KYBERNETIKA — VOLUME *29* (1993), NUMBER 4, PAGES 305 – 324

INEXACT TRUST REGION METHOD FOR LARGE SPARSE NONLINEAR LEAST SQUARES

LADISLAV LUKŠAN

The main purpose of this paper is to show that linear least squares methods based on bidiagonalization, namely the LSQR algorithm, can be used for generation of trust region path. This property is a basis for an inexact trust region method which uses the LSQR algorithm for direction determination. This method is very efficient for large sparse nonlinear least squares as it is supported by numerical experiments.

1. INTRODUCTION

Inexact trust region methods are frequently used for general large-scale unconstrained minimization where we find the local minimizer $x^* \in R^n$ of the function $f : R^n \to R$ which has continuous second-order derivatives. A typical inexact trust region method can be represented by the following algorithm.

Algorithm 1.1.

Data: $0 < \beta_1 < \beta_2 < 1 < \gamma_1 < \gamma_2$, $0 < \rho_1 < \rho_2 < 1$, $0 < \varepsilon_2 < 1$, $0 < \Delta_{\max}$.

Step 1: Choose an initial point $x \in R^n$ and an initial trust region bound $0 < \Delta \le \Delta_{\max}$. Compute the value $f := f(x)$ of the objective function $f : R^n \to R$ at the point $x \in R^n$.

Step 2: Compute the gradient $g := g(x)$ of the objective function $f : R^n \to R$ at the point $x \in R^n$. If $\|g\| \le \varepsilon_2$ then stop, otherwise determine the matrix B which is an approximation of the Hessian matrix of the objective function $f : R^n \to R$ at the point $x \in R^n$.

Step 3: Determine the current precision $0 < \omega < 1$ and compute the vector $d \in R^n$ so that

(a) $\|d\| \le \Delta$

(b) $\|d\| < \Delta \Rightarrow \|Bd + g\| \le \omega \|g\|$

(c) $Q(d) \le -\frac{1}{2}\|g\| \min(\|d\|, \|g\|/\|B\|)$

where

$$Q(d) = \frac{1}{2} d^{\mathrm{T}} B d + d^{\mathrm{T}} g \tag{1.1}$$

is a local quadratic approximation of the objective function $f : R^n \to R$.

Step 4: Set $x^+ := x + d$. Compute the value $f^+ := f(x^+)$ of the objective function $f : R^n \to R$ at the point $x^+ \in R^n$ and the ratio

$$\rho = \frac{f^+ - f}{Q(d)} \tag{1.2}$$

If $\rho < \rho_1$ then compute the value Δ^+ using the quadratic interpolation and set

$$\begin{array}{ll} \Delta := \beta_1 \|d\| & \text{if } \Delta^+ < \beta_1 \|d\|, \\ \Delta := \beta_2 \|d\| & \text{if } \Delta^+ > \beta_2 \|d\|, \\ \Delta := \Delta^+ & \text{otherwise.} \end{array}$$

If $\rho_1 \le \rho \le \rho_2$ then set $\Delta^+ := \Delta$ and $\Delta := \min(\Delta^+, \gamma_2 \|d\|)$. If $\rho_2 < \rho$ then set $\Delta^+ := \max(\Delta, \gamma_1 \|d\|\}$ and $\Delta := \min(\Delta^+, \gamma_2 \|d\|, \Delta_{\max})$.

Step 5: If $\rho \le 0$ then go to Step 3, otherwise set $x := x^+$, $f := f^+$ and go to Step 2.

Inexact trust region methods have strong convergence properties (see [6], [7], [8]). Even if they also work well for indefinite matrices, we confine our attention to positive semidefinite case which appears in nonlinear least squares.

The most complicated part of Algorithm 1.1 is computation of the vector $d \in R^n$ satisfying the conditions (a), (b), (c). There exist three basic possibilities for positive semidefinite case. First, the vector $d \in R^n$ can be obtained as a solution of the subproblem

$$d = \operatorname*{arg\,min}_{\|d(\lambda)\| \le \Delta} Q(d(\lambda))$$

which leads to the repeated solution of the equation $(B + \lambda I) d(\lambda) + g = 0$ for selected values of λ. This way gives well-convergent algorithms, especially in connection with the Newton method, but for large number of variables, it is time consuming.

The second possibility, so-called dog-leg strategy, consists in computation of two vectors $d_1 \in R^n$ and $d_2 \in R^n$ such that $g^{\mathrm{T}} B g d_1 + \|g\|^2 g = 0$ and $B d_2 + g = 0$. The resulting vector $d \in R^n$ is then obtained as $d = \lambda d_1$ if $\|d_1\| \ge \Delta$, $d = d_1 + \lambda(d_2 - d_1)$ if $\|d_1\| < \Delta < \|d_2\|$, and $d = \|d_2\|$ if $\|d_2\| \le \Delta$, where the scaling factor $\lambda > 0$ is chosen so that $\|d\| = \Delta$. This way is more economical since the equation $B d_2 + g = 0$ can be solved inaccurately ($\|B d_2 + g\| \le \omega \|g\|$) by some iterative method.

The third possibility is very natural. The equation $Bd + g = 0$ is solved by some iterative method which generates the vectors $d_i \in R^n$, $i \in N$, having the following properties:

(A) There exists an index $k \in N$, such that

$$\|Bd_k + g\| \le \omega \|g\| \tag{1.3}$$

for a given $0 < \omega < 1$.

(B) The sequence $Q(d_i)$, $1 \le i \le k$, is decreasing, i. e.

$$Q(d_{i+1}) < Q(d_i) \tag{1.4}$$

for $1 \le i < k$.

(C) The sequence $\|d_i\|$, $1 \le i \le k$, is increasing, i. e.

$$\|d_{i+1}\| > \|d_i\| \tag{1.5}$$

for $1 \le i < k$.

(D) It holds that

$$Q(\lambda d_1) \le -\frac{1}{2}\|g\| \, \|\lambda d_1\| \tag{1.6}$$

for $0 \le \lambda \le 1$, and

$$Q(d_i) \le -\frac{1}{2}\frac{\|g\|^2}{\|B\|} \tag{1.7}$$

for $1 \le i \le k$.

The resulting vector $d \in R^n$ is then obtained as $d = \lambda d_1$ if $\|d_1\| \ge \Delta$, $d = d_i + \lambda(d_{i+1} - d_i)$ if $\|d_i\| < \Delta < \|d_{i+1}\|$ for some $1 \le i < k$, and $d = \|d_k\|$ if $\|d_k\| \le \Delta$, where the scaling factor $\lambda > 0$ is chosen so that $\|d\| = \Delta$.

Steihaug [8] has proved that all above conditions are satisfied for the conjugate gradient method. Our main purpose is to prove that these conditions are also satisfied for more complicated iterative methods appearing in least squares solutions.

