# A Dai-Liao conjugate gradient algorithm with clustering the eigenvalues


## Neculai Andrei

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


## January 4, 2017


**Abstract.** A new value for the parameter in Dai and Liao conjugate gradient algorithm is presented. This is based on the clustering the eigenvalues of the matrix which determine the search direction of this algorithm. This value of the parameter lead us to a variant of the Dai and Liao algorithm which is more efficient and more robust than the variants of the same algorithm based on the minimizing the condition number of the matrix associated to the search direction. Global convergence of this variant of the algorithm is briefly discussed.



**Key words:** Unconstrained optimization; Conjugate gradient algorithms; Eigenvalues clustering; Condition number; Wolfe conditions; Convergence


### 1. Introduction

For solving the unconstrained nonlinear optimization problem

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

where $f:\mathbb{R}^n\to\mathbb{R}$ is continuously differentiable and bounded below, using an extended conjugacy condition, Dai and Liao [9] (DL), suggested the following conjugate gradient algorithm:

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

the stepsize $\alpha_k$ is positive and the search directions $d_k$ are computed as:

$$d_{k+1}=-g_{k+1}+\beta_k^{DL}s_k,\quad d_0=-g_0, \tag{3}$$

$$\beta_k^{DL}=\frac{y_k^T g_{k+1}}{y_k^T s_k}-t_k\frac{s_k^T g_{k+1}}{y_k^T s_k}, \tag{4}$$

where $t_k$ is a nonnegative parameter, $g_k=\nabla f(x_k)$, $y_k=g_{k+1}-g_k$ and $s_k=x_{k+1}-x_k$. Usually, the steplength $\alpha_k$ is computed according to the Wolfe line search conditions:

$$f(x_k+\alpha_k d_k)\le f(x_k)+\delta\alpha_k g_k^T d_k, \tag{5}$$

$$g_{k+1}^T d_k\ge\sigma g_k^T d_k, \tag{6}$$

where $0<\delta\le\sigma<1$. A characteristic of this algorithm is that its numerical performances are very dependent on the parameter $t_k$ [3].

Observe that if $t_k=0$, then $\beta_k^{DL}$ reduces to the conjugate gradient parameter proposed by Hestenes and Stiefel [12]. Besides, we see that if $t_k=2\|y_k\|^2/y_k^T s_k$, then the conjugate gradient algorithm proposed by Hager and Zhang [11] is obtained. Also, if

$t_k = \tau_k + \|y_k\|^2 / y_k^T s_k - y_k^T s_k / \|s_k\|^2$, where $\tau_k$ is a parameter corresponding to the scaling factor in the scaled memoryless BFGS method, then we get the conjugate gradient of Dai and Kou [8].

Based on a singular values study, Babaie-Kafaki and Ghanbari [6] presented two optimal choices of the parameter $t_k$ in (4). The idea of these selections of the parameter $t_k$ is to minimize the condition number of the matrix representing the search direction. Intensive numerical experiments proved that the resulting algorithms are indeed more efficient and more robust than the conjugate gradient algorithms suggested by Hager and Zhang and that of Dai and Kou.

In this paper we propose another approach for the selection of the parameter $t_k$ in (4). The idea of this approach is to cluster the eigenvalues of the matrix representing the search direction. This is taken from linear conjugate gradient algorithms where clustering the eigenvalues is very benefic.

As in [6], in section 2, using the minimization of the condition number of the matrix representing the search direction, two optimal values of the parameter $t_k$ in (4) are presented. Section 3 is dedicated to present a new selection procedure based on clustering the eigenvalues of the matrix representing the search direction. In section 4 the numerical experiments and comparisons between these two approaches and CG-DESCENT are presented.

## 2. The values of the parameter $t_k$ based on minimizing the condition number [6]

Observe that from (3) and (4) the search direction of the DL algorithm can be very easy written as:

$$d_{k+1} = -H_{k+1} g_{k+1}, \tag{7}$$

where

$$H_{k+1} = I - \frac{s_k y_k^T}{y_k^T s_k} + t_k \frac{s_k s_k^T}{y_k^T s_k}. \tag{8}$$

Therefore, the DL method can be viewed as a quasi-Newton one in which the inverse Hessian is approximated by the nonsymmetric matrix $H_{k+1}$.

