supported by INdAM-GNCS(2013 Project *Numerical methods and software for large-scale optimization with applications to image processing* and 2014 Project *First-order optimization methods for image restoration and analysis*), by the National Science Foundation (grants 1016204 and 1115568), and by the Office of Naval Research (grant N00014-11-1-0068).

Roberta De Asmundis
Department of Computer, Control and Management Engineering "Antonio Ruberti", Sapienza University of Rome, Via L. Ariosto 25, 00185 Roma, Italy, E-mail: roberta.deasmundis@uniroma1.it

Daniela di Serafino
Department of Mathematics and Physics, 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, E-mail: daniela.diserafino@unina2.it

William W. Hager
Department of Mathematics, University of Florida, Gainesville, FL 32611-8105, USA, E-mail: hager@math.ufl.edu

Gerardo Toraldo
Department of Mathematics and Applications, University of Naples Federico II, Via Cintia, Monte S. Angelo, I-80126 Naples, Italy, E-mail: toraldo@unina.it

Hongchao Zhang
Department of Mathematics, Louisiana State University, Baton Rouge, LA 70803-4918, USA, E-mail: hozhang@math.lsu.edu

further applications are listed in [26,21]. Moreover, since the theoretical basis of gradient methods for general unconstrained optimization derives from the minimization of convex quadratic functions, problem (1.1) is a simple setting to design effective methods for more general problems. In addition, problem (1.1) allows to study the relevance of the eigenvalues of the Hessian of the objective function to the algorithms we consider.

Gradient methods generate a sequence $\{x_k\}$ as follows:

$$x_{k+1} = x_k - \alpha_k g_k, \quad k = 0, 1, 2, \dots \tag{1.2}$$

where $g_k = \nabla f(x_k)$ and the steplength $\alpha_k > 0$ depends on the method under consideration. In particular, in the classical Steepest Descent (SD) method [5] α_k is chosen as

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

It is well known that, although based on exact line searches, the SD method behaves poorly in most cases, because the sequence $\{x_k\}$ tends to zigzag between two orthogonal directions and this usually deteriorates convergence [1]. Therefore, the SD method, despite of the minimal storage requirements and the very low computational cost per iteration, has long been considered very bad and ineffective.

However, in the last years there has been a renewed interest for gradient methods, starting from the innovative approach of Barzilai and Borwein [2], who proposed two novel choices for α_k $(k > 0)$:

$$\alpha_k^{BB1} = \frac{\|s_{k-1}\|^2}{s_{k-1}^T y_{k-1}}, \quad \alpha_k^{BB2} = \frac{s_{k-1}^T y_{k-1}}{\|y_{k-1}\|^2},$$

where $s_{k-1} = x_k - x_{k-1}$, $y_{k-1} = g_k - g_{k-1}$, and $\|\cdot\|$ is the L_2 vector norm.

Global and R-linear convergence of the gradient method using the Barzilai-Borwein (BB) steplengths was proved in [28,9]. However, despite these steplengths cannot guarantee a decrease in the objective function at each iteration, they turned to be, in practice, much more efficient than (1.3). Actually, some authors argued about the relationship between the non-monotonicity and the surprising computational performance of the BB methods for strictly convex quadratic programming (see, e.g., [18] and the references therein).

It is interesting to note that α_k^{BB1} is equal to α_{k-1}^{SD}, i.e., the SD steplength at the previous iteration, while α_k^{BB2} is equal to α_{k-1}^{MG}, where

$$\alpha_k^{MG} = \operatorname*{argmin}_{\alpha} \|\nabla f(x_k - \alpha g_k)\| = \frac{g_k^T A g_k}{g_k^T A^2 g_k}$$

is the so-called minimal gradient steplength. Therefore, α_k^{BB1} and α_k^{BB2} can be seen as steplengths with one-step delay. In [21] the use of larger delays has been proposed in the gradient methods with retards, extending convergence results that hold for the BB methods. In [24] it is pointed out that the use of larger delays increases non-monotonicity and hence may speed up convergence, but it also increases the loss of precision which is observed in the intermediate computations of the method. Such a drawback can be reduced by using an adaptive strategy to choose the retard, but this still cannot guarantee monotonicity.

Effective extensions of the BB methods to general optimization problems were proposed by Raydan in [29] and then by several authors (see [3] and the references therein). In these cases, a nonmonotone line-search strategy, such as the one presented in [22], is applied to guarantee global convergence, whereas the constraints are dealt with by using classical gradient projection techniques (see, e.g., [12] and the references therein).

The advantage of retaining monotonicity has been pointed out by several authors (see [19] and the references therein), especially when dealing with non-quadratic problems. Several works have been recently devoted to design faster gradient methods [6,10,11,31,32,19,14], many of which are monotone, whose common basic idea is to combine Cauchy steps with other steplengths.

Yuan supports this approach through a theoretical and computational analysis leading to the conclusion that “a good gradient method would use at least one exact line search (the Cauchy step) in every few iterations" [32]. In particular, in [31] he proposed the following interesting formula for the steplength:

$$\alpha_k^Y = 2\left(\sqrt{\left(\frac{1}{\alpha_{k-1}^{SD}} - \frac{1}{\alpha_k^{SD}}\right)^2 + 4\frac{\|g_k\|^2}{\left(\alpha_{k-1}^{SD}\|g_{k-1}\|\right)^2}} + \frac{1}{\alpha_{k-1}^{SD}} + \frac{1}{\alpha_k^{SD}}\right)^{-1}; \tag{1.4}$$

this steplength was determined by imposing finite termination for two-dimensional quadratic problems and satisfies the inequalities

$$\left(\frac{1}{\alpha_{k-1}^{SD}} + \frac{1}{\alpha_k^{SD}}\right)^{-1} < \alpha_k^Y < \min\{\alpha_{k-1}^{SD}, \alpha_k^{SD}\}. \tag{1.5}$$

Based on (1.4), Dai and Yuan designed some methods which appear very competitive with the BB method [11]. These methods alternate exact line searches with one or more steplengths defined by (1.4).

In the same line, but through a quite different theoretical approach, in [14] De Asmundis et al. proposed the steplength

$$\widetilde{\alpha}_k = \left(\frac{1}{\alpha_{k-1}^{SD}} + \frac{1}{\alpha_k^{SD}}\right)^{-1}. \tag{1.6}$$

They presented a monotone gradient method, named SDA, which combines (1.6) and the Cauchy steplength, and showed that it tends to align the search direction with the eigendirection corresponding to the smallest eigenvalue of A. As a consequence, the search is asymptotically reduced to the one-dimensional subspace spanned by that eigendirection. More generally, De Asmundis et al. suggested that a better understanding of some new gradient methods can be achieved through a deeper analysis of asymptotic spectral properties of the steplength rules used by those methods. In this context, they also showed how the behaviour of the relaxed steepest descent method in [30] can be better understood, and its surprising computational performance further improved. The relevance of the eigenvalues of A to a deeper and more satisfying understanding of the gradient methods was also pointed out in [17,18,11], by explaining the effectiveness of the BB and related methods in terms of the relationship between steplength and Hessian eigenvalues, rather than of decrease in the objective function.

