A hybrid conjugate gradient method based on the self-scaled memoryless BFGS update

Ioannis E. Livieris
Department of Computer Engineering & Informatics
Technological Educational Institute of Western Greece
Antirrio, Greece GR 263-34
livieris@teiwest.gr

Stamatis Karlos
Department of Computer Engineering & Informatics
Technological Educational Institute of Western Greece
Antirrio, Greece GR 263-34
stkarlos@upatras.gr

Vassilis Tampakas
Department of Computer Engineering & Informatics
Technological Educational Institute of Western Greece
Antirrio, Greece GR 263-34
vtampakas@teimes.gr

Panagiotis Pintelas
Department of Mathematics
University of Patras
Patras, Greece GR 265-00
ppintelas@gmail.com

ABSTRACT

In this work, we present a new conjugate gradient method adapting the approach of the hybridization of the conjugate gradient update parameters of DY and HS+ convexly, which is based on a quasi-Newton philosophy. The computation of the hybrization parameter is obtained by minimizing the distance between the hybrid conjugate gradient direction and the self-scaling memoryless BFGS direction. Our numerical experiments indicate that our proposed method is preferable and in general superior to classic conjugate gradient methods in terms of efficiency and robustness.

CCS CONCEPTS

• **Theory of computation** → *Mathematical optimization*; • **Mathematics of computing** → *Continuous optimization*;

KEYWORDS

Unconstrained optimization, conjugate gradient method, Frobenious norm, self-scaled memoryless BFGS.

ACM Reference format:
Ioannis E. Livieris, Stamatis Karlos, Vassilis Tampakas, and Panagiotis Pintelas. 2017. A hybrid conjugate gradient method based on the self-scaled memoryless BFGS update. In *Proceedings of PCI 2017, Larissa,Greece, September 28–30, 2017,* 5 pages.
DOI: 10.1145/3139367.3139384

PCI 2017, Larissa,Greece

978-1-4503-5355-7/17/09…$15.00
DOI: 10.1145/3139367.3139384

1 INTRODUCTION

Let us consider the unconstrained optimization problem

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

where $f:\mathbb{R}^n\to\mathbb{R}$ is a continuously differentiable function and its gradient is denoted by $g(x)=\nabla f(x)$. Conjugate gradient methods is probably the most popular class of unconstrained optimization algorithms, characterized by their low memory requirements, simple computations and strong global convergence properties. Generally, a nonlinear conjugate gradient method generates a sequence of points $\{x_k\}$, starting from an initial point $x_0\in\mathbb{R}^n$, using the iterative formula

$$x_{k+1}=x_k+\alpha_k d_k,\quad k=0,1,\ldots \tag{2}$$

where x_k is the k-th approximation to the solution of (1), $\alpha_k>0$ is the stepsize obtained by a line search and d_k is the search direction which is defined by

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

where $g_k=g(x_k)$. Conjugate gradient methods differ in their way of defining the update parameter β_k. If f is a strictly convex quadratic function, and the performed line search is exact, all these methods are equivalent, but for a general function different choices of β_k give rise to distinct conjugate gradient methods with quite different computational efficiency and convergence properties.

Hager and Zhang [25] presented an excellent survey in which the essential conjugate gradient methods are divided in two main categories. The first category includes the Fletcher-Reeves (FR) method [21], the Dai-Yuan (DY) method [16] and the Conjugate Descent (CD) method [20] with the following update parameters

$$\beta_k^{\mathrm{FR}}=\frac{\|g_{k+1}\|^2}{\|g_k\|^2},\qquad \beta_k^{\mathrm{DY}}=\frac{\|g_{k+1}\|^2}{d_k^T y_k},\qquad \beta_k^{\mathrm{CD}}=-\frac{\|g_{k+1}\|^2}{d_k^T g_k},$$

which all share the common numerator $\|g_{k+1}\|^2$ in β_k. The second category includes the Polak-Ribière (PR) method [39], the Hestenes-Stiefel (HS) method [26] and the Liu and Storey (LS) method [30] which all have the same numerator $g_{k+1}^T y_k$ in β_k. The update parameters of these methods are respectively specified as follows

$$\beta_k^{\text{PR}} = \frac{g_{k+1}^T y_k}{\|g_k\|^2}, \qquad \beta_k^{\text{HS}} = \frac{g_{k+1}^T y_k}{d_k^T y_k}, \qquad \beta_k^{\text{LS}} = -\frac{g_{k+1}^T y_k}{d_k^T g_k}.$$