Consider the nonlinear least squares problem which is a special minimization problem where the objective function $f := R^n \to R$ has the form

$$f(x) = \frac{1}{2}\sum_{i=1}^{m} f_i^2(x) \tag{1.8}$$

and the functions $f_i : R^n \to R$, $1 \le i \le m$, have continuous second-order derivatives. Denote $f = f(x)$, $f_i = f_i(x)$, $1 \le i \le m$ and $g = g(x)$, $g_i = g_i(x)$, $1 \le i \le m$, the values and the gradients of the functions $f : R^n \to R$, $f_i : R^n \to R$, $1 \le i \le m$, at the point $x \in R^n$ respectively and set

$$A = A(x) = \begin{bmatrix} g_1^{\mathrm{T}}(x) \\ \dots \\ g_m^{\mathrm{T}}(x) \end{bmatrix}, \qquad b = b(x) = -\begin{bmatrix} f_1(x) \\ \dots \\ f_m(x) \end{bmatrix}. \tag{1.9}$$

Then

$$f = \frac{1}{2}b^{\mathrm{T}}b\,, \qquad g = -A^{\mathrm{T}}b\,, \tag{1.10}$$

and if we denote $x^+ = x + d$ as a new vector of variables, we get after linearization

$$f(x^+) = \frac{1}{2}\sum_{i=1}^{m} f_i^2(x^+) \approx \frac{1}{2}\|Ad - b\|^2$$

Therefore the optimal direction vector $d^* \in R^n$ can be obtained as a solution of the linearized problem

$$d^* = \underset{d\in R^n}{\arg\min}\ \|Ad - b\| \tag{1.11}$$

Since the function $\|Ad-b\|$ is convex the vector $d^* \in R^n$ is a solution of the problem (1.11) if and only if

$$A^{\mathrm{T}}(Ad^* - b) = 0 \tag{1.12}$$

If we denote $B = A^{\mathrm{T}}A$ and if we use (1.10), we get the equation $Bd^* + g = 0$ which is equivalent to (1.12). Therefore it suffices to substitute $B = A^{\mathrm{T}}A$ in Algorithm 1.1 to adapt it for nonlinear least squares. Especially the quadratic function (1.1) takes the form

$$Q(d) = \frac{1}{2}d^{\mathrm{T}}A^{\mathrm{T}}Ad - d^{\mathrm{T}}A^{\mathrm{T}}b \tag{1.13}$$

Using the substitution $B = A^{\mathrm{T}}A$ we can transform the conjugate gradient method to solve the normal equation (1.12). The resulting method is the CGLS algorithm (see [5] as an example) which is represented by the following iterative process

$$d_o = 0, \qquad r_o = b, \tag{1.14a}$$

$$v_1 = A^{\mathrm{T}}r_o, \qquad \gamma_1 = \|v_1\|^2 \tag{1.14b}$$

$$p_1 = v_1 \tag{1.14c}$$

and

$$u_i = Ap_i, \qquad \delta_i = \|u_i\|^2 \tag{1.14d}$$

$$d_i = d_{i-1} + \frac{\gamma_i}{\delta_i}p_i, \qquad r_i = r_{i-1} - \frac{\gamma_i}{\delta_i}u_i \tag{1.14e}$$

$$v_{i+1} = A^{\mathrm{T}}r_i, \qquad \gamma_{i+1} = \|v_{i+1}\|^2 \tag{1.14f}$$

$$p_{i+1} = v_{i+1} + \frac{\gamma_{i+1}}{\gamma_i}p_i \tag{1.14g}$$

for $i \in N$. As it was proved by Steihaug [8] for the CG method, the vectors $d_i \in R^n$, $i \in N$, obtained by (1.14) satisfy the conditions (A), (B), (C), (D). The inequality

(1.3) has the form $\gamma_k \le \omega^2 \gamma_1$ since $A^{\mathrm{T}}(Ad_k - b) = v_k$ and $\|A^{\mathrm{T}}(Ad_k - b)\|^2 = \gamma_k$ by (1.14f).

The CGLS algorithm is not the best one for linear least squares. Methods based on bidiagonalization [1], [4], namely the LSQR algorithm proposed in [5], were proved to be numerically more stable. In the next section we shall study properties of such methods with regard to conditions (A), (B), (C), (D) which have to be satisfied.

2. BIDIAGONALIZATION AND LINEAR LEAST SQUARES

Consider the problem which consists in finding a vector $d^* \in R^n$ such that

$$d^* = \underset{d \in R^n}{\arg\min} \; \|Ad - b\|. \tag{2.1}$$

Since the function $\|Ad-b\|$ is convex, the vector $d^* \in R^n$ is a solution of the problem (2.1) if and only if

$$A^{\mathrm{T}}(Ad^* - b) = 0. \tag{2.2}$$

The problem (2.1) can be solved iteratively using a bidiagonalization procedure proposed in [1] and [4]. In this case

$$\beta_1 u_1 = b, \tag{2.3a}$$

$$\alpha_1 v_1 = A^{\mathrm{T}} u_1, \tag{2.3b}$$

and

$$\beta_{i+1} u_{i+1} = A v_i - \alpha_i u_i, \tag{2.3c}$$

$$\alpha_{i+1} v_{i+1} = A^{\mathrm{T}} u_{i+1} - \beta_{i+1} v_i, \tag{2.3d}$$

for $i \in N$, where the right hand sides are assumed to be nonzero and the coefficients on the left hand sides are chosen so that the corresponding vectors have unit norms. If some right hand side becomes zero then we formally set both the coefficient and the vector on the left hand side equal to zero and we stop the iterative process. Namely if $b = 0$ or $A^{\mathrm{T}}b = 0$ we set $\beta_1 = 0$, $u_1 = 0$ or $\alpha_1 = 0$, $v_1 = 0$ respectively.

It can be easily proved by induction (see [1] and [4]) that for $\alpha_i > 0$, $\beta_i > 0$, $1 \le i \le k$, the vectors $v_i \in R^n$, $1 \le i \le k$, are nonzero and mutually orthogonal and the vectors $u_i \in R^m$, $1 \le i \le k$, have the same property.

The iterative process (2.3) can be written in the matrix form

$$U_{i+1}(\beta_1 e_1) = b, \tag{2.4a}$$

$$A V_i = U_{i+1} B_i, \tag{2.4b}$$

$$A^{\mathrm{T}} U_{i+1} = V_i B_i^{\mathrm{T}} + \alpha_{i+1} v_{i+1} e_{i+1}^{\mathrm{T}}, \tag{2.4c}$$

for $i \in N$, where $V_i = [v_1, \dots, v_i] \in R^{n\times i}$, $V_i^{\mathrm{T}} V_i = I$, $U_{i+1} = [u_1, \dots, u_{i+1}] \in R^{n\times(i+1)}$, and

$$B_i = \begin{bmatrix} \alpha_1, & 0, & \dots, & 0 \\ \beta_2, & \alpha_2, & \dots, & 0 \\ 0, & \beta_2, & \dots, & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0, & 0, & \dots, & \alpha_i \\ 0, & 0, & \dots, & \beta_{i+1} \end{bmatrix}. \tag{2.5}$$

If $\alpha_i > 0$, $\beta_i > 0$, $1 \le i \le k$, then the lower bidiagonal matrices $B_i \in R^{(i+1)\times i}$, $1 \le i \le k$, have full column rank. If $\beta_{i+1} > 0$ then $U_{i+1}^{\mathrm{T}} U_{i+1} = I$. In the other case $U_{i+1} = [U_i, 0]$, $B_i = [L_i^{\mathrm{T}}, 0]^{\mathrm{T}}$, where $L_i \in R^{i\times i}$ is a nonsingular square lower bidiagonal matrix, and (2.4) can be rewritten in the form

$$U_i(\beta_1 e_1) = b, \tag{2.6a}$$

$$AV_i = U_i L_i, \tag{2.6b}$$

$$A^{\mathrm{T}} U_i = V_i L_i^{\mathrm{T}} + \alpha_{i+1} v_{i+1} e_{i+1}^{\mathrm{T}}, \tag{2.6c}$$

for $i \in N$, where $U_i = [u_1, \dots, u_i] \in R^{n\times i}$, $U_i^{\mathrm{T}} U_i = I$.

Together with the iterative process (2.3) we consider the sequence of vectors $d_i \in R^n$, $1 \le i \le k$, such that

$$d_i = \operatorname*{arg\,min}_{d\in\mathcal{R}(V_i)} \|Ad - b\|. \tag{2.7}$$

Lemma 2.1. Consider the iterative process (2.3) with $\alpha_i > 0$, $\beta_i > 0$, $1 \le i \le k$. Let $d_i \in R^n$, $1 \le i \le k$, be the sequence of vectors defined by (2.7). Then, for $1 \le i \le k$,

$$d_i = V_i y_i \tag{2.8a}$$

where

$$y_i = \operatorname*{arg\,min}_{y\in R^i} \|B_i y - \beta_1 e_1\|. \tag{2.8b}$$

If $\beta_{i+1} = 0$ (it can be satisfied only for $i = k$) then $\|Ad_i - b\| = 0$.