We note that although the Conjugate Gradient (CG) method is still the method of choice for convex quadratic programming, numerical experiments have pointed out some circumstances under which BB and other new gradient methods may be competitive with CG. This is the case, e.g., when low accuracy is required in the solution of the problem, or single rather than double precision is used in the computations, as pointed out in [21,18,15]. Gradient methods become even more competitive with the CG method when applied to more general problems than (1.1), e.g., when used with projection techniques in box-constrained problems or applied to non-quadratic objective functions, as in [7,8,4]. Furthermore, one of the main reasons which stimulates the research about gradient methods is their successful use in some applications, such as image deblurring and denoising, where they show a smoothing, regularizing effect, and where a "strict" optimal solution is not necessary (see, e.g., [23]).

Motivated by the previous considerations, in this paper we propose a new gradient method for problem (1.1), which exploits the Yuan steplength (1.4). Specifically, we first analyse the asymptotic behaviour of the Yuan steplength, showing that it tends to approximate the reciprocal of the largest eigenvalue of the Hessian matrix A. Then, based on this theoretical result, we propose a gradient method, named SDC, in which, cyclically, a certain number of SD iterates is followed by a certain number of gradient steps that use a constant steplength computed through the formula (1.4). We note that our scheme is different from the one in [11] and so is the motivation of our method. Its main computational feature is to foster the reduction of the gradient components along the eigenvectors of A in a selective way, in order to reduce the search in subspaces of smaller and smaller dimensions, and hence to deal with problems having better and better condition numbers.

The paper is organized as follows. In Section 2 we report some results that provide the theoretical basis for the analysis carried out in the sequel of the paper. In Section 3 we study the asymptotic behaviour of the steplength (1.4), highlighting its relationship with (1.6), and then, based on this study, we introduce the SDC method. We also illustrate the behaviour of this method through numerical experiments on a selected test problem, confirming the properties entailed by the theoretical results. In Section 4 we establish the R-linear convergence of our method. In Section 5 we analyse through numerical experiments the performance of SDC and make a comparison with the the most efficient Dai-Yuan method presented in [11], showing that SDC tends to outperform it. Finally, in Section 6, we draw some conclusions.

In the rest of the paper we denote by $\kappa(A)$ the spectral condition number of the Hessian matrix A, by $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ the eigenvalues of A, and by $\{d_1, d_2, \ldots, d_n\}$ a set of associated orthonormal eigenvectors. We also make the following assumptions:

Assumption 1 *The eigenvalues $\lambda_1, \ldots, \lambda_n$ are such that*

$$\lambda_1 > \lambda_2 > \cdots > \lambda_n > 0.$$

Assumption 2 *Any starting point x_0 considered in this work is such that*

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

Furthermore, we denote by μ_i^k, $i = 1, \ldots, n$, the component of g_k along d_i, i.e.,

$$g_k = \sum_{i=1}^{n} \mu_i^k d_i. \tag{1.7}$$

We point out that the above assumptions are not restrictive. For the first one see, e.g., [19, Section 2]; concerning the second one, we note that it is equivalent to state that the components of g_0 along d_0 and d_n, i.e., μ_1^0 and μ_n^0, are both nonzero at the starting point. More generally, if $g_0 \neq 0$, there exist i_1 and i_n, with $1 \leq i_1 \leq i_n \leq n$, such that $g_0^T d_{i_1} \neq 0$, $g_0^T d_{i_n} \neq 0$ and $g_0^T d_i = 0$ for $i < i_1$ and $i > i_n$. Then, as observed at the end of Section 2, for $i < i_1$ and $i > i_n$ the components of the gradient along d_i will be equal to zero at each iteration, and all the results presented in the sequel will hold with the indices i_1 and i_n in place of 1 and n, respectively.

2 Preliminary results

We now recall some theoretical results that will be useful to the analysis in Section 3. The next theorems summarise results in [1,27] about the behaviour of the sequences $\{\mu_i^k\}$, $\{\alpha_k^{SD}\}$, and $\{\|g_k\|\}$ generated by the SD method (see [27, Lemmas 3.3 and 5.5, and Theorem 5.1]).

Theorem 2.1 *Consider the SD method applied to problem (1.1), starting from any point x_0, and suppose that Assumptions 1-2 hold. Then*

$$\lim_{k\to\infty} \alpha_{2k}^{SD} = \frac{1 + c^2}{\lambda_n(1 + c^2\kappa(A))}, \tag{2.1}$$

$$\lim_{k\to\infty} \alpha_{2k+1}^{SD} = \frac{1 + c^2}{\lambda_n(\kappa(A) + c^2)}, \tag{2.2}$$

where c is the constant such that

$$c = \lim_{k\to\infty} \frac{\mu_1^{2k}}{\mu_n^{2k}} = -\lim_{k\to\infty} \frac{\mu_n^{2k+1}}{\mu_1^{2k+1}}. \tag{2.3}$$

Furthermore,

$$\lim_{k\to\infty} \frac{(\mu_i^k)^2}{\sum_{j=1}^{n} (\mu_j^k)^2} = 0 \quad \textit{for } 1 < i < n.$$

Theorem 2.2 *Consider the SD method applied to problem (1.1), starting from any point x_0, and suppose that Assumptions 1-2 hold. Then*

$$\lim_{k\to\infty} \frac{\|g_{2k+1}\|}{\|g_{2k}\|} = \frac{c^2(\kappa(A)-1)^2}{(1+c^2\kappa(A))^2},$$
$$\lim_{k\to\infty} \frac{\|g_{2k+2}\|}{\|g_{2k+1}\|} = \frac{c^2(\kappa(A)-1)^2}{(c^2+\kappa(A))^2},$$

where c is the same constant as in Theorem 2.1

As a consequence of Theorem 2.1, in the SD method eventually

$$g_k \approx \mu_1^k d_1 + \mu_n^k d_n,$$

i.e., the SD method asymptotically reduces its search in the two-dimensional subspace spanned by d_1 and d_n, zig-zagging between the two directions without being able to eliminate any of them.

In [14] the authors show how, moving from some theoretical properties of the SD method, second order information provided by the steplength (1.3) can be exploited to dramatically improve the usually poor practical behaviour of the Cauchy method, achieving computational results comparable with those of the BB method. A key issue in their reasoning is the following result ([14, p. 1422]).

Theorem 2.3 *Consider the SD method applied to problem (1.1), starting from any point x_0, and suppose that Assumptions 1-2 hold. Then*

$$\lim_{k\to\infty} \left(\frac{1}{\alpha_{2k}^{SD}} + \frac{1}{\alpha_{2k+1}^{SD}} \right) = \lambda_1 + \lambda_n \; . \tag{2.4}$$

They also show that a gradient method with constant steplength

$$\hat{\alpha}_k = \frac{1}{\lambda_1 + \lambda_n} \tag{2.5}$$

tends to align the search direction with the eigendirection of A corresponding to the minimum eigenvalue λ_n (see Proposition 3.2 in [14]), thus forcing the search in the one-dimensional subspace spanned by the eigendirection d_n. By combining this result with Theorem 2.3, they propose a modified version of the SD method, called SDA (SD with Alignment), in which a constant steplength computed using (1.6) is used at selected iterations.