The conjugate gradient methods in the first category possess strong global convergence properties [1, 17, 33] while the methods in the second category lack convergence in certain circumstances and as a result they can cycle infinitely without presenting any substantial progress [41]. However, the methods in the first category usually exhibit poor computational performance due to the jamming phenomenon [40] i.e. the algorithms can take many short steps without making significant progress to the solution. In contrast, the methods in the second category possess an automatic approximate restart procedure which avoids jamming from occurring, hence, their numerical performance is often superior to the performance of the methods with $\|g_{k+1}\|^2$ in the numerator of β_k.

In the literature, much effort has been devoted to develop new conjugate gradient methods which possess strong convergence properties and are also computationally superior to classical methods by hybridizing the above two approaches. The main idea behind the hybridization approach is to exploit the convergence properties of a conjugate gradient method from the first category and switch to a conjugate gradient from the second category when the iterations jam. Along this line, sample works include the hybridizations of FR and PR methods [22, 27, 43], the hybridizations of HS and DY methods [18, 44] and the hybridization of LS and CD methods [43]. Notice that, in these methods the update parameter is calculated based on discrete combinations of update parameters of the two categories.

In more recent works, Andrei [5–7] proposed a new class of hybrid conjugate gradient algorithms which is based on the concept of convex combination of classical conjugate gradient algorithms. Based on his numerical experiments he concluded the performance of the hybrid variants based on the concept of convex combination is better than that of the constituents. Following Andrei's approach, Babaie-Kafaki et al. [9–11, 13] proposed some globally convergent conjugate gradient methods in which the update parameter β_k is determined as the convex combination of β_k^{DY} and $\beta_k^{\text{HS+}}$, namely

$$\beta_k^{\text{HCG+}} = \lambda_k \beta_k^{\text{DY}} + (1-\lambda_k)\beta_k^{\text{HS+}}, \tag{4}$$

with $\beta_k^{\text{HS+}} = \max\{\beta_k^{\text{HS}}, 0\}$ and the scalar $\lambda_k \in [0,1]$ is the hybridization parameter. Notice that if $\lambda_k = 0$, then $\beta_k^{\text{HCG+}} = \beta_k^{\text{HS+}}$ and if $\lambda_k = 1$, then $\beta_k^{\text{HCG+}} = \beta_k^{\text{DY}}$. Moreover, the authors presented some promising numerical results in case the hybrization parameter is adaptively calculated by

$$\lambda_k = -2\frac{\|y_k\|^2}{s_k^T y_k}\frac{s_k^T g_{k+1}}{g_k^T g_{k+1}}, \tag{5}$$

which is based on a modified secant equation.

Motivated by the computational efficiency of the HCG+ method, we propose a new adaptive choice for the computation of the parameter λ_k based on a quasi-Newton philosophy. More specifically, the value of the hybrization parameter is obtained by minimizing the distance between the hybrid conjugate gradient direction and the self-scaling memoryless BFGS direction.

The remainder of this paper is organized as follows. In Section 2, we present our new hybrid conjugate gradient method and in Section 3, we report the numerical experiments using the performance profiles of Dolan and Morè. Finally, Section 4 presents our concluding remarks.

2 MODIFIED HYBRID CONJUGATE GRADIENT METHOD

2.1 Self-scaling memoryless BFGS

The self-scaling memoryless BFGS (SSML-BFGS) method is generally considered as one of the most efficient method for solving large-scale optimization problems [8, 28, 33] due to its strong theoretical properties and favorable computational performance. Moreover, it provides a good understanding about the relationship between nonlinear conjugate gradient methods and quasi-Newton methods [4, 38, 42].

