Scaled Conjugate Gradient Algorithms for Unconstrained Optimization

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

Abstract. In this work we present and analyze a new scaled conjugate gradient algorithm and its implementation, based on an interpretation of the secant equation and on the inexact Wolfe line search conditions. The best spectral conjugate gradient algorithm SCG by Birgin and Martínez [3], which is mainly a scaled variant of Perry's [13], is modified in such a manner to overcome the lack of positive definiteness of the matrix defining the search direction. This modification is based on the quasi-Newton BFGS updating formula. The computational scheme is embedded in the restart philosophy of Beale-Powell. The parameter scaling the gradient is selected as spectral gradient or in an anticipative manner by means of a formula using the function values in two successive points. In very mild conditions it is shown that, for strongly convex functions, the algorithm is global convergent. Preliminary computational results, for a set consisting of 500 unconstrained optimization test problems, show that this new scaled conjugate gradient algorithm substantially outperforms the spectral conjugate gradient SCG algorithm.

Keywords: Unconstrained optimization, conjugate gradient method, spectral gradient method, BFGS formula, numerical comparisons
AMS Subject Classification: 49M07, 49M10, 90C06, 65K

1. Introduction

The conjugate gradient methods represent an important innovation for solving large-scale unconstrained optimization problems

$$min\, f(x), \tag{1}$$

where $f:R^n \to R$ is continuously differentiable and its gradient is available. These methods generate a sequence x_k of approximations to the minimum x^* of f, in which

$$x_{k+1} = x_k + \alpha_k d_k, \tag{2}$$
$$d_{k+1} = -g_{k+1} + \beta_k d_k, \tag{3}$$

where $g_k = \nabla f(x_k)$, α_k is selected to minimize $f(x)$ along the search direction d_k, and β_k is a scalar parameter. The iterative process is initialized with an initial point x_0 and $d_0 = -g_0$. A lot of versions of conjugate gradient methods which correspond to the selection procedure of parameter β_k are already known. When the function f is quadratic and α_k is selected to minimize $f(x)$ along the direction d_k, then all choices of β_k are equivalent, but for general nonlinear functions different choices of β_k give algorithms with very different convergence performances. A history of conjugate gradient and Lanczos

[1] The author was awarded the Romanian Academy Grant 168/2003.

algorithms from their very beginning until 1976 is presented by Golub and O'Leary [9]. An excellent survey of nonlinear conjugate gradient methods with special attention to global convergence properties is made by Hager and Zhang [11].

This paper is motivated by a variant of the conjugate gradient algorithm, called spectral conjugate gradient (SCG), given by Birgin and Martínez [3]. Preserving the nice geometrical properties of Perry's direction, Birgin and Martínez present a conjugate gradient algorithm in which the parameter scaling the gradient defining the search direction is selected by means of a spectral formula suggested for the first time by Barzilai and Borwein [2]. Numerical experiments with this algorithm proved that this computational scheme outperforms Polak-Ribière and Fletcher-Reeves and is competitive with CONMIN of Shanno and Phua [19] and SGM of Raydan [16]. Another recent conjugate gradient scheme related to the Perry/Shanno method is CG_DESCENT of Hager and Zhang [10].

In this paper we modify the best algorithm of Birgin and Martínez in order to overcome the lack of positive definiteness of the matrix defining the search direction. This is done using the quasi-Newton BFGS updating philosophy, thus obtaining a new descent direction. Using the restart technology of Beale-Powell we get a new scaled conjugate gradient algorithm in which the scaling parameter is selected as spectral gradient or in an anticipative manner using the function values in two successive points. The algorithm implements both Wolfe line search conditions.

The paper is organized as follows: In section 2 we present the scaled conjugate gradient method with restart. A complete description of the scaled conjugate gradient algorithm is shown in section 3. The algorithm performs two types of steps: a normal one in which a double quasi-Newton updating scheme is used, and a restart one where the current information is used to define the search direction. The convergence analysis of the algorithm for strongly convex functions is described in section 4. In section 5 we present preliminary computational results on a set of 500 unconstrained optimization test problems and compare our algorithm with SCG algorithm by Birgin and Martínez [3].

2. Scaled conjugate gradient method with restart

For solving (1) we consider the iterative process (2), where for $k = 0,1,\ldots$ the stepsize α_k is positive and the directions d_k are generated by:

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \beta_k s_k, \tag{4}$$

in which θ_{k+1} and β_k are parameters which are to be determined.

Observe that if $\theta_{k+1} = 1,$ then we get the classical conjugate gradient algorithms according to the value of the scalar parameter β_k. On the other hand, if $\beta_k = 0,$ then we get another class of algorithms according to the selection of the parameter θ_{k+1}. There are two possibilities for θ_{k+1} : a positive scalar or a positive definite matrix. If $\theta_{k+1} = 1$ we have the steepest descent (Cauchy [5]) algorithm. If $\theta_{k+1} = \nabla^2 f(x_{k+1})^{-1},$ or an approximation of it, then we get the Newton or the quasi-Newton algorithms, respectively. Therefore, we can see that in the general case, when $\theta_{k+1} \neq 0$ is selected in a quasi-Newton manner and $\beta_k \neq 0,$ then (4) represents a combination between the quasi-Newton and the conjugate gradient methods.

