Ladislav Lukšan

Variable metric method with limited storage for large-scale unconstrained minimization

Kybernetika, Vol. 18 (1982), No. 6, 517--528

Persistent URL: `http://dml.cz/dmlcz/124855`

KYBERNETIKA — VOLUME 18 (1982), NUMBER 6

VARIABLE METRIC METHOD WITH LIMITED STORAGE FOR LARGE-SCALE UNCONSTRAINED MINIMIZATION

LADISLAV LUKŠAN

This contribution contains a description of a variable metric method with limited storage for large-scale unconstrained minimization. The quadratic termination of this method is proved and an algorithm which implements this method is presented. Efficiency of the algorithm is demonstrated on test functions.

1. INTRODUCTION

We are concerned with the problem of finding a local unconstrained minimum of a real-valued function $F(x)$ defined in the n-dimensional vector space R_n and having continuous second-order derivatives. The variable metric methods are widely used for solving this problem when $n \leqq 100$, say. They construct a sequence of symmetric positive definite matrices of the order n, so it is necessary to have $n(n+1)/2$ locations in the high-speed computer storage. As n increases, $n(n+1)/2$ becomes too large and the variable metric methods cannot be used.

Probably the first efficient method for large-scale unconstrained minimization was the method of conjugate gradients. It is an iterative method, whose k-th iteration $(k = 0, 1, 2, \ldots)$ has the form

$$x_{k+1} = x_k + \alpha_k s_k \tag{1.1}$$

where s_k is a direction vector and α_k is a steplength. The direction vector s_k must satisfy the condition

$$-s_k^{\mathrm{T}} g_k \geqq \varepsilon_0 \|s_k\| \, \|g_k\| \tag{1.2}$$

where $g_k = g(x_k)$ is the gradient of the objective function $F(x)$ at the point x_k and $0 < \varepsilon_0 < 1$ is a small positive number. The steplength α_k is taken to satisfy conditions

$$\begin{cases} F_{k+1} - F_k \leqq \varepsilon_1 \alpha_k s_k^{\mathrm{T}} g_k \\ s_k^{\mathrm{T}} g_{k+1} \geqq (1 - \varepsilon_2)\, s_k^{\mathrm{T}} g_k \end{cases} \tag{1.3}$$

where $F_{k+1} = F(x_{k+1})$, $F_k = F(x_k)$, $g_{k+1} = g(x_{k+1})$, $g_k = g(x_k)$ and where $0 < 2\varepsilon_1 < 1$ and $0 < 2\varepsilon_2 < 1$. Note that $s_k^T g_{k+1} = 0$ when perfect line search is performed. The direction vector s_k is computed recursively by the rule

$$s_{k+1} = -g_{k+1} + \beta_k s_k \tag{1.4}$$

where β_k can have three alternative values

$$\beta_k = \frac{g_{k+1}^T y_k}{s_k^T y_k} \quad \text{(Hestenes - Stiefel [9])} \tag{1.5a}$$

$$\beta_k = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k} \quad \text{(Fletcher - Reeves [6])} \tag{1.5b}$$

$$\beta_k = \frac{g_{k+1}^T y_k}{g_k^T g_k} \quad \text{(Polak - Ribiere [16])} \tag{1.5c}$$

(we use notation $d_k = x_{k+1} - x_k = \alpha_k s_k$ and $y_k = g_{k+1} - g_k$ through this paper).

The method of conjugate gradients was introduced by Hestenes and Stiefel for solving systems of linear algebraic equations and by Fletcher and Reeves for unconstrained minimization. Since that time it has been improved by many authors. An important modification of the method of conjugate gradients is based on the addition of one or several terms into (1.4) which vanish for $g_{k+1}^T s_k = 0$ (perfect line search). A suitable selection of these terms causes that (1.4) can be expressed in the form

$$s_{k+1} = -H_{k+1} g_{k+1} \tag{1.6}$$

where H_{k+1} is a symmetric positive definite matrix of the order n which satisfies the so-called quasi-Newton condition

$$H_{k+1} y_k = d_k \tag{1.7}$$

This condition is satisfied when H_{k+1} is the inverse of the Hessian matrix of the objective function $F(x)$ at the point x_{k+1} and it forms a basis for the broad class of quasi-Newton methods.