Generally, the SSML-BFGS matrices are computed based on the L-BFGS philosophy [29, 32] using information from the most recent iteration. Given an initial matrix $B_0 = \theta I$, $\theta \in \mathbb{R}\backslash\{0\}$, and the BFGS update formula

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k},$$

the resulting minimal memory BFGS update scheme takes the form

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

where $\theta_k \in \mathbb{R}\backslash\{0\}$ is the scaling parameter. Additionally, the search direction in this method is generated by

$$d_{k+1} = -B_{k+1}^{-1} g_{k+1},$$

where B_{k+1}^{-1} is the inverse of Hessian approximation which can be easily calculated by the following expression [32]

$$B_{k+1}^{-1} = \frac{1}{\theta_k} I - \frac{1}{\theta_k}\frac{s_k y_k^T + y_k s_k^T}{s_k^T y_k} + \left(1 + \frac{1}{\theta_k}\frac{\|y_k\|^2}{s_k^T y_k}\right)\frac{s_k s_k^T}{s_k^T y_k}. \tag{6}$$

It is pointed out by many researchers [2, 3, 31, 34–37] that efficiency of the SSML-BFGS is heavily depended on the selection of the scaling parameter θ_k. The idea behind scaling is to achieve an ideal distribution of the eigenvalues of update formula (6), improving its condition number and consequently increasing the numerical stability of the method [33]. Based on the analysis of quadratic objective functions, there have been proposed two very popular and effective adaptive formulae for the computation of θ_k. The first one has been proposed by Oren and Luenberger [36]

$$\theta_k^{\text{OL}} = \frac{s_k^T y_k}{\|s_k\|^2}, \tag{7}$$

while the second one by Oren and Spedicato [37]

$$\theta_k^{\text{OS}} = \frac{\|y_k\|^2}{s_k^T y_k}. \tag{8}$$

However, Nocedal and Yuan [34] reported some very disappointing numerical experiments in which the best self-scaling BFGS algorithm of Oren and Luenberger [36] performs badly compared to the classical BFGS algorithm when applied with inexact line search to a simple quadratic function of two variables. Later, Al-Baali [2] presented a globally and superlinearly convergent with inexact line search under the additional condition that

$$\theta_k \leq 1. \tag{9}$$

Condition (9) is motivated by the fact that the eigenvalues of Hessian approximation B_{k+1} can be reduced if $\theta_k < 1$ and hence smaller eigenvalues are introduced in B_{k+1} if the eigenvalues of B_k are large. Numerical evidences [2, 3] presented that the performance of the self-scaling BFGS was improved substantially and concluded that the proposed scaled method was computationally superior to the original one.

For more choices and information on scalar θ_k, we refer to [2, 3, 31, 34–37] and the references therein.

2.2 New hybrization parameter based on the SSML-BFGS update

Motivated by the computational efficiency of the SSML-BFGS, we propose and an adaptive choice for parameter λ_k following a similar methodology of that in [12, 15]. More specifically, we consider to define parameter λ_k in such a way to reduce the distance between the search direction matrix of the HCG+ and the SSML-BFGS direction.

For this purpose, following Perry's point of view, it is notable that from (3) and (4), the search direction of the HCG+ method can be written as

$$d_{k+1} = -Q_{k+1}g_{k+1}, \tag{10}$$

where

$$Q_{k+1} = I - \lambda_k \frac{d_k g_{k+1}^T}{d_k^T y_k} - (1-\lambda_k)\frac{d_k y_k^T}{d_k^T y_k}.$$