As we know, the numerical performances and the efficiency of the quasi-Newton methods are determined by the condition number $\kappa(H_{k+1})$ of the successive approximations of the inverse Hessian $H_{k+1}^{-1}$. A matrix with a large condition number is called an ill-conditioned matrix. Ill-conditioned matrices may produce instability in numerical computation with them, i.e. if $\kappa(H_{k+1})$ is large, then small values of relative error of $g_{k+1}$ in (7) may produce large relative error of the search direction $d_{k+1}$. For minimizing the condition number of the matrix $H_{k+1}$, representing the DL search direction, Babaie-Kafaki and Ghanbari [6] suggested two choices. Both of them are based on the estimations of the upper bound of the condition number of $H_{k+1}$. Using two special upper bounds of the condition number of $H_{k+1}$, in [6] the following choices of the parameter $t_k$ are determined:

$$t_{k1}^* = \frac{y_k^T s_k}{\|s_k\|^2} + \frac{\|y_k\|}{\|s_k\|} \tag{9}$$

and

$$t_{k2}^* = \frac{\|y_k\|}{\|s_k\|}, \tag{10}$$

which can be considered as optimal values for the parameter $t_k$ in DL conjugate gradient algorithm. Numerical experiments presented in [6] illustrate that both these variants of DL conjugate gradient algorithms (2)-(4), denoted as M1 for the first choice of $t_k$ given by (9) and

M2 for the second selection of $t_k$ given by (10) are more efficient and more robust than CG-DESCENT by Hager and Zhang [11] and DK+ by Dai and Kou [8].

### 3. The value of parameter $t_k$ based on clustering the eigenvalues

As we know, in a small neighborhood of the current point, the nonlinear objective function in (1) behaves like a quadratic one for which the results from linear conjugate gradient can apply. In fact, the first conjugate gradient algorithm was introduced by Hestenes and Stiefel [12] to minimize positive definite quadratic objective functions. This algorithm for solving positive definite linear algebraic systems of equations $Ax = b,\ A \in \mathbb{R}^{n\times n},\ b \in \mathbb{R}^n,$ is known as *linear conjugate gradient*. In exact arithmetic the linear conjugate gradient algorithm gives the correct solution within $n$ steps. In this case this can be considered as a direct method. However, in practice, this algorithm is regarded as an iterative method (see Reid [17]) because a sufficiently accurate approximation solution is often obtained in far fewer than $n$ steps. In absence of rounding errors, the theoretical convergence rate has been studied by many authors. The conclusion is that the rate of convergence of linear conjugate gradient depends strongly on the distribution of eigenvalues of the matrix $A$. Further insights concerning this problem of convergence was studied by many researchers, see for example: Axelsson [4], Axelsson and Lindskog [5], Strakoš [18], Van der Sluis and Van der Vorst [20], Meurant [15], Winther [21]. For faster convergence of linear conjugate gradient algorithms some approaches can be considered like: the presence of isolated smallest and/or largest eigenvalues of the matrix $H_{k+1}$, as well as gaps inside the eigenvalues spectrum [5], clustering of the eigenvalues about one point [21] or about several points [14], or preconditioning [13]. If the matrix has a number of certain distinct eigenvalues contained in $m$ disjoint intervals of very small length, then the linear conjugate gradient method will produce a very small residual after $m$ iterations. This is an important property of linear conjugate gradient method and we try to use it in nonlinear case. The idea of this variant of the DL algorithm, we present in this paper, is to determine $t_k$ by clustering the eigenvalues of $H_{k+1}$, by minimizing the largest eigenvalue of the matrix $H_{k+1}$. The structure of the eigenvalues of the matrix $H_{k+1}$ is given by the following theorem.