P r o o f. If $d \in \mathcal{R}(V_i)$ then necessarily $d = V_i y$ for some $y \in R^i$. If $\beta_{i+1} > 0$ then

$$\|Ad - b\| = \|AV_i y - b\| = \|U_{i+1}(B_i y - \beta_1 e_1)\| = \|B_i y - \beta_1 e_1\|,$$

by (2.4a) and (2.4b), since $U_{i+1}^{\mathrm{T}} U_{i+1} = I$, so that (2.7) is equivalent to (2.8b). If $\beta_{i+1} = 0$ then

$$\|Ad - b\| = \|AV_i y - b\| = \|U_i(L_i y - \beta_1 e_1)\| = 0,$$

by (2.6a) and (2.6b), since the lower bidiagonal square matrix L_i is nonsingular and, therefore, there exists a solution $y_i \in R^i$ of the equation $L_i y = \beta_1 e_1$. □

Corollary 2.1. Let the assumptions of Lemma 2.1 be satisfied. Then, for $1 \le i \le k$,

$$d_i = V_i y_i \tag{2.9a}$$

where

$$y_i = \alpha_1 \beta_1 (B_i^{\mathrm{T}} B_i)^{-1} e_1. \tag{2.9b}$$

Proof. Since the function $\|B_i y - \beta_1 e_1\|$ is convex, the vector $y_i \in R^i$ is a solution of the problem (2.8b) if and only if $B_i^{\mathrm{T}}(B_i y - \beta_1 e_1) = 0$. Because the lower bidiagonal matrices $B_i \in R^{(i+1)\times i}$, $1 \le i \le k$, have full column rank, we can write $y_i = \beta_1 (B_i^{\mathrm{T}} B_i)^{-1} B_i^{\mathrm{T}} e_1$ which together with $B_i^{\mathrm{T}} e_1 = \alpha_1 e_1$ (see (2.5)) gives (2.9b). □

Theorem 2.1. Consider the iterative process (2.3) and sequence of vectors (2.7). Then either $d^* = 0$ is a solution of the problem (2.1) or there exists an index $k \le n$ such that $d^* = d_k \in \mathcal{R}(V_k)$ is a solution of the problem (2.1) and, moreover, $\alpha_i > 0$, $\beta_i > 0$ for $1 \le i \le k$.

Proof. If either $b = 0$ or $A^{\mathrm{T}} b = 0$ then $d^* = 0$ is a trivial solution of the problem (2.1). In this case either $\beta_1 = 0$ or $\alpha_1 = 0$. Suppose now that $\alpha_i > 0$, $\beta_i > 0$ for $1 \le i \le k \le n$. If $k = n$ then $\mathcal{R}(V_n) = R^n$ since the vectors v_i, $1 \le i \le n$, are nonzero and mutually orthogonal. Therefore

$$d_n = \operatorname*{arg\,min}_{d \in \mathcal{R}(V_n)} \|Ad - b\| = \operatorname*{arg\,min}_{d \in R^n} \|Ad - b\| = d^*$$

is a solution of the problem (2.1). If $k < n$ and $\beta_{k+1} = 0$ then $\|Ad_k - b\| = 0$ by Lemma 2.1 so that $d^* = d_k$ is a solution of the problem (2.1). If $k < n$ and $\alpha_{k+1} = 0$ then $A^{\mathrm{T}} U_{k+1} = V_k B_k^{\mathrm{T}}$ by (2.4c) so that

$$A^{\mathrm{T}}(Ad_k - b) = A^{\mathrm{T}}(AV_k y_k - b) = A^{\mathrm{T}} U_{k+1}(B_k y_k - \beta_1 e_1) = V_k B_k^{\mathrm{T}}(B_k y_k - \beta_1 e_1) = 0,$$

by (2.4a) and (2.4b), since $B_k^{\mathrm{T}}(B_k y_k - \beta_1 e_1) = 0$ by (2.8b), and $d^* = d_k$ is a solution of the problem (2.1) by (2.2). □

Theorem 2.1 shows that $d^* = d_i$ is a solution of the problem (2.1) whenever $\alpha_{i+1} = 0$ or $\beta_{i+1} = 0$. The next lemma gives an important estimation in case $\alpha_{i+1} > 0$ and $\beta_{i+1} > 0$.

Lemma 2.2. Let the assumptions of Lemma 2.1 be satisfied. Then, for $1 \le i \le k$,

$$\|A^{\mathrm{T}}(Ad_i - b)\| = \alpha_{i+1}\beta_{i+1}|v_i^{\mathrm{T}} d_i|. \tag{2.10}$$

P r o o f. Let $\alpha_{i+1} > 0$ and $\beta_{i+1} > 0$. Then using (2.4a) and (2.4c) we get

$$\begin{aligned} A^{\mathrm{T}}(Ad_i - b) &= A^{\mathrm{T}}(AV_i y_i - b) = A^{\mathrm{T}}U_{i+1}(B_i y_i - \beta_1 e_1) = \\ &= (V_i B_i^{\mathrm{T}} + \alpha_{i+1} v_{i+1} e_{i+1}^{\mathrm{T}})(\beta_i y_i - \beta_1 e_1) = \\ &= \alpha_{i+1} v_{i+1} e_{i+1}^{\mathrm{T}}(B_i y_i - \beta_1 e_1) = \alpha_{i+1}\beta_{i+1} v_{i+1} e_i^{\mathrm{T}} y_i \end{aligned}$$

since $B_i^{\mathrm{T}}(B_i y_i - \beta_1 e_1) = 0$ by (2.8b), $e_{i+1}^{\mathrm{T}} B_i = \beta_{i+1} e_i^{\mathrm{T}}$ by (2.5), and $e_{i+1}^{\mathrm{T}} e_i = 0$. But $V_i^{\mathrm{T}} V_i = I$ and, therefore, $V_i^{\mathrm{T}} d_i = V_i^{\mathrm{T}} V_i y_i = y_i$ so that $e_i^{\mathrm{T}} y_i = e_i^{\mathrm{T}} V_i^{\mathrm{T}} d_i = v_i^{\mathrm{T}} d_i$ which together with $\|v_{i+1}\| = 1$ gives (2.10). If $\alpha_{i+1} = 0$ or $\beta_{i+1} = 0$ then $\|A^{\mathrm{T}}(Ad_i - b)\| = 0$ by Theorem 2.1. □

Now we shall study properties of the vectors $d_i \in R^n$, $1 \le i \le k$, defined by (2.7). We shall use the notation (1.13).

