A Subspace Study on Conjugate Gradient Algorithms*

J. Stoer
Institut für Angewandte Mathematik, Universität Würzburg
D-97074 Würzburg, Germany

Ya-xiang Yuan†
Computing Center, Academia Sinica
P.O. Box 2719, Beijing 10080, China

Abstract

In this paper, we analyse techniques of computing a search direction by minimizing the approximate quadratic model in the 2 dimensional subspace spanned by the current gradient and the last search direction. The classical conjugate gradient methods are only the special cases where the objective function is quadratic and line searches are exact. Based on our analyses on the case where line searches are not exact, we construct new conjugate direction type algorithms.

Key words: unconstrained optimization, conjugate gradient, subspace.

Running title: Conjugate gradient algorithms.

*This work was partially supported by the Chinese National Science Foundation Grant 18901024

†Alexander von Humblodt fellow, visiting university of Würzburg, Germany from July 1992 to June 1993

1. Introduction

Conjugate gradient algorithms for the nonlinear optimization are a class of numerical algorithms for the unconstrained optimization problem:

$$\min_{x\in\Re^n} f(x) \tag{1.1}$$

where $f(x)$ is a general nonlinear function. Normally the initial direction d_1 is given by

$$d_1 = -g_1 \tag{1.2}$$

where we use the notations $g_1 = g(x_1) = \nabla f(x_1)$. In the $k-$th iteration, a step-length α_k is calculated by a line search technique and the next iterate is set to

$$x_{k+1} = x_k + \alpha_k d_k \ . \tag{1.3}$$

The search direction for the next iteration has the following form:

$$d_{k+1} = -g_{k+1} + \beta_k d_k \ , \tag{1.4}$$

where β_k is a parameter. Two famous ways of choosing β_k are

$$\beta_k = ||g_{k+1}||_2^2/||g_k||_2^2 \ , \tag{1.5}$$

and

$$\beta_k = g_{k+1}^T y_k/||g_k||_2^2 \ , \tag{1.6}$$

where $y_k = g_{k+1} - g_k$. (1.5) was given by Fletcher and Reeves [5], and (1.6) by Polak and Ribiére [8], and Polyak [9], independently.

If (1.2) is satisfied, the objective function is a convex quadratic function

$$f(x) = \frac{1}{2}x^T Ax + b^T x + c \ , \tag{1.7}$$

and exact line searches are used, that is

$$\alpha_k = \ \arg \ \min_{\alpha\in\Re} f(x_k + \alpha d_k) \ , \tag{1.8}$$

then (1.5) and (1.6) are equivalent, and the conjugacy relation

$$d_i^T A d_j = 0 \ . \tag{1.9}$$

holds for all $i \neq j$. In fact, relation (1.9) is the original condition that was used to derive conjugate gradient algorithms, and the name "conjugate gradient" also comes from this relation. Under the conditions (1.2), (1.4), and (1.7)-(1.9), it can be shown that x_{k+1} is the minimum of the objective function in the subspace $x_k + span\{g_1, g_2, ..., g_k\}$, and $g_1, g_2, ..., g_k$ are mutually orthogonal unless that $g_k = 0$. (for example see [4]). Hence the solution will be found after at most n iterations. Some properties of conjugate gradient

methods, including that conjugate gradient algorithms generate points that minimize the objective function on subspaces, are given in [3].

If the first search direction is not the steepest descent direction, then even for quadratic functions (1.4), it was first shown by Powell [10] that the conjugate gradient method normally does not terminate within finitely many iterations. It is well known that the conjugate gradient method converges at least linearly (for example, see [7]). However, an upper bound for the rate of convergence is obtained by [13]. Therefore, unless finite termination happens within $n + 1$ iterations, the conjugate gradient method converges exactly linearly.

For general nonlinear functions, various choices of β_k give different conjugate algorithms. If the objective function is uniformly convex, then the Fletcher-Reeves method and the Polak-Rebiére-Polyak method both converge to the unique solution if exact line searches are carried out at every iteration (for example, see [7]).

Practical numerical algorithms normally make inexact line searches instead of exact line searches. For inexact line searches, a descent search direction d_k is needed, namely

$$d_k^T g_k < 0 \tag{1.10}$$

and a step-length α_k is computed to satisfy:

$$f(x_k + \alpha_k d_k) \leq f(x_k) + b_1 \alpha_k d_k^T g_k \tag{1.11}$$

and

$$d_k^T g(x_k + \alpha_k d_k) \geq b_2 d_k^T g_k \ , \tag{1.12}$$

where $0 < b_1 \leq b_2 < 1$ are two constants. A direct corollary of (1.11) is that

$$d_k^T (g_{k+1} - g_k) > 0 \tag{1.13}$$

is always true. And it is worth to mention that (1.13) does not require any convexity assumption of $f(x)$.

Without the convexity assumption of $f(x)$, Powell [12] proved the global convergence of the Fletcher-Reeves method provided exact line searches are used. Powell's results are extended to inexact line searches such that α_k satisfies (1.11) and

$$|d_k^T g(x_k + \alpha_k d_k)| \leq -b_2 d_k^T g_k \ , \tag{1.14}$$

with $b_2 \in [b_1, 0.5)$ (see, [1]). Condition (1.14) is slightly stronger than (1.12), but there exist techniques to compute α_k that satisfies (1.11) and (1.14) (for example, see [4]).

Conjugate gradient methods are based on the conjugacy condition (1.9). A main reason for constructing conjugate directions is that to minimize a convex quadratic function in a subspace spanned by a set of mutually conjugate directions is equivalent to minimize the objective function along each conjugate direction in turn. Hence it is not unexpected that the performance of conjugate gradient algorithms is dependent on the accuracy of the line searches. It can be said that conjugate direction methods are so constructed

that exact line searches are expected, though in real computations normally inexact line searches are used.