**Theorem 3.1.** *Let $H_{k+1}$ be defined by (8). Suppose that the stepsize $\alpha_k$ in (2) is selected by the Wolfe line search conditions (5) and (6). Then $H_{k+1}$ is a nonsingular matrix and its eigenvalues consist of 1 ($n-2$ multiplicity), $\lambda_{k+1}^{+}$ and $\lambda_{k+1}^{-}$, where*

$$\lambda_{k+1}^{\pm} = \frac{1}{2}\left[(1+t_k b_k) \pm \left|1-t_k b_k\right|\right], \tag{11}$$

*or 1 ($n-1$ multiplicity) and $t_k b_k$, where*

$$b_k = \frac{\|s_k\|^2}{y_k^T s_k} \geq 0. \tag{12}$$

***Proof*** By the Wolfe line search conditions (5) and (6) we have $y_k^T s_k > 0.$ Therefore, the vectors $y_k$ and $s_k$ are nonzero vectors. Let $V$ be the vector space spanned by $\{s_k, y_k\}$. We must consider two cases, as follows:

*Case 1.* $\dim(V) = 2.$ Hence $\dim(V^{\perp}) = n-2,$ where $V^{\perp}$ is the orthogonal complement of $V$ in $\mathbb{R}^n$. Thus, there exist a set of mutually unit orthogonal vectors $\{u_k^i\}_{i=1}^{n-2} \subset V^{\perp}$ such that

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

which from (8) leads to

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

Therefore, the matrix $H_{k+1}$ has $n-2$ eigenvalues equal to 1, which corresponds to $\{u_k^i\}_{i=1}^{n-2}$ as eigenvectors. Now, we are interested to find the rest of the two remaining eigenvalues, denoted as $\lambda_{k+1}^+$ and $\lambda_{k+1}^-$, respectively. From the formula of algebra (see for example [19]) we have $\det(H_{k+1}) = t_k b_k$. Hence, if $t_k > 0$, and the line search guarantees that $y_k^T s_k > 0$, then $H_{k+1}$ is nonsingular. On the other hand from (8) we see that $tr(H_{k+1}) = n - 1 + t_k b_k$. By the relationships between the determinant and the trace of a matrix and its eigenvalues, it follows that the other eigenvalues of $H_{k+1}$ are the roots of the following quadratic polynomial

$$\lambda^2 - (1 + t_k b_k)\lambda + t_k b_k = 0. \tag{13}$$

Clearly, the other two eigenvalues of the matrix $H_{k+1}$ are determined from (13) as in (11).

*Case 2.* $\dim(V) = 1$. Hence $\dim(V^\perp) = n-1$. Observe that $H_{k+1}s_k = (t_k b_k)s_k$. Therefore, $(t_k b_k, s_k)$ is an eigenpair of the matrix $H_{k+1}$. But, the determinant of a matrix is equal to the product of all its eigenvalues. In our case we have:

$$\det H_{k+1} = \lambda_1 \times \lambda_2 \times \cdots \times \lambda_{n-2} \times \lambda_{n-1} \times \lambda_n = \underbrace{1 \times 1 \times \cdots \times 1}_{n-2} \times t_k b_k \times \lambda_n.$$

But, $\det(H_{k+1}) = t_k b_k$. Hence, $\lambda_n = 1$. The matrix $H_{k+1}$ has as eigenvalues: 1 of $n-1$ multiplicity and $t_k b_k$. ■

Observe that if $t_k = 1/b_k$, then in both the above cases all the eigenvalues of $H_{k+1}$ are clustered in a point, i.e. the distance among them is minimized. Hence, another value for the parameter $t_k$ in DL algorithm, different from $t_{k1}^*$ and $t_{k2}^*$ given by Babaie-Kafaki and Ghanbari [6], can be computed as:

$$t_k^* = \frac{y_k^T s_k}{\|s_k\|^2}. \tag{14}$$

The value $t_k^*$ given by (14) can be considered as optimal subject to the criterion to minimize the distance among the eigenvalues of the iteration matrix $H_{k+1}$. Therefore, considering in (4) $t_k = t_k^*$, then we get the DLE conjugate gradient parameter $\beta_k^{DL*}$ as:

$$\beta_k^{DL*} = \frac{y_k^T g_{k+1}}{y_k^T s_k} - \frac{s_k^T g_{k+1}}{\|s_k\|^2} \tag{15}$$

for which the eigenvalues of $H_{k+1}$ are clustered in a point.

