A NUMERICALLY STABLE REDUCED–GRADIENT TYPE ALGORITHM FOR SOLVING LARGE–SCALE LINEARLY CONSTRAINED MINIMIZATION PROBLEMS

Hermínio Simões Gomes
and
José Mario Martínez

RELATÓRIO TÉCNICO Nº 09/88

ABSTRACT. In this paper we present a Reduced–Gradient–type algorithm for solving large–scale linearly constrained minimization problems. During each iteration of the algorithm linear systems are solved using a Preconditioned Conjugate Gradient scheme. The Preconditioning scheme uses orthogonal transformations, thus providing numerical stability. The total storage used by the algorithm may be predicted before beginning the calculations. We present some numerical experiments which seem to confirm the reliability of the algorithm.

Universidade Estadual de Campinas
Instituto de Matemática, Estatística e Ciência da Computação
IMECC – UNICAMP
Caixa Postal 6065
13.081 - Campinas, SP
Brasil

Maio – 1988

1. INTRODUCTION

Many practical problems require the solution of large-scale linearly constrained minimization problems:

$$\begin{aligned} &\text{Minimize } f(x) \\ &\text{s.t. } x \in P \end{aligned} \tag{1}$$

where P is a polytope in $\mathbb{R}^n$.

The best-known algorithm for solving problems of this type is the system MINOS [23,24], developed by the Systems Optimization Laboratory of Stanford University. MINOS represents the polytope P in the standard form:

$$\begin{aligned} Ax &= b \\ \ell \leq x &\leq u \end{aligned} \tag{2}$$

and classifies the variables of the problem during each iteration in basic, superbasic and nonbasic, in such a way that the matrix A is partitioned in the form $A = (B\ S\ N)$ and linear systems involving the square matrices B and B^T need to be solved. These systems are solved using the $L-U$ factorization of B, and the factorization is updated from one iteration to another, using well-known schemes [3,4,5,27,28].

Other authors improved the techniques used in MINOS for special structures of A. For example, the $L-U$ factorization of staircase matrices is considered in [12] and its updating is studied in [13]. Staircase structures are important because of their application in control problems (see [13]).

Our work was motivated by the following observations:

a) We found many problems where the representation of P in the form (2), using slack variables, is inconvenient because it leads to much larger systems of linear equations than those which

must be solved considering only the original variables (see [1]).

b) The storage locations used by codes based on the updating of L - U factorizations are not predictable a priori, and, in fact, the storage requirements may be prohibitive for moderate computer environments.

c) In spite of the numerical stability of the calculation and updating of L - U factorization based on classical schemes, this stability is severely challenged in critically ill-conditioned problems.

In order to deal with problems where we had difficulties using [23,24], we decided to develop a system where:

a) The constraints which define P are considered in their original form, and many of them may be dropped from the active set simultaneously (see [16]).

b) The linear systems are solved using a preconditioned conjugate gradient scheme, so that the storage used by the method may be predicted. The same preconditioning matrix is used during some iterations.

c) The preconditioning scheme uses plane rotations [14,15] to generate an upper-triangular banded matrix, in the sense of [2,22]. This scheme, combined with the conjugate gradient approach, proved to be numerically stable.

The organization of the paper is as follows:

In Section 2, we explain the active-set strategy which underlies the development of the method. The active-set strategy is used in many algorithms for linearly constrained optmization. We decided to write a nearly self-contained paper, so we included a convergence proof for this strategy. This proof turns to be a generalization of Theorem 1 of [25].

In Section 3, we show how to use the change of variables of Bertsekas [7] to identify Kuhn-Tucker points, and to transform locally the original problem into a problem with only bounded variables,

in order to generate efficient descent directions.

In Section 4, we describe the basic algorithm.

In Section 5, we explain our strategy for solving the linear systems which appear at each iteration.

In Section 6, we make some remarks concerning the computational implementation of the method.

In Section 7, we describe our numerical experiments.

Finally, in Section 8, we draw some conclusions, and suggest some lines for future work.