The addition of auxiliary terms into (1.4) was introduced by Perry [15]. Shanno [20] has shown that the matrix H_{k+1} in (1.6) can be chosen in such a way that

$$H_{k+1} = \gamma_k \left(\bar{H}_k + \frac{1}{\gamma_k \sigma_k} d_k d_k^T - \frac{1}{\tau_k} \bar{H}_k y_k (\bar{H}_k y_k)^T + \right.$$
$$\left. + \frac{1}{\tau_k} \left(\frac{\tau_k}{\sigma_k} d_k - \bar{H}_k y_k \right) \left(\frac{\tau_k}{\sigma_k} d_k - \bar{H}_k y_k \right)^T \right)$$

where $\bar{H}_k = I$ (the unit matrix of the order n) and $\gamma_k > 0$ is a free parameter. Furthermore $\sigma_k = y_k^T d_k$ and $\tau_k = y_k^T \bar{H}_k y_k$. The above expression for H_{k+1} is just the generalized one-step BFGS update (see Broyden [3], Fletcher [7], Goldfarb [8] and

Shanno [19]). Shanno [20] has also proposed an algorithm which uses the two-step BFGS update. The relationship between the method of conjugate gradients and the BFGS method was also studied in [4], [5] and [11]. It gave rise to the combined conjugate gradient quasi-Newton algorithm. All these methods work with limited storage.

We are proposing a variable metric method with limited storage, which uses the m-step BFGS update. It is very close to the method proposed in [13] but it uses general values of free parameters. Moreover, an efficient algorithm is presented and its efficiency is demonstrated on the standard test problems.

2. PROPERTIES OF THE NEW METHOD

The variable metric method with limited storage (or the m-step BFGS method) studied in this section is an iterative method, whose k-th iteration has the form (1.1)–(1.3), where

$$s_k = -H_{k,0}g_k \tag{2.1}$$

and where $H_{k,0}$ is a symmetric positive definite matrix of the order n obtained by the m-step BFGS update

$$\begin{aligned} H_{k,j-1} = \gamma_{k-j}\Bigg(H_{k,j} &+ \frac{\varrho_{k-j}}{\gamma_{k-j}\sigma_{k-j}}\, d_{k-j}d_{k-j}^{\mathrm{T}} - \\ &- \frac{1}{\tau_{k-j}}\, H_{k,j}y_{k-j}(H_{k,j}y_{k-j})^{\mathrm{T}} + \\ &+ \frac{1}{\tau_{k-j}}\left(\frac{\tau_{k-j}}{\sigma_{k-j}}\, d_{k-j} - H_{k,j}y_{k-j}\right)\left(\frac{\tau_{k-j}}{\sigma_{k-j}}\, d_{k-j} - H_{k,j}y_{k-j}\right)^{\mathrm{T}}\Bigg) \end{aligned} \tag{2.2}$$

for $1 \leqq j \leqq \bar{k}$, $\bar{k} = \min(k, m)$ where $H_{k,\bar{k}} = \mathrm{I}$ (I is the unit matrix of the order n), $\gamma_{k-j} > 0$ and $\varrho_{k-j} > 0$ are free parameters and $\sigma_{k-j} = y_{k-j}^{\mathrm{T}}d_{k-j}$, $\tau_{k-j} = y_{k-j}^{\mathrm{T}}H_{k,j} \cdot y_{k-j}$. Then for an arbitrary vector v

$$\begin{aligned} H_{k,j-1}v = \gamma_{k-j}\Bigg(H_{k,j}v &- \frac{d_{k-j}^{\mathrm{T}}v}{\sigma_{k-j}}\, H_{k,j}y_{k-j} + \\ &+ \left(\left(\frac{\varrho_{k-j}}{\gamma_{k-j}} + \frac{\tau_{k-j}}{\sigma_{k-j}}\right)\frac{d_{k-j}^{\mathrm{T}}v}{\sigma_{k-j}} - \frac{(H_{k,j}y_{k-j})^{\mathrm{T}}v}{\sigma_{k-j}}\right) d_{k-j}\Bigg) \end{aligned} \tag{2.3}$$

holds. This expression contains only n-dimensional vectors and it can be used for consecutive evaluation of the term $H_{k,0}g_k$ which appears in (2.1).

Now we are proving the main results about the behaviour of the m-step BFGS method applied to the quadratic function

$$F(x) = \tfrac{1}{2}(x - \tilde{x})^{\mathrm{T}}\, G(x - \tilde{x}) \tag{2.4}$$

supposing perfect line search is performed.

Theorem 2.1. If the m-step BFGS method is applied to the quadratic function (2.4) and if the steplengths are chosen by perfect line searches, then

$$s_k = -\left(\prod_{j=1}^{\bar{k}} \gamma_{k-j}\right)\left(g_k - \frac{y_{k-1}^T g_k}{y_{k-1}^T d_{k-1}} d_{k-1}\right) \tag{2.5}$$

for $1 \leqq k \leqq n$, where $\bar{k} = \min(k, m)$.