We finally report some known formulas, which hold for any gradient method (1.2). We have

$$g_{k+1} = g_k - \alpha_k A g_k = \prod_{j=0}^{k} (I - \alpha_j A) g_0, \tag{2.6}$$

and, if

$$g_0 = \sum_{i=1}^{n} \mu_i^0 d_i,$$

then by (2.6) we have

$$g_{k+1} = \sum_{i=1}^{n} \mu_i^{k+1} d_i, \tag{2.7}$$

where

$$\mu_i^{k+1} = \mu_i^0 \prod_{j=0}^{k} (1 - \alpha_j \lambda_i). \tag{2.8}$$

Formulas (2.6)-(2.8) are significant in the study of gradient methods, since they allow to analyse 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 (2.7)-(2.8) that for $h > k$ it will be $\mu_i^h = 0$, i.e., 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 and only if $\mu_i^0 = 0$ or $\alpha_j = 1/\lambda_i$ for some $j \leq k$. Furthermore, from (2.6) it follows that the SD method has finite termination if and only if at some iteration the gradient is an eigenvector of A.

3 A new gradient method

In this section we study the asymptotic behaviour of the Yuan steplength (1.6). Motivated by our analysis, we propose a modification of the SD method that combines Cauchy steplengths and Yuan steplengths, in a different way from the Dai and Yuan methods in [11], but similarly to the SDA method in [14].

To find a relationship between the asymptotic behaviour of α_k^Y and the eigenvalues of the matrix A, we provide a different expression of it, which also highlights some connections between α_k^Y and $\widetilde{\alpha}_k$.

Lemma 3.1 *The Yuan steplength (1.4) can be written as*

$$\alpha_k^Y = \frac{2}{(\widetilde{\alpha}_k)^{-1} + \sqrt{(\widetilde{\alpha}_k)^{-2} - 4\rho_k}} = \frac{2\widetilde{\alpha}_k}{1 + \sqrt{1 - 4\rho_k(\widetilde{\alpha}_k)^2}}, \tag{3.1}$$

where $\widetilde{\alpha}_k$ is defined in (1.6) and

$$\rho_k = \frac{1}{\alpha_{k-1}^{SD}\alpha_k^{SD}} - \frac{\|g_k\|^2}{\|g_{k-1}\|^2}\frac{1}{\left(\alpha_{k-1}^{SD}\right)^2}. \tag{3.2}$$

Proof We first observe that

$$\alpha_k^Y = 2\left(\sqrt{\left(\frac{1}{\alpha_{k-1}^{SD}} - \frac{1}{\alpha_k^{SD}}\right)^2 + 4\frac{\|g_k\|^2}{\left(\alpha_{k-1}^{SD}\|g_{k-1}\|\right)^2}} + \frac{1}{\alpha_{k-1}^{SD}} + \frac{1}{\alpha_k^{SD}}\right)^{-1} =$$

$$2\left(\sqrt{\left(\frac{1}{\alpha_{k-1}^{SD}} + \frac{1}{\alpha_k^{SD}}\right)^2 - 4\frac{1}{\alpha_{k-1}^{SD}\alpha_k^{SD}} + 4\frac{\|g_k\|^2}{\left(\alpha_{k-1}^{SD}\|g_{k-1}\|\right)^2}} + \frac{1}{\alpha_{k-1}^{SD}} + \frac{1}{\alpha_k^{SD}}\right)^{-1}.$$

Then, the thesis trivially follows from the definition of $\widetilde{\alpha}_k$. □

We are now ready to analyse the asymptotic behaviour of α_k^Y.

Theorem 3.1 *Let $\{\alpha_k^{SD}\}$ and $\{g_k\}$ be the sequences generated by the SD method applied to problem (1.1), starting from any point x_0, and suppose that Assumptions 1-2 hold. Then,*

$$\lim_{k\to\infty} \alpha_k^Y = \frac{1}{\lambda_1}, \tag{3.3}$$

$$\lim_{k\to\infty} \rho_k = \lambda_1\lambda_n, \tag{3.4}$$

where α_k^Y and ρ_k are defined in (1.4) and (3.2), respectively.

Proof By Theorems 2.1 and 2.2 we have

$$\lim_{k\to\infty} \frac{1}{\alpha_k^{SD}\alpha_{k-1}^{SD}} = \frac{\lambda_n^2(\kappa(A)+c^2)(1+c^2\kappa(A))}{(1+c^2)^2},$$

$$\lim_{k\to\infty} \frac{\|g_k\|^2}{(\alpha_{k-1}^{SD}\|g_{k-1}\|)^2} = \frac{\lambda_n^2 c^2(\kappa(A)-1)^2}{(1+c^2)^2},$$

where c is the constant in (2.3), and therefore

$$\lim_{k\to\infty} \rho_k = \frac{\lambda_n^2}{(1+c^2)^2}\left[(\kappa(A)+c^2)(1+c^2\kappa(A)) - c^2(\kappa(A)-1)^2\right] = \tag{3.5}$$

$$\frac{\lambda_n^2}{(1+c^2)^2}\left(\kappa(A) + c^4\kappa(A) + 2c^2\kappa(A)\right) = \lambda_1\lambda_n. \tag{3.6}$$

By Lemma 3.1 and Theorem 2.3, we get

$$\lim_{k\to\infty} \alpha_k^Y = \frac{2}{\lambda_1 + \lambda_n + \sqrt{(\lambda_1+\lambda_n)^2 - 4\lambda_1\lambda_n}} = \frac{1}{\lambda_1}.$$

□

It trivially follows from Theorem 3.1 that, under Assumptions 1-2, the largest and the smallest eigenvalues of A can be approximated through α_k^Y, $\widetilde{\alpha}_k$ and ρ_k. More precisely,

$$\lim_{k\to\infty} \frac{1}{\alpha_k^Y} = \lambda_1, \qquad \lim_{k\to\infty} \rho_k \alpha_k^Y = \lim_{k\to\infty} \frac{1}{\widetilde{\alpha}_k} - \frac{1}{\alpha_k^Y} = \lambda_n. \tag{3.7}$$

In other words, Theorem 3.1 shows that the SD method asymptotically reveals some second order information, which can be conveniently exploited to speed up the convergence.

In Section 2 we observed that, for any gradient method, if at the k-th iteration $\alpha_k = 1/\lambda_i$ for some i, then the component of the gradient along the eigenvector d_i of A will be zero at all subsequent iterations. In particular, if we take the steplength $\alpha_k = 1/\lambda_1$, the component of the gradient along the corresponding eigenvector of A will be eliminated. Furthermore, a decrease in the objective function is guaranteed by this choice of α_k, since $1/\lambda_1 \leq \alpha_k^{SD}$. Of course, computing the exact value of λ_1 is unrealistic, but (3.7) suggests how to get an approximation of it.

The approach we propose is based on the idea of using a finite sequence of Cauchy steps with a twofold goal: forcing the search in a two-dimensional space and getting a suitable approximation of $1/\lambda_1$ through α_k^Y. Once a "good" approximation of $1/\lambda_1$ is obtained, the Yuan steplength providing such approximation is used, with the aim of driving toward zero the component μ_1^k of the gradient along d_1. Of course, μ_1^k cannot generally vanish, but it follows from (2.8) that if the approximation of $1/\lambda_1$ is accurate enough, then taking the same value of α_k^Y for multiple steps can significantly reduce μ_1^k. We also note that, in the ideal case where the component along d_1 is completely removed, the quadratic problem reduces to a $(n-1)$-dimensional problem and a new sequence of Cauchy steps followed by some steps with a constant value of α_k^Y can drive toward zero the component along the eigenvector d_2. Therefore, a cyclic alternation of SD steplengths and constant Yuan steplengths can be performed with the aim of eliminating the components of the gradient, according to the decreasing order of the eigenvalues of A. In other words, our strategy is aimed at reducing the search in subspaces of smaller and smaller dimensions, and forcing the gradient method to deal with problems with better and better condition numbers. The use of SD steplengths should also help in reducing the components of the gradient that are not addressed by the current Yuan steplength.