Therefore, the HCG+ method can be considered as a quasi-Newton method [17, 33] in which the inverse Hessian is approximated by the nonsymmetric matrix Q_{k+1}. Subsequently, based on the above discussion, we consider to compute parameter λ_k as the solution of the following minimization problem

$$\min_{\lambda_k > 0} \|D_{k+1}\|_F \tag{11}$$

where $D_{k+1} = Q_{k+1}^T - B_{k+1}^{-1}$ and $\|\cdot\|_F$ is the Frobenius matrix norm. Since $\|D_{k+1}\|_F^2 = \mathrm{tr}(D_{k+1}^T D_{k+1})$ and after some algebra, we obtain

$$\|D_{k+1}\|_F^2 = \alpha\lambda_k^2 + \beta\lambda_k + \gamma.$$

where

$$\alpha = \frac{\|s_k\|^2\|g_k\|^2}{(s_k^T y_k)^2}$$

$$\beta = -2\left[\frac{s_k^T g_k}{s_k^T y_k} + \left(\frac{1}{\theta_k}-1\right)\frac{\|s_k\|^2(y_k^T g_k)}{(s_k^T y_k)^2} - \left(1+\frac{1}{\theta_k}\frac{\|y_k\|^2}{s_k^T y_k}\right)\frac{\|s_k\|^2(s_k^T g_k)}{(s_k^T y_k)^2}\right]$$

and γ is a real constant, independent of λ_k. Clearly, the computation of $\|D_{k+1}\|_F^2$ can be considered as a second degree polynomial of variable λ_k where the coefficient of λ_k^2 is always positive. Therefore, the unique solution of the minimization problem (11) is given by

$$\lambda_k^* = \frac{(s_k^T g_k)(s_k^T y_k)}{\|g_k\|^2\|s_k\|^2} + \left(\frac{1}{\theta_k}-1\right)\frac{y_k^T g_k}{\|g_k\|^2} - \left(1+\frac{1}{\theta_k}\frac{\|y_k\|^2}{s_k^T y_k}\right)\frac{s_k^T g_k}{\|g_k\|^2}. \tag{12}$$

Clearly, an important property of the value of λ_k^* is that matrix Q_{k+1} is as close as possible to the SSML-BFGS matrix (6). Moreover, in order to have a convex combination in (4), we restrict the values of λ_k in the interval $[0, 1]$, namely if $\lambda_k^* < 0$ then we set $\lambda_k^* = 0$ and also, if $\lambda_k^* > 1$ then we set $\lambda_k^* = 1$.

3 EXPERIMENTAL RESULTS

In this section, we compare the numerical performance of our proposed modified hybrid conjugate gradient method, denoted MHCG+ to CG-DESCENT method [23] and HCG+ method [10] on a set of 134 problems from the CUTEr [14]. The implementation code was written in C on a 2.66GHz Quad-Core processor with 4GB RAM running Linux operating system. In our experiments, we use the condition $\|g_k\|_\infty \leq 10^{-6}$ as stopping criterion and all algorithms were implemented with the same line search as CG-DESCENT [23] with its default parameters. The detailed numerical results can be found in http://www.math.upatras.gr/~livieris/Results/MHCG.zip.

Efficiency comparisons were made utilizing the performance profile introduced by Dolan and Morè [19], on the total number of function and gradient evaluations being equal to $N_f + 3N_g$ [24], where N_f and N_g denote the number of function and gradient evaluations, respectively and on the CPU time (in seconds). The use of profiles provide a wealth of information such as solver efficiency, robustness and probability of success in compact form and eliminate the influence of a small number of problems on the benchmarking process and the sensitivity of results associated with the ranking of solvers [19]. The performance profile plots the fraction P of problems for which any given method is within a factor τ of the best solver. The horizontal axis of the figure gives the percentage of the test problems for which a method is the fastest (efficiency); while the vertical axis gives the percentage of the test problems that were successfully solved by each method (robustness). The top curve is the method that solved the most problems in a time that was within a factor τ of the best time. The curves in the following figures have the following meaning:

- "MHCG$_1^+$" stands for the conjugate gradient method with the update parameter $\beta_k^{\mathrm{HCG+}}$ in which λ_k is defined by (12) and $\theta_k = \min\left\{\theta_k^{\mathrm{OL}}, 1\right\}$.
- "MHCG$_2^+$" stands for the conjugate gradient method with the update parameter $\beta_k^{\mathrm{HCG+}}$ iin which λ_k is defined by (12) and $\theta_k = \min\left\{\theta_k^{\mathrm{OS}}, 1\right\}$.
- "CG-DESCENT" stands for the CG-DESCENT method (version 5.3) [23].
- "HCG+" stands for the conjugate gradient method with the update parameter $\beta_k^{\mathrm{HCG+}}$ in which λ_k is defined by (5) as in [10].