2. AN ACTIVE SET STRATEGY

Let us consider the problem (1), where P is defined by the set of linear inequalities:

$$a_i^T x \geq b_i \quad , \quad i=1,\ldots,m \tag{3}$$

For each subset I of $\{1,\ldots,m\}$, we define:

$$F_I = \{x \in P \,/\, a_i^T x = b_i \text{ for } i \in I \,,\, a_j^T x > b_j \text{ for } j \notin I \,,\, i,j = 1,\ldots,m\} \tag{4}$$

Let us call $\dim(F_I)$ the dimension of the smaller linear manifold which contains F_I.

Obviously, $F_I \cap F_J = \phi$ if $I \neq J$, and $P = \bigcup_{I \subset \{1,\ldots,m\}} F_I$.

Suppose that (x^k) is a sequence generated by an algorithm intended to solve (1), $x^o \in P$, and assume that (x^k) satisfies the following axioms:

(i) If x^k is not a Kuhn-Tucker point ([16,21]), then x^{k+1} is defined and $f(x^{k+1}) < f(x^k)$.

(ii) If x^{k+1} is defined, and $x^k \in F_I$, then one of the following possibilities hold:

(a) $x^{k+1} \in F_I$

(b) $x^{k+1} \in F_J$ and $\dim(F_J) < \dim(F_I)$

(c) x^k is the minimum of f on F_I.

(iii) For each k, I, $x^k \in F_I$, there exists $\ell > k$ such that $x^\ell \notin F_I$, or x^ℓ is a Kuhn-Tucker point.

Using the axioms above, we may prove the following theorem, which turns out to be a generalization of the Theorem 1 of [25].

THEOREM 2.1. Any sequence generated by an algorithm satisfying the rules (i)-(iii) stops after a finite number of steps. (That is, there exists a $\bar{k}$ such that $x^{\bar{k}}$ is a Kuhn-Tucker point).

PROOF. Let us prove that, for each $I \subset \{1,\ldots,m\}$ the set of iterates x^k which lie in F_I is finite. We prove the assertion by induction on ν, the dimension of F_I. For $\nu = 0$, F_I contains exactly one point, so the proposition is true.

Let us suppose that, for each F_I with $\dim F_I \leq \nu - 1$, the set of x^k which belong to F_I is finite. Suppose now that $\dim F_I = \nu$ and $x^k \in F_I$ for all $k \in \mathbb{Z}_1$, where $\mathbb{Z}_1$ is an infinite set of integers. Let k_1 be the first index of $\mathbb{Z}_1$. By (iii) there exists $q_1 > 0$ such that $x^{k_1+q_1} \notin F_I$. If $x^{k_1+q_1-1}$ is the minimum of f in F_I, then $f(x^\ell) < f(x^{k_1+q_1-1})$ for all $\ell > k_1$ and so, $x^\ell \notin F_I$ for all $\ell > k_1$. Now, if $x^{k_1+q_1-1}$ is not the minimum of f in F_I, it turns out, by (ii), that $x^{k_1+q_1} \in F_{J_1}$, where $\dim F_{J_1} \leq \nu - 1$. Now let $k_2 > k_1+q_1-1$ such

that $k_2 \in \mathbb{Z}_1$. Using the same reasoning as above, we find $x^{k_2+q_2}$ $\in F_{J_2}$, where $\dim F_{J_2} \leq \nu - 1$. Repeating this argument, we construct an infinite set $x^{k_i+q_i}$ contained in $\cup \{F_I / \dim F_I \leq \nu-1\}$. This contradicts the inductive hypotheses. ∎

REMARK. The problem of minimizing on the set F_I is, essentially, an unconstrained minimization problem. The axiom (iii) states that the subalgorithm used to handle this problem, either finds the solution in a finite number of steps or finds a point in an F_J with $\dim F_J < \dim F_I$. Usually, F_J is contained in the boundary of F_I. Of course, the existence of such an algorithm is restricted to some classes of functions, such as linear, quadratic [25], homogeneous [19,20], polynomials, etc (See [29]). However, the algorithmic consequences of the principles above is that a powerful minimization algorithm and a strict convergence criterion should be used for the unconstrained minimization on each face.