**Remark 3.1.** Assume that the level set $S = \{x \in \mathbb{R}^n : f(x) \le f(x_0)\}$ is bounded, where $x_0$ is the starting point of the iterative method (2). Suppose that in a neighborhood $N$ of $S$ the function $f$ is continuously differentiable and its gradient is Lipschitz continuous, i.e. there exists a constant $L > 0$ such that $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|$, for all $x, y \in N$. Under these assumptions, by Cauchy-Schwarz inequality, from (14) it is simple to see that $t_k^* \le L$. Hence, if the search directions are descent and the stepsizes are determined to satisfy the strong Wolfe conditions, then Theorem 3.3 of Dai and Liao [9] ensures the global convergence of the algorithm for uniformly convex objective functions. On the other hand, if (15) is modified as

$$\beta_k^{DL*} = \max\left\{\frac{y_k^T g_{k+1}}{y_k^T s_k}, 0\right\} - \frac{s_k^T g_{k+1}}{\|s_k\|^2}, \tag{16}$$

and the search directions satisfy the sufficient descent condition, then Theorem 6 of Dai and Liao [9] ensures the global convergence of the algorithm for general nonlinear objective functions. ■

**Remark 3.2**. Observe that $H_{k+1}$ given by (8) is non-symmetric. In [16] it is proved that in linear conjugate gradient method clustering the eigenvalues for non-symmetric matrices do not necessarily lead to fast convergence. However, in our case, by construction, the vast majority of eigenvalues of $H_{k+1}$ are equal to 1, only one or at most two being different by 1. Therefore, clustering the eigenvalues of $H_{k+1}$ in a point means modifying at most two eigenvalues. This is the reason we get a fast convergence of DLE algorithm. ■

**4. Numerical results and comparisons**

The algorithm given by (2) and (3), where $\beta_k^{DL}$ is replaced by $\beta_k^{DL*}$ from (16), is called DLE. We selected a number of 80 large-scale unconstrained optimization test functions in generalized or extended form, presented in [2], where the vast majority of problems are taken from CUTEr collection [7]. For each test function we have considered 10 numerical experiments with the number of variables increasing as $n = 1000, 2000, \ldots, 10000$. Therefore, the numerical experiments with these algorithms (DLE, M1 and M2) include a set of 800 unconstrained optimization test functions, of different structures and complexities.

The algorithms we compare in these numerical experiments find local solutions. Therefore, the comparisons of algorithms are given in the following context. Let $f_i^{ALG1}$ and $f_i^{ALG2}$ be the optimal value found by ALG1 and ALG2, for problem $i = 1, \ldots, 800$, respectively. We say that, in the particular problem $i$, the performance of ALG1 was better than the performance of ALG2 if:

$$\left| f_i^{ALG1} - f_i^{ALG2} \right| < 10^{-3} \tag{17}$$

and the number of iterations (#iter), or the number of function-gradient evaluations (#fg), or the CPU time of ALG1 was less than the number of iterations, or the number of function-gradient evaluations, or the CPU time corresponding to ALG2, respectively. The test problems where the algorithms do not converge to the same function value, according to criterion (17), are discarded from comparisons.

All algorithms considered in these numerical experiments DLE, M1 and M2 use the Wolfe line search conditions with cubic interpolation, $\rho = 0.0001$, $\sigma = 0.8$ and the same stopping criterion $\|g_k\|_\infty \leq 10^{-6}$, where $\|.\|_\infty$ represents the maximum absolute component of a vector. The algorithms, equipped with an acceleration procedure (see [1]) and the Powell-Beale restart criterion, were implemented in double precision Fortran using loop unrolling of depth 5 and compiled with f77 (default compiler settings) and run on a Workstation Intel Pentium 4 with 1.8 GHz. All codes are authored by Andrei.