Proof. Let $1 \leqq k \leqq n$ and $\bar{k} = \min(k, m)$. If the perfect line search is performed in all subsequent iterations we have $d_{k-j}^T g_{k+1-j} = 0$ for $1 \leqq j \leqq \bar{k}$ and from (2.3) we obtain

$$H_{k,j-1} g_{k+1-j} = \gamma_{k-j}\left(H_{k,j} g_{k+1-j} - \frac{y_{k-j}^T H_{k,j} g_{k+1-j}}{\sigma_{k-j}} d_{k-j}\right) \tag{2.6}$$

for $1 \leqq j \leqq \bar{k}$. Since $H_{k,\bar{k}} = I$, (2.5) holds for $k = 1$. Now we use induction. Let (2.5) holds for $k = l - 1 \geqq 1$. Then search directions s_i, $1 \leqq i < l$ are parallel to the search directions of the method of conjugate gradients, so that

$$d_i^T y_j = y_i^T d_j = 0\,, \quad 1 \leqq i < j < l \tag{2.7}$$

$$d_i^T g_j = 0\,, \quad 1 \leqq i < j \leqq l \tag{2.8}$$

$$g_i^T g_j = 0\,, \quad 1 \leqq i < j \leqq l \tag{2.9}$$

Now we shall prove that

$$H_{l,0} g_l = \left(\prod_{i=1}^{\bar{l}} \gamma_{l-i}\right)\left(g_l - \sum_{i=1}^{\bar{l}} \frac{y_{l-i}^T g_l}{\sigma_{l-i}} d_{l-i}\right) \tag{2.10}$$

where $\bar{l} = \min(l, m)$. We prove it by induction again. Let

$$H_{l,0} g_l = \left(\prod_{i=1}^{j} \gamma_{l-i}\right)\left(H_{l,j} g_l - \sum_{i=1}^{j} \frac{y_{l-i}^T H_{l,j} g_l}{\sigma_{l-i}} d_{l-i}\right) \tag{2.11}$$

for $j < \bar{l}$. From (2.6) it follows that (2.11) holds for $j = 1$. Since $d_{l-j-1}^T g_l = 0$ for $j < \bar{l}$ by (2.8), we obtain

$$H_{l,j} g_l = \gamma_{l-j-1}\left(H_{l,j+1} g_l - \frac{y_{l-j-1}^T H_{l,j+1} g_l}{\sigma_{l-j-1}} d_{l-j-1}\right) \tag{2.12}$$

from (2.3). Since $y_{l-i}^T d_{l-j-1} = 0$ for $1 \leqq i \leqq j$ by (2.7), we have $y_{l-i}^T H_{l,j} g_l = \gamma_{l-j-1} y_{l-i}^T H_{l,j+1} g_l$ for $1 \leqq i \leqq j$. Setting it together with (2.12) into (2.11), we obtain

$$H_{l,0} g_l = \left(\prod_{i=1}^{j} \gamma_{l-i}\right)\left(\gamma_{l-j-1} H_{l,j+1} g_l - \gamma_{l-j-1} \frac{y_{l-j-1}^T H_{l,j+1} g_l}{\sigma_{l-j-1}} d_{l-j-1} - \right.$$

$$\left. - \sum_{i=1}^{j} \gamma_{l-j-1} \frac{y_{l-i}^T H_{l,j+1} g_l}{\sigma_{l-i}} d_{l-i}\right) =$$

$$= \left(\prod_{i=1}^{j+1} \gamma_{l-i}\right)\left(H_{l,j+1} g_l - \sum_{i=1}^{j+1} \frac{y_{l-i}^T H_{l,j+1} g_l}{\sigma_{l-i}} d_{l-i}\right)$$

which is just (2.11) with j increased by 1, so that (2.10) is proved since $H_{l,l} = I$. Now $y_{l-i}^T g_l = (g_{l+1-i}^T g_l - g_{l-i}^T g_l) = 0$ for $2 \leqq i \leqq l$ by (2.9), so that (2.10) gives

$$s_l = -H_{l,0} g_l = -\left(\prod_{i=1}^{l} \gamma_{l-i}\right)\left(g_l - \frac{y_{l-1}^T g_l}{\sigma_{l-1}} d_{l-1}\right)$$

which is just (2.5) with k increased by 1, so that (2.5) is proved for all $1 \leqq k \leqq n$. □

Theorem 2.1 implies that the m-step BFGS method applied to the quadratic function (2.4) is equivalent to the method of conjugate gradients when the perfect line search is performed, so that it finds a minimum of the quadiatic function (2.4) after at most n iterations.

Theorem 2.2. If the assumptions of Theorem 2.1 are satisfied, then

$$H_{k,0} y_{k-j} = \left(\prod_{i=1}^{j} \gamma_{k-i}\right) \frac{\varrho_{k-j}}{\gamma_{k-j}} d_{k-j}, \quad 1 \leqq j \leqq \bar{k} \tag{2.13}$$

for $1 \leqq k \leqq n$, where $\bar{k} = \min(k, m)$.