Lemma 2.3. Let the assumptions of Lemma 2.1 be satisfied. Then, for $1 \le i \le k$,

$$Q(d_i) = -\frac{1}{2}\alpha_1^2\beta_1^2 e_1^{\mathrm{T}} C_i e_1 \tag{2.11}$$

and

$$\|d_i\|^2 = \alpha_1^2\beta_1^2 e_1^{\mathrm{T}} C_i^2 e_1 \tag{2.12}$$

where

$$C_i = (B_i^{\mathrm{T}} B_i)^{-1}. \tag{2.13}$$

P r o o f. Using (1.13) and (2.8a) we can write

$$Q(d_i) = \frac{1}{2} y_i^{\mathrm{T}} V_i^{\mathrm{T}} A^{\mathrm{T}} A V_i y_i - y_i^{\mathrm{T}} V_i^{\mathrm{T}} A^{\mathrm{T}} b$$

which together with (2.4a), (2.4b) and (2.9b) gives

$$\begin{aligned} Q(d_i) &= \frac{1}{2} y_i^{\mathrm{T}} B_i^{\mathrm{T}} U_{i+1}^{\mathrm{T}} U_{i+1} B_i y_i - y_i^{\mathrm{T}} B_i^{\mathrm{T}} U_{i+1}^{\mathrm{T}} b = \frac{1}{2} y_i^{\mathrm{T}} B_i^{\mathrm{T}} B_i y_i - \beta_1 y_i^{\mathrm{T}} B_i^{\mathrm{T}} e_1 = \\ &= \frac{1}{2}\alpha_1^2\beta_1^2 e_1^{\mathrm{T}} C_i B_i^{\mathrm{T}} B_i C_i e_1 - \alpha_1^2\beta_1^2 e_1^{\mathrm{T}} C_i e_1 = -\frac{1}{2}\alpha_1^2\beta_1^2 e_1^{\mathrm{T}} C_i e_1 \end{aligned}$$

since $B_i^{\mathrm{T}} e_1 = \alpha_1 e_1$ by (2.5). Similarly we get

$$\|d_i\|^2 = y_i^{\mathrm{T}} V_i^{\mathrm{T}} V_i y_i = y_i^{\mathrm{T}} y_i = \alpha_1^2\beta_1^2 (C_i e_1)^{\mathrm{T}} C_i e_1 = \alpha_1^2\beta_1^2 e_1^{\mathrm{T}} C_i^2 e_1$$

since $V_i^{\mathrm{T}} V_i = I$ and the matrix (2.13) is symmetric. □

Lemma 2.4. Let the assumptions of Lemma 2.1 be satisfied. Then, for $1 \le i < k$,

$$C_{i+1} = \begin{bmatrix} C_i + \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} C_i e_i e_i^{\mathrm{T}} C_i, & -\alpha_{i+1}\beta_{i+1}\gamma_{i+1} C_i e_i \\ -\alpha_{i+1}\beta_{i+1}\gamma_{i+1} e_i^{\mathrm{T}} C_i, & \gamma_{i+1} \end{bmatrix} \tag{2.14}$$

where

$$\gamma_{i+1} = \frac{1}{\alpha_{i+1}^2 + \beta_{i+2}^2 - \alpha_{i+1}^2 \beta_{i+1}^2 e_i^{\mathrm{T}} C_i e} > 0 \tag{2.15}$$

P r o o f . Using (2.5) we can write

$$B_i^{\mathrm{T}} B_i = \begin{bmatrix} \alpha_1, & \beta_2, & 0, & \dots, & 0, & 0 \\ 0, & \alpha_2, & \beta_3, & \dots, & 0, & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0, & 0, & 0, & \dots, & \alpha_i, & \beta_{i+1} \end{bmatrix} \begin{bmatrix} \alpha_1, & 0, & \dots, & 0 \\ \beta_2, & \alpha_2, & \dots, & 0 \\ 0, & \beta_3, & \dots, & 0 \\ \vdots & \vdots & & \vdots \\ 0, & 0, & \dots, & \alpha_i \\ 0, & 0, & \dots, & \beta_{i+1} \end{bmatrix} =$$

$$= \begin{bmatrix} \alpha_1^2 + \beta_2^2, & \alpha_2\beta_2, & 0, & \dots, & 0 \\ \alpha_2\beta_2, & \alpha_2^2 + \beta_3^2, & \alpha_3\beta_3, & \dots, & 0 \\ 0, & \alpha_3\beta_3, & \alpha_3^2 + \beta_4^2, & \dots, & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0, & 0, & 0, & \dots, & \alpha_i^2 + \beta_{i+1}^2 \end{bmatrix}$$

Therefore

$$B_{i+1}^{\mathrm{T}} B_{i+1} = \begin{bmatrix} B_i^{\mathrm{T}} B_i, & \alpha_{i+1}\beta_{i+1} e_i \\ \alpha_{i+1}\beta_{i+1} e_i^{\mathrm{T}}, & \alpha_{i+1}^2 + \beta_{i+2}^2 \end{bmatrix} \tag{2.16}$$

Since the matrix $B_{i+1}^{\mathrm{T}} B_{i+1}$ is nonsingular, it suffices to prove that $B_{i+1}^{\mathrm{T}} B_{i+1} C_{i+1} = I$ for matrices (2.14) and (2.16), which leads to straightforward computations. □

Lemma 2.5. Let the assumptions of Lemma 2.1 be satisfied. Then, for $1 \le i \le k$,

$$e_1^{\mathrm{T}} C_i^2 e_i e_1^{\mathrm{T}} C_i e_i > 0 \tag{2.17}$$

P r o o f . (By induction): Since both matrices C_1 and C_1^2 are positive definite we have $e_1^{\mathrm{T}} C_1 e_1 > 0$ and $e_1^{\mathrm{T}} C_1^2 e_1 > 0$ so that $e_1^{\mathrm{T}} C_1^2 e_1 e_1^{\mathrm{T}} C_1 e_1 > 0$. Suppose that (2.17) holds for some $i < k$. Then, using (2.14), we get

$$e_1^{\mathrm{T}} C_{i+1} e_{i+1} = -\alpha_{i+1}\beta_{i+1}\gamma_{i+1} e_1^{\mathrm{T}} C_i e_i$$

and

$$\begin{aligned} e_1^{\mathrm{T}} C_{i+1}^2 e_{i+1} &= -e_1^{\mathrm{T}}(C_i + \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} C_i e_i e_i^{\mathrm{T}} C_i)(\alpha_{i+1}\beta_{i+1}\gamma_{i+1} C_i e_i) - \\ &\quad -e_1^{\mathrm{T}}(\alpha_{i+1}\beta_{i+1}\gamma_{i+1}^2 C_i e_i) = \\ &= -\alpha_{i+1}\beta_{i+1}\gamma_{i+1}(e_1^{\mathrm{T}} C_i^2 e_i + \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} e_1^{\mathrm{T}} C_i e_i e_i^{\mathrm{T}} C_i^2 e_i + \\ &\quad +\gamma_{i+1} e_1^{\mathrm{T}} C_i e_i). \end{aligned}$$

Therefore

$$\begin{aligned} e_1^{\mathrm{T}} C_{i+1}^2 e_{i+1} e_1^{\mathrm{T}} C_{i+1} e_{i+1} &= \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1}^2 (e_1^{\mathrm{T}} C_i^2 e_i e_1^{\mathrm{T}} C_i e_i + \gamma_{i+1}(e_1^{\mathrm{T}} C_i e_i)^2 + \\ &\quad +\alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} (e_1^{\mathrm{T}} C_i e_i)^2 e_i^{\mathrm{T}} C_i^2 e_i). \end{aligned}$$

But $e_1^{\mathrm{T}} C_i^2 e_i e_1^{\mathrm{T}} C_i e_i > 0$ by inductive assumption, $\gamma_{i+1} > 0$ from positive definiteness of the matrix C_{i+1}, and $e_i^{\mathrm{T}} C_i^2 e_i > 0$ from positive definiteness of the matrix C_i^2. This together with $\alpha_{i+1} > 0$ and $\beta_{i+1} > 0$ gives $e_1^{\mathrm{T}} C_{i+1}^2 e_{i+1} e_1^{\mathrm{T}} C_{i+1} e_{i+1} > 0$. □