Actually, the good property of conjugate directions, namely that the minimization over a multi-dimensional subspace is equivalent to that over each conjugate directions in turn, is only true for exact line searches. Hence when line searches are not exact, the conjugacy property may have disadvantages. Suppose we minimize the convex quadratic function (1.7) on a subspace spanned by a set of mutually conjugate directions $\{d_1, ..., d_k\}$. Suppose that the line search along d_1 is not exact, that is $\alpha_1 \neq \alpha_1^*$ where α_1^* is the step-length that solves (1.8). Then no matter what line search searches that are used in the following iterations, it is true that

$$(x_{k+1} - x^*)^T A(x_{k+1} - x^*) \geq (\alpha_1 - \alpha_1^*)^2 d_1^T A d_1 \ , \tag{1.15}$$

where $x^* = -A^{-1}b$ is the minimum of the objective function (1.7). Hence we see that the error left in the current iteration can not be eliminated in the following iterations as long as the following search directions are conjugate to the current search direction.

In this paper, we compute search directions by minimizing an approximate quadratic model in the 2-dimensional subspace spanned by the current gradient and the previous search direction. The conjugate gradient method is a special case of our method when the objective function is quadratic and line searches are exact. However, our main object is to construct numerical algorithms that suit inexact line searches and that is also as simple as conjugate gradient algorithms, namely the search direction is a linear combination of the steepest descent direction and the previous search direction.

2. Quadratic Model for $Span(g_{k+1}, d_k)$

In this section, we derive a formula for search direction d_{k+1} by minimizing the approximate quadratic function on the subspace spanned by g_{k+1} and d_k.

Assume that at the k-th iteration an inexact line search is carried out, that is the stepsize α_k satisfies (1.11)-(1.12). We use the notations

$$s_k = \alpha_k d_k = x_{k+1} - x_k \tag{2.1}$$

$$y_k = g_{k+1} - g_k \ . \tag{2.2}$$

We consider the quadratic approximate function:

$$\phi_{k+1}(d) = g_{k+1}^T d + \frac{1}{2} d^T B_{k+1} d \ , \tag{2.3}$$

where $B_{k+1} \in \Re^{n \times n}$ is an approximation to the Hessian $\nabla^2 f(x_{k+1})$. Because we want the search direction d_{k+1} to be a linear combination of g_{k+1} and d_k, we study the problem:

$$\min_{d \in \Omega_k} \ \phi_{k+1}(d) \ , \tag{2.4}$$

where $\Omega_k = span\{g_{k+1}, d_k\}$. Notice that standard conjugate gradient algorithms choose search directions in the form of (1.4). This seems reasonable, because of any vector $d \in \Omega_k$

not parallel to s_k is parallel to a vector in the form of (1.4) with a suitable β. However, line search subroutines usually try $\alpha_k = 1$ first. Hence it is important to have a good prediction of the length of search directions. Therefore, we compute d_{k+1} in the form of $\mu_k g_{k+1} + \nu_k s_k$.

If the vectors g_{k+1} and d_k are collinear, $d_{k+1} \in \Omega_k$ implies that d_{k+1} is parallel to the steepest descent direction. We can give an prediction of the initial steplength by considering problem (2.4). Because g_{k+1} and d_k are collinear, (2.4) is equivalent to

$$\min_{t \in \Re} t g_{k+1}^T s_k + \frac{1}{2} t^2 s_k^T B_{k+1} s_k. \tag{2.5}$$

Remembering that B_{k+1} is an approximation to $\nabla^2 f(x_{k+1})$, and because

$$\nabla^2 f(x_{k+1}) s_k \approx y_k \ , \tag{2.6}$$

we let B_{k+1} satisfy the quasi-Newton equation:

$$B_{k+1} s_k = y_k. \tag{2.7}$$

From the above relation and (2.5), the next search direction can be set to be

$$d_{k+1} = -\frac{g_{k+1}^T s_k}{s_k^T y_k} s_k \ . \tag{2.8}$$

Now we assume that g_{k+1} and d_k are not collinear. Substitute d by $\mu g_{k+1} + \nu s_k$, (2.4) gives that

$$\min_{(\mu,\nu) \in \Re^2} \begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \end{pmatrix}^T \begin{pmatrix} \mu \\ \nu \end{pmatrix} + \frac{1}{2}(\mu, \nu) \begin{pmatrix} g_{k+1}^T B_{k+1} g_{k+1} & g_{k+1}^T B_{k+1} s_k \\ s_k^T B_{k+1} g_{k+1} & s_k^T B_{k+1} s_k \end{pmatrix} \begin{pmatrix} \mu \\ \nu \end{pmatrix} \ . \tag{2.9}$$