Notice that although the descent property may not always hold for the methods of HCG+, MHCG$_1^+$ and MHCG$_2^+$, uphill search direction seldom occurred in our experiments; when occurred, we restarted the algorithm with $d_k = -g_k$ as in [10].

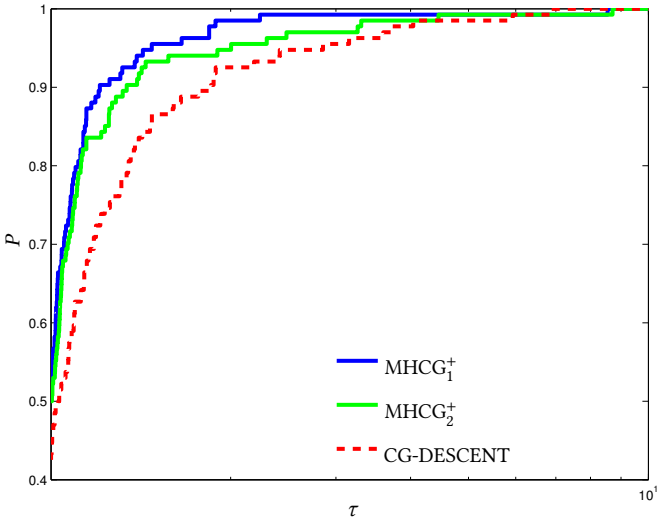

Figure 1: Log_{10} **scaled performance profiles for MHCG$_1^+$, MHCG$_2^+$ and CG-DESCENT based on total number of function and gradient evaluations**

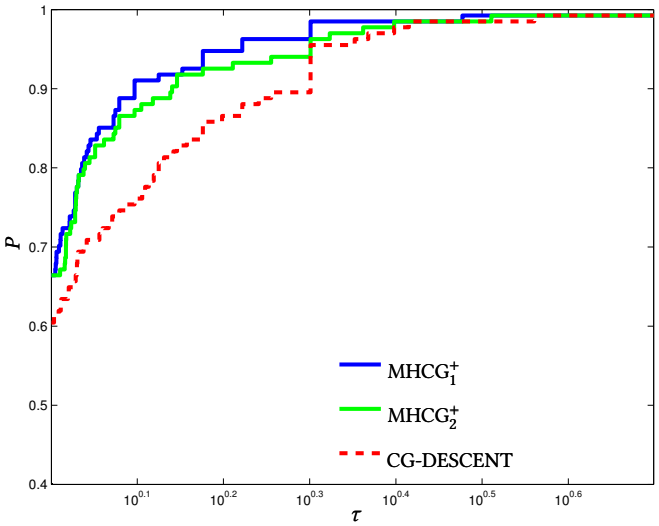

Figure 2: Log_{10} **scaled performance profiles for MHCG$_1^+$, MHCG$_2^+$ and CG-DESCENT based on CPU time**

Figure 1 presents the performance profiles of MHCG$_1^+$, MHCG$_2^+$ and CG-DESCENT in the perspectives of the total number of function and gradient evaluations. MHCG$_1^+$ and MHCG$_2^+$ solve about 50.7% and 49.3% of the test problems with the least number of function and gradient evaluations, respectively while CG-DESCENT solves about 42.5% of the test problems. As regards the CPU time, our proposed methods illustrate the highest probability of being the optimal solver since they corresponds to the top curves. The interpretation of Figure 2 illustrates that MHCG$_1^+$ and MHCG$_2^+$ exhibit the best performance with respect to CPU time since they both solve (66.5%) 89 out of 134 of the test problems, with the least computational time while CG-DESCENT solves only (60.4%) 81 out of 134 test problems. Based on the above observations, we conclude that both our proposed methods outperform CG-DESCENT in terms of efficiency and robustness, regarding all performance metrics.