Figure 1 presents the Dolan and Moré [10] performance profile subject to CPU computing time metric of DLE algorithm versus M1 variant of DL conjugate gradient algorithm where in (4) $t_k = t_{k1}^*$. Similarly, Figure 2 presents a comparison between DLE versus M2 variant of DL conjugate gradient algorithm where in (4) $t_k = t_{k2}^*$. Form Figure 1, we see that out of 800 problems, we considered in this numerical study, only for 789 problems does the criterion (17) hold. For example, comparing DLE versus M1, in Figure 1, subject to the number of iterations, we see that DLE was better in 187 problems (i.e. it achieved the minimum number of iterations

for solving 187 problems), M1 was better in 161 problems and they achieved the same number of iterations in 441 problems. Similarly, subject to the number of function and gradient evaluations, DLE was better in 227 problems, M1 was better in 192 problems and they achieved the same number of evaluations in 370 problems. Subject to CPU computing time, DLE was better in 187 problems, M1 was better in 156 problems and they achieved the same computing time, for solving 789 problems, in 446 problems. In these figures we plot the fraction $P$ of problems for which any given method is within a factor $\tau$ of the best time.

In the performance profile plots the top curve corresponds to the method that solved the most problems in a time that was within a factor $\tau$ of the best time. The percentage of the test problems for which a method is the fastest is given on the left axis of the plot. The right-hand side of the plot gives the percentage of the test problems that were successfully solved by these algorithms, respectively. Mainly, the right side is a measure of the robustness of an algorithm.