Due to relation (2.7), (2.9) can be rewritten as

$$\min_{(\mu,\nu) \in \Re^2} \begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \end{pmatrix}^T \begin{pmatrix} \mu \\ \nu \end{pmatrix} + \frac{1}{2}(\mu, \nu) \begin{pmatrix} \rho_k & g_{k+1}^T y_k \\ y_k^T g_{k+1} & s_k^T y_k \end{pmatrix} \begin{pmatrix} \mu \\ \nu \end{pmatrix} \ , \tag{2.10}$$

where $\rho_k = g_{k+1}^T B_{k+1} g_{k+1}$. We assume that ρ_k satisfies the relation

$$\rho_k s_k^T y_k - (g_{k+1}^T y_k)^2 > 0 \ , \tag{2.11}$$

which is always true if the approximate Hessian B_{k+1} is positive definite. Under condition (2.11), the unique solution of (2.10) can be easily computed:

$$\begin{aligned} \begin{pmatrix} \mu_{k+1} \\ \nu_{k+1} \end{pmatrix} &= -\begin{pmatrix} \rho_k & g_{k+1}^T y_k \\ y_k^T g_{k+1} & s_k^T y_k \end{pmatrix}^{-1} \begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \end{pmatrix} \\ &= \frac{-1}{\rho_k s_k^T y_k - (g_{k+1}^T y_k)^2} \begin{pmatrix} s_k^T y_k & -g_{k+1}^T y_k \\ -y_k^T g_{k+1} & \rho_k \end{pmatrix} \begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \end{pmatrix} \\ &= \frac{-1}{\rho_k s_k^T y_k - (g_{k+1}^T y_k)^2} \begin{pmatrix} s_k^T y_k ||g_{k+1}||_2^2 - g_{k+1}^T y_k g_{k+1}^T s_k \\ \rho_k g_{k+1}^T s_k - g_{k+1}^T y_k ||g_{k+1}||_2^2 \end{pmatrix} \ . \end{aligned} \tag{2.12}$$

Thus, the search direction d_{k+1} can be chosen as

$$\begin{aligned} d_{k+1} &= (g_{k+1}, s_k)\begin{pmatrix}\mu_{k+1}\\ \nu_{k+1}\end{pmatrix} \\ &= \frac{1}{\rho_k s_k^T y_k - (g_{k+1}^T y_k)^2}[(g_{k+1}^T y_k g_{k+1}^T s_k - s_k^T y_k ||g_{k+1}||_2^2) g_{k+1} \\ &\quad +(g_{k+1}^T y_k ||g_{k+1}||_2^2 - \rho_k g_{k+1}^T s_k) s_k] \ . \end{aligned} \tag{2.13}$$

If line searches are exact, that is $g_{k+1}^T s_k = 0$, from (2.13) we have that

$$d_{k+1} = \frac{||g_{k+1}||_2^2 s_k^T y_k}{\rho_k s_k^T y_k - g_{k+1}^T y_k}\left(-g_{k+1} + \frac{g_{k+1}^T y_k}{s_k^T y_k} s_k\right) \ , \tag{2.14}$$

Hence for any choice of ρ_k, the search direction d_{k+1} is parallel to a vector that can be written in the form of (1.4) with

$$\beta_k = \frac{g_{k+1}^T y_k}{d_k^T y_k}, \tag{2.15}$$

which is apparently equivalent to the Fletcher-Reeves formula (1.5) and the Polak-Ribiére-Polyak formula (1.6) if (1.2), (1.7) and (1.8) are satisfied.

If line searches are not exact, different values of ρ_k gives different d_{k+1}. First, (2.11) is satisfied as long as

$$\rho_k \in ((g_{k+1}^T y_k)^2 / s_k^T y_k, +\infty) \ . \tag{2.16}$$

First we consider the two extrem cases. When $\rho_k \to +\infty$, it follows from (2.13) that $\mu_k \to 0$ and the direction $d_{k+1}/||d_{k+1}||_2$ converges to $-\text{sign}(g_{k+1}^T s_k) s_k$. Similarly, when $\rho_k \to (g_{k+1}^T y_k)^2 / s_k^T y_k$, we have that $||d_{k+1}||_2 \to +\infty$ and the direction $d_{k+1}/||d_{k+1}||_2$ converges to the unit length vector parallel to the vector defined in (2.14). Hence it is reasonable to choose ρ_k not too large and not too close to $(g_{k+1}^T y_k)^2 / s_k^T y_k$. Actually, due to the fact that ρ_k should be an approximation to $g_{k+1}^T B_{k+1} g_{k+1}$, it is reasonable to require $\rho_k / ||g_{k+1}||_2^2$ to be bounded. Using relation (2.7), we have that

$$\begin{aligned} \rho_k &= \left(\frac{g_{k+1}^T B_{k+1} g_{k+1} s_k^T B_{k+1} s_k}{(s_k^T B_{k+1} g_{k+1}^T)^2}\right)\left(\frac{(g_{k+1}^T B_{k+1} s_k)^2}{s_k^T B_{k+1} s_k}\right) \\ &= \frac{1}{\cos^2 < B_{k+1}^{\frac{1}{2}} g_{k+1}, B_{k+1}^{\frac{1}{2}} s_k >}\left(\frac{(g_{k+1}^T y_k)^2}{s_k^T y_k}\right) . \end{aligned} \tag{2.17}$$

The quantity $\cos^2 < B_{k+1}^{\frac{1}{2}} g_{k+1}, B_{k+1}^{\frac{1}{2}} s_k >$ in the above equation is unknown, as B_{k+1} is unknown. Because the mean value of $\cos^2\theta$ is $\frac{1}{2}$, it seems reasonable to replace $\cos^2 < B_{k+1}^{\frac{1}{2}} g_{k+1}, B_{k+1}^{\frac{1}{2}} s_k >$ by $\frac{1}{2}$ in (2.17). Thus, we obtain a particular formula for computing ρ_k:

$$\rho_k = 2(g_{k+1}^T y_k)^2 / s_k^T y_k. \tag{2.18}$$

Another simple way is to let B_{k+1} be the BFGS update from the scaled matrix $\frac{s_k^T y_k}{||s_k||_2^2} I$, that is

$$B_{k+1} = \frac{s_k^T y_k}{||s_k||_2^2} \left(I - \frac{s_k s_k^T}{||s_k||_2^2} \right) + \frac{y_k y_k^T}{s_k^T y_k} \ , \tag{2.19}$$

which gives that

$$\rho_k = \frac{s_k^T y_k}{||s_k||_2^2} (||g_{k+1}||_2^2 - (g_{k+1}^T s_k)^2 / ||s_k||_2^2) + (g_{k+1}^T y_k)^2 / s_k^T y_k \ . \tag{2.20}$$