The above strategy for the choice of the steplength can be formalized as follows:

$$\alpha_k^{SDC} = \begin{cases} \alpha_k^{SD} \text{ if } \operatorname{mod}(k, h+m) < h, \\ \alpha_s^Y \quad \text{otherwise, with } \ s \ = \max\{i \leq k \ : \ \operatorname{mod}(i, h+m) = h\}, \end{cases} \tag{3.8}$$

where $h \geq 2$ and $m \geq 1$ (the superscript SDC indicates that SD steplengths are alternated with Constant ones, computed through the Yuan formula). In other words, we make h consecutive exact line searches and then, using the last two SD steplengths, we compute the Yuan steplength to be applied in m consecutive gradient iterations. It is clear that the two parameters h and m play complementary roles. Large values of h, which provide more accurate approximations of $1/\lambda_1$, are likely to work well with small values of m. Conversely, rough approximations of $1/\lambda_1$, due to small values of h, should be balanced by large values of m.

We note that the approach described so far is similar to the one in the SDA method presented in [14], where a sequence of SD steplenghts is combined with a sequence of steplengths of the form (1.6) in order to force the search into the one-dimensional subspace spanned by d_n, i.e., to asymptotically eliminate the gradient components along the remaining eigenvectors. We also observe that the steplength (3.8) generally differs from the one proposed in [11], i.e.,

$$\alpha_k^{DY} = \begin{cases} \alpha_k^{SD} \text{ if } \operatorname{mod}(k, h+m) < h, \\ \alpha_k^Y \quad \text{otherwise,} \end{cases} \tag{3.9}$$

since in the latter case α_k^Y is recomputed at each iteration.

A possible drawback of a gradient method with steplength (3.8), henceforth called SDC, could be the non-monotonicity, which is more likely to show up when small values of h and/or large values of m are adopted. Therefore, it is interesting to devise some strategy aimed to force monotonicity, to have a better picture of the effect of non-monotonicity on the performance of SDC. The most straightforward way to force the method to generate a decreasing sequence $\{f(x_k)\}$ is to impose that the steplength does not exceed $2\alpha_k^{SD}$, as in the SDA method. The resulting method, called SDC

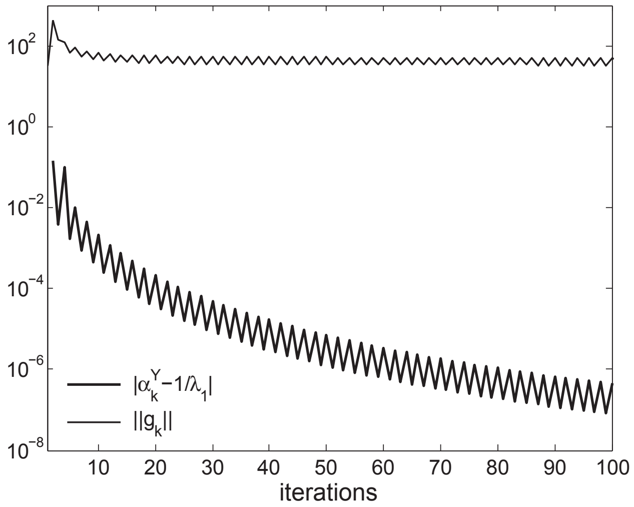

Fig. 1 Problem (3.10): convergence history of the sequences $\{|\alpha_k^Y - 1/\lambda_1|\}$ and $\{\|g_k\|\}$ for the first 100 iterations of the SD method.

with Monotonicity (SDCM), uses a steplength α_k^{SDCM} that is obtained from (3.8) by substituting α_s^Y with

$$\min\left\{\alpha_s^Y,\, 2\alpha_k^{SD}\right\}.$$

Note that SDCM can be seen as a special case of the relaxed steepest descent method, whose convergence is stated in [30]. In the next section we also prove the convergence of SDC, showing that it can be placed in the very general algorithmic framework proposed in [6].

To illustrate and support our approach, we apply the SDC and SDCM methods to a test problem of type (1.1), with 1000 variables and A diagonal, defined as follows:

$$A_{ii} = \frac{1}{i\sqrt{i}}, \quad b_i = 0. \tag{3.10}$$

The starting point x_0 is such that

$$Ax_0 = e, \quad e = (1, 1, \ldots, 1)^T,$$

and the following stopping condition is used:

$$\|g_k\| < tol\, \|g_0\|, \tag{3.11}$$

where $tol = 10^{-3}, 10^{-6}, 10^{-9}, 10^{-12}$. The experiments are performed by using Matlab. We notice that the SD method takes 5954 iterations to satisfy (3.11) with $tol = 10^{-3}$.

In Figure 1 we plot, on a log scale, the values of $|\alpha_k^Y - 1/\lambda_1|$ and $\|g_k\|$, for $k = 1, \ldots, 100$, against the number of iterations. It is evident that a quite accurate approximation of λ_1 is achieved after few iterations, despite the very slow convergence of the SD method. Therefore, we expect that a small value of h can be effectively used to build the Yuan steplength needed to selectively reduce the eigencomponents of the gradient.

In Tables 1-2 we report the results obtained by running the SDC and SDCM methods on the selected problem, with 9 possible choices for the pair (h, m), obtained by varying h and m in $\{2, 8, 16\}$ and $\{2, 4, 6\}$, respectively. These tests are aimed at understanding whether and how h and m affect the methods, in terms of number of iterations and spectral properties. For each run we show the overall number of iterations and, for SDC, the number of non-monotone steps, i.e., the steps where the objective function increases. For comparison purposes, in Table 1 we also report the number of iterations required by the Dai-Yuan (DY) method using (3.9) with $h = 2$ and $m = 2$, which corresponds to the most effective choice in [11].

SDC appears very competitive with DY, regardless of the choice of the parameters. However, the role of h and m is not negligible in the numerical behaviour of SDC. For the smallest value of h, the performance strongly improves as m increases, while this tendency is less evident for

Table 1 Problem (3.10): number of iterations of the SDC and DY methods. The number of non-monotone SDC steps is reported in brackets.

	SDC (h, m)									DY
tol	(2,2)	(2,4)	(2,6)	(8,2)	(8,4)	(8,6)	(16,2)	(16,4)	(16,6)	
10^{-3}	763 (11)	543 (53)	499 (102)	879	628 (6)	583 (2)	1154	822 (2)	808 (5)	848
10^{-6}	1517 (23)	1130 (98)	898 (162)	1471	1089 (12)	1247 (20)	1781	1352 (2)	1035 (9)	1612
10^{-9}	1853 (32)	1599 (152)	1345 (220)	2526	1513 (16)	1766 (36)	2393	1761 (3)	1540 (13)	2711
10^{-12}	2439 (39)	1996 (180)	1643 (264)	2869	2091 (21)	2048 (40)	2879	2108 (3)	2099 (20)	3612

Table 2 Problem (3.10): number of iterations of the SDCM method.