Using a geometric interpretation for the quadratic function minimization Birgin and Martínez [3] suggest the following expression for parameter β_k in (4):

$$\beta_k = \frac{(\theta_{k+1} y_k - s_k)^T g_{k+1}}{y_k^T s_k}. \tag{5}$$

With this, the corresponding direction is:

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \frac{(\theta_{k+1} y_k - s_k)^T g_{k+1}}{y_k^T s_k} s_k. \tag{6}$$

The following particularizations are obvious. If $\theta_{k+1} = 1,$ then (6) is the direction considered by Perry [13]. At the same time we see that (6) is the direction given by Dai and Liao [6] for $t = 1.$ Additionally, if $s_j^T g_{j+1} = 0, \quad j = 0,1,\ldots,k,$ then from (6) we get:

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \frac{\theta_{k+1} y_k^T g_{k+1}}{\alpha_k \theta_k g_k^T g_k} s_k, \tag{7}$$

which is the direction corresponding to a generalization of the Polak and Ribière formula. Of course, if $\theta_{k+1} = \theta_k = 1$ in (7), we get the classical Polak and Ribière formula [14]. If $s_j^T g_{j+1} = 0, \quad j = 0,1,\ldots,k,$ and additionally the successive gradients are orthogonal, then from (6)

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \frac{\theta_{k+1} g_{k+1}^T g_{k+1}}{\alpha_k \theta_k g_k^T g_k} s_k, \tag{8}$$

which is the direction corresponding to a generalization of the Fletcher and Reeves formula. Therefore, (6) is a general formula for direction computation in a conjugate gradient manner including the classical Fletcher and Reeves [8], and Polak and Ribière [14] formulas. Computational experiments given by Birgin and Martínez, with a spectral gradient selection choice of parameter θ_{k+1}, on a set of 40 unconstrained optimization problems, show that the algorithm (6) of Perry outperforms variant (7) of Polak and Ribière and variant (8) of Fletcher and Reeves and compare favourable with CONMIN computational scheme of Shanno and Phua [19].

Shanno [17, 18] proved that the conjugate gradient method is exactly the BFGS quasi-Newton method where at every step the approximation to the inverse Hessian is restarted as the identity matrix. Now we extend this result for the scaled conjugate gradient. We see that the direction given by (6) can be written as:

$$d_{k+1} = -\left[\theta_{k+1} I - \theta_{k+1} \frac{s_k y_k^T}{y_k^T s_k} + \frac{s_k s_k^T}{y_k^T s_k}\right] g_{k+1} \equiv -Q_{k+1} g_{k+1}, \tag{9}$$

where

$$Q_{k+1} = \theta_{k+1} I - \theta_{k+1} \frac{s_k y_k^T}{y_k^T s_k} + \frac{s_k s_k^T}{y_k^T s_k}. \tag{10}$$

If $\theta_{k+1} = 1,$ we have:

$$d_{k+1} = -\left[I - \frac{s_k y_k^T}{y_k^T s_k} + \frac{s_k s_k^T}{y_k^T s_k}\right] g_{k+1}, \tag{11}$$

which is exactly the Perry formula. By direct computation we can prove:

Proposition 1.

$$y_k^T Q_{k+1} = s_k^T. \ \blacksquare \tag{12}$$

Observe that (12) is similar but not identical to the quasi-Newton equation, which requires that an update of the inverse Hessian H_{k+1} should be in such a way as to satisfy:

$$H_{k+1} y_k = s_k. \tag{13}$$

A major difficulty with (9) is that the matrix Q_{k+1}, defined by (10), is not symmetric and hence not positive definite. Thus, the directions d_{k+1} from (9) are not necessarily descent directions and therefore numerical instability can result. Besides, another difficulty arising from this lack of symmetry is that the true quasi-Newton equation (13) is not satisfied.

In order to overcome this difficulty and to get a true quasi-Newton updating we first make the matrix Q_{k+1} from (10) symmetric as follows:

$$Q_{k+1} = \theta_{k+1} I - \theta_{k+1} \frac{s_k y_k^T + y_k s_k^T}{y_k^T s_k} + \frac{s_k s_k^T}{y_k^T s_k}. \quad (14)$$

Now, we force Q_{k+1} to satisfy the quasi-Newton equation (13) yielding to the following symmetric update:

$$Q_{k+1}^* = \theta_{k+1} I - \theta_{k+1} \frac{y_k s_k^T + s_k y_k^T}{y_k^T s_k} + \left[1 + \theta_{k+1} \frac{y_k^T y_k}{y_k^T s_k}\right] \frac{s_k s_k^T}{y_k^T s_k}. \quad (15)$$

By direct computation it is very easy to prove that Q_{k+1}^* satisfies the quasi-Newton equation, i.e.

Proposition 2.

$$Q_{k+1}^* y_k = s_k. \ \blacksquare \quad (16)$$

Notice that

$$d_{k+1} = -Q_{k+1}^* g_{k+1} \quad (17)$$

does not actually require the matrix Q_{k+1}^*, i.e. the direction d_{k+1} can be computed as:

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \theta_{k+1} \left(\frac{g_{k+1}^T s_k}{y_k^T s_k}\right) y_k - \left[\left(1 + \theta_{k+1} \frac{y_k^T y_k}{y_k^T s_k}\right) \frac{g_{k+1}^T s_k}{y_k^T s_k} - \theta_{k+1} \frac{g_{k+1}^T y_k}{y_k^T s_k}\right] s_k, \quad (18)$$

involving only 4 scalar products. Again observe that if $g_{k+1}^T s_k = 0,$ then (18) reduces to:

$$d_{k+1} = -\theta_{k+1} g_{k+1} + \theta_{k+1} \frac{g_{k+1}^T y_k}{y_k^T s_k} s_k. \quad (19)$$

Thus, in this particular case the effect is simply one of multiplying the Hestenes and Stiefel [12] search direction by a positive scalar.