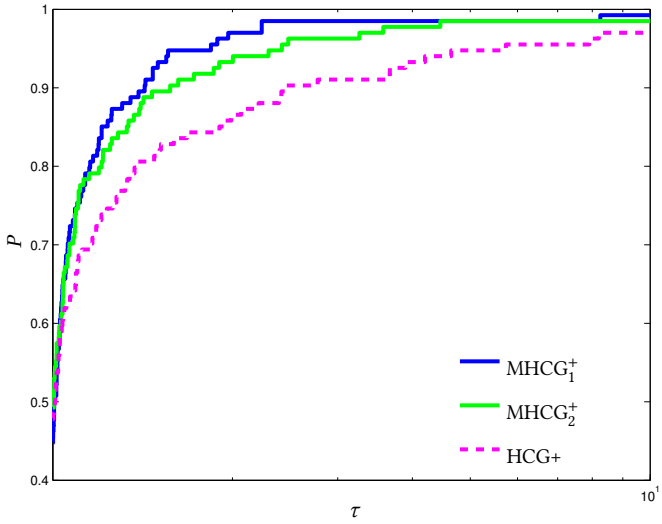

Figure 3: Log_{10} **scaled performance profiles for MHCG$_1^+$, MHCG$_2^+$ and HCG+ based on total number of function and gradient evaluations**

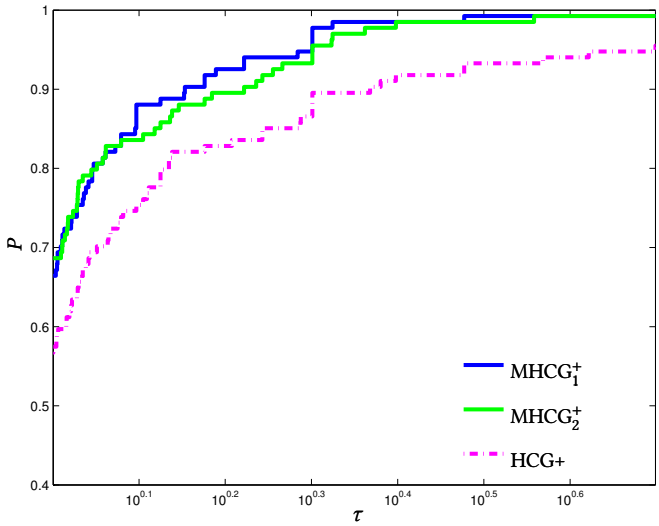

Figure 4: Log_{10} **scaled performance profiles for MHCG$_1^+$, MHCG$_2^+$ and HCG+ based on CPU time**

Figures 3 and 4 present the performance profiles of MHCG$_1^+$, MHCG$_2^+$ and HCG+, relative to both performance metrics. Clearly,

both our proposed methods outperform HCG+ illustrating the highest probability of being the most robust solver with $MHCG_2^+$ presenting slightly better performance. More specifically, Figure 3 reports that $MHCG_2^+$ solves 49.3% of the test problems with the least computational cost while HCG+ solves 45.5%. Furthermore, the interpretation of Figure 4 shows that $MHCG_1^+$ and $MHCG_2^+$ solve about 66.4% and 68.7% of the test problems, respectively with the least CPU time while HCG+ solves about 56.7% of the test problems.

Since all conjugate gradient methods have been implemented with the same line search, we conclude that our proposed methods generate the best search directions on average.

4 CONCLUSIONS

In this work, we presented a new conjugate gradient method incorporating the approach of the hybridization of the update parameters of DY and HS+ convexly in which the computation of the hybrization parameter is based on a quasi-Newton philosophy. More specifically, the value of the parameter is obtained by minimizing the distance between the hybrid conjugate gradient direction and the self-scaling memoryless BFGS direction. Numerical comparisons have been made between our proposed method and the classical conjugate gradient methods CG-DESCENT [23] and the HCG+ [10] on a set of unconstrained optimization problems of the CUTEr collection. The reported numerical results illustrated the computational efficiency and robustness of our proposed method.