	SDCM (h, m)								
tol	(2,2)	(2,4)	(2,6)	(8,2)	(8,4)	(8,6)	(16,2)	(16,4)	(16,6)
10^{-3}	1039	591	579	879	633	505	1154	851	684
10^{-6}	1275	1079	1053	1471	1149	1025	1781	1249	1249
10^{-9}	1951	1753	1467	2526	1689	1451	2393	1781	1631
10^{-12}	2401	2179	1961	2869	2145	1969	2879	2229	2223

larger values of h. Making several consecutive exact linesearches ($h = 16$) fosters monotonicity, but the overall number of iterations tends to increase, due to the slow convergence of the SD method. Conversely, the monotonicity of the method deteriorates as m grows, especially when few SD steps are performed. Since the effectiveness of the BB methods has been related to their non-monotonicity [18], we wonder if non-monotonicity also plays a critical role in determining the nice behaviour of the SDC method. A comparison between the results in Table 1 and Table 2 seems to suggest that forcing the monotonicity does not deteriorate the performance too much, and actually it may also lead to some improvement (see, e.g., the case $h = 8$ and $m = 6$).

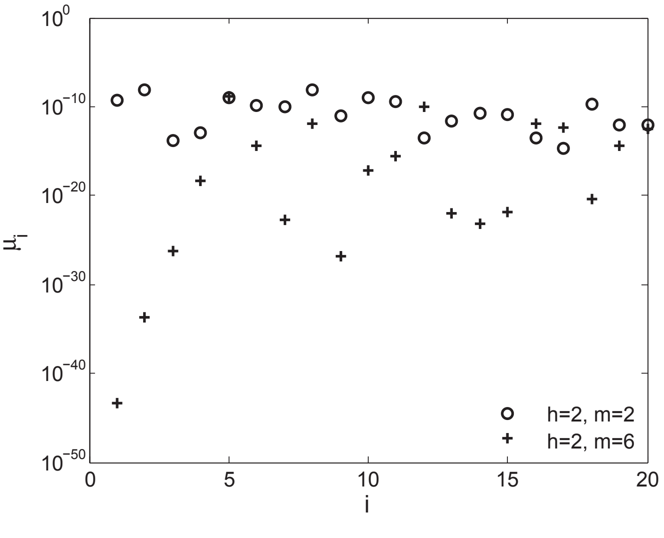

Fig. 2 Problem (3.10): values of the eigencomponents μ_i^k ($i = 1, \ldots, 20$) of the gradient at the solution computed by SDC, for $h = 2$ and $m = 2$ and for $h = 2$ and $m = 6$.

To gain further insight into the behaviour of the SDC method, we also analyse how the eigencomponents of the gradient are affected by m. In Figure 2, we plot the values of the first 20 eigencomponents of the gradient at the solution computed by SDC with $tol = 10^{-9}$, for $h = 2$ and $m = 2, 6$ (the smallest value of h is considered to better highlight the effects produced by different values of m). It clearly emerges that the order of magnitude of the eigencomponents is much smaller for $m = 6$, and, in this case, the first eigencomponents are practically zero. This confirms

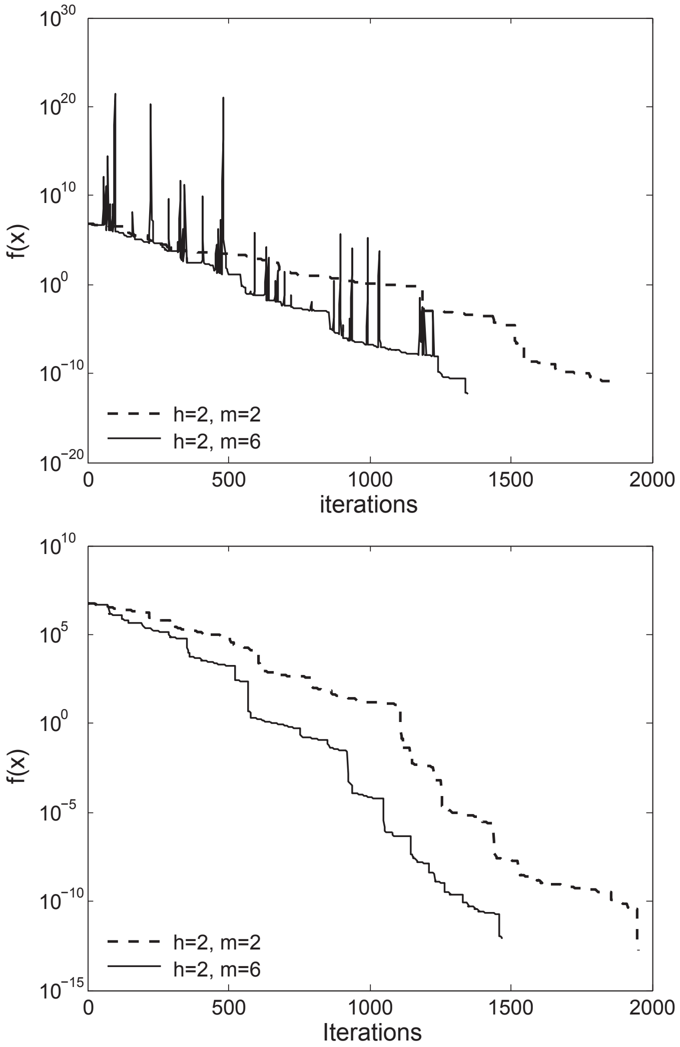

Fig. 3 Problem (3.10): convergence history of $\{f(x_k)\}$ in the SDC and SDCM methods (top and bottom, respectively), for $h = 2$ and $m = 2$ and for $h = 2$ and $m = 6$.

the role played by multiple constant Yuan steplengths in driving toward zero the eigencomponents corresponding to the largest eigenvalues.

Finally, in Figure 3 we compare the convergence histories of the objective function in the SDC and SDCM methods with $tol = 10^{-9}$, for $h = 2$ and $m = 2$ and for $h = 2$ and $m = 6$. The oscillating behaviour of SDC for $m = 6$ clearly appears, as well as its faster convergergence with respect to SDCM for both values of m. The figure also confirms that imposing monotonicity does not yield a severe deterioration of convergence.

4 Convergence analysis of the SDC method

We study the convergence of the SDC method following the approach proposed by Dai to establish the global and R-linear convergence of the Alternate Step Gradient method [6]. According to this approach, we assume without loss of generality that

$$A = \mathrm{diag}(\lambda_1, \lambda_2, \dots, \lambda_n) \tag{4.1}$$

and that $\lambda_n = 1$. From (4.1) and Assumption 1 it trivially follows that μ_i^k is the i-th component of g_k in the standard basis. Furthermore, we define

$$G(k,l) = \sum_{i=l}^{n} \left(\mu_i^k\right)^2.$$

Extending convergence results in [28,21,9], Dai establishes a convergence theory for gradient methods with steplengths satisfying the following property [6]:

Property A *There exists a positive integer m_0 and positive constants M_1 and M_2, with $M_1 \geq \lambda_n$, such that*

(i) $\lambda_n \leq \alpha_k^{-1} \leq M_1$;

(ii) for any $l \in \{2, \ldots, n\}$ and any $\varepsilon > 0$, if $G(k-j, l) \leq \varepsilon$ and $\left(\mu_{l-1}^{k-j}\right)^2 \geq M_2 \varepsilon$ hold for $j \in \{0, \ldots, \min\{k, m_0\}\}$, then $\alpha_k^{-1} \geq \frac{2}{3}\lambda_{l-1}$.

Dai's convergence theorem can be stated as follows.