As we know, the BFGS update to the inverse Hessian, which is currently the best update of the Broyden class, is defined by:

$$H_{k+1} = H_k - \frac{H_k y_k s_k^T + s_k y_k^T H_k}{y_k^T s_k} + \left[1 + \frac{y_k^T H_k y_k}{y_k^T s_k}\right] \frac{s_k s_k^T}{y_k^T s_k}. \quad (20)$$

Therefore, we can immediately see that the conjugate gradient method (17), where Q_{k+1}^* is given by (15), is exactly the BFGS quasi-Newton method, where at every step the approximation of the inverse Hessian is restarted as the identity matrix multiplied by the scalar θ_{k+1}.

In order to ensure the convergence of the algorithm (2), with d_{k+1} given by (18), we need to constrain the choice of α_k. We consider line searches that satisfy the Wolfe conditions [20, 21]:

$$f(x_k + \alpha_k d_k) - f(x_k) \le \sigma_1 \alpha_k g_k^T d_k, \quad (21)$$

$$\nabla f(x_k + \alpha_k d_k)^T d_k \ge \sigma_2 g_k^T d_k, \quad (22)$$

where $0 < \sigma_1 \le \sigma_2 < 1.$

Theorem 1. *Suppose that α_k in (2) satisfies the Wolfe conditions (21) and (22), then the direction d_{k+1} given by (18) is a descent direction.*

Proof: Since $d_0 = -g_0,$ we have $g_0^T d_0 = -\|g_0\|^2 \le 0.$ Multiplying (18) by $g_{k+1}^T,$ we have

$$g_{k+1}^T d_{k+1} = \frac{1}{(y_k^T s_k)^2} \left[-\theta_{k+1} \|g_{k+1}\|^2 (y_k^T s_k)^2 + 2\theta_{k+1} (g_{k+1}^T y_k)(g_{k+1}^T s_k)(y_k^T s_k)\right.$$

$$-(g_{k+1}^T s_k)^2 (y_k^T s_k) - \theta_{k+1}(y_k^T y_k)(g_{k+1}^T s_k)^2\Big].$$

Applying the inequality $u^T v \le \frac{1}{2}(\|u\|^2 + \|v\|^2)$ to the second term of the right hand side of the above equality, with $u = (s_k^T y_k) g_{k+1}$ and $v = (g_{k+1}^T s_k) y_k$ we get:

$$g_{k+1}^T d_{k+1} \le -\frac{(g_{k+1}^T s_k)^2}{y_k^T s_k}. \tag{23}$$

But, by Wolfe condition (22), $y_k^T s_k > 0.$ Therefore, $g_{k+1}^T d_{k+1} < 0$ for every $k = 0,1,\ldots$ ■

Observe that the second Wolfe condition (22) is crucial for the descent character of direction (18). Besides, the estimation (23) is independent of the parameter θ_{k+1}.

Usually, all conjugate gradient algorithms are periodically restarted. The standard restarting point occurs when the number of iterations is equal to the number of variables, but some other restarting methods can be considered as well. The Powell restarting procedure [15] is to test if there is very little orthogonality left between the current gradient and the previous one. At step r when:

$$\left|g_{r+1}^T g_r\right| \ge 0.2\left\|g_{r+1}\right\|^2, \tag{24}$$

we restart the algorithm using the direction given by (18). Another restarting procedure, considered by Birgin and Martínez [3], consists of testing if the angle between the current direction and $-g_{k+1}$ is not acute enough. Therefore, at step r when:

$$d_r^T g_{r+1} > -10^{-3}\left\|d_r\right\|_2\left\|g_{r+1}\right\|_2, \tag{25}$$

the algorithm is restarted using the direction given by (18).

At step r when one of the two criteria (24) or (25) is satisfied, the direction is computed as in (18). For $k \ge r+1,$ we consider the same philosophy used to get (15), i.e. that of modifying the gradient g_{k+1} with a positive definite matrix which best estimates the inverse Hessian without any additional storage requirements. Therefore, the direction d_{k+1}, for $k \ge r+1,$ is computed using a double update scheme as:

$$d_{k+1} = -H_{k+1} g_{k+1}, \tag{26}$$

where

$$H_{k+1} = H_{r+1} - \frac{H_{r+1} y_k s_k^T + s_k y_k^T H_{r+1}}{y_k^T s_k} + \left[1 + \frac{y_k^T H_{r+1} y_k}{y_k^T s_k}\right]\frac{s_k s_k^T}{y_k^T s_k}. \tag{27}$$

and

$$H_{r+1} = \theta_{r+1} I - \theta_{r+1}\frac{y_r s_r^T + s_r y_r^T}{y_r^T s_r} + \left(1 + \theta_{r+1}\frac{y_r^T y_r}{y_r^T s_r}\right)\frac{s_r s_r^T}{y_r^T s_r}. \tag{28}$$