Proof. We prove this theorem by induction. Let

$$H_{k,k-l} y_{l-j} = \left(\prod_{i=1}^{j} \gamma_{l-i}\right) \frac{\varrho_{l-j}}{\gamma_{l-j}} d_{l-j}, \quad 1 \leqq j \leqq \bar{l} \tag{2.14}$$

for some $l < k$, where $\bar{l} = \min(l, m - k + l)$. This relation is true for $l = k - \bar{k}$, where $\bar{k} = \min(k, m)$, since the condition $1 \leqq j \leqq \bar{l}$ cannot be satisfied for any index $j(\bar{l} = \min(k - \bar{k}, m - k + k - \bar{k}) = 0$ for $l = k - \bar{k})$. Since $d_l^T y_{l-j} = 0$ by (2.7) and $y_l^T H_{k,k-l} y_{l-j} = 0$ by (2.7) and (2.14), we obtain

$$H_{k,k-l-1} y_{l-j} = \gamma_l H_{k,k-l} y_{l-j} = \left(\prod_{i=0}^{j} \gamma_{l-i}\right) \frac{\varrho_{l-j}}{\gamma_{l-j}} d_{l-j}, \quad 1 \leqq j \leqq \bar{l}$$

from (2.3) and (2.14). After changing the indices (increasing j by 1) we obtain

$$H_{k,k-l-1} y_{l+1-j} = \left(\prod_{i=1}^{j} \gamma_{l+1-i}\right) \frac{\varrho_{l+1-j}}{\gamma_{l+1-j}} d_{l+1-j}, \; 2 \leqq j \leqq \overline{l+1} \tag{2.15}$$

where $\overline{l+1} = \min(l+1, m - k + l + 1)$. Furthermore, we have

$$H_{k,k-l-1} y_l = \gamma_l \left(H_{k,k-l} y_l - \frac{\varrho_l}{\sigma_l} H_{k,k-l} y_l + \left(\left(\frac{\varrho_l}{\gamma_l} + \frac{\tau_l}{\sigma_l}\right)\frac{\sigma_l}{\sigma_l} - \frac{\tau_l}{\sigma_l}\right) d_l\right) =$$

$$= \gamma_l \frac{\varrho_l}{\gamma_l} d_l$$

from (2.3). This expression and (2.15) give

$$H_{k,k-l-1} y_{l+1-j} = \left(\prod_{i=1}^{j} \gamma_{l+1-i}\right) \frac{\varrho_{l+1-j}}{\gamma_{l+1-j}} d_{l+1-j}, \quad 1 \leqq j \leqq \overline{l+1}$$

which is just (2.14) with l increased by 1, so that (2.13) is proved. □

Theorem 2.2 implies that setting $\gamma_{k-j} = 1$, $1 \leqq j < \bar{k}$, $\gamma_{k-\bar{k}} = \gamma$ and $\varrho_{k-j} = \varrho$, we have satisfied $\bar{k}$ generalized quasi-Newton conditions

$$H_{k,0}y_{k-j} = \varrho d_{k-j}\,, \quad 1 \leqq j \leqq \bar{k}$$

Parameter γ introduced in [14] serves for conditioning and improving stability of the BFGS update (see also Shanno and Phua [21]). Parameter ϱ was introduced in [2]. For the quadratic function (2.4) the best choice is $\varrho = 1$. Special choices of the parameter ϱ can improve the behaviour of the m-step BFGS method for non-quadratic objective function.

3. IMPLEMENTATION OF THE NEW METHOD

The m-step BFGS method uses recurrence relation (2.3) for consecutive evaluation of the search direction (2.1). This recurrence relation can be rewritten in the form

$$H_{k,j-1}v = \varphi(d_{k-j}, H_{k,j}y_{k-j}, v, H_{k,j}v, \sigma_{k-j}, \tau_{k-j})$$

where

$$\varphi(d, u, v, w, \sigma, \tau) = \gamma\left(w - \frac{d^{\mathrm{T}}v}{\sigma}u + \left(\left(\frac{\varrho}{\gamma} + \frac{\tau}{\sigma}\right)\frac{d^{\mathrm{T}}v}{\sigma} - \frac{u^{\mathrm{T}}v}{\sigma}\right)d\right) \tag{3.1}$$

Note that γ and ϱ are not parameters of the function φ but they are implicitly assumed to appear in (3.1). For $m = 3$ we can write the chart of computation in the following form

$$\begin{array}{llll}
H_k g_k & & & \\
\uparrow & & & \\
H_{k-1}g_k & \boxed{H_{k-1}y_{k-1} \quad d_{k-1}} & & \\
\uparrow & \uparrow & & \\
H_{k-2}g_k & H_{k-2}y_{k-1} & \boxed{H_{k-2}y_{k-2} \quad d_{k-2}} & \\
\uparrow & \uparrow & \uparrow & \\
H_{k-3}g_k & H_{k-3}y_{k-1} & H_{k-3}y_{k-2} & \boxed{H_{k-3}y_{k-3} \quad d_{k-3}} \\
g_k & y_{k-1} & y_{k-2} & y_{k-3}
\end{array}$$