Theorem 4.1 *Consider any gradient method (1.2) applied to problem (1.1) and assume that the matrix A satisfies (4.1) and Assumption 1 with $\lambda_n = 1$. If the steplength α_k has Property A, then, for any starting point x_0, either $g_k = 0$ for some k or the sequence $\{\|g_k\|\}$ converges to zero R-linearly.*

As pointed out in [6], Property A holds for many gradient methods. In the next theorem we show that it holds for the SDC method too. Then, the global and R-linear convergence of SDC follows from Theorem 4.1.

Theorem 4.2 *Consider the SDC method applied to problem (1.1) and assume that the matrix A satisfies (4.1) and Assumption 1. Then the steplength α_k^{SDC} satisfies Property A.*

Proof Since the SD steplength α_k^{SD} is such that $\left(\alpha_k^{SD}\right)^{-1} \in [\lambda_n, \lambda_1]$ and the Yuan steplength α_k^Y satisfies (1.5), condition (ii) holds for $M_1 = 2\lambda_1$. To prove that condition (ii) holds, we take $m_0 = m$, where m is the number of consecutive constant Yuan steplengths in the SDC method, and $M_2 = 2$. Now we assume that

$$G(k-j, l) \leq \varepsilon, \quad \left(\mu_{l-1}^{k-j}\right)^2 \geq M_2 \varepsilon, \tag{4.2}$$

for $j \in \{0, \ldots, \min\{k, m_0\}\}$, $l \in \{2, \ldots, n\}$ and $\varepsilon > 0$. It follows from (1.5) that

$$\alpha_k^{-1} \geq \left(\alpha_r^{SD}\right)^{-1},$$

where $r = \max\left\{i \leq k : \alpha_i = \alpha_i^{SD}\right\}$, i.e., r is the index of the last iteration in which the steplength was computed by using (1.3). Then, by recalling (1.7) we have

$$\begin{aligned}
\alpha_k^{-1} &\geq \left(\alpha_r^{SD}\right)^{-1} = \frac{g_r^T A g_r}{g_r^T g_r} = \frac{\displaystyle\sum_{i=1}^{n} \left(\mu_i^r\right)^2 \lambda_i}{\displaystyle\sum_{i=1}^{n} \left(\mu_i^r\right)^2} \\
&\geq \frac{\displaystyle\sum_{i=1}^{l-1} \left(\mu_i^r\right)^2 \lambda_i}{\displaystyle\sum_{i=1}^{l-1} \left(\mu_i^r\right)^2 + \sum_{i=l}^{n} \left(\mu_i^r\right)^2} \geq \frac{\lambda_{l-1} \displaystyle\sum_{i=1}^{l-1} \left(\mu_i^r\right)^2}{\displaystyle\sum_{i=1}^{l-1} \left(\mu_i^r\right)^2 + \varepsilon} = \frac{\lambda_{l-1}}{1 + \varepsilon \Big/ \left(\displaystyle\sum_{i=1}^{l-1} \left(\mu_i^r\right)^2\right)}.
\end{aligned} \tag{4.3}$$

Since $r \in \{\max\{k - m_0, 0\}, \ldots, k\}$, because of the second inequality in (4.2) we get

$$\sum_{i=1}^{l-1} \left(\mu_i^r\right)^2 \geq \left(\mu_{l-1}^r\right)^2 \geq 2\varepsilon,$$

and from (4.3) it follows that

$$\alpha_k^{-1} \geq \frac{2}{3}\lambda_{l-1}.$$

This completes the proof. □

Remark 4.1 *By reasoning as in the previous theorem we can prove the global and R-linear convergence of a variant of the SDA method proposed in [14], obtained by removing the requirement that the steplength not be greater than* $2\alpha_k^{SD}$.

5 Numerical experiments

In order to provide a clearer picture of the numerical behaviour of the SDC and SDCM methods, we performed extensive numerical experiments, especially aimed at verifying whether and how h and m affect the behaviour of the method, and if monotonicity can be enforced without significantly degrading performance.

We considered two sets of test problems of type (1.1), with A diagonal, $b = 0$ and dimension $n = 10^4$. For each set we defined

$$A_{11} = \xi, \quad A_{nn} = 1,$$

with $\xi = 10^4, 10^5, 10^6$, and built the remaining diagonal entries of A so that $\kappa(A) = \xi$. In the first set of problems, referred to as RAND, the diagonal matrix entries A_{jj}, with $j = 2, \ldots, n-1$, were randomly generated in $[A_{11},\ A_{nn}]$, while in the second set, referred to as NONRAND, they were set as follows:

$$A_{jj} = 10^{\frac{ncond}{n-1}(n-j)},$$

with $ncond = \log_{10}\xi = \log_{10}\kappa(A)$. For each problem, 10 starting points were randomly generated with entries in $[-5, 5]$. We note that the RAND problems, as well as the starting points, are those used in [11] to compare different gradient methods (actually, larger condition numbers are considered here). However, we also decided to use the NONRAND problems in order to test our methods on a class of quadratic problems on which the SD method exhibits very slow convergence (by running the SD method on an instance of the RAND and NONRAND problems with condition number $\kappa(A) = 10^6$, we got, for $tol = 10^{-6}$, 3321 iterations in the first case, and more than 100,000 iterations in the second case).

The SDC and SDCM methods were run with different values of h and m, i.e., all the combinations of $h = 10, 20, 30, 40, 50$ and $m = 2, 4, 8$. We neglected smaller values of h to avoid too strong non-monotonicity. For comparison purposes, the DY method considered in the previous section was also run. We did not run other gradient methods, since their performance is usually not superior than the one of the DY method [11,14]. The stopping criterion (3.11) was used by all the methods, with $tol = 10^{-6}, 10^{-9}, 10^{-12}$; a maximum number of 25,000 iterations was also set, but it was never reached in our experiments.

All the methods were implemented in Matlab (v. 8.0 - R2012b). The random diagonal entries of the matrix A in the first set of problems, as well as the starting points, were generated by using the Matlab `rand` function.

In Tables 3 and 4 we report the number of iterations performed by the SDC and DY methods on each RAND and NONRAND problem, averaged over the 10 runs with different starting points. The last row in each table is obtained by adding up the iterations performed on all the problems. We see that SDC exhibits a quite surprising monotonic behaviour for all the selected combinations of h and m (i.e., SDC and SDCM coincide), except $h = 10$ and $m = 8$. In the latter case, the number of SDCM iterations, reported in brackets, is comparable with the number of SDC iterations.

As expected, the RAND problems are easier to solve than the NONRAND ones. However, SDC does not perform as bad as SD on the NONRAND problems, and actually the number of total iterations increases by at most a factor of 2.2 when moving from the RAND to the NONRAND problems. For the SDC method, the overall worst performance occurs for $h = 10$, which is likely to produce a Yuan steplength providing an approximation of $1/\lambda_1$ that is not accurate enough. In this case the choice of m appears crucial, as we can see by comparing the results for $m = 2, 4$ with those for $m = 8$. The largest value of m is able to make up for the effects of using a small value of h, though producing non-monotonicity. For the other values of h, $m = 8$ is often less favorable

Table 3 RAND problems: mean number of iterations of the SDC and DY methods. The number of SDCM iterations is also reported in brackets if it is different from the SDC one.