As above, observe that this computational scheme does not involve any matrix. Indeed, $H_{r+1} g_{k+1}$ and $H_{r+1} y_k$ can be computed as:

$$v \equiv H_{r+1} g_{k+1} = \theta_{r+1} g_{k+1} - \theta_{r+1}\left(\frac{g_{k+1}^T s_r}{y_r^T s_r}\right) y_r + \left[\left(1 + \theta_{r+1}\frac{y_r^T y_r}{y_r^T s_r}\right)\frac{g_{k+1}^T s_r}{y_r^T s_r} - \theta_{r+1}\frac{g_{k+1}^T y_r}{y_r^T s_r}\right] s_r, \tag{29}$$

and

$$w \equiv H_{r+1} y_k = \theta_{r+1} y_k - \theta_{r+1}\left(\frac{y_k^T s_r}{y_r^T s_r}\right) y_r$$

$$+\left[\left(1+\theta_{r+1}\frac{y_r^T y_r}{y_r^T s_r}\right)\frac{y_k^T s_r}{y_r^T s_r}-\theta_{r+1}\frac{y_k^T y_r}{y_r^T s_r}\right]s_r, \tag{30}$$

involving 6 scalar products. With these the direction (26) at any nonrestart step can be computed as:

$$d_{k+1}=-v+\frac{(g_{k+1}^T s_k)w+(g_{k+1}^T w)s_k}{y_k^T s_k}-\left(1+\frac{y_k^T w}{y_k^T s_k}\right)\frac{g_{k+1}^T s_k}{y_k^T s_k}s_k, \tag{31}$$

involving only 4 scalar products. We see that d_{k+1} from (31) is defined as a double quasi-Newton update scheme. It is useful to note that $y_k^T s_k > 0$ is sufficient to ensure that the direction d_{k+1} given by (31) is well defined and it is always a descent direction.

We shall now consider some formulas for the computation of θ_{k+1}. As we have already seen, in our algorithm θ_{k+1} is defined as a scalar approximation of the inverse Hessian. According to the procedures for a scalar estimation of the inverse Hessian we get a family of scaled conjugate gradient algorithms. The following procedures can be used.

θ_{k+1} **spectral.** Motivated by the spectral gradient method introduced by Barzilai and Borwein [2] and analyzed by Raydan [16] and Fletcher [7], we can consider a spectral gradient choice for θ_{k+1} as:

$$\theta_{k+1}=\frac{s_k^T s_k}{y_k^T s_k}. \tag{32}$$

The parameter θ_{k+1} given by (32) is the inverse of the Rayleigh quotient. Again we notice that $y_k^T s_k > 0$ is sufficient to ensure that θ_{k+1} in (32) is well defined.

θ_{k+1} **anticipative.** Recently, Andrei [1], using the information in two successive points of the iterative process, developed another scalar approximation of the Hessian of function f obtaining a new algorithm which compares favourable with Barzilai-Borwein's. Indeed, in point $x_{k+1}=x_k+\alpha_k d_k$ we can write

$$f(x_{k+1})=f(x_k)+\alpha_k g_k^T d_k+\frac{1}{2}\alpha_k^2 d_k^T \nabla^2 f(z) d_k,$$

where z is on the line segment connecting x_k and x_{k+1}. Having in view the local character of the searching procedure and that the distance between x_k and x_{k+1} is small enough we can choose $z=x_{k+1}$ and consider γ_{k+1} as a scalar approximation of $\nabla^2 f(x_{k+1})$, where $\gamma_{k+1}\in R$. This is an anticipative viewpoint in which a scalar approximation of the Hessian at point x_{k+1} is computed using only the local information from two successive points: x_k and x_{k+1}. Therefore, we can write:

$$\gamma_{k+1}=\frac{2}{d_k^T d_k}\frac{1}{\alpha_k^2}\left[f(x_{k+1})-f(x_k)-\alpha_k g_k^T d_k\right]. \tag{33}$$

Observe that for convex functions $\gamma_{k+1}>0$. If $f(x_{k+1})-f(x_k)-\alpha_k g_k^T d_k<0$, then the reduction $f(x_{k+1})-f(x_k)$ in function value is smaller than $\alpha_k g_k^T d_k$. In these cases the idea is to change a little the stepsize α_k as $\alpha_k-\eta_k$, maintaining the other quantities at their values, in such a way so that γ_{k+1} is positive. To get a value for η_k let us select a real $\delta>0$, "small enough", but comparable with the value of the function, and take

$$\eta_k=\frac{1}{g_k^T d_k}\left[f(x_k)-f(x_{k+1})+\alpha_k g_k^T d_k+\delta\right], \tag{34}$$

with which a new value for γ_{k+1} can be computed as:

$$\gamma_{k+1} = \frac{2}{d_k^T d_k} \frac{1}{(\alpha_k - \eta_k)^2} \left[f(x_{k+1}) - f(x_k) - (\alpha_k - \eta_k) g_k^T d_k \right]. \tag{35}$$

With these, the value for parameter θ_{k+1} is selected as:

$$\theta_{k+1} = \frac{1}{\gamma_{k+1}}, \tag{36}$$

where γ_{k+1} is given by (33) or (35).