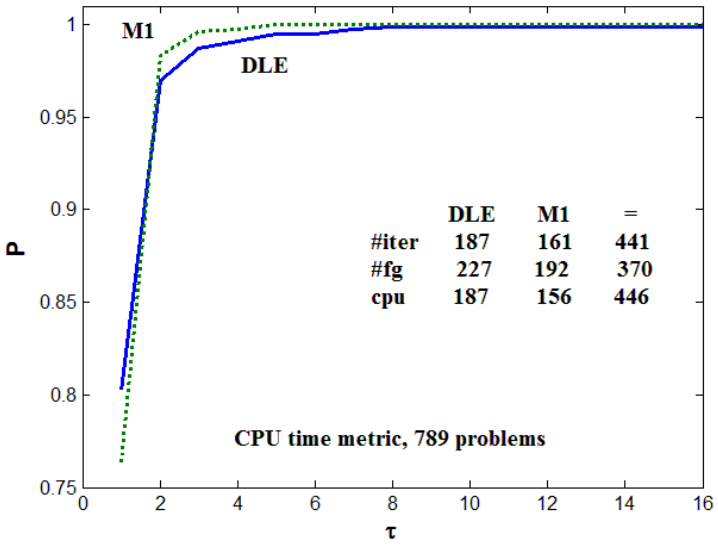


**Fig. 1.** DLE versus M1 subject to cpu time metric.

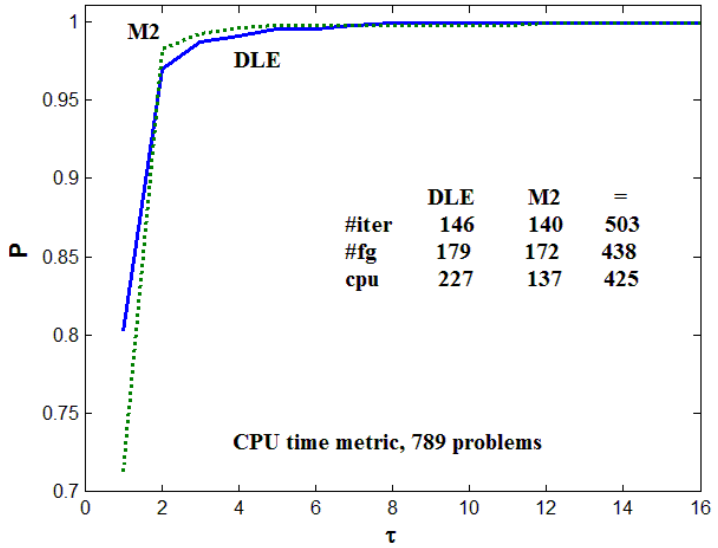


**Fig. 2.** DLE versus M2 subject to cpu time metric.

We see that DLE algorithm is more efficient than both M1 and M2 variants of DL algorithm suggested by Babaie-Kafaki and Ghanbari [6]. Also, DLE is slightly more robust than M2. Therefore, in comparison with M1 and M2, on average, DLE appears to generate the best search direction. We see that this computational scheme based on clustering the eigenvalues of the matrix representing the search direction constitutes a very serious alternative to those algorithms based on minimizing the condition number of the same matrix. Similar performance plots are obtained subject to the number of iterations and the number of function and gradient evaluations metrics.

In the second set of numerical experiments, in order to see the importance of an optimal selection of the parameter $t_k$ in the Dai and Liao algorithm, we compare DLE versus CG-DESCENT by Hager and Zhang [11] (version 1.4, Wolfe line search, default settings, $\|\nabla f(x_k)\|_\infty \leq 10^{-6}$). Observe that CG-DESCENT, which is among the best nonlinear conjugate gradient algorithms proposed in the literature, but not necessarily the best, is an ad hoc conjugate gradient algorithm obtained by taking *ex abrupto* $t_k = 2\|y_k\|^2 / y_k^T s_k$ in (4). Figure 3 presents the Dolan and Moré performance profiles of these algorithms.

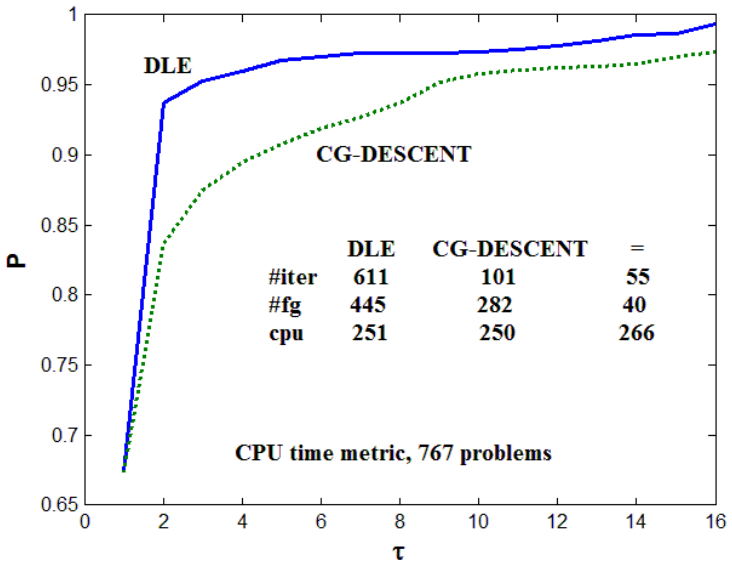


**Fig. 3.** DLE versus CG-DESCENT subject to cpu time metric.

From Figure 3 we see that DLE is top performer versus CG-DESCENT, and the difference is especially significant subject to robustness. Both these algorithms have similar efficiency. Therefore, we have the computational evidence that clustering the eigenvalues of the iteration matrix in Dai and Liao conjugate gradient algorithm lead us to a more efficient and clearly more robust algorithm. Selection of an optimal value for the parameter $t_k$ in the Dai and Liao algorithm has a crucial effect on its performances.