Problem		SDC (h, m)															DY
$\kappa(A)$	*tol*	(10,2)	(10,4)	(10,8)	(20,2)	(20,4)	(20,8)	(30,2)	(30,4)	(30,8)	(40,2)	(40,4)	(40,8)	(50,2)	(50,4)	(50,8)	
10^4	10^{-6}	242	229	235	238	234	232	260	272	253	270	309	268	280	341	291	218
10^4	10^{-9}	840	882	678 (698)	772	658	707	740	690	719	779	734	726	894	779	855	701
10^4	10^{-12}	1324	1407	1186 (1206)	1181	1105	1092	1147	1069	1135	1197	1141	1164	1224	1122	1256	1142
10^5	10^{-6}	255	255	225	246	243	229	297	262	253	267	284	253	300	335	290	227
10^5	10^{-9}	2209	2742	1872 (1767)	1742	1666	1619	1555	1448	1775	1576	1408	1665	1507	1398	1606	2003
10^5	10^{-12}	4421	5168	3279 (3167)	2861	2739	3050	2816	2563	3025	2319	2744	2846	2607	2633	2774	3736
10^6	10^{-6}	227	221	204	224	230	221	245	249	227	239	281	255	244	315	277	208
10^6	10^{-9}	4999	6845	3587 (3309)	2576	2415	5118	2477	2897	3583	2614	2914	3401	2461	2520	4079	4346
10^6	10^{-12}	13274	18219	8099 (7735)	5869	7620	10866	5668	7168	8774	5690	6615	8671	5345	4944	7165	12899
Total iters		27791	35968	19365 (18546)	15709	16910	23134	15205	16618	19744	14951	16430	19249	14862	14387	18593	25480

Table 4 NONRAND problems: mean number of iterations of the SDC and DY methods. The number of SDCM iterations is also reported in brackets if it is different from the SDC one.

Problem		SDC (h, m)															DY
$\kappa(A)$	*tol*	(10,2)	(10,4)	(10,8)	(20,2)	(20,4)	(20,8)	(30,2)	(30,4)	(30,8)	(40,2)	(40,4)	(40,8)	(50,2)	(50,4)	(50,8)	
10^4	10^{-6}	597	559	555 (553)	579	555	515	586	558	548	625	557	551	657	584	580	551
10^4	10^{-9}	1117	1176	1030 (1012)	995	1039	974	1066	999	1030	1086	1008	1001	1122	1051	1063	1011
10^4	10^{-12}	1652	1707	1487 (1457)	1409	1413	1468	1443	1410	1441	1561	1451	1464	1545	1448	1495	1477
10^5	10^{-6}	1377	1421	1154 (1171)	1241	1245	1134	1227	1178	1251	1234	1256	1247	1290	1219	1166	1182
10^5	10^{-9}	3686	3795	2965 (3022)	2827	2866	3077	2791	2672	2756	2858	2805	2789	2840	2759	2705	2980
10^5	10^{-12}	5395	6136	4660 (4559)	4326	4303	4935	4128	4116	4131	4226	4193	4316	4186	4117	4149	4805
10^6	10^{-6}	2880	2470	2084 (1961)	2130	2047	1985	2049	1917	2031	2173	1914	1958	2038	2019	2024	1930
10^6	10^{-9}	12224	13754	7972 (8262)	7148	7737	8146	7348	7262	7934	7388	7406	7836	7420	7140	6989	10275
10^6	10^{-12}	21544	24038	13110 (14045)	11945	11873	14356	11656	12212	13938	11786	12148	12149	11482	11991	12224	19058
Total iters		50472	55056	35017 (36042)	32600	33078	36590	32294	32324	35060	32937	32738	33311	32580	32328	32395	43269

than the other values of m, especially if high accuracy in the solution is required. This is in line with the previous observation that a large value of m is unnecessary, or even counterproductive, when a sufficiently large value of h is selected. Our conjecture is that once a Yuan steplength has been able to eliminate a gradient eigencomponent, a further use of such steplength (which is likely to be smaller than the values of $1/\lambda_i$ corresponding to the remaining eigencomponents) can only slow down the method. For $h \geq 20$, setting $m = 2, 4$ leads to comparable results, with differences of less than 10% in the total number of iterations; using $m = 8$ generally does not pay, especially for the NONRAND problems. The largest value of h generally becomes more effective on the most ill-conditioned problems when high accuracy is required in the solution. On the other hand, when the condition number is not too large and the accuracy requirement is not too high, smaller values of h seem to be preferable. We believe that, when the problems are "easy" (i.e., they just require a few hundred iterations), a large value of h is unnecessary to get a suitable Yuan steplength, and actually it tends to increase the overall number of iterations. In order to support this argument, we define a SDC sweep as the sequence of h SD steps followed by m gradient steps with constant Yuan steplengths, and analyse the number of sweeps required by the SDC method with different values of h. For instance, on the RAND problem with $\kappa(A) = 10^6$ and $\texttt{tol} = 10^{-6}$, SDC performs, on average, 5.8 sweeps for $h = 50$ and $m = 4$, and 7.3 sweeps for $h = 30$ and $m = 4$. However, the number of SDC iterations is much larger in the first case (315 vs 249). In other words, increasing h increases the number of iterations despite the reduction of the number of sweeps.

Further experiments carried out on test problems with different dimensions (not reported here for the sake of brevity) are consistent with the previous findings. In particular, they show that the performance of SDC can seriously deteriorate only if very small values of h are used with large problems and high accuracy requirements. On the other hand, choosing $m = 2$ or $m = 4$ appears to be quite effective, independently of the problem dimension.

Concerning the comparison between the DY and SDC methods, we note that SDC performs worse than DY when the lowest accuracy is required. In the remaining cases, SDC tends to outperform DY for $h \geq 20$, with a saving in the number of iterations which increases significantly with the condition number and the accuracy requirement. This is a remarkable result, since the literature shows that, among the gradient methods, DY exhibits the best overall numerical performance [11, 14].

We conclude this section by observing that the ability of the SDC method to eliminate the eigencomponents corresponding to the largest eigenvalues, already pointed out in Section 3, is confirmed by the experiments on the RAND and NONRAND problems. A clear picture of this feature of SDC is provided in Figure 4, where the scalars

$$\beta_i^k = \sqrt{\sum_{j=1}^{i} \left(\mu_j^k\right)^2}, \quad i = 1, \ldots, n,$$

are plotted at the last iteration of the SDC method, with $h = 30$ and $m = 8$, applied to specific instances of the RAND and NONRAND problems with $\kappa(A) = 10^6$ and $tol = 10^{-12}$. Such ability is especially apparent for the RAND problem, for which the size of the gradient at the last iteration ($\|g_k\| \simeq 10^{-5}$) is mainly determined by the eigencomponents μ_i^k associated with few smallest eigenvalues.