Proposition 3. *Assume that $f(x)$ is continuously differentiable and $\nabla f(x)$ is Lipschitz continuous, with a positive constant L. Then at point x_{k+1},*

$$\gamma_{k+1} \le 2L. \tag{37}$$

Proof: From (33) we have:

$$\gamma_{k+1} = \frac{2\left[f(x_k) + \alpha_k \nabla f(\xi_k)^T d_k - f(x_k) - \alpha_k \nabla f(x_k)^T d_k \right]}{\|d_k\|^2 \alpha_k^2},$$

where ξ_k is on the line segment connecting x_k and x_{k+1}. Therefore

$$\gamma_{k+1} = \frac{2\left[\nabla f(\xi_k) - \nabla f(x_k) \right]^T d_k}{\|d_k\|^2 \alpha_k}.$$

Using the inequality of Cauchy and the Lipschitz continuity it follows that

$$\gamma_{k+1} \le \frac{2\|\nabla f(\xi_k) - \nabla f(x_k)\|}{\|d_k\| \alpha_k} \le \frac{2L\|\xi_k - x_k\|}{\|d_k\| \alpha_k} \le \frac{2L\|x_{k+1} - x_k\|}{\|d_k\| \alpha_k} = 2L. \ \blacksquare$$

Therefore, from (36) we get a lower bound for θ_{k+1} as:

$$\theta_{k+1} \ge \frac{1}{2L},$$

i.e. it is bounded away from zero.

3. The algorithm

Having in view the above developments and the definitions of g_k, s_k and y_k, as well as the selection procedures for θ_{k+1} computation, the following family of scaled conjugate gradient algorithms can be presented.

Algorithm SCALCG

Step 1. Select $x_0 \in R^n$, and the parameters $0 < \sigma_1 \le \sigma_2 < 1$. Compute $f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $\alpha_0 = 1/\|g_0\|$. Set $k = 0$.

Step 2. Line search. Compute α_k satisfying the Wolfe conditions (21) and (22). Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1}), g_{k+1}$ and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step 3. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k + 1$.

Step 4. Compute θ_k using a spectral (32) or an anticipative (36) approach.

Step 5. Compute the (restart) direction d_k as in (18).

Step 6. Line search. Compute the initial guess of the step length as $\alpha_k = \alpha_{k-1} \|d_{k-1}\|_2 / \|d_k\|_2$. Using this initialization compute α_k satisfying the Wolfe

conditions (21) and (22). Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1}), g_{k+1}$ and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step 7. Store $\theta = \theta_k$, $s = s_k$ and $y = y_k$.

Step 8. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k + 1$.

Step 9. If the Powell restart criterion (24) or the angle restart criterion (25) is satisfied, then go to step 4 (a restart step); otherwise continue with step 10 (a normal step).

Step 10. Compute

$$v = \theta g_k - \theta\left(\frac{g_k^T s}{y^T s}\right) y + \left[\left(1 + \theta \frac{y^T y}{y^T s}\right)\frac{g_k^T s}{y^T s} - \theta \frac{g_k^T y}{y^T s}\right] s,$$

$$w = \theta y_k - \theta\left(\frac{y_{k-1}^T s}{y^T s}\right) y + \left[\left(1 + \theta \frac{y^T y}{y^T s}\right)\frac{y_{k-1}^T s}{y^T s} - \theta \frac{y_{k-1}^T y}{y^T s}\right] s,$$

and

$$d_k = -v + \frac{(g_k^T s_{k-1})w + (g_k^T w)s_{k-1}}{y_{k-1}^T s_{k-1}} - \left(1 + \frac{y_{k-1}^T w}{y_{k-1}^T s_{k-1}}\right)\frac{g_k^T s_{k-1}}{y_{k-1}^T s_{k-1}} s_{k-1}. \tag{38}$$

Step 11. Line search. Compute the initial guess of the step length as $\alpha_k = \alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. Using this initialization compute α_k satisfying the Wolfe conditions (21) and (22). Update the variables $x_{k+1} = x_k + \alpha_k d_k$. Compute $f(x_{k+1}), g_{k+1}$ and $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.

Step 12. Test for continuation of iterations. If this test is satisfied the iterations are stopped, else set $k = k + 1$ and go to step 9. ■

It is well known that if f is bounded below along the direction d_k, then there exists a step length α_k satisfying the Wolfe conditions. The initial selection of the step length crucially affects the practical behavior of the algorithm. At every iteration $k \geq 1$ the starting guess for the step α_k in the line search is computed as $\alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. This selection, was considered for the first time by Shanno and Phua in CONMIN [19].

4. Convergence analysis for strongly convex functions

Throughout this section we assume that f is strongly convex and Lipschitz continuous on the level set

$$L_0 = \{x \in R^n : f(x) \leq f(x_0)\}.$$

That is, there exists constants $\mu > 0$ and L such that

$$(\nabla f(x) - \nabla f(y))^T (x - y) \geq \mu \|x - y\|^2 \tag{39}$$

and

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \tag{40}$$

for all x and y from L_0. For the convenience of the reader we include here the following lemma.