3. THE REDUCED GRADIENT STATEGY

Let us consider again the problem (1), with the constraints given by (3), and, additionally:

$$c_i^T x = d_i \quad , \quad i=1(1)q \tag{5}$$

Let x^k be the k-th approximation to the solution. Suppose, without loss of generality, that:

$$a_i^T x = b_i \quad , \quad i=1(1)p \tag{6}$$

Suppose, further, that $\{a_1,\ldots,a_p, c_1,\ldots,c_q\}$ is a linearly independent set of vectors, so x^k is a regular feasible point on P.

The following change of variables, due to Bertsekas [7], transforms locally the original problem in a minimization problem with bounded variables.

Define the new variables:

$$y_i = a_i^T x \quad , \quad i=1(1)p$$
$$y_{p+j} = c_j^T x \quad , \quad j=1(1)q \tag{7}$$

Define

$$A = \begin{bmatrix} a_1^T \\ \vdots \\ a_p^T \\ c_1^T \\ \vdots \\ c_q^T \end{bmatrix} .$$

Suppose, without loss of generality, that the first $p+q$ columns of A are linearly independent. So, $A = (B,N)$, where B is a square nonsingular matrix. We complete the definition of the variables y_i setting:

$$y_{p+q+i} = e_{p+q+i} \quad , \quad i = 1(1)n-p-q \tag{8}$$

So, by (7) and (8),

$$y = Mx \quad , \quad M = \begin{bmatrix} B & N \\ O & I \end{bmatrix} \tag{9}$$

and M is a nonsingular $n \times n$ matrix.

Therefore, the original problem may be written, in a neighborhood of x^k, as

$$\begin{aligned} &\text{Minimize} \quad \Phi(y) \\ \text{s/t} \quad & y_i \geq b_i \quad , \quad i=1(1)p \\ & y_i = d_{i-p} \quad , \quad i=p+1(1)p+q \\ & y_i \text{ free for } \quad i=p+q+1(1)n \end{aligned} \tag{10}$$

where $\Phi(y) = f(M^{-1}y)$.

Setting $y^k = M^{-1}x^k$, it is clear that the K-T conditions for (1), (3), (5) are:

$$\frac{\partial \Phi}{\partial y_i}(y^k) \geq 0 \quad \text{for} \quad i=1(1)p \tag{11}$$

and

$$\frac{\partial \Phi}{\partial y_i}(y^k) = 0 \quad \text{for} \quad i=p+q+1(1)n \tag{12}$$

So, if (11) and (12) hold, x^k is a Kuhn-Tucker point of (1).

If (12) holds, but $\frac{\partial \Phi}{\partial y_i}(y^k) < 0$ for some $i \in \{1,\ldots,p\}$, then y^k is a stationary point for the problem:

$$\begin{aligned} &\text{Minimize} \quad \Phi(y) \\ \text{s/t} \quad & y_i = b_i \quad , \quad i=1(1)p \\ & y_i = d_{i-p} \quad , \quad i=p+1(1)p+q \\ & y_i \text{ free for } \quad i=p+q+1(1)n \end{aligned} \tag{13}$$

and $z = (z_1,\ldots,z_n)^T$ is a descent direction for (10), provided

$$z_i = -\frac{\partial \Phi}{\partial y_i}(y^k) \text{ if } \frac{\partial \Phi}{\partial y_i}(y^k) < 0 \ , \ i = 1(1)p$$

$$= 0 \quad \text{otherwise} \tag{14}$$

If (12) does not hold, then any direction z satisfying:

$$z_i = 0 \ , \ i = 1(1)p+q$$

and

$$\sum_{i=p+q+1}^{n} z_i \frac{\partial \Phi}{\partial y_i}(y^k) < 0 \tag{15}$$

is a descent direction for (10), and any point $\bar{y} = y^k + \lambda z$ satisfies the constraints of (13).