REFERENCES

[1] M. Al-Baali. 1985. Descent property and global convergence of the Fletcher-Reeves method with inexact line search. *IMA J. Numer. Anal.* 5 (1985), 121–124.

[2] M. Al-Baali. 1998. Analysis of a family self-scaling quasi-Newton methods. *Computational Optimization and Applications* 9 (1998), 191–203.

[3] M. Al-Baali. 1998. Numerical Experience with a class of self-scaling quasi-Newton algorithms. *Journal of Optimization Theory* 96 (1998), 533–553.

[4] N. Andrei. 2007. Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization. *Optimization Methods and Software* 22 (2007), 561–571.

[5] N. Andrei. 2008. Another hybrid conjugate gradient algorithm for unconstrained optimization. *Numerical Algorithms* 47 (2008), 143–156. Issue 2.

[6] N. Andrei. 2009. Hybrid conjugate gradient algorithm for unconstrained optimization. *Journal of Optimization Theory and Applications* 141 (2009), 249–264.

[7] N. Andrei. 2010. Accelerated hybrid conjugate gradient algorithm with modified secant condition for unconstrained optimization. *Numerical Algorithms* 54 (2010), 23–46. Issue 1.

[8] M.S. Apostolopoulou, D.G. Sotiropoulos, I.E. Livieris, and P. Pintelas. 2009. A memoryless BFGS neural network training algorithm. In *7th IEEE International Conference on Industrial Informatics (INDIN'09)*. 216–221.

[9] S. Babaie-Kafaki, M. Fatemi, and N. Mahdavi-Amiri. 2011. Two effective hybrid conjugate gradient algorithms based on modified BFGS updates. *Numerical Algorithms* 58 (2011), 315–331.

[10] S. Babaie-Kafaki and R. Ghanbari. 2012. Two hybrid nonlinear conjugate gradient methods based on a modified secant equation. *Optimization: A journal of mathematical programming and operations research* (2012), 1–16.

[11] S. Babaie-Kafaki and R. Ghanbari. 2015. A hybridization of the Hestenes–Stiefel and Dai–Yuan conjugate gradient methods based on a least-squares approach. *Optimization Methods and Software* 30, 4 (2015), 673–681.

[12] S. Babaie-Kafaki and R. Ghanbari. 2016. A class of adaptive Dai-Laio conjugate gradient methods based on scaled memoryless BFGS update. *4OR* (2016), 1–8.

[13] S. Babaie-Kafaki and N. Mahdavi-Amiri. 2013. Two modified hybrid conjugate gradient methods based on a hybrid secant equation. *Mathematical Modelling and Analysis* 18, 1 (2013), 32–52.

[14] I. Bongartz, A. Conn, N. Gould, and P. Toint. 1995. CUTEr: constrained and unconstrained testing environments. *ACM Transaction on Mathematical Software* 21, 1 (1995), 123–160.

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

[16] Y.H. Dai and Y.X. Yuan. 1999. A nonlinear conjugate gradient with a strong global convergence properties. *SIAM Journal of Optimization* 10 (1999), 177–182.

[17] Y.H. Dai and Y.X. Yuan. 2000. *Nonlinear conjugate gradient methods.* Shanghai Scientific and Technical Publishers, Shanghai.

[18] Y.H. Dai and Y.X. Yuan. 2001. An efficient hybrid conjugate gradient method for unconstrained optimization. *Annuals Operational Research* 103 (2001), 33–47.

[19] E. Dolan and J.J. Moré. 2002. Benchmarking optimization software with performance profiles. *Mathematical Programming* 91 (2002), 201–213.

[20] R. Fletcher. 1987. *Practical Methods of Optimization, Volume 1: Unconstrained Optimization* (1st ed.). John Wiley and Sons, New York.

[21] R. Fletcher and C.M. Reeves. 1964. Function minimization by conjugate gradients. *Computer Journal* 7 (1964), 149–154.