Lemma 1. *Assume that d_k is a descent direction and ∇f satisfies the Lipschitz condition*

$$\|\nabla f(x) - \nabla f(x_k)\| \leq L\|x - x_k\|, \tag{41}$$

for every x on the line segment connecting x_k and x_{k+1}, where L is a constant. If the line search satisfies the second Wolfe condition (22), then

$$\alpha_k \geq \frac{1-\sigma_2}{L}\frac{\left|g_k^T d_k\right|}{\left\|d_k\right\|^2}. \tag{42}$$

Proof: Subtracting $g_k^T d_k$ from both sides of (22) and using the Lipschitz condition we have

$$(\sigma_2 - 1)g_k^T d_k \leq (g_{k+1} - g_k)^T d_k \leq L\alpha_k \left\|d_k\right\|^2. \tag{43}$$

Since d_k is a descent direction and $\sigma_2 < 1,$ (42) follows immediately from (43). ■

Lemma 2. *Assume that ∇f is strongly convex and Lipschitz continuous on L_0. If θ_{k+1} is selected by spectral gradient, then the direction d_{k+1} given by (18) satisfies:*

$$\left\|d_{k+1}\right\| \leq \left(\frac{2}{\mu} + \frac{2L}{\mu^2} + \frac{L^2}{\mu^3}\right)\left\|g_{k+1}\right\|. \tag{44}$$

Proof: By Lipschitz continuity (40) we have

$$\left\|y_k\right\| = \left\|g_{k+1} - g_k\right\| = \left\|\nabla f(x_k + \alpha_k d_k) - \nabla f(x_k)\right\| \leq L\alpha_k \left\|d_k\right\| = L\left\|s_k\right\|. \tag{45}$$

On the other hand, by strong convexity (39)

$$y_k^T s_k \geq \mu \left\|s_k\right\|^2. \tag{46}$$

Selecting θ_{k+1} as in (32), it follows that

$$\theta_{k+1} = \frac{s_k^T s_k}{y_k^T s_k} \leq \frac{\left\|s_k\right\|^2}{\mu\left\|s_k\right\|^2} = \frac{1}{\mu}. \tag{47}$$

Now, using the triangle inequality and the above estimates (45)-(47), after some algebra on $\left\|d_{k+1}\right\|,$ where d_{k+1} is given by (18), we get (44). ■

Lemma 3. *Assume that ∇f is strongly convex and Lipschitz continuous on L_0. If θ_{k+1} is selected by the anticipative procedure, then the direction d_{k+1} given by (18) satisfies:*

$$\left\|d_{k+1}\right\| \leq \left(\frac{1}{m} + \frac{2L}{m\mu} + \frac{1}{\mu} + \frac{L^2}{m\mu^2}\right)\left\|g_{k+1}\right\|. \tag{48}$$

Proof: By strong convexity on L_0, there exists the constant $m > 0$, such that $\nabla^2 f(x) \geq mI,$ for all $x \in L_0$. Therefore, for every k, $\gamma_{k+1} \geq m.$ Now, from (36) we see that, for all k,

$$\theta_{k+1} \leq \frac{1}{m}. \tag{49}$$

With this, like in lemma 2, we get (48). ■

The convergence of the scaled conjugate gradient algorithm (SCALCG) when f is strongly convex is given by

Theorem 2. *Assume that f is strongly convex and Lipschitz continuous on the level set L_0. If at every step of the conjugate gradient (2) with d_{k+1} given by (18) and the step length α_k selected to satisfy the Wolfe conditions (21) and (22), then either $g_k = 0$ for some $k,$ or $\lim_{k\to\infty} g_k = 0.$*

Proof: Suppose $g_k \neq 0$ for all $k.$ By strong convexity we have

$$y_k^T d_k = (g_{k+1} - g_k)^T d_k \geq \mu\alpha_k \left\|d_k\right\|^2. \tag{50}$$

By theorem 1, $g_k^T d_k < 0$. Therefore, the assumption $g_k \neq 0$ implies $d_k \neq 0$. Since $\alpha_k > 0$, from (50) it follows that $y_k^T d_k > 0$. But f is strongly convex over L_0, therefore f is bounded from below. Now, summing over k the first Wolfe condition (21) we have

$$\sum_{k=0}^{\infty} \alpha_k g_k^T d_k > -\infty.$$

Considering the lower bound for α_k given by (42) in lemma 1 and having in view that d_k is a descent direction it follows that

$$\sum_{k=1}^{\infty} \frac{\left| g_k^T d_k \right|^2}{\left\| d_k \right\|^2} < \infty. \tag{51}$$

Now, from (23), using the inequality of Cauchy and (46) we get

$$g_{k+1}^T d_{k+1} \leq -\frac{(g_{k+1}^T s_k)^2}{y_k^T s_k} \leq -\frac{\left\| g_{k+1} \right\|^2 \left\| s_k \right\|^2}{\mu \left\| s_k \right\|^2} = -\frac{\left\| g_{k+1} \right\|^2}{\mu}.$$

Therefore, from (51) it follows that

$$\sum_{k=0}^{\infty} \frac{\left\| g_k \right\|^4}{\left\| d_k \right\|^2} < \infty. \tag{52}$$

Now, inserting the upperbound (44), or (48), for d_k in (52) yields

$$\sum_{k=0}^{\infty} \left\| g_k \right\|^2 < \infty,$$

which completes the proof. ■

For general functions the convergence of the algorithm is coming from theorem 1 and the restart procedure. Therefore, for strongly convex functions and under inexact line search it is global convergent. If restarts are employed, the algorithm is convergent, but the speed of convergence can decrease. To a great extent, however, SCALCG algorithm is very close to Perry/Shanno computational scheme [17, 18]. In fact SCALCG is a scaled memoryless BFGS preconditioned algorithm where the scaling factor is the inverse of a scalar approximation of the Hessian. Although a global convergence result has not been established for SCALCG, recall that for the Perry/Shanno scheme, the iterates either converge to a stationary point or the iterates cycle.