The discussion above outlines the procedures which we shall use to detect a K-T point and to obtain a descent direction from a non-stationary point. Of course, once a descent direction has been obtaining using (14) or (15), it must be transformed to the original space by means of

$$w_k = M^{-1} z \tag{16}$$

and, consequently,

$$x^{k+1} = x^k + \lambda_k w_k \tag{17}$$

is a feasible point for λ_k small enough.

It is easy to see that the calculations used to find $\nabla \Phi$, and to calculate z, w_k, need only the resolution of linear systems where the matrices involved are B and B^T.

Of course, for many problems, (12) is not going to hold exactly at any point, so the annihilation of $\frac{\partial \Phi}{\partial y_i}(y^k)$ is going to be declared up to a tolerance ε.

4. BASIC STEPS OF THE ALGORITHM

Suppose that x^0 is an arbitrary regular (see [21]) initial approximation in P. Once a regular x^k is calculated, the steps used to obtain x^{k+1} are the following:

STEP 1. Let I_k be the set of indexes i such that $a_i^T x^k = b_i$, $i \in I_k$, $a_i^T x^k > b_i$ for $i \notin I_k$. Using the change of variables defined in the previous section, test the K-T conditions in x^k. If they are satisfied, stop.

STEP 2. If the equations (12) are satisfied, calculate w_k using (14) and (16). Go to Step 5.

STEP 3. If $I_k \neq I_{k-1}$ or $k = 0$, set $z_i = -\frac{\partial \phi}{\partial y_i}(y^k)$ in (15). Compute w_k using (16). Go to Step 5.

STEP 4. Compute z satisfying (15) using some conjugate gradient type formula [11,16,17,18,21] in the space y and use (16) to compute w_k.

STEP 5. Define:

$$\lambda_k^{MAX} = \max \{ \lambda \ / \ [x^k, x^k + \lambda w_k] \subset P \}$$

Find $\lambda_k \in (0, \lambda_k^{MAX}]$ such that

$$f(x^k + \lambda_k w_k) < < f(x^k) \qquad (18)$$

STEP 6.

$$x^{k+1} \longleftarrow x^k + \lambda_k w_k \ .$$

REMARKS. In our implementation of the algorithm we used the Fletcher-Reeves formula in Step 4 because it is the CG - type formula

with less storage requirements [6,11,16,21].

The hypotheses on the regularity of x^k may be relaxed allowing a steplength $\lambda_k = 0$ in Step 5. However, a very careful roundoff analysis is needed in this case to prevent loss of feasibility, and, theoretically, cycling may occur.

For quadratic functions we used

$$f(x^k + \lambda_k w_k) = \min\{f(x^k + \lambda w_k) \ , \ \lambda \in [0, \lambda_k^{MAX}]\}$$

in order to take advantage of the finite termination properties of the methods of Conjugate Gradients [11,16,21].

For more general functions we used the Armijo condition:

$$f(x^k + \lambda_k w_k) \leq f(x^k) + 10^{-4} \lambda_k \langle \nabla f(x^k), w_k \rangle \qquad (19)$$

with a first initial estimate for λ_k given by $\lambda_k^o = 2(\Phi(y^k) - \Phi(y^{k-1}))/\langle \nabla\Phi(y^k), w_k \rangle$ and a safeguarded cubic interpolation scheme for achieving (19) (see [11,16,21]).

5. THE RESOLUTION OF THE SYSTEMS WITH B^T AND B

As we pointed out in Section 4, we need to solve, during each iteration, linear systems of the form

$$B^T \pi = \beta \ , \quad B\delta = \gamma \ . \qquad (20)$$

Since our objective is a method with predictable storage requirements, the schemes based on the L-U factorization are not well suited for this purpose.

Moreover, we desire numerical stability, so orthogonal factorizations seem to be more adequate. However, although the fill - in produced by orthogonal factorizations may

be predicted [15, 22], it turns out to be very high in many cases. So, we decided to use an iterative scheme for solving (19), where the preconditioning strategy, based on orthogonal factorizations was the one successfully used in [22].