The one step limited memory BFGS method also uses matrix (2.19) to generate search directions. In (2.20), we use (2.19) implicitly to define ρ_k. However, our search direction defined by (2.20) and (2.13) is not the same as that of the limited BFGS method. The latter is

$$\begin{aligned} d_{k+1} &= -B_{k+1}^{-1} g_{k+1} \\ &= -\frac{||s_k||_2^2}{s_k^T y_k} \left(g_{k+1} - y_k \frac{g_{k+1}^T s_k}{s_k^T y_k} \right) \\ &\quad + \left[\frac{||s_k||_2^2 g_{k+1}^T y_k}{(s_k^T y_k)^2} - \left(1 + \frac{||s_k||_2^2 ||y_k||_2^2}{(s_k^T y_k)^2} \right) \frac{g_{k+1}^T s_k}{s_k^T y_k} \right] s_k \end{aligned} \tag{2.21}$$

which is normally not in the subspace Ω_k.

3. Algorithm and Convergence Analysis

In this section, we give a general numerical algorithm for unconstrained optimization, and prove the global convergence of the algorithm.

The algorithm is as follows:

Algorithm 3.1

Step 1 *Given* $x_1 \in \Re^n$, $\epsilon \geq 0$,
given $1 > b_2 > b_1 > 0$;
$k = 1$, *choose* d_1 *such that* $d_1^T g_1 < 0$.

Step 2 *Calculate step-length* α_k *satisfying (1.11)-(1.12);*
set $x_{k+1} = x_k + \alpha_k d_k$;
compute $g_{k+1} = \nabla f(x_{k+1})$;
if $||g_{k+1}||_2 \leq \epsilon$ *then stop.*

Step 3 *If* g_{k+1} *and* d_k *not collinear then go to Step 4;*
define d_{k+1} *by (2.8);*
go to Step 5.

Step 4 *Choosing* ρ_k *satisfying (2.11);*
compute d_{k+1} *by (2.13);*

Step 5 $k := k + 1$, *go to Step 2.*

The above algorithm has the following convergence property:

Theorem 3.2 *Assume that the objective function* $f(x)$ *is convex,* $\nabla^2 f(x)$ *is uniformly bounded, and assume that* $\epsilon = 0$ *is chosen in Algorithm 3.1. If there exist two positive constants* $\hat{M}$ *and* $\hat{\delta}$ *such that*

$$\hat{\delta} \min[1, ||g_{k+1}||_2] s_k^T y_k \leq \rho_k s_k^T y_k - (g_{k+1}^T y_k)^2 \leq \hat{M} s_k^T y_k \tag{3.1}$$

holds for all k*, then the algorithm either stops at a stationary point such that* $\nabla f(x_k) = 0$ *or generates a sequence* $\{x_k\}$ *such that either*

$$\liminf_{k \to \infty} ||\nabla f(x_k)||_2 = 0. \tag{3.2}$$

or

$$\liminf_{k \to \infty} f(x_k) = -\infty. \tag{3.3}$$

Proof Assume the theorem is not true, the algorithm would generate a sequence $\{x_k\}$ such that $f(x_k)$ is bounded below and that

$$||g_k||_2 \geq \delta \tag{3.4}$$

holds for all k, where δ is a positive constant. Because $f(x_k)$ is bounded below, from line search condition (1.11) we have that

$$\sum_{k=1}^{\infty} -s_k^T g_k < \infty. \tag{3.5}$$

The other line search condition (1.12) implies that

$$s_k^T y_k \geq -(1 - b_2) s_k^T g_k. \tag{3.6}$$

It follows from (3.5) and (3.6) that

$$\sum_{k=1}^{\infty} \frac{(s_k^T g_k)^2}{s_k^T y_k} < \infty. \tag{3.7}$$

Because $\nabla^2 f(x)$ is bounded, it is easy to see that $s_k^T y_k = O(||s_k||_2^2)$. Therefore (3.7) implies that

$$\sum_{k=1}^{\infty} \frac{(d_k^T g_k)^2}{||d_k||_2^2} < \infty. \tag{3.8}$$

Since d_{k+1} is the solution of problem (2.4), it follows that

$$\begin{aligned} -d_{k+1}^T g_{k+1} &= 2[\phi_{k+1}(0) - \phi_{k+1}(d_{k+1})] \\ &\geq 2[\phi_{k+1}(0) - \phi_{k+1}(-||g_{k+1}||_2^2 g_{k+1}/\rho_k)] \\ &= ||g_{k+1}||_2^4/\rho_k \end{aligned} \tag{3.9}$$

Line search condition (1.12) implies that

$$|g_{k+1}^T s_k| \leq \frac{b_2}{1-b_2} s_k^T y_k \tag{3.10}$$

It follows from the convexity of $f(x)$ and the boundedness of matrix $\nabla^2 f(x)$ that $||y_k||_2^2/s_k^T y_k$ are uniformly bounded (see, [11]). Therefore, it follows from (3.9), (2.13) and (3.1) that

$$\begin{aligned}
\frac{||d_{k+1}||_2}{-d_{k+1}^T g_{k+1}} &\leq \frac{1}{s_k^T y_k}[O(||s_k||_2||y_k||_2) + O(\rho_k |g_{k+1}^T s_k| ||s_k||_2)] \\
&\leq O(\frac{||s_k||_2}{\sqrt{-s_k^T g_k}}) + O(||s_k||_2) \\
&\leq O\left(\sqrt{||s_k||_2}\sqrt{\frac{||d_k||_2}{-d_k^T g_k}}\right) \\
&\leq O\left(\sqrt{-s_k^T g_k}\frac{||d_k||_2}{-d_k^T g_k}\right). \qquad (3.11)
\end{aligned}$$