5. Preliminary computational results and comparisons

In this section we present the preliminary computational performance of a Fortran implementation of the *SCALCG - scaled conjugate gradient algorithm* on a set of 500 unconstrained optimization test problems. At the same time, we compare the performance of SCALCG to *the best spectral conjugate gradient algorithm, SCG* (betatype=1, Perry-M1), by Birgin and Martínez [3]. The SCALCG code is authored by Andrei, while the SCG is co-authored by Birgin and Martínez. SCG uses spectral selection of θ_{k+1} and angle restart criterion. All codes are written in double precision Fortran using the same style of programming and compiled with f77 (default compiler settings) on an Intel Pentium 4, 1.8GHz workstation. The SCALCG code implements both the scaled conjugate gradient with spectral choice of scaling parameter θ_{k+1}, as well as with the anticipative choice of this parameter. In order to compare SCALCG with SCG we manufactured a new SCG code of Birgin and Martínez by introducing a sequence of code implementing the same stopping criteria of the algorithms used in SCALCG.

The test problems are the unconstrained problems in the CUTE [4] library, along with other large-scale optimization test problems. We selected 50 large-scale unconstrained optimization test problems in extended or generalized form. For each test function we have considered 10 numerical experiments with number of variables $n = 1000, 2000, \ldots, 10000$.

Concerning the stopping criterion used in steps 3, 8 and 12 we consider the following tests:

$$\text{c1:} \quad \|g_k\|_\infty \leq \varepsilon_g \tag{53}$$

$$\text{c2:} \quad \|g_k\|_\infty \leq max\{\varepsilon_g, \varepsilon_f \|g_0\|_\infty\}, \tag{54}$$

where $\|.\|_\infty$ denotes the maximum absolute component of a vector and $\varepsilon_g = 10^{-6}$ and $\varepsilon_f = 10^{-10}$.

In all algorithms the Wolfe line search conditions are implemented with $\sigma_1 = 0.0001$ and $\sigma_2 = 0.9$. SCALCG and SCG use exactly the same implementation of Wolfe conditions. The initial guess of the step length at the first iteration is $\alpha_0 = 1/\|g_0\|$. At the following iteration, in all algorithms, the starting guess for the step α_k is computed as $\alpha_{k-1}\|d_{k-1}\|_2 / \|d_k\|_2$. This proved to be one of the best selection of the initial guess of the step length.

The numerical results concerning the number of iterations, the number of restart iterations, the number of function and gradient evaluations, cpu time in seconds, for each of the methods are posted at the following web site:

http://www.math.ufl.edu/~coap/ (Journal Software).

In the following we present the numerical performances of these two codes and comparisons between SCALCG and SCG using the same stopping criterion. Let $f_i^{SCALCG_{cj}}$ be the optimal functional value found by SCALCG algorithm and $f_i^{SCG_{cj}}$ the optimal functional value found by SCG algorithm for test problem $i = 1,\ldots,500$ using the stoppping criterion cj, $j = 1,2$. We say that, in the particular problem i, the performance of $SCALCG_{cj}$ was better than the performance of SCG_{cj} if $\left|f_i^{SCALCG_{cj}} - f_i^{SCG_{cj}}\right| < 10^{-3}$ and the number of iterations, or the number of function-gradient evaluations, or the CPU time of $SCALCG_{cj}$ was less than the number of iterations, or the number of function-gradient evaluations, or the CPU time corresponding to SCG_{cj} respectively.

In the first set of numerical experiments we compare $SCALCG_{cj}$ with θ_{k+1} spectral (θ^s) and $SCALCG_{cj}$ with θ_{k+1} anticipative (θ^a) using the Powell restart criterion. The Tables 1 and 2 present the comparisons between $SCALCG_{cj}$ (θ^s) and $SCALCG_{cj}$ (θ^a) using stopping criterion cj, $j = 1,2$, respectively. In these Tables we find the number of problems, out of 500, for which an algorithm achieved the minimum number of iterations (#iter), or the minimum number of function-gradient evaluations (#fg) or the minimum CPU time. For example when comparing $SCALCG_{c1}$ (θ^s) and $SCALCG_{c1}$ (θ^a), using the stopping criterion $c1$, subject to the number of iterations, $SCALCG_{c1}$ (θ^s) was better in 177 problems (i.e. it achieved the minimum number of iterations in 177 problems), $SCALCG_{c1}$ (θ^a) was better in 150 problems, and they had the same number of iterations in 159 problems, etc.

Table 1. Comparisons between $SCALCG_{c1}$ (θ^s) and $SCALCG_{c1}$ (θ^a).

	$SCALCG_{c1}$ (θ^s)	$SCALCG_{c1}$ (θ^a)	=
# iter	177	150	159
# fg	194	170	122
CPU time	179	193	114

Table 2. Comparisons between $SCALCG_{c2}(\theta^s)$ and $SCALCG_{c2}(\theta^a)$.

	$SCALCG_{c2}(\theta^s)$	$SCALCG_{c2}(\theta^a)$	=
# iter	174	139	173
# fg	191	151	144
CPU time	176	182	128

In the second set of numerical experiments we compare $SCALCG_{cj}$ with θ_{k+1} anticipative (θ^a) using the Powell restart criterion and SCG_{cj} with θ_{k+1} spectral (θ^s) and angle restart criterion for $j = 1,2$. As above, in Tables 3 and 4 we find the number of problems, out of 500, for which $SCALCG_{cj}(\theta^a)$ and $SCG_{cj}(\theta^s)$ achieved the minimum number of iterations or the minimum number of function-gradient evaluations or the minimum CPU time, using the stopping criterion cj, $j = 1,2$, respectively.