Some vector in this chart is computed by means of four vectors. They are the closest vector in the same column (see arrows), the vector on the bottom of the same column (see rings) and the framed vectors in the previous row. Therefore 9 n-dimensional vectors must be stored simultaneously for $m = 3$ (the method of conjugate gradients uses 5 n-dimensional vectors). These 9 n-dimensional vectors are denoted x, g, s, x_1, g_1, x_2, g_2, x_3, g_3 in the description of the algorithm. Vectors x, g, s, x_1, g_1 represent the vectors x_k, g_k, s_k, x_{k-1}, g_{k-1} and vectors x_1, g_1, x_2, g_2, x_3, g_3 represent the vectors d_{k-1}, y_{k-1}, d_{k-2}, y_{k-2}, d_{k-3}, y_{k-3}. Note that vectors x_1, g_1 represent both x_{k-1}, g_{k-1} and d_{k-1}, y_{k-1}.

Now we are in a position to describe the algorithm of m-step BFGS method. We use $\varrho = 1$ in (3.1). The choice of the parameter γ is controlled by the integer l.

Algorithm 3.1.

Step 1: Determine the initial vector x and compute values $F := F(x)$ and $g := g(x)$.

Step 2: Test for convergence. If the termination criteria are satisfied (for example if $\|g\|$ is sufficiently small) then stop.

Step 3: In the first iteration go to step 4 else go to step 5.

Step 4: Set $s := -g$. Set $k := 0$ and go to step 17.

Step 5: Set $\bar{l} := l$ and $\bar{k} := \min(k, m)$. Set $x_1 := x - x_1$, $g_1 := g - g_1$ and $s := g$.

Step 6: If $\bar{k} \geqq 3$ go to step 7 else go to step 8.

Step 7: Compute $\tau_3 = g_3^T g_3$. Compute $s := \varphi(x_3, g_3, g, s, \sigma_3, \tau_3)$. Set $x_0 := g_2$, $g_0 := g_1$, compute $x_0 := \varphi(x_3, g_3, g_2, x_0, \sigma_3, \tau_3)$, $g_0 := \varphi(x_3, g_3, g_1, g_0, \sigma_3, \tau_3)$ and set $x_3 := x_0$, $g_3 := g_0$ (vectors x_0 and g_0 need not be stored if the computation runs by coordinates). Function φ is defined by (3.1) where $\gamma = 1$ if $\bar{l} = 0$ or $\gamma = \sigma_3/\tau_3$ if $\bar{l} = 1$. Set $\bar{l} := 0$ and go to step 10.

Step 8: If $\bar{k} \geqq 2$ go to step 9 else go to step 11.

Step 9: Set $x_3 := g_2$ and $g_3 := g_1$.

Step 10: Compute $\tau_2 := g_2^T x_3$. Compute $s := \varphi(x_2, x_3, g, s, \sigma_2, \tau_2)$ and $g_3 := := \varphi(x_2, x_3, g_1, g_3, \sigma_2, \tau_2)$. Function φ is defined by (3.1) where $\gamma = 1$ if $\bar{l} = 0$ or $\gamma = \sigma_2/\tau_2$ if $\bar{l} = 1$. Set $\bar{l} := 0$ and go to step 12.

Step 11: Set $g_3 := g_1$.

Step 12: Compute $\sigma_1 := g_1^T x_1$ and $\tau_1 := g_1^T g_3$. If $\sigma_1 \leqq 0$ or $\tau_1 \leqq 0$ go to step 4 else go to step 13.

Step 13: Compute $s := \varphi(x_1, g_3, g, s, \sigma_1, \tau_1)$. Function φ is defined by (3.1) where $\gamma = 1$ if $\bar{l} = 0$ or $\gamma = \sigma_1/\tau_1$ if $\bar{l} = 1$. Set $s := -s$.

Step 14: If $m \geqq 3$ set $g_3 := g_2$, $x_3 := x_2$ and $\sigma_3 := \sigma_2$.

Step 15: If $m \geqq 2$ set $g_2 := g_1$, $x_2 := x_1$ and $\sigma_2 := \sigma_1$.

Step 16: If $-s^T g \geqq \varepsilon_0 \|s\| \|g\|$ go to step 17 else go to step 4.

Step 17: Set $x_1 := x$, $g_1 := g$, $F_1 := F$. Use a standard procedure to determine the steplength α so that $F - F_1 \leqq \varepsilon_1 \alpha s^T g_1$ and $s^T g \geqq (1 - \varepsilon_2) s^T g_1$ holds, where F and g are new values $F := F(x)$ and $g := g(x)$ at the point $x := := x_1 + \alpha s$. (These values are determined in present step by use of a standard procedure.)

Step 18: Set $k := k + 1$ and go to step 2.