## 5. Conclusions

For the parameter $t_k$ in Dai and Liao conjugate gradient algorithm, Babaie-Kafaki and Ghanbari [6] proposed two optimal values. These are determined by minimizing two estimations of the upper bound of the condition number of the matrix which generates the search directions in this

algorithm. Observe that $t_{k1}^*$ and $t_{k2}^*$ given by (9) and (10) respectively are not optimal in the real sense of the word, because they are not minimizing the condition number of the iteration matrix, but instead two estimations of the upper bound of it. On the other hand, in this paper, using the idea of clustering the eigenvalues of the matrix determined by the search direction of the Dai and Liao conjugate gradient algorithm a new optimal value of the parameter $t_k$ is obtained. Global convergence of this new variant of the Dai and Liao algorithm is discussed. Using 800 large-scale unconstrained optimization test problems, of different structures and complexities, we have the computational evidence that the algorithm based on clustering the eigenvalues is more efficient than its variants using the minimization of the estimates of the upper bound of the condition number. All these algorithms have similar robustness. The first conclusion of this paper, confirmed by intensive numerical experiments, shows that both these approaches based on minimizing the condition number of the matrix which generates the search directions on one hand, and by clustering the eigenvalues of the same matrix, on the other hand, lead us to efficient and robust algorithms. The approach based on clustering the eigenvalues is clearly more efficient. On the other hand, the second conclusion is that the Dai-Liao algorithm with parameter $t_k$ selected in an optimal manner by clustering the eigenvalues of the iteration matrix, i.e. by minimizing the distance among the eigenvalues of the iteration matrix, is more efficient and clearly more robust than the CG-DESCENT algorithm.

**References**


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

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

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

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

[5] Axelsson, O., Lindskog, G.: On the rate of convergence of the preconditioned conjugate gradient methods. Numer. Math. 48, 499-523 (1986)

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

[7] Bongartz, I., Conn, A.R., Gould, N.I.M., Toint, Ph.L.: CUTEr: constrained and unconstrained testing environments. ACM Trans. Math. Software 21, 123-160 (1995)

[8] Dai, Y.H., Kou, C.X.: A nonlinear conjugate gradient algorithm with an optimal property and an improved Wolfe line search. SIAM Journal on Optimization 23, 296-320 (2013)

[9] Dai, Y.H., Liao, L.Z.: New conjugate conditions and related nonlinear conjugate gradient methods Appl. Math. Optim. 43, 87-101 (2001)

[10] Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Mathematical Programming 91, 201-213 (2002)

[11] Hager, W.W., Zhang, H.: A new conjugate gradient method with guaranteed descent and an efficient line search SIAM Journal on Optimization 16, 170-192 (2005)

[12] Hestenes, M.R., Steifel, E.: Metods of conjugate gradients for solving linear systems. J. Research Nat. Bur. Standards Sec. B. 48, 409-436 (1952)

[13] Kaporin, I.E.: New convergence results and preconditioning strategies for the conjugate gradient methods. Numerical Linear Algebra with Applications 1, 179-210 (1994)

[14] Kratzer, D., Parter, S.V., Steuerwalt, M.: Bolck splittings for the conjugate gradient method. Comp. Fluid. 11, 255-279 (1983)

[15] Meurant, G.: Computer Solution of Large Linear Systems. Studies in Mathematics and its Applications, volume 28, North Holland, Elsevier, Amsterdam (1999)
[16] Pestana, J., Wathen, A.J.: On the choice of preconditioner for minimum residual methods for non-Hermitian matrices. Journal of Computational and Applied Mathematics 249, 57-68 (2013)
[17] Reid, J.K.: On the method of conjugate gradients for solution of large sparse systems of linear equations. In: J.K. Reid (Ed.) Large Sparse Sets of Linear Equations, Academic Press, London 231-254 (1971)
[18] Strakoš, Z.: On the real convergence rate of the conjugate gradient method. Linear Algebra and its Applications 154-156, 535-549 (1991)
[19] Sun, W., Yuan, Y.X.: Optimization Theory and Methods. Nonlinear Programming. Springer Science + Business Media, New York (2006)
[20] Van der Sluis, A., Van der Vorst, H.A.: The rate of convergence of conjugate gradients. Numer. Math. 48, 543-560 (1986)
[21] Winther, R.: Some superlinear convergence results for the conjugate gradient method. SIAM J. Numer. Anal. 17, 14-17 (1980)

---oooOooo---