Let R be an upper triangular matrix such that BR^{-1}, is "nearly orthogonal". The system $B\delta = \gamma$ may be decomposed as

$$BR^{-1}\delta_1 = \gamma \tag{21}$$

$$R\delta = \delta_1 . \tag{22}$$

The system (21) may be solved using some specially oriented conjugate gradient algorithm, such as LSQR [26]. If BR^{-1} is close to an orthogonal matrix, it is expected that only a few iterations should to necessary to solve (21) up to a reasonable accuracy. We use the same preconditioning matrix and iterative procedure for solving $B^T\pi = \beta$, transforming this system into

$$R^{-T}B^T\pi = R^{-T}\beta . \tag{23}$$

Now, suppose that $Nonz(R) \subset \{1,\ldots,m\} \times \{1,\ldots,m\}$ represents the "Non-zero structure" which is desired for the upper-triangular preconditioning matrix $R = (r_{ij})$. That is, r_{ij} will be necessarly equal to zero if $(i,j) \notin Nonz(R)$.

We obtain R performing the orthogonal factorization of B using plane rotations as in [15,22] and neglecting r_{ij} when $(i,j) \notin Nonz(R)$.

If $Nonz(R)$ is carefully chosen, we expect that the preconditioning matrix R obtained with this algorithm should not be very different from the upper-triangular matrix $\bar{R}$ which arises when performing the complete orthogonal factorization of B. The advantage of this scheme over preconditioning schemes based on the Cholesky factorization of B^TB is that it provides a numerically

stable direct method to solve the systems when Nonz(R) is large enough to contain the true non-zero structure of $\bar{R}$. In our implementation, we used a band-matrix structure for R. That is, $(i,j) \notin$ Nonz(R) if $j > i+b$, where b is an integer given by the user.

Now, in fact, B changes from one iteration to another, and repeating the preconditioning procedure at each iteration should be very expensive. So, we decided to use the same preconditioning matrix B at consecutive iterations whenever this is possible. If the order of B_{k+1} is lower than the order of B_k, we "cut" rows and columns of R so that B_{k+1} and R turn to have the same order. Conversely, if the order of B_{k+1} is greater than the order of B_k, we add rows and columns corresponding to an identity matrix.

Usually B_{k+1} tends to be close to B_k and so the preconditioning matrix used for B_k tends to be efficient for B_{k+1}. Of course, we do not expect the matrix R to be a good preconditioner throughout the whole process. Therefore, the preconditioner R is re-computed when the number of iterations used to solve (20) exceed a given integer IGAT.

Ideally, we choose IGAT with the expectancy that the computer time used by IGAT C-G iterations for solving (21)-(22) is similar to the computer time used by the preconditioning process.

6. REMARKS CONCERNING THE COMPUTATIONAL IMPLEMENTATION OF THE METHOD

During some stages by the algorithm, given the full-rank $m \times n$ matrix A, whose rows are the gradients of the active constraints at the current point, we need to choose a full-rank $m \times m$ submatrix B. In some cases this is very simple, for instance, when n-m rows of A are vectors of the form (0... 0 1 0 ... 0), but in less structured cases we need a special algorithm to detect m

linearly independent columns of A.

We used the following heuristic strategy:

Let $i_j = j$, j=1(1)m. $I = \phi$. Perform the following steps:

STEP 1. For j = 1(1)m, perform Step 2-3.

STEP 2. Let $k = \underset{\ell}{\text{argmax}}\{|a_{i_j \ell}| , \ell = 1(1)n , \ell \notin I\}$

STEP 3. $I = I \cup \{k\}$

STEP 4. Construct B as the matrix whose columns are the columns of A with indexes in I, in increasing order.

If the procedure above fails to find a nonsingular matrix B, we repeat it with $i_j = m-j+1$, j = 1(1)m.

This heuristic strategy worked very well in practice.

We must stress that singularity, or severe ill-conditioning of B is going to be detected in the phase of the iterative solution of the linear systems. In this case, more than IGAT iterations of LSQR are used and, so, we turn to the preconditioning scheme. If the heuristic strategies for finding linearly independent columns fail, the code stops with an appropriate error message.