Algorithm 3.1 uses two integers l and m. Here l is a parameter controlling whether we use the value $\gamma = 1$ $(l = 0)$ or the value $\gamma = \sigma/\tau (l = 1)$ and m is a maximum number of BFGS updates in each iteration ($m \leqq 3$). In the step 17 of Algorithm 3.1 we can use any standard procedure for the determination of the steplength α. The safeguarded cubic interpolation has been used in our realization of the algorithm. Values ε_0, ε_1 and ε_2 in steps 16 and 17 of Algorithm 3.1 are usually small. Numerical experiments were carried out with the values $\varepsilon_0 = 10^{-3}$ and $\varepsilon_1 = \varepsilon_2 = 10^{-2}$.

4. NUMERICAL EXPERIMENTS

Efficiency of Algorithm 3.1 was tested by means of 18 standard problems

1) $F(x) = (10(x_1 - x_2)^2 + (x_1 - 1)^2)^4$
   $x = [-1{\cdot}2;\ 1{\cdot}0]^T$
2) $F(x) = (10(x_1 - x_2)^2 + (x_1 - 1)^2)^{1/4}$
   $x = [-1{\cdot}2;\ 1{\cdot}0]^T$
3) $F(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2$
   $x = [-1{\cdot}2;\ 1{\cdot}0]^T$
4) $F(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + 90(x_3^2 - x_4)^2 + (x_3 - 1)^2 +$
   $+ 10{\cdot}1((x_2 - 1)^2 + (x_4 - 1)^2) + 19{\cdot}8(x_2 - 1)(x_4 - 1)$
   $x = [-3{\cdot}0;\ -1{\cdot}0;\ -3{\cdot}0;\ -1{\cdot}0]^T$
5) $F(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$
   $x = [3{\cdot}0;\ -1{\cdot}0;\ 0{\cdot}0;\ 1{\cdot}0]^T$
6) $F(x) = (\exp(x_1) - x_2)^4 + 100(x_2 - x_3)^6 + \mathrm{tg}^4(x_3 - x_4) + x_1^8 + (x_4 - 1)^2$
   $x = [1{\cdot}0;\ 2{\cdot}0;\ 2{\cdot}0;\ 2{\cdot}0]^T$
7) $F(x) = \sum_{i=1}^{13} ((x_4 \exp(-x_1 z_i) - x_5 \exp(-x_2 z_i) + x_6 \exp(-x_3 z_i)) - y_i)^2$
   $y_i = \exp(-z_i) - 5\exp(-10z_i) + 3\exp(-4z_i);\ z_i = i/10$
   $x = \sum_{i=1}^{6} e_i + e_2$
8) $F(x) = \frac{1}{2}(20 \sum_{i=1}^{6} (16\text{-}i)(x_i - 1)^2)$
   $x = 0$
9) $F(x) = \frac{1}{2}(20 \sum_{i=1}^{6} (16 - i)(x_i - 1)^2) + \frac{1}{40}(20 \sum_{i=1}^{6} (16 - i)(x_i - 1)^2)^2$
   $x = 0$
10) $F(x) = (1 - x_1)^2 + (1 - x_{10})^2 + 10 \sum_{i=1}^{9} (10\text{-}i)(x_i^2 - x_{i+1})^2$
   $x = e_{10} - 1{\cdot}2e_1$
11) $F(x) = (\sum_{i=1}^{10} i^3 (x_i - 1)^2)^3$
   $x = 0$
12) $F(x) = (\sum_{i=1}^{10} i^3 (x_i - 1)^2)^{1/3}$
   $x = 0$
13) $F(x) = \sum_{i=1}^{10} (100(x_i^2 - x_{i+10})^2 + (x_i - 1)^2)$
   $x = \sum_{i=1}^{10} (e_{i+10} - 1{\cdot}2e_i)$

14) $F(x) = \sum_{i=1}^{5} (100(x_i^2 - x_{i+5})^2 + (x_i - 1)^2 + 90(x_{i+10}^2 + x_{i+15})^2 + (x_{i+10} - 1)^2 +$
$+ 10{\cdot}1((x_{i+5} - 1)^2 + (x_{i+15} - 1)^2) + 19{\cdot}8(x_{i+5} - 1)(x_{i+15} - 1))$

$x = -\sum_{i=1}^{5} (3e_i + e_{i+5} + 3e_{i+10} + e_{i+15})$

15) $F(x) = \sum_{i=1}^{5} ((x_i + 10x_{i+5})^2 + 5(x_{i+10} - x_{i+15})^2 + (x_{i+5} - 2x_{i+10})^4 +$
$+ 10(x_i - x_{i+15})^4)$

$x = \sum_{i=1}^{5} (3e_i - e_{i+5} + e_{i+15})$

16) $F(x) = \sum_{i=1}^{30} f_i^2(x)$

$f_i(x) = 420x_i + (i - 15)^3 + \sum_{\substack{j=1 \\ j \neq i}}^{30} \left(x_j^2 + \frac{i}{j}\right)^{1/2} \left(\sin^5 \log \left(x_j^2 + \frac{i}{j}\right)^{1/2} +\right.$