6 Conclusions

We introduced a new gradient method, named SDC, for convex quadratic programming, in which some SD iterates are alternated with some gradient iterates that use a constant steplength based on the Yuan formula. The use of this constant steplength is justified by its spectral properties, which dramatically speed up the convergence of the SD method, by forcing a selective elimination of the eigencomponents of the gradient (i.e., the components of the gradient along the eigenvectors of the Hessian), starting from the one relative to the eigenvector with largest eigenvalue and proceeding toward the eigencomponents associated with smaller eigenvalues. This behaviour is quite different from that of other fast gradient methods, where the ability to reduce all the eigencomponents at the same rate has been experimentally observed and considered as one of the main reasons

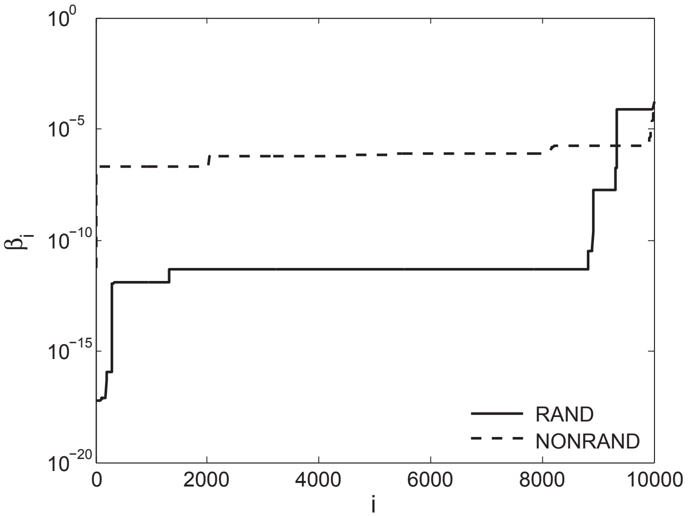

Fig. 4 RAND and NONRAND problems with $\kappa(A) = 10^6$: values of the scalars β_i, $i = 1, \ldots, n$, at the last SDC iteration ($tol = 10^{-12}$).

for the effectiveness of such methods [11]. Actually, such ability has been fostered by alternating long steplengths with short steplengths [18,33,20]. This is in contrast with the behaviour of the SDC method, because the Yuan formula tends to approximate the inverse of the largest eigenvalue of the Hessian, and hence to produce quite small steplengths. In our opinion, such feature may provide a twofold advantage to the method. Firstly, the monotonicity is quite naturally satisfied if the number of SD iterates is not too small, as confirmed by our numerical results. Secondly, it may improve the smoothing and regularizing effect observed for the SD method used for certain ill-posed inverse problems [13].

Current work is devoted to extend the SDC method to box-constrained quadratic problems and more general nonlinear optimization problems.

Acknowledgments We wish to thank the anonymous referees for their constructive and detailed comments, which helped to improve the quality of this paper.

References

1. Akaike, H.: 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 (1959)
2. Barzilai, J., Borwein, J.: Two-point step size gradient methods. IMA J. Numer. Anal. **8**, 141–148 (1988)
3. Birgin, E., Martínez, J., Raydan, M.: Spectral projected gradient methods: review and perspectives (2012). To appear in Journal of Statistical Software, available from http://www.ime.usp.br/∼egbirgin/publications/bmr5.pdf
4. Bonettini, S., Landi, G., Loli Piccolomini, E., Zanni, L.: Scaling techniques for gradient projection-type methods in astronomical image deblurring. Int. J. Comput. Math. **90**(1), 9–29 (2013)
5. Cauchy, A.: Méthodes générales pour la résolution des systèmes d'équations simultanées. CR. Acad. Sci. Par. **25**, 536–538 (1847)
6. Dai, Y.H.: Alternate step gradient method. Optimization **52**(4-5), 395–415 (2003)
7. Dai, Y.H., Fletcher, R.: Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. Numer. Math. **100**(1), 21–47 (2005)
8. Dai, Y.H., Hager, W., Schittkowski, K., Zhang, H.: The cyclic Barzilai-Borwein method for unconstrained optimization. IMA J. Numer. Anal. **26**(3), 604–627 (2006)
9. Dai, Y.H., Liao, L.Z.: R-linear convergence of the Barzilai and Borwein gradient method. IMA J. Numer. Anal. **22**(1), 1–10 (2002)
10. Dai, Y.H., Yuan, Y.: Alternate minimization gradient method. IMA J. Numer. Anal. **23**, 377–393 (2003)
11. Dai, Y.H., Yuan, Y.: Analysis of monotone gradient methods. J. Ind. Manag. Optim. **1**, 181–192 (2005)
12. De Angelis, P., Toraldo, G.: On the identification property of a projected gradient method. SIAM J. Numer. Anal. **30**(5), 1483–1497 (1993)
13. De Asmundis, R., di Serafino, D., Landi, G.: On the regularizing behavior of recent gradient methods in the solution of linear ill-posed problems (2014). Submitted

14. De Asmundis, R., di Serafino, D., Riccio, F., Toraldo, G.: On spectral properties of steepest descent methods. IMA J. Numer. Anal. **33**, 1416–1435 (2013)
15. van den Doel, K., Ascher, U.: The chaotic nature of faster gradient descent methods. J. Sci. Comput. **51**(3), 560–581 (2012)
16. Figueiredo, M., Nowak, R., Wright, S.: Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems. IEEE Journal of Selected Topics in Signal Processing **1**, 586–598 (2007)
17. Fletcher, R.: Low storage method for uncostrained optimization, *Lectures in Applied Mathematics*, vol. 26. AMS Publications (1990)
18. Fletcher, R.: On the Barzilai–Borwein method. In: L. Qi, K. Teo, X. Yang, P.M. Pardalos, D. Hearn (eds.) Optimization and Control with Applications, *Applied Optimization*, vol. 96, pp. 235–256. Springer (2005)
19. Fletcher, R.: A limited memory steepest descent method. Math. Program., Ser. A **135**, 413–436 (2012)
20. Frassoldati, G., Zanni, L., Zanghirati, G.: New adaptive stepsize selections in gradient methods. J. Ind. Manag. Optim. **4**(2), 299–312 (2008)
21. Friedlander, A., Martínez, J., Molina, B., Raydan, M.: Gradient method with retards and generalizations. SIAM J. Numer. Anal. **36**, 275–289 (1999)
22. Grippo, L., Lampariello, F., Lucidi, S.: A nonmonotone line search technique for Newton's method. SIAM J. Num. Anal. **23**, 707–716 (1986)
23. Huang, H.: Efficient reconstruction of 2D images and 3D surfaces. Ph.D. thesis, University of BC, Vancouver (2008)
24. Lamotte, J.L., Molina, B., Raydan, M.: Smooth and adaptive gradient method with retards. Math. Comput. Model. **36**(9–10), 1161–1168 (2002)
25. Loosli, G., Canu, S.: Optimization in Signal and Image Processing, chap. Quadratic Programming and Machine Learning – Large Scale Problems and Sparsity, pp. 111–135. Wiley (2010)
26. Moré, J., Toraldo, G.: Algorithms for bound constrained quadratic programming problems. Numer. Math. **55**, 377–400 (1989)
27. Nocedal, J., Sartenaer, A., Zhu, C.: On the behavior of the gradient norm in the steepest descent method. Comput. Optim. Appl. **22**, 5–35 (2002)
28. Raydan, M.: On the Barzilai and Borwein choice of steplength for the gradient method. IMA J. Numer. Anal. **13**, 321–326 (1993)
29. Raydan, M.: The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. SIAM J. Optim. **7**, 26–33 (1997)
30. Raydan, M., Svaiter, B.: Relaxed steepest descent and Cauchy-Barzilai-Borwein method. Comput. Optim. Appl. **21**, 155–167 (2002)
31. Yuan, Y.: A new stepsize for the steepest descent method. J. Comp. Math. **24**, 149–156 (2006)
32. Yuan, Y.: Step–sizes for the gradient method. AMS/IP Studies in Advanced Mathematics **42**(2), 785–796 (2008)
33. Zhou, B., Gao, L., Dai, Y.H.: Gradient methods with adaptive step-sizes. Comput. Optim. Appl. **35**(1), 69–86 (2006)