The above inequality and relation (3.5) imply that for all sufficiently large k, the inequality $-||d_{k+1}||_2/d_{k+1}^T g_{k+1} \leq -||d_k||_2/d_k^T g_k$ holds, which indicates that the sequence $\{||d_k||_2^2/(d_k^T g_k)^2,\ k = 1, 2, ...\}$ is bounded above. Therefore $(d_k^T g_k)^2/||d_k||_2^2$ is bounded away from zero, which contradicts (3.8). This shows that the theorem is true. QED

For general nonlinear functions, the global convergence can also be proved if

$$\lim_{k\to\infty} s_k = 0 \tag{3.12}$$

and

$$\lim_{k\to\infty} \rho_k g_{k+1}^T s_k = 0. \tag{3.13}$$

Theorem 3.3 *Assume that the objective function $f(x)$ is twice continuously differentiable and $\nabla^2 f(x)$ is uniformly bounded, assume that $\epsilon = 0$ is chosen in Algorithm 3.1. If (3.12) and (3.13) are satisfied, then the algorithm either stops at a stationary point such that $\nabla f(x_k) = 0$ or generates a sequence $\{x_k\}$ such that either (3.2) or (3.3) holds.*

Proof Assume the theorem does not hold, then (3.4) and (3.5) are true. Because the analysis from (3.5) to (3.8) is independent of the convexity of $f(x)$, it can be seen that (3.8) is still true.

Limit (3.12) and the boundedness of $\nabla^2 f(x)$ implies that

$$\lim_{k\to\infty} ||y_k||_2 = 0. \tag{3.14}$$

It follows from (2.13), (3.13), (3.14) and (3.10) that

$$\frac{||d_{k+1}||_2}{-d_{k+1}^T g_{k+1}} \leq \frac{3||g_{k+1}||_2^3||s_k||_2||y_k||_2 + \rho_k||s_k||_2|g_{k+1}^T s_k|}{s_k^T y_k||g_{k+1}||_2^4 - 2g_{k+1}^T y_k||g_{k+1}||_2^2 g_{k+1}^T s_k + \rho_k(g_{k+1}^T s_k)^2}$$

$$
\begin{aligned}
&\leq \frac{||s_k||_2}{s_k^T y_k}\left[\frac{||y_k||_2 + \delta^{-3}\rho_k|g_{k+1}^T s_k|}{\delta - 2||y_k||_2|g_{k+1}^T s_k|/s_k^T y_k}\right] \\
&\leq \frac{||d_k||_2}{-d_k^T g_k} \qquad (3.15)
\end{aligned}
$$

holds for all large k. This, due to our arguments in the proof of Theorem 3.2, contradicts (3.8). Therefore, our theorem is true. QED.

4. Numerical Results and Discussion

Numerical tests have been done on a DECstation 2100. Programs are written in FORTRAN with double precision. Our line search subroutine computes α_k such that (1.11) and (1.14) hold for $b_1 = 0.01$ and $b_2 = 0.9$. For all algorithms, the first search direction is the steepest descent direction $d_1 = -g_1$.

We tested two choices for ρ_k. namely (2.18) and (2.20). In Algorithm A, we compute ρ_k by (2.18). Hence $\rho_k s_k^T y_k - (g_{k+1}^T y_k)^2 = (g_{k+1}^T y_k)^2$. In order to avoid possible numerical overflow in computing d_{k+1} by (2.12), we modified ρ_k if needed to satisfy the following inequality:

$$\rho_k s_k^T y_k - (g_{k+1}^T y_k)^2 \geq 0.1 s_k^T y_k ||g_{k+1}||_2^2. \tag{4.1}$$

The motivation for the above inequality is due to (3.1). Hence ρ_k is given by

$$\rho_k = \max[2(g_{k+1}^T y_k)^2/s_k^T y_k,\ (g_{k+1}^T y_k)^2/s_k^T y_k + 0.1||g_{k+1}||_2^2]. \tag{4.2}$$

In Algorithm B, the parameter ρ_k is computed by (2.20). If g_{k+1} and s_k are collinear, then $\rho_k s_k^T y_k - (g_{k+1}^T y_k)^2 = 0$. Hence we choose d_{k+1} by (2.8) whenever $1 - (g_{k+1}^T s_k)^2/||g_{k+1}||_2^2||s_k||_2^2 < 10^{-8}$

We compared the numerical results of our algorithms with the one step limited memory BFGS method, the Fletcher-Reeves method and the Polak-Ribiére-Polyak method. For the Fletcher-Reeves method and the Polak-Ribiére-Polak method, we restart the algorithm by setting $d_k = -g_k$ whenever an up-hill search direction is given.