Theorem 2.2. Consider the iterative process (2.3) and the sequence of vectors (2.7). Let k be the index from Theorem 2.1. Then, for $1 \le i < k$,

$$Q(d_{i+1}) < Q(d_i) \tag{2.18}$$

and

$$\|d_{i+1}\| > \|d_i\|. \tag{2.19}$$

P r o o f . Using (2.14) we get

$$\begin{aligned} e_1^{\mathrm{T}} C_{i+1} e_1 &= e_1^{\mathrm{T}}(C_i + \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} C_i e_i e_i^{\mathrm{T}} C_i) e_1 = \\ &= e_1^{\mathrm{T}} C_i e_1 + \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} (e_1^{\mathrm{T}} C_i e_i)^2 > e_1^{\mathrm{T}} C_i e_1 \end{aligned}$$

since $\alpha_{i+1} > 0$, $\beta_{i+1} > 0$ by the assumption, $\gamma_{i+1} > 0$ from positive definiteness of the matrix C_{i+1}, and $(e_1^{\mathrm{T}} C_i e_i)^2 > 0$ by Lemma 2.5. This together with (2.11) gives $Q(d_{i+1}) < Q(d_i)$. Similarly using (2.14) we get

$$\begin{aligned} e_1^{\mathrm{T}} C_{i+1}^2 e_1 &= e_1^{\mathrm{T}}(C_i + \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} C_i e_i e_i^{\mathrm{T}} C_i)^2 e_1 + (\alpha_{i+1}\beta_{i+1}\gamma_{i+1} e_1^{\mathrm{T}} C_i e_i)^2 = \\ &= e_1^{\mathrm{T}} C_i^2 e_1 + 2\alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1} e_1^{\mathrm{T}} C_i^2 e_i e_1^{\mathrm{T}} C_i e_i + \\ &\quad +\alpha_{i+1}^4 \beta_{i+1}^4 \gamma_{i+1}^2 (e_1^{\mathrm{T}} C_i e_i)^2 e_i^{\mathrm{T}} C_i^2 e_i + \alpha_{i+1}^2 \beta_{i+1}^2 \gamma_{i+1}^2 (e_1^{\mathrm{T}} C_i e_i)^2 > \\ &> e_1^{\mathrm{T}} C_i^2 e_1 \end{aligned}$$

since $\alpha_{i+1} > 0$, $\beta_{i+1} > 0$ by the assumption, $\gamma_{i+1} > 0$ from positive definiteness of the matrix C_{i+1}, $e_i^{\mathrm{T}} C_i^2 e_i > 0$ from positive definiteness of the matrix C_i^2, and $(e_1^{\mathrm{T}} C_i e_i)^2 > 0$, $e_1^{\mathrm{T}} C_i^2 e_i e_1^{\mathrm{T}} C_i e_i > 0$ by Lemma 2.5. This together with (2.12) gives $\|d_{i+1}\|^2 > \|d_i\|^2$. □

Theorem 2.3. Let the assumptions of Theorem 2.2 be satisfied. Then

$$Q(\lambda d_1) \le -\frac{1}{2}\|A^{\mathrm{T}}b\| \ \|\lambda d_1\|, \tag{2.20}$$

for $0 \le \lambda \le 1$, and

$$Q(d_i) \le -\frac{1}{2}\frac{\|A^{\mathrm{T}}b\|^2}{\|A^{\mathrm{T}}A\|}, \tag{2.21}$$

for $1 \le i \le k$.

P r o o f. The equalities (2.11) and (2.12) imply

$$Q(d_1) = -\frac{1}{2}\alpha_1^2\beta_1^2 e_1^{\mathrm{T}} C_1 e_1 = -\frac{1}{2}\alpha_1^2\beta_1^2\sqrt{e_1^{\mathrm{T}}C_1^2e_1} = -\frac{1}{2}\|A^{\mathrm{T}}b\| \ \|d_1\|$$

since from (2.3a) and (2.3b) $A^{\mathrm{T}}b = \alpha_1\beta_1 v_1$ follows, which together with $\|v_1\| = 1$ gives $\|A^{\mathrm{T}}b\| = \alpha_1\beta_1$. But the function (1.13) is convex and $Q(0) = 0$ so that (2.20) holds for $0 \le \lambda \le 1$. Furthermore using (2.3c) we can write

$$\beta_2^2 = (Av_1 - \alpha_1 u_1)^{\mathrm{T}}(Av_1 - \alpha_1 u_1) = v_1^{\mathrm{T}}A^{\mathrm{T}}Av_1 - \alpha_1^2$$

since $\|u_1\| = 1$ and $v_1^{\mathrm{T}}A^{\mathrm{T}}u_1 = \alpha_1\|v_1\|^2 = \alpha_1$ (see (2.3b)). Therefore

$$\alpha_1^2 + \beta_2^2 = v_1^{\mathrm{T}}A^{\mathrm{T}}Av_1 \le \|A^{\mathrm{T}}A\| \ \|v_1\|^2 = \|A^{\mathrm{T}}A\|.$$

Now, if we use (2.11) and (2.15), we get

$$Q(d_1) = -\frac{1}{2}\alpha_1^2\beta_1^2 e_1^{\mathrm{T}}C_1e_1 = -\frac{1}{2}\frac{\alpha_1^2\beta_1^2}{\alpha_1^2+\beta_2^2} \le -\frac{1}{2}\frac{\|A^{\mathrm{T}}b\|^2}{\|A^{\mathrm{T}}A\|}$$

which together with (2.18) gives (2.21). □

We have proved that the vectors $d_i \in R^n$, $1 \le i \le k$, defined by (2.7) satisfy the conditions (A), (B), (C), (D) stated in Section 1. This fact will be used in the next section for construction of an inexact trust region algorithm. It remains to derive simple recurrence relations for the vectors $d_i \in R^n$, $1 \le i \le k$.

The most widely used iterative method for linear least squares is the LSQR algorithm proposed in [5]. This algorithm uses orthogonal matrices Q_i, $1 \le i \le k$, such that

$$Q_iB_i = \begin{bmatrix} R_i \\ 0 \end{bmatrix}, \quad Q_i(\beta_1 e_1) = \begin{bmatrix} h_i \\ \overline{\eta}_{i+1} \end{bmatrix} \tag{2.22}$$

where

$$R_i = \begin{bmatrix} \rho_1, & \sigma_2, & 0, & \dots, & 0 \\ 0, & \rho_2, & \sigma_3, & \dots, & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0, & 0, & 0, & \dots, & \rho_i \end{bmatrix}, \quad h_i = \begin{bmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \eta_i \end{bmatrix} \tag{2.23}$$

At the same time Q_i, $1 \le i \le k$, are products of Givens plane rotations and $R_i \in R^{i \times i}$, $1 \le i \le k$, are regular square upper bidiagonal matrices. The iterative process for computing elements of both the matrices R_i, $1 \le i \le k$, and the vectors h_i, $1 \le i \le k$, has the form

$$\overline{\rho}_1 = \alpha_1, \qquad \overline{\eta}_1 = \beta_1 \tag{2.24a}$$

and

$$\rho_i = \sqrt{\overline{\rho}_i^2 + \beta_{i+1}^2}, \qquad c_i = \frac{\overline{\rho}_i}{\rho_i}, \qquad s_i = \frac{\beta_{i+1}}{\rho_i}, \tag{2.24b}$$

$$\overline{\rho}_{i+1} = c_i \alpha_{i+1}, \qquad \sigma_{i+1} = s_i \alpha_{i+1}, \tag{2.24c}$$

$$\eta_i = c_i \overline{\eta}_i, \qquad , \overline{\eta}_{i+1} = -s_i \overline{\eta}_i \tag{2.24d}$$

for $1 \le i \le k$ (see [5] for detailed description).

The values $\rho_i > 0$ and η_i, $1 \le i \le k$, can be used in estimation (2.10).