[22] J.C. Gilbert and J. Nocedal. 1992. Global convergence properties of conjugate gradient methods for optimization. *SIAM Journal of Optimization* 2, 1 (1992), 21–42.

[23] W.W. Hager and H. Zhang. 2005. A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM of Journal Optimization* 16 (2005), 170–192.

[24] W.W. Hager and H. Zhang. 2006. CG DESCENT: A conjugate gradient method with guarantee descent and an efficient line search. *ACM Trans. Math. Software* 32 (2006), 113–137.

[25] W.W. Hager and H. Zhang. 2006. A Survey of Nonlinear Conjugate Gradient Methods. *Pacific of Journal Optimization* 2 (2006), 35–58.

[26] M.R. Hestenes and E. Stiefel. 1952. Methods for conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Standards* 49 (1952), 409–436.

[27] Y.F. Hu and C. Storey. 1991. Global convergence result for conjugate gradient methods. *Journal of Optimization Theory and Applications* 71 (1991), 399–405.

[28] C.X. Kou and Y. H. Dai. 2014. A Modified Self-Scaling Memoryless Broyden-Fletcher-Goldfarb-Shanno Method for Unconstrained Optimization. *Journal of Optimization Theory and Applications* (2014).

[29] D.C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization methods. *Mathematical Programming* 45 (1989), 503–528.

[30] Y. Liu and C. Storey. 1991. Efficient generalized conjugate gradient algorithms, part 1: theory. *Journal of Optimization Theory and Applications* 69 (1991), 129–137.

[31] E. Spedicato M. Al-Baali and F. Maggioni. 2014. Broyden's quasi-newton methods for a nonlinear system of equations and unconstrained optimization: a review and open problems. *Optimization Methods and Software* 29, 5 (2014), 937–954.

[32] J. Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Mathematical Computing* 35, 151 (1980), 773–782.

[33] J. Nocedal and S. J. Wright. 1999. *Numerical Optimization.* Springer-Verlag, New York.

[34] J. Nocedal and Y. Yuan. 1993. Analysis of a self-scaling quasi-Newton method. *Mathematical Programming* 61 (1993), 19–37.

[35] S.S. Oren. 1972. *Self-scaling variable metric algorithms for unconstrained minimization.* Ph.D. Dissertation. Stanford University, California, USA.

[36] S.S. Oren and D.G. Luenberger. 1974. Self-scaling variable metric (SSVM) algorithms, Part I: Criteria and sufficient conditions for scaling a class of algorithms. *Management Science* 20 (1974), 845–862.

[37] S.S. Oren and E. Spedicato. 1976. Optimal conditioning of self-scaling variable metric algorithms. *Mathematical Programming* 10 (1976), 70–90.

[38] J.M. Perry. 1977. A class of conjugate gradient algorithms with a two-step variable-metric memory. *Center for Mathematical Studies in Economies and Management Science. Evanston Illiois: Northwestern University Press* (1977).

[39] E. Polak and G. Ribière. 1969. Note sur la convergence de methods de directions conjuguees. *Revue Francais d'Informatique et de Recherche Operationnelle* 16 (1969), 35–43.

[40] M.J.D. Powell. 1977. Restart Procedures for the Conjugate Gradient Method. *Mathematical Programming* 12 (1977), 241–254.

[41] M.J.D. Powell. 1984. Nonconvex minimization calculations and the conjugate gradient method. In *Numerical Analysis.* Lecture Notes in Mathematics, Vol. 1066. Springer Verlag, Berlin, 122–141.

[42] D.F. Shanno. 1978. On the convergence of a new conjugate gradient algorithm. *SIAM J. Numer. Anal.* 15, 6 (1978), 1247–1257.

[43] D. Touati-Ahmed and C. Storey. 1990. Efficient hybrid conjugate gradient techniques. *Journal of Optimization Theory and Applications* 64 (1990), 379–397.

[44] L. Zhang and W. Zhou. 2008. Two descent hybrid conjugate gradient methods for optimization. *J. Comput. Appl. Math.* 216 (2008), 251–164.