We tested the algorithms on the 18 examples given by Moré, Garbow and Hillstrom [6]. The results are given in Table 4.1, where n is the number of variables, I, F, G are numbers of iterations, function evaluations, and gradient evaluations respectively. The stopping condition is $||g_k||_2 \leq 10^{-6}$. The algorithms are also terminated if the number of function evaluation exceeds 500. We also terminate the calculation if the function value improvement is too small. More exactly, algorithms are terminated whenever

$$[f(x_k) - f(x_{k+1})]/(1 + |f(x_k)|) < 10^{-16}. \tag{4.3}$$

In the table, a supscript "*" indicates that the algorithm terminated due to (4.3) but condition $||g_k||_2 \leq 10^{-6}$ is not satisfied, and "*Failed*" means that d_k is so large that an numerical overflow happens while the algorithm tries to compute $f(x_k + d_k)$.

From Table 4.1, we found that our algorithm with ρ_k given by (2.20) performs similar to the Limited Memory BFGS method. Both our algorithm and the Limited Memory BFGS method are much better than the Fletcher-Reeves method and the Polak-Ribiére-Polyak method. And our algorithm is slightly better than the Limited Memory BFGS method when ρ_k is computed by (4.2).

Results of (2.18)

	n	IT	NF	NG	FVAL	GNORM	INFO
1	3	61	104	69	1.465D-13	8.003D-07	0
2	6	248	381	286	5.655D-03	7.057D-07	0
3	3	4	8	5	1.128D-08	6.728D-10	0
4	2	188	307	236	7.824D-18	1.923D-04	1
5	3	54	89	65	3.355D-12	2.606D-07	0
6	6	17	21	18	7.242D-19	1.633D-08	0
7	9	329	501	360	6.014D-05	3.178D-03	-1
8	8	43	73	60	5.422D-05	7.286D-07	0
9	3	12	21	13	3.200D-06	7.780D-07	0
10	2	26	34	28	1.336D-18	2.116D-03	2
11	4	51	87	57	8.582D+04	6.762D-04	3
12	3	307	502	383	5.275D-06	5.265D-05	-1
13	20	63	89	67	6.862D-06	8.833D-07	0
14	14	35	69	47	2.391D-17	1.526D-07	0
15	16	214	332	246	1.930D-10	3.604D-07	0
16	2	21	35	24	3.651D-20	1.637D-09	0
17	4	155	241	169	1.324D-13	9.149D-07	0
18	8	34	55	37	3.517D-03	3.260D-07	0

Results of LBFGS

	n	IT	NF	FVAL	GNORM	IFLAG
1	3	61	74	6.946D-16	5.604D-07	0
2	6	210	272	5.656D-03	1.414D-5	0
3	3	9	11	1.128D-08	9.946D-07	0
4	2	203	306	2.299D-23	3.793D-07	0
5	3	73	106	7.808D-12	2.076D-06	0
6	6	15	16	7.350D-27	1.645D-12	0
7	9	374	501	5.827D-05	4.804D-03	-1
8	8	52	69	5.422D-05	5.569D-07	0
9	3	14	18	3.200D-06	7.482D-08	0
10	2	31	52	7.509D-15	3.924D-02	0
11	4	66	90	8.582D+04	2.163D-04	-1
12	3	112	194	1.429D-06	2.925D-05	0
13	20	67	82	6.862D-06	7.016D-07	0
14	14	44	76	4.647D-17	4.332D-08	0
15	16	168	215	4.989D-10	6.183D-07	0
16	2	16	21	1.399D-15	2.673D-07	0
17	4	233	313	8.047D-14	6.846D-07	0
18	8	40	60	3.517D-03	7.848D-07	0

TABLE 4.1

		L-BFGS	F-R	P-R-P	(2.18)	(2.20)
	n	I-F-G	I-F-G	I-F-G	I-F-G	I-F-G
1	3	62-142-65	96-293-100	95-290-98	85-145-94	45-88-50
2	6	Failed	>500	>500	>500	Failed
3	3	4-10-6	25-56-25	9-16-11	4-8-5	4-10-6
4	2	Failed	>500	Failed	188-307-236*	Failed
5	3	102-259-109	52-89-56	162-264-186	54-89-65	61-147-68
6	6	17-21-18	33-122-34	23-83-25	17-21-18	17-21-18
7	9	>500	>500	>500	>500	>500
8	8	58-169-75	39-76-49	>500	43-73-60	50-139-62
9	3	13-21-14	18-36-18	30-64-33	12-21-13	13-21-14
10	2	31-188-35*	>500	15-86-20*	26-34-28*	31-188-35*
11	4	57-107-59*	>500	64-265-71*	47-74-49*	48-87-52*
12	3	Failed	>500	>500	>500	Failed
13	20	60-87-65	>500	124-143-135	59-85-62	70-108-73
14	14	43-126-57	116-379-122	50-155-58	35-69-47	45-129-59
15	16	122-331-128	>500	158-423-164	117-169-129	119-320-132
16	2	19-38-20	94-265-95	17-42-19	21-35-24	19-38-20
17	4	>500	49-147-49	>500	155-241-169	>500
18	8	42-70-45	>500	37-101-41	34-55-37	49-78-52

A straightforward generalization of our approach is to compute the search direction d_{k+1} by minimization the quadratic function (2.3) in the subspace $Span(g_{k+1}, d_k, d_{k-1})$. Similar to our analysis from (2.7) to (2.13), we can choose

$$d_{k+1} = \mu_k g_{k+1} + \nu_k s_k + \tau_k s_{k-1}, \tag{4.4}$$

and compute μ_k, ν_k and τ_k by minimizing a quadratic function in $\Re^3$.