$\left. + \cos^5 \log \left(x_j^2 + \frac{i}{j}\right)^{1/2}\right)$

$x = -2{\cdot}8742711 \,.\, \sum_{i=1}^{30} e_i f_i(0)$

17) $F(x) = \sum_{i=1}^{30} (y_i - \sum_{j=1}^{30} (a_{ij} \sin x_j + b_{ij} \cos x_j))^2$

$y_i = \sum_{j=1}^{30} (a_{ij} \sin \xi_j + b_{ij} \cos \xi_j)$

a_{ij}, b_{ij} – random coefficients uniformly distributed within the interval $\langle -100, +100 \rangle$

ξ_j, δ_j – random coefficients uniformly distributed within the interval $\langle -\pi, +\pi \rangle$

$x = \xi + 0{\cdot}1\delta$

18) $F(x) = 1 - \exp(-\frac{1}{60} \sum_{i=1}^{30} x_i^2)$

$x = \sum_{i=1}^{30} (-1)^i \left(1 + \frac{i}{30}\right) e_i$

The objective function $F(x)$ and the initial vector x are given for each problem. Here e_i is ith column of the unit matrix of a desired order. The minimal value of the objective function is always zero. Results of the tests are shown in Table 1.

Columns in Table 1 correspond to combinations of values l (choice of γ) and m (number of BFGS updates). Rows in Table 1 correspond to the test problems given above. Table 1 contains two values for each run, which are separated by the stroke. The first is the number of iterations and the second is the number of function evaluations. An asterisk in the row 7 shows, that an alternative local minimum was

Table 1.

	$l = 0$			$l = 1$		
	$m = 1$	$m = 2$	$m = 3$	$m = 1$	$m = 2$	$m = 3$
1	169—185	290—327	69— 91	48— 50	49— 52	48— 51
2	86—219	39—101	28— 95	54—111	35— 87	36—116
3	30— 76	30— 57	34— 54	40— 58	46— 60	39— 53
4	55—112	45— 94	47— 126	102—120	54— 65	42— 48
5	118—232	106—214	36— 60	195—238	111—137	148—188
6	A	244—255	81— 84	125—179	55— 64	65— 74
7*	331—527	208—281	51— 72	A	146—194	86—111
8	6— 13	6— 13	6— 13	11— 13	11— 13	10— 12
9	22— 44	22— 51	19— 43	13— 14	15— 16	15— 16
10	A	A	385—1106	A	A	275—320
11	A	A	A	110—119	113—128	101—107
12	240—688	266—650	265— 791	256—279	242—282	154—183
13	21— 52	16— 25	22— 30	51— 64	28— 36	28— 43
14	58—119	45— 94	47— 126	124—150	55— 65	42— 48
15	261—518	276—554	36— 60	149—182	A	204—243
16	10— 21	12— 25	16— 33	10— 12	10— 12	10— 12
17	240—481	246—493	252— 505	A	294—319	357—404
18	2— 7	2— 7	2— 7	2— 7	2— 7	2— 7

found (instead of global minimum). The letter A shows that 300 iterations did not suffice to find a minimum.

To compare known methods for large-scale unconstrained minimization Table 2 has been set. Columns of Table 2 correspond to the PARTAN method [18], the method of conjugate gradients (CG method) with formula (1.5a) and with restart after each $2n$ iterations, the method of Nazareth [12], the method of Beale [1] modified as in [17], the two step BFGS method of Shanno [20] and our method with $l = 1$ and $m = 3$. The meaning of numbers in Table 2 is the same as in Table 1.

The same termination criteria, namely $\|g_k\| \leqq 10^{-8}$ or $F_k \leqq 10^{-16}$ or $\|x_k - x_{k-1}\| \leqq 10^{-8}$ and $\|x_{k-1} - x_{k-2}\| \leqq 10^{-8}$ were used for all methods in both tables. The results slightly differ since different initial estimates of the steplength α_k were used. Results in Table 1 correspond to initial estimate

$$\alpha_k = \min\left(1, 4\,\frac{\tilde{F} - F_k}{s_k^{\mathrm{T}} g_k}\right)$$

while results in Table 2 correspond to initial estimate

$$\alpha_k = 2\,\frac{\tilde{F} - F_k}{s_k^{\mathrm{T}} g_k}$$

(here $\tilde{F}$ is a lower bound for minimum value of objective function $F(x)$).

Table 2.