Lemma 2.6. Let the assumptions of Lemma 2.1 be satisfied and let $\rho_i > 0$ and η_i, $1 \le i \le k$, are the values generated by (2.24). Then, for $1 \le i \le k$,

$$\|A^{\mathrm{T}}(Ad_i - b)\| = \alpha_{i+1}\beta_{i+1}\frac{|\eta_i|}{\rho_i} \tag{2.25}$$

Proof. Using (2.8) and (2.22) we can write

$$d_i = V_i y_i \tag{2.26a}$$

where

$$R_i y_i = h_i \tag{2.26b}$$

Then

$$v_i^{\mathrm{T}} d_i = v_i^{\mathrm{T}} V_i R_i^{-1} h_i = e_i^{\mathrm{T}} R_i^{-1} h_i = \frac{1}{\rho_i} e_i^{\mathrm{T}} h_i = \frac{\eta_i}{\rho_i}$$

which together with (2.10) gives (2.25). □

Recurrence relations for the vectors $d_i \in R^n$, $1 \le i \le k$, can be derived from (2.26). We do not give this derivation here because it is fully contained in [5]. The resulting formulas have the form

$$d_o = 0, \tag{2.27a}$$

$$p_1 = v_1, \tag{2.27b}$$

and

$$d_i = d_{i-1} + \frac{\eta_i}{\rho_i} p_i, \tag{2.27c}$$

$$p_{i+1} = v_{i+1} - \frac{\sigma_{i+1}}{\rho_i} p_i, \tag{2.27d}$$

for $1 \le i \le k$. Note that, in contrast with the CGSL method (1.14), the coefficients η_i/ρ_i, $1 \le i \le k$, in (2.27c) are not all positive (they alternate signs).

3. INEXACT TRUST REGION METHOD FOR NONLINEAR LEAST SQUARES

Now we are in a position to describe complete inexact trust region method which is a combination of Algorithm 1.1 together with the LSQR algorithm investigated in Section 2.

Algorithm 3.1.

Data: $0 < \beta_1 < \beta_2 < 1 < \gamma_1 < \gamma_2$, $0 < \rho_1 < \rho_2 < 1$, $0 < \varepsilon_1 < \varepsilon_2 < 1$, $0 < \tau_1 < 1$, $0 < \omega_{\max} < 1$, $0 < \Delta_{\max}$, $k_1 \in N$, $\ell_1 \in N$

Step 1: Choose an initial point $x \in R^n$. Compute the values $f_i := f_i(x)$ of the functions $f_i : R^n \to R$, $1 \le i \le m$, at the point $x \in R^n$. Determine the vector $b \in R^m$ using (1.9). Compute the value $f := f(x)$ of the objective function $f : R^n \to R$ by (1.10). Set $\Delta := 0$ and $\tau := (\tau_1)^{1/n}$. Set $k := 1$.

Step 2: Compute the gradients $g_i := g_i(x)$ of the functions $f_i : R^n \to R$, $1 \le i \le m$, at the point $x \in R^n$. Determine the matrix $A \in R^{m\times n}$ using (1.9). Compute the gradient $g := g(x)$ of the objective function $f : R^n \to R$ by (1.10). If either $f \le \varepsilon_1$ or $\|g\| \le \varepsilon_2$ then stop, otherwise set $\ell := 1$.

Step 3: If $\Delta = 0$ then set $\Delta := \min(\|g\|^3/\|Ag\|^2, 4f/\|g\|, \Delta_{\max})$. Set $\omega := \min(\sqrt{\|g\|}, \tau^k, \omega_{\max})$. Compute the vector $d \in R^n$ by the following subalgorithm:

Step 3.1: Set $d := 0$. Compute $\beta := \|b\|$ and $u := b/\beta$. Compute $\alpha := \|g\|/\beta$ and $v := -g/\|g\|$. Set $\overline{\rho} := \alpha$, $\overline{\eta} := \beta$ and $p := v$. Set $i := 1$.

Step 3.2: Compute $\beta := \|Av - \alpha u\|$. If $\beta = 0$ then go to Step 3.3, otherwise set $u := (Av - \alpha u)/\beta$. Compute $\alpha := \|A^{\mathrm{T}}u - \beta v\|$. If $\alpha = 0$ then go to Step 3.3, otherwise set $v := (A^{\mathrm{T}}u - \beta v)/\alpha$.

Step 3.3: Compute $\rho := \sqrt{\overline{\rho}^2 + \beta^2}$, $c = \overline{\rho}/\rho$, $s = \beta/\rho$ and $\eta = c\overline{\eta}$. If $\|d + (\eta/\rho)p\| > \Delta$ then determine $0 < \lambda < 1$ so that $\|d + \lambda(\eta/\rho)p\| = \Delta$, set $d := d + \lambda(\eta/\rho)p$ and go to Step 4. Otherwise set $d := d + (\eta/\rho)p$.

Step 3.4: If either $i = n + 3$ or $\alpha\beta|\eta|/\rho \le \omega\|g\|$ then go to Step 4, otherwise compute $\rho := c\alpha$, $\sigma := s\alpha$, $\eta := -s\overline{\eta}$ and set $p := v - (\sigma/\rho)p$. Set $i := i + 1$ and go to Step 3.2.

Step 4: Set $x^+ := x + d$. Compute the values $f_i^+ := f_i(x^+)$ of the functions $f_i : R^n \to R$, $1 \le i \le m$, at the point $x^+ \in R^n$. Determine the vector $b^+ \in R^m$ using (1.9). Compute the value $f^+ := f(x^+)$ of the objective function $f : R^n \to R$ by (1.10). Compute the value $Q(d)$ by (1.13) and set $\rho := (f^+ - f)/Q(d)$. When $\rho < \rho_1$ then compute $\alpha := (f^+ - f)/d^{\mathrm{T}}g$, $\beta := 1/(2(1-\alpha))$ and set $\Delta := \beta_1\|d\|$ if $\beta < \beta_1$, $\Delta := \beta\|d\|$ if $\beta_1 \le \beta \le \beta_2$, $\Delta := \beta_2\|d\|$ if $\beta_2 < \beta$. When $\rho_1 \le \rho \le \rho_2$ then set $\Delta := \min(\Delta, \gamma_2\|d\|)$. When $\rho_2 < \rho$ then compute $\Delta := \max(\Delta, \gamma_1\|d\|)$ and set $\Delta := \min(\Delta, \gamma_2\|d\|, \Delta_{\max})$.

Step 5: If $\rho \le 0$ and $\ell \ge \ell_1$ then stop (too many reductions). If $\rho \le 0$ and $\ell < \ell_1$ then set $\ell := \ell + 1$ and go to Step 3. If $\rho > 0$ and $k \ge k_1$ then stop (too many iterations). If $\rho > 0$ and $k < k_1$ then set $x := x^+$, $b := b^+$, $f ;= f^+$, set $k := k + 1$ and go to Step 2.

The maximum number of iterations $k_1 \in N$ serves as an alternative termination criterion in the case when the convergence is too slow. The maximum number of reductions $\ell_1 \in N$ serves as a safeguard against possible infinite cycle which can arise for large residual problems when present round-off errors do not allow us to obtain a solution with the required gradient norm ($\|g\| \le \varepsilon_2$).

We suppose, in the subsequent considerations, that all computations were performed accurately and that $k_1 = \ell_1 = \infty$. Furthermore we denote

$$g(x) = \sum_{i=1}^m f_i(x) g_i(x) \tag{3.1}$$

and

$$G(x) = \sum_{i=1}^m g_i(x) g_i^{\mathrm{T}}(x) + \sum_{i=1}^m f_i(x) G_i(x) \tag{3.2}$$

the gradient and the Hessian matrix of the objective function (1.8) respectively.