Assume the approximation matrix B_{k+1} satisfy the quasi-Newton equation (2.7) and also the following relation

$$B_{k+1} s_{k-1} = y_{k-1} \ . \tag{4.5}$$

Then minimizing the quadratic function (2.3) is equivalent to

$$\min_{(\mu,\nu,\tau)\in\Re^3} \begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \\ g_{k+1}^T s_{k-1} \end{pmatrix}^T \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix} + \frac{1}{2} \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix}^T \begin{pmatrix} \rho_k & g_{k+1}^T y_k & g_{k+1}^T y_{k-1} \\ y_k^T g_{k+1} & s_k^T y_k & s_{k-1}^T y_k \\ g_{k+1}^T y_{k-1} & s_{k-1}^T y_k & s_{k-1}^T y_{k-1} \end{pmatrix} \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix} \ . \tag{4.6}$$

Again, here ρ_k is an approximation to $g_{k+1}^T B_{k+1} g_{k+1}$. However one difficulty is that the quadratic function in (4.6) may not be convex. For example, if the submatrix

$$\begin{pmatrix} s_k^T y_k & s_k^T y_{k-1} \\ s_k^T y_{k-1} & s_{k-1}^T y_{k-1} \end{pmatrix} \tag{4.7}$$

has a negative eigenvalue, then the quadratic function in (4.6) can never be convex no matter how large the the parameter ρ_k is.

One way to overcome this difficulty is to replacing the terms $g_{k+1}^T y_{k-1}$ and $s_{k-1}^T y_k$ by zeros. In this case, the new search direction d_{k+1} is given by

$$d_{k+1} = \bar{d}_{k+1} - s_{k-1} g_{k+1}^T s_{k-1} / s_{k-1}^T y_{k-1} \tag{4.8}$$

where $\bar{d}_{k+1}$ is the right hand side of (2.13).

If we replace only $s_{k-1}^T y_k$ by zero, then the objective function in (4.6) reduces to

$$\begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \\ g_{k+1}^T s_{k-1} \end{pmatrix}^T \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix} + \frac{1}{2} \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix}^T \begin{pmatrix} \rho_k & g_{k+1}^T y_k & g_{k+1}^T y_{k-1} \\ y_k^T g_{k+1} & s_k^T y_k & 0 \\ g_{k+1}^T y_{k-1} & 0 & s_{k-1}^T y_{k-1} \end{pmatrix} \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix} . \tag{4.9}$$

We find that the above quadratic function is strictly convex if and only if

$$D_k = \rho_k s_k^T y_k s_{k-1}^T y_{k-1} - s_k^T y_k (g_{k+1}^T y_{k-1})^2 - s_{k-1}^T y_{k-1} (g_{k+1}^T y_k)^2 > 0. \tag{4.10}$$

Hence, when ρ_k satisfies the above inequality, the unique solution $(\mu_k, \nu_k, \tau_k)^T$ of (4.9) can be easily computed as follows:

$$\frac{-1}{D_k} \begin{pmatrix} s_k^T y_k s_{k-1}^T y_{k-1} & -s_{k-1}^T y_{k-1} g_{k+1}^T y_k & -s_k^T y_k g_{k+1}^T y_{k-1} \\ -s_{k-1}^T y_{k-1} g_{k+1}^T y_k & \theta_k & g_{k+1}^T y_k g_{k+1}^T y_{k-1} \\ -s_k^T y_k g_{k+1}^T y_{k-1} & g_{k+1}^T y_k g_{k+1}^T y_{k-1} & \kappa_k \end{pmatrix} \begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \\ g_{k+1}^T s_{k-1} \end{pmatrix} \tag{4.11}$$

where $\theta_k = \rho_k s_{k-1}^T y_{k-1} - (g_{k+1}^T y_{k-1})^2$ and $\kappa_k = \rho_k s_k^T y_k - (g_{k+1}^T y_k)^2$. Similar to (4.2), ρ_k can be set to

$$\rho_k = \hat{\rho}_k + \max[\hat{\rho}_k, 0.1||g_{k+1}||_2^2] \tag{4.12}$$

where $\hat{\rho}_k = (g_{k+1}^T y_k)^2 / s_k^T y_k + (g_{k+1}^T y_{k-1})^2 / s_{k-1}^T y_{k-1}$.

Another way is to use a two-step limited memory BFGS update matrix:

$$B_k = \frac{s_{k-1}^T y_{k-1}}{||s_{k-1}||_2^2} \left(I - \frac{s_{k-1} s_{k-1}^T}{||s_{k-1}||_2^2} \right) + \frac{y_{k-1} y_{k-1}^T}{s_{k-1}^T y_{k-1}} \tag{4.13}$$

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k} . \tag{4.14}$$

In this case, the minimization of quadratic function (2.3) reduces to

$$\min_{(\mu,\nu,\tau)\in\Re^3} \begin{pmatrix} ||g_{k+1}||_2^2 \\ g_{k+1}^T s_k \\ g_{k+1}^T s_{k-1} \end{pmatrix}^T \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix} + \frac{1}{2} \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix}^T \begin{pmatrix} \chi_k & g_{k+1}^T y_k & \eta_k \\ y_k^T g_{k+1} & s_k^T y_k & s_{k-1}^T y_k \\ \eta_k & s_{k-1}^T y_k & \xi_k \end{pmatrix} \begin{pmatrix} \mu \\ \nu \\ \tau \end{pmatrix} \tag{4.15}$$

where $\chi_k = g_{k+1}^T B_{k+1} g_{k+1}$, $\eta_k = g_{k+1}^T B_{k+1} s_{k-1}$ and $\xi_k = s_{k-1}^T B_{k+1} s_{k-1}$ can be easily computed.

In table 4.2, we give numerical results of our algorithm with different choices of d_{k+1} from $Span(g_{k+1}, s_k, s_{k-1})$. As in Table 4.1, for comparison, we also given the 2-step limited memory BFGS method that uses $d_{k+1} = -B_{k+1}^{-1} g_{k+1}$ and B_{k+1} is defined by (4.13)-(4.14).