Table 3. Comparisons between $SCALCG_{c1}(\theta^a)$ and $SCG_{c1}(\theta^s)$.

	$SCALCG_{c1}(\theta^a)$	$SCG_{c1}(\theta^s)$	=
# iter	349	75	51
# fg	271	126	78
CPU time	408	52	15

Table 4. Comparisons between $SCALCG_{c2}(\theta^a)$ and $SCG_{c2}(\theta^s)$.

	$SCALCG_{c2}(\theta^a)$	$SCG_{c2}(\theta^s)$	=
# iter	346	69	41
# fg	257	113	86
CPU time	394	50	12

From these Tables we see that, for these criteria, the top performer is SCALCG algorithm with anticipative selection of scaling parameter (θ^a).

6. Conclusion

The best algorithm of Birgin and Martínez, which mainly is a scaled variant of Perry's, was modified in order to overcome the lack of positive definiteness of the matrix defining the search direction. This modification takes advantage of the quasi-Newton BFGS updating formula. Using the restart technology of Beale-Powell, we get a scaled conjugate gradient algorithm in which the parameter scaling the gradient is selected as spectral gradient or in an anticipative manner by means of a formula using the function values in two successive points. Although the update formulas (18) and (29)-(31) are more complicated this computational scheme proved to be more efficient and more robust in numerical experiments. The algorithm implements the Wolfe conditions, and we prove that the steps are along the descent directions. A preliminary computational study shows that our scaled conjugate gradient algorithm performs better than Birgin and Martínez's SCG algorithm for a test set consisting of 500 problems.

References

1. N. Andrei, "*A new gradient descent method for unconstrained optimization*", ICI Technical Report, March 2004.
2. J. Barzilai and J.M. Borwein, "*Two point step size gradient method*", IMA J. Numer. Anal., 8, pp.141-148, 1988.

3. E. Birgin and J.M. Martínez, "*A spectral conjugate gradient method for unconstrained optimization*", Applied Math. and Optimization, 43, pp.117-128, 2001.
4. I. Bongartz, A.R. Conn, N.I.M. Gould and P.L. Toint, "*CUTE: constrained and unconstrained testing environments*", ACM Trans. Math. Software, 21, pp.123-160, 1995.
5. A. Cauchy, "*Méthodes générales pour la résolution des systèmes déquations simultanées*", C.R. Acad. Sci. Par., 25, pp.536-538, 1847.
6. Y.H. Dai and L.Z. Liao, "*New conjugate conditions and related nonlinear conjugate gradient methods*", Appl. Math. Optim., vol. 43 pp.87-101, 2001.
7. R. Fletcher, "*On the Barzilai-Borwein method*", Numerical Analysis Report NA/207, 2001.
8. R. Fletcher and C.M. Reeves, "*Function minimization by conjugate gradients*", Comput. J. 7, pp. 149-154, 1964.
9. G.H. Golub and D.P. O'Leary, "*Some history of the conjugate gradient and Lanczos algorithms: 1948-1976*", SIAM Rev., 31 pp.50-102, 1989.
10. W.W. Hager and H. Zhang, "*A new conjugate gradient method with guaranteed descent and an efficient line search*" SIAM Journal on Optimization, 16 (2005), 170-192.
11. W.W. Hager and H. Zhang, "*A survey of nonlinear conjugate gradient methods*", Pacific Journal of Optimization, 2 (2006), pp. 35-58.
12. M.R. Hestenes and E. Stiefel, "*Methods of conjugate gradients for solving linear systems*", J. Research Nat. Bur. Standards Sec. B. 48, pp. 409-436, 1952.
13. J.M. Perry, "*A class of conjugate gradient algorithms with a two step variable metric memory*", Discussion paper 269, Center for Mathematical Studies in Economics and Management Science, Northwestern University, 1977.
14. E. Polak and G. Ribière, "*Note sur la convergence de methods de directions conjugres*", Revue Francaise Informat. Reserche Opérationnelle 16, pp. 35-43, 1969.
15. M.J.D. Powell, "*Restart procedures for the conjugate gradient method*", Math. Programming, 12, pp.241-254, 1977.
16. M. Raydan, "*The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*", SIAM J. Optim., 7, 26-33, 1997.
17. D.F. Shanno, "*Conjugate gradient methods with inexact searches*", Mathematics of Operations Research, vol. 3, pp.244-256, 1978.
18. D.F. Shanno, "*On the convergence of a new conjugate gradient algorithm*", SIAM J. Numer. Anal. vol. 15, pp.1247-1257, 1978.
19. D.F. Shanno and K.H. Phua, *"Algorithm 500, Minimization of unconstrained multivariate functions [E4]*", ACM Trans. on Math. Soft., 2, pp.87-94, 1976.
20. P. Wolfe, "*Convergence conditions for ascent methods*", SIAM Rev., 11, pp.226-235, 1969.
21. P. Wolfe, "*Convergence conditions for ascent methods II: some corrections*", SIAM Rev. 13, pp.185-188, 1971.

February 16, 2006