Theorem 3.1. Let the functions $f_i : R^n \to R$, $1 \le i \le m$, have continuous second-order derivatives and let there exist constants $C_1 > 0$, $C_2 > 0$, $C_3 > 0$ so that $|f_i(x)| \le C_1$, $\|g_i(x)\| \le C_2$, $\|G_i(x)\| \le C_3$, $1 \le i \le n$, for all $x \in R^n$. Let $x_k \in R^n$, $k \in N$, be the sequence generated by the Algorithm 3.1. Then

$$\lim_{k\to\infty}\inf \|g(x_k)\| = 0 \tag{3.3}$$

Proof. From (1.9) we have

$$\|A^{\mathrm{T}}(x)A(x)\| \le \sum_{i=1}^m \|g_i(x) g_i^{\mathrm{T}}(x)\| = \sum_{i=1}^m \|g_i(x)\|^2 \le m C_2^2$$

and (3.2) implies

$$\begin{aligned}\|G(x)\| &\le \|A^{\mathrm{T}}(x)A(x)\| + \left\|\sum_{i=1}^{m} f_i(x)G_i(x)\right\| \le \\ &\le mC_2^2 + \sum_{i=1}^{m} |f_i(x)|\ \|G_i(x)\| \le m(C_2^2 + C_1C_3)\end{aligned}$$

Therefore both matrices $B(x) = A^{\mathrm{T}}(x)A(x)$ and $G(x)$ are bounded from above so that (3.3) holds (see [6], [7], [8]). □

Theorem 3.2. Let the assumptions of Theorem 3.1 be satisfied with

$$\lim_{k\to\infty} x_k = x^* \tag{3.4}$$

Let the matrix $A(x^*)$ has full column rank and

$$\sum_{i=1}^{m} f_i(x^*)G_i(x^*) = 0 \tag{3.5}$$

Then the rate of convergence of the sequence $x_k \in R^n$, $k \in N$ is superlinear.

P r o o f . We have to prove that

$$\lim_{k\to\infty} \omega_k = 0 \tag{3.6}$$

and

$$\lim_{k\to\infty} \frac{\|(G(x_k) - A^{\mathrm{T}}(x_k)A(x_k))d_k\|}{\|d_k\|} = 0 \tag{3.7}$$

since these conditions are sufficient for the superlinear rate of convergence if the matrix $G(x^*)$ is positive definite (see [6], [7], [8]). But $\omega_k \to 0$ since $0 < \omega_k \le \|g(x_k)\|$ in Step 3 of Algorithm 3.1 and $g(x_k) \to 0$ by (3.3) and (3.4). From (3.2) we get

$$\frac{\|(G(x_k) - A^{\mathrm{T}}(x_k)A(x_k))d_k\|}{\|d_k\|} \le \left\|\sum_{i=1}^{m} f_i(x_k)G_i(x_k)\right\|$$

and continuity assumptions imply

$$\lim_{k\to\infty} \sum_{i=1}^{m} f_i(x_k)G(x_k) = \sum_{i=1}^{m} f_i(x^*)G_i(x^*)$$

which together with (3.5) gives (3.7). The matrix $G(x^*)$ is positive definite since

$$G(x^*) = \sum_{i=1}^{m} g_i(x^*)g_i^{\mathrm{T}}(x^*) + \sum_{i=1}^{m} f_i(x^*)G_i(x^*) = A^{\mathrm{T}}(x^*)A(x^*)$$

by (3.2) and (3.5) and since the matrix $A(x^*)$ has full column rank. □

4. COMPUTATIONAL EXPERIMENTS

In this section we present results of a comparative study of three trust region methods for nonlinear least squares: the exact trust region method with the double dog-leg step (DDLS) subalgorithm proposed in [3], the inexact trust region method with the CGLS subalgorithm described in Section 1 and the inexact trust region method with the LSQR subalgorithm studied in Section 2. All these trust region methods were realized by algorithms which differ from Algorithm 3.1 only in Steps 3.1–3.4 (Algorithm 3.1 uses the LSQR subalgorithm).

Algorithm 3.1 contains several parameters. We have used the values $\beta_1 = 0.05$, $\beta_2 = 0.75$, $\gamma_1 = 2$, $\gamma_2 = 10^6$, $\rho_1 = 0.1$, $\rho_2 = 0.9$, $\varepsilon_1 = 10^{-16}$, $\varepsilon_2 = 10^{-8}$, $\tau_1 = 10^{-3}$, $\omega_{\max} = 0.4$, $\Delta_{\max} = 10^3$, $k_1 = 500$, $l_1 = 20$ in all numerical experiments.

All test results were obtained by means of the 9 problems given in the Appendix. All these problems were considered with 100 variables. Therefore a sparse matrix technology was used (for instance the DDLS subalgorithm contained a sparse Choleski factorization procedure). Summary results for all problems are given in Table 1. Rows of this table correspond to individual problems and columns correspond to selected algorithms (DDLS, CGLS, LSQR). The results are presented in the form IT-IF-IG (P) where IT is number of iterations IF is number of different points at which the values $f_i(x)$, $1 \le i \le m$, were computed, IG is number of different points at which the gradients $g_i(x)$, $1 \le i \le m$, were computed and (P) is the logarithm of the obtained gradient norm.

Numerical results contained in Table 1 show that the LSQR algorithm is most efficient, measured by both numbers of iterations and numbers of functions evaluations, in comparison with other tested algorithms.

Table 1.

n=100	DDLS	CGLS	LSQR
1	218-221-219 (-11)	135-150-136 (-8)	117-121-118 (-11)
2	166-180-167 (-8)	152-188-153 (-11)	111-131-112 (-7)
3	13-14-14 (-8)	17-18-18 (-8)	14-15-15 (-8)
4	29-60-30 (-7)	199-230-200 (-7)	81-109-82 (-6)
5	5-6-6 (-14)	9-10-10 (-10)	6-7-7 (-8)
6	5-6-6 (-10)	10-11-11 (-10)	8-9-9 (-13)
7	25-61-26 (-4)	38-69-39 (-4)	38-72-39 (-4)
8	15-17-16 (-8)	15-16-16 (-8)	15-16-16 (-8)
9	69-108-70 (-6)	53-80-54 (-6)	50-71-51 (-6)
10	405-458-406 (-6)	26-61-27 (-7)	28-66-29 (-7)
$\sum$	950-1131-960	654-833-664	468-617-478

APPENDIX

Our test problems consist in searching local minimum of the objective function

$$F(x) = \frac{1}{2}\sum_{k=1}^{m} f_k(x)$$

from the starting point $\bar{x}$. We suppose n is even. We use functions div (integer division) and mod (remainder after integer division).

Problem 1. Chained Rosenbrock function.

$$\begin{aligned} m &= 2(n-1), & i &= \operatorname{div}(k+1,2) \\ f_k(x) &= 10(x_i^2 - x_{i+1}), & & k - \text{odd} \\ f_k(x) &= x_i - 1, & & k - \text{even} \\ \bar{x}_\ell &= -1.2, & & \ell - \text{odd} \\ \bar{x}_\ell &= 1.0, & & \ell - \text{even} \end{aligned}$$

Problem 2. Chained Wood function.

$$m = 3(n-2), \quad i = 2\operatorname{div}(k+5,6) - 1$$

$$\begin{aligned} f_k(x) &= 10(x_i^2 - x_{i+1}), & \operatorname{mod}(k,6) &= 1 \\ f_k(x) &= x_i - 1, & \operatorname{mod}(k,6) &= 2 \\ f_k(x) &= \sqrt{90}\,(x_{i+2}^2 - x_{i+3}), & \operatorname{mod}(k,6) &= 3 \\ f_k(x) &= x_{i+2} - 1, & \operatorname{mod}(k,6) &= 4 \\ f_k(x) &= \sqrt{10}\,(x_{i+1} + x_{i+3} - 2), & \operatorname{mod}(k,6) &= 5 \\ f_k(x) &= (x_{i+1} - x_{i+3})/\sqrt{10}, & \operatorname{mod}(k,6) &= 0 \end{aligned}$$

$$\begin{aligned} \bar{x}_\ell &= -3, & \ell &- \text{odd}, & \ell &\le 4 \\ \bar{x}_\ell &= -2, & \ell &- \text{odd}, & \ell &> 4 \\ \bar{x}_\ell &= -1, & \ell &- \text{even}, & \ell &\ge 4 \\ \bar{x}_\ell &= 0, & \ell &- \text{even}, & \ell &< 4 \end{aligned}$$