The column under heading (4.8) are the numerical results of the algorithm with d_{k+1} given by (4.8), $\bar{d}_{k+1}$ by (2.13) and ρ_k by (4.2). The column under heading (4.12) are the numerical results of our algorithm with d_{k+1} given be (4.5), (4.11) and (4.12). The last column of Table 4.2 are the numerical results of our algorithm with d_{k+1} given by (4.5) where the parameters μ_k, ν_k, τ_k are computed by solving (4.15), and $\chi_k = g_{k+1}^T B_{k+1} g_{k+1}$, $\eta_k = g_{k+1}^T B_{k+1} s_{k-1}$ and $\xi_k = s_{k-1}^T B_{k+1} s_{k-1}$ with B_{k+1} defined by (4.13)-(4.14).

TABLE 4.2

		L-BFGS	(4.8)	(4.12)	(4.15)
	n	I-F-G	I-F-G	I-F-G	I-F-G
1	3	31-63-33	92-140-100	45-66-47	35-86-40
2	6	Failed	>500	206-256-222	Failed
3	3	5-10-7	4-8-5	4-8-5	3-5-4
4	2	Failed	151-270-173*	219-348-244*	Failed
5	3	22-47-28	55-95-64	62-83-72	23-54-29
6	6	17-21-18	13-18-15	11-16-13	12-21-13
7	9	>500	>500	>500	>500
8	8	48-97-57	59-100-64	58-92-68	43-84-50
9	3	12-18-14	20-30-23	12-21-14	11-18-13
10	2	15-49-17*	33-57-35*	42-59-44*	15-45-16*
11	4	64-114-66*	68-117-73*	58-77-62*	32-61-36*
12	3	Failed	>500	119-172-135	Failed
13	20	60-96-63	78-111-80	81-122-84	65-93-67
14	14	35-72-43	46-77-48	55-85-62	34-72-44
15	16	40-93-41	215-326-230	144-183-153	142-478-147
16	2	16-23-17	20-32-22	26-36-27	15-23-16
17	4	157-415-177	103-171-116	224-318-244	124-323-141
18	8	34-59-37	51-90-55	30-49-32	30-47-34

First, comparing the column under heading (4.8) with that under heading (2.18) in Table 4.1, we found that the technique of adding the term $-\frac{g_{k+1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} s_{k-1}$ to the search direction obtained from $Span(g_{k+1}, s_k)$ did not provide any improvement. Not surprisingly, we found that (4.15) performs very similar to the two step limited memory BFGS method, as in (4.15) we use the two step limited memory BFGS matrix to compute χ_k, η_k and ξ_k. Both algorithms are slightly better than (4.8). Our algorithm with d_{k+1} given by (4.11) and ρ_k given by (4.12) seems the best among all the algorithms listed in Table 4.2. It successfully solved problems 2 and 12 while all other algorithms failed.

We have presented a new numerical method for unconstrained optimization. Our approach is to generate search directions based on minimizations on subspaces. The new method can be viewed as a generalization of the conjugate gradient method as it reduces to the conjugate gradient method when line searches are exact and the objective function is strict convex and quadratic. In some sense, the search direction of the conjugate gradient method is an optimal search direction when line searches are exact, as quadratic

termination is ensured. However, when line searches are not exact, conjugate directions may not be the best choice. Our method is based on minimizations on $Span(g_{k+1}, s_k)$, it makes use of the accuracy of line searches. Numerical results indicate that our approach provides an improvement over conjugate directions.

References

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

[2] A. Buckley, "Conjugate gradient methods", in: M.J.D. Powell, ed., *Nonlinear Optimization 1981* (Academic Press, London, 1982) pp. 17-22.

[3] J.E. Dennis Jr., and K. Turner, "Generalized conjugate directions", Technical Report 85-11, Department of Mathematical Sciences, Rice University, USA, 1985.

[4] R. Fletcher, *Practical Optimization: Vol 1, Unconstrained Optimization* John Wiley and Sons, Chichester, 1980.

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

[6] J.J. Moré, B.S. Garbow and K.E. Hillstrom (1981). "Testing unconstrained optimization software" *ACM Transactions on Mathematical Software* **7**, pp. 17–41.

[7] E. Polak, *Computational Methods in Optimization: A Unified Approach* (Academic Press, New York, 1971).

[8] E. Polak and Ribiére G., "Note sur la convergence de méthodes de directions conjugees" *Rev. Fr. Inform. Rech. Oper.* 16 (1969) 35-43.

[9] B.T. Polyak, " The conjugate gradient method in extremal problems" *USSR Comp. Maths. and Math. Phys.* 9 (1969) 94-112.

[10] M.J.D. Powell, "Some convergence properties of the conjugate gradient method" *Mathematical Programming* 11(1976) 42-49. (1976a)

[11] M.J.D. Powell, "Some global convergence properties of a variable metric algorithm for minimization without exact line searches", in: R.W. Cottle and C.E. Lemke, eds., *Nonlinear Programming, SIAM-AMS Proceedings vol. IX* (SIAM publications, Philadelphia, 1976) pp. 53-72. (1976b)

[12] M.J.D. Powell, "Nonconvex minimization calculations and the conjugate gradient method", in: G.A. Watson, ed., *Numerical Analysis, Lecture Notes in Mathematics No. 1066* (Springer-Verlag, London, 1984) pp. 121-141.

[13] Y. Yuan, "Analysis on the conjugate gradient method", Technical Report, Computing Center, Academia Sinica, China, 1990.