	PARTAN method	CG method	Nazareth [12]	Beale [1]	Shanno [20]	m-step BFGS method
1	27— 38	29— 92	27— 79	31— 94	24— 28	24— 27
2	10— 40	56—123	43—115	24— 63	28— 73	37— 100
3	41— 90	27— 47	25— 48	27— 50	23— 44	23— 37
4	45— 79	150—446	130—377	120— 356	34— 53	30— 46
5	117— 218	93—177	77—168	51— 91	89—153	82— 159
6	103— 226	34— 53	55—110	34— 56	49— 80	39— 56
7*	A	231—649	91—252	98— 277	131—345	55— 140
8	14— 15	16— 17	24— 39	21— 22	10— 11	9— 10
9	17— 19	18— 26	30— 72	20— 28	14— 18	14— 18
10	A	A	A	378—1535	A	266—1015
11	22— 701	132—309	126—320	105— 219	102—221	106— 174
12	391—1107	270—710	298—830	260— 688	225—564	198— 468
13	30— 57	31— 43	28— 62	31— 43	22— 38	18— 28
14	45— 79	186—484	139—307	126— 307	42— 74	30— 46
15	123— 229	72—132	62—120	57— 105	123—267	89— 173
16	12— 13	36— 37	24— 37	36— 37	10— 11	10— 11
17	A	225—442	276—544	213— 397	189—352	243— 475
18	6— 7	7— 11	7— 8	9— 14	6— 7	6— 7

5. CONCLUSION

The numerical experiments show high efficiency of the m-step BFGS method when complicated problems are solved (problems 10—18). This method has been implemented in the software package for optimization and nonlinear approximation SPONA (see [10]) as program POPT 96.

(Received October 20, 1981.)

REFERENCES

[1] E. M. L. Beale: A derivation of conjugate gradients. In: Numerical Methods for Non-linear Optimization (F. A. Lootsma ed.), Academic Press, London 1972, 39—43.

[2] M. C. Biggs: Minimization algorithms making use of non-quadratic properties of the objective function. J. Inst. Math. Appl. *8* (1971), 3, 315—327.

[3] C. G. Broyden: The convergence of a class of double rank minimization algorithms 2. The new algorithm. J. Inst. Math. Appl. *6* (1970), 3, 222—231.

[4] A. G. Buckley: A combined conjugate gradient quasi-Newton minization algorithm. Math. Programming *15* (1978), 2, 200—210.

[5] A. G. Buckley: Extending the relationship between the conjugate gradient and BFGS algorithms. Math. Programming *15* (1978), 3, 343—348.

[6] R. Fletcher, C. M. Reeves: Function minimization by conjugate gradients. Comput. J. *7* (1964), 2, 149—154.
[7] R. Fletcher: A new approach to variable metric algorithms. Comput. J. *13* (1970), 3, 317—322.
[8] D. Goldfarb: A family of variable metric algorithms derived by variational means. Math. Comp. *24* (1970), 109, 23—26.
[9] M. R. Hestenes, E. Stiefel: Methods of conjugate gradients for solving linear systems. J. Res. Nat. Bur. Standards *49* (1952), 6, 409—439.
[10] L. Lukšan: Software package for optimization and nonlinear approximation. Proc. of 2nd IFAC/IFIP Symposium on software for computer control, Prague 1979.
[11] L. Nazareth: A relationship between the BFGS and conjugate gradient algorithms. SIAM J. Numer. Anal. *16* (1979), 5, 794—800.
[12] L. Nazareth: A conjugate direction algorithm without line searches. J. Optim. Theory Appl. *23* (1977), 3, 373—387.
[13] J. Nocedal: Updating quasi-Newton matrices with limited storage. Math. Comp. *35* (1980), 151, 773—782.
[14] S. S. Oren, D. G. Luenberger: Self-scaling variable metric SSVM algorithms 1. Criteria and sufficient conditions for scaling a class of algorithms. Management Sci. *20* (1974), 5, 845—862.
[15] A. Perry: A modified conjugate gradient algorithm. Oper. Res. *26* (1978), 6, 1073—1078.
[16] E. Polak, G. Ribiere: Note sur la convergence de methodes des directions conjugees. Revue Fr. Inf. Rech. Oper. *16-R1* (1969), 35—43.
[17] M. J. D. Powell: Restart procedure for the conjugate gradient method. Math. Programming *12* (1977), 2, 241—254.
[18] B. V. Shah, R. J. Buehler, O. Kempthorne: Some algorithms for minimizing a function of several variables. SIAM J. *12* (1964), 1, 74—92.
[19] D. F. Shanno: Conditioning of quasi-Newton methods for function minimization. Math. Comp. *24* (1970), 111, 647—656.
[20] D. F. Shanno: Conjugate gradient methods with inexact searches. Math. Oper. Res. *3* (1978), 3, 244—256.
[21] D. F. Shanno, K. H. Phua: Matrix conditioning and nonlinear optimization. Math. Programming *14* (1978), 2, 149—160.

Ing. Ladislav Lukšan, CSc., Středisko výpočetní techniky ČSAV (General Computing Centre — Czechoslovak Academy of Sciences), Pod vodárenskou věží 4, 182 07 Praha 8. Czechoslovakia.