Problem 3. Chained Powell singular function.

$$m = 2(n-2), \quad i = 2\operatorname{div}(k+3,4) - 1$$

$$\begin{aligned} f_k(x) &= x_i + 10x_{i+1}, & \operatorname{mod}(k,4) &= 1 \\ f_k(x) &= \sqrt{5}\,(x_{i+2} - x_{i+3}), & \operatorname{mod}(k,4) &= 2 \\ f_k(x) &= (x_{i+1} - 2x_{i+2})^2, & \operatorname{mod}(k,4) &= 3 \\ f_k(x) &= \sqrt{10}\,(x_i - x_{i+3})^2, & \operatorname{mod}(k,4) &= 0 \end{aligned}$$

$$\begin{aligned}
\bar{x}_\ell &= 3, \quad \mathrm{mod}(\ell,4)=1\\
\bar{x}_\ell &= -1, \quad \mathrm{mod}(\ell,4)=2\\
\bar{x}_\ell &= 0, \quad \mathrm{mod}(\ell,4)=3\\
\bar{x}_\ell &= 1, \quad \mathrm{mod}(\ell,4)=0
\end{aligned}$$

Problem 4. Chained Cragg and Levy function.

$$m = 5(n-2)/2, \quad i = 2\,\mathrm{div}(k+4,5)-1$$

$$\begin{aligned}
f_k(x) &= (\exp(x_i)-x_{i+1})^2, && \mathrm{mod}(k,5)=1\\
f_k(x) &= 10(x_{i+1}-x_{i+2})^3, && \mathrm{mod}(k,5)=2\\
f_k(x) &= \tfrac{\sin^2(x_{i+2}-x_{i+3})}{\cos^2(x_{i+2}-x_{i+3})}, && \mathrm{mod}(k,5)=3\\
f_k(x) &= x_i^4, && \mathrm{mod}(k,5)=4\\
f_k(x) &= x_{i+3}-1, && \mathrm{mod}(k,5)=0
\end{aligned}$$

$$\begin{aligned}
\bar{x}_\ell &= 1, \quad \ell=1\\
\bar{x}_\ell &= 2, \quad \ell>1
\end{aligned}$$

Problem 5. Generalized Broyden tridiagonal function.

$$m=n, \quad x_0=0, \quad x_{n+1}=0$$
$$f_k(x) = (3-2x_k)\,x_k + 1 - x_{k-1} - x_{k+1}$$
$$\bar{x}_\ell = -1, \quad \ell \ge 1$$

Problem 6. Generalized Broyden banded function.

$$m=n, \quad k_1=\max(1,k-5), \quad k_2=\min(n,k+1)$$
$$f_k(x) = (2+5x_k^2)x_k + 1 + \sum_{j=k_1}^{k_2} x_j(1+x_j)$$
$$\bar{x}_\ell = -1, \quad \ell \ge 1$$

Problem 7. Extended Freudenstein and Roth function.

$$\begin{aligned}
m &= 2(n-1), \quad i=\mathrm{div}(k+1,2)\\
f_k(x) &= x_i + x_{i+1}((5-x_{i+1})x_{i+1}-2)-13\ , \quad k-\text{odd}\\
f_k(x) &= x_i + x_{i+1}((1+x_{i+1})x_{i+1}-14)-29, \quad k-\text{even}\\
\bar{x}_\ell &= 0.5, \quad \ell<n\\
\bar{x}_\ell &= -2, \quad \ell=n
\end{aligned}$$

Problem 8. Wright and Holt zero residual problem (n is multiple of 4).

$$m = 5n, \quad i = \mathrm{mod}(k, n/2) + 1, \quad j = i + n/2$$

$$a = 1, \quad k \le m/2$$

$$a = 2, \quad k > m/2$$

$$b = 5 - \mathrm{div}(k, m/4)$$

$$c = \mathrm{mod}(k, 5) + 1$$

$$f_k(x) = (x_i^a - x_j^b)^c$$

$$\bar{x}_\ell = \sin^2(\ell)$$

Problem 9. Toint quadratic merging problem.

$$m = 3(n-2), \quad i = 2\,\mathrm{div}(k+5, 6) - 1$$

$$\begin{aligned}
f_k(x) &= x_i + 3x_{i+1}(x_{i+2} - 1) + x_{i+3}^2 - 1, && \mathrm{mod}(k,6) = 1\\
f_k(x) &= (x_i + x_{i+1})^2 + (x_{i+2} - 1)^2 - x_{i+3} - 3, && \mathrm{mod}(k,6) = 2\\
f_k(x) &= x_i x_{i+1} - x_{i+2} x_{i+3}, && \mathrm{mod}(k,6) = 3\\
f_k(x) &= 2x_i x_{i+2} + x_{i+1} x_{i+3} - 3, && \mathrm{mod}(k,6) = 4\\
f_k(x) &= (x_i + x_{i+1} + x_{i+2} + x_{i+3})^2 + (x_i - 1)^2, && \mathrm{mod}(k,6) = 5\\
f_k(x) &= x_i x_{i+1} x_{i+2} x_{i+3} + (x_{i+3} - 1)^2 - 1, && \mathrm{mod}(k,6) = 0
\end{aligned}$$

$$\bar{x}_\ell = 5, \quad \ell \ge 1$$

Problem 10.

$$m = 2n - 1, \quad i = \mathrm{div}(k+1, 2)$$

$$\begin{aligned}
f_k(x) &= 4 - \exp(x_i) - \exp(x_{i+1}), && \mathrm{mod}(k,2) = 1, \quad i = 1\\
f_k(x) &= 8 - \exp(3x_{i-1}) - \exp(3x_i)\\
&\qquad + 4 - \exp(x_i) - \exp(x_{i+1}), && \mathrm{mod}(k,2) = 1, \quad 1 < i < n\\
f_k(x) &= 8 - \exp(3x_{i-1}) - \exp(3x_i), && \mathrm{mod}(k,2) = 1, \quad i = n\\
f_k(x) &= 6 - \exp(2x_i) - \exp(2x_{i+1}), && \mathrm{mod}(k,2) = 0
\end{aligned}$$

$$\bar{x}_\ell = 0.2, \quad \ell \ge 1$$

(Received January 12, 1993.)

REFERENCES

[1] G. Golub and W. Kahan: Calculating the singular values and pseudo-inverse of a matrix. SIAM J. Numer. Anal. *2* (1965), 205–224.

[2] J. J. Moré, B. S. Garbow and K. E. Hillström: Testing unconstrained optimization software. ACM Trans. Math. Software *7* (1981), 17–41.

[3] J. E. Dennis and H. H.W. Mei: An Unconstrained Optimization Algorithm which Uses Function and Gradient Vlues. Report No. TR-75-246. Dept. of Computer Sci., Cornell University 1975.

[4] C. C. Paige: Bidiagonalization of matrices and solution of linear equations. SIAM J. Numer. Anal. *11* (1974), 197–209.

[5] C. C. Paige and M. A. Saunders: LSQR: An algorithm for sparse linear equations and sparse least squares. ACM Trans. Math. Software *8* (1982), 43–71.

[6] M. J. D. Powell: Convergence properties of a class of minimization algoritms. In: Nonlinear Programming 2 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), Academic Press, London 1975.

[7] G. A. Shultz, R. B. Schnabel and R. H. Byrd: A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties. SIAM J. Numer. Anal. *22* (1985), 47–67.

[8] T. Steihaug: The conjugate gradient method and trust regions in large-scale optimization. SIAM J. Numer. Anal. *20* (1983), 626–637.

Ing. Ladislav Lukšan, DrSc., Ústav informatiky a výpočetní techniky AV ČR (Institute of Computer Science – Academy of Sciences of the Czech Republic), Pod vodárenskou věží 2, 182 07 Praha 8. Czech Republic.