7. NUMERICAL EXPERIMENTS

We ran many experiments concerning linearly constrained optimization problems with staircase structure [12,13]. For small problems (up to 150 constraints and/or variables), our system, BUGRE, behaved well, but in most cases MINOS used less computer time for arriving to the solution. The situation changed dramatically when considering large problems (more than 1000 constraints and/or variables). In fact, none of the problems which we are

going to describe next could be solved using MINOS, some of them because of storage reasons (more than 250000 REAL*8 positions were needed by MINOS), and others because of stability reasons (ill-conditioning stopping in subroutine CHUZR).

The first set of tests concerns problems of the type:

$$\text{Minimize} \quad f(x)$$

$$\text{s.t.} \quad Ax \leq b$$

$$\ell \leq x \leq u$$

where $A \in \mathbb{R}^{m \times n}$ has a staircase structure with blocks of the same size. The elements of A are generated randonly between -20 and 20. The elements of b are generated randonly between -100 and 100, in order that x^0 be a feasible point.

In some cases, we generated the elements of A independently, and in other cases, all the blocks of A are equal. We consider three objective functions: "Linear" ($f = -\Sigma x_i$), "Quadratic" ($f = (1/2)\ \Sigma x_i^2$) and "Entropy" ($f = \Sigma x_i \log x_i$). In all cases we used the tolerance $\varepsilon = 10^{-4}$ and BUGRE stopped with a diagnostic of "Convergence".

The tests were run in double precision using a VAX785, and the codes were written in FORTRAN.

Problem	1	2	3	4	5	6
m	1200	1200	1050	1050	1000	1000
n	1202	1202	1052	1052	1602	1602
Type of Blocks	3×5, equal	3×5, different	3×5, equal	3×5, different	5×10, equal	5×10, different
Number of Blocks	400	400	350	350	200	200
Density of A	0,42%	0,42%	0,47%	0,47%	0,62%	0,62%
Bounds	$0 \leq x \leq 50$	$0 \leq x \leq 50$	$1 \leq x \leq 10$	$1 \leq x \leq 10$	$2 \leq x \leq 6$	$2 \leq x \leq 6$
Objective Function	LINEAR	LINEAR	QUADRATIC	QUADRATIC	ENTROPY	ENTROPY
Initial Point	0	0	5	5	6	6
Number of bands of R	8	8	10	10	15	15
IGAT	4	4	5	5	4	4
Iterations	450	282	78	290	425	963
Number of different B_k	451	283	65	266	426	964
Calls to the Preconditioning routine	59	94	6	45	23	92
Evaluations of f and ∇f	451	283	95	332	427	965
Mean Size of B	155 × 155	152 × 152	103 × 103	106 × 106	33 × 33	42 × 42
REAL*8 positions used	38802	38802	33893	33893	53785	53785
CPU Time	52.5'	42.9'	5.09'	23.2'	21.4'	53.6'

TABLE 1 - First Set of Experiments with BUGRE.

The second set of tests involves problems of the type

$$\text{Minimize} \quad f(x)$$

$$\text{s.t.} \quad Ax = b$$

$$\ell \leq x \leq u$$

where, as in the first set, A has staircase structure with blocks of the same size, and its elements are randonly generated in the interval $[-20,20]$. b is calculated so that x^0 is a feasible point. We also consider cases where the blocks are independent and other cases where the blocks are equal. The linear function considered in this set of tests is $f(x) = -x_1$. We also show the results with the Quadratic function $(1/2)\ \Sigma\ x_i^2$ and the entropy function $\Sigma\ x_i \log x_i$. As in the previous set of tests, $\varepsilon = 10^{-4}$ and BUGRE converged in all cases.

A final test concerns a linear problem with 100 constraints of the type $a_i^T x = b_i$ and 3900 constraints of the type $a_i^T x \leq b_i$. The results for this problem are shown in Table 3.

Problem	7	8	9	10	11	12
m	1000	1000	800	800	900	900
n	1801	1801	1204	1204	905	905
Type of Blocks	5 × 10, equal	5 × 10, different	4 × 10, equal	4 × 10, different	5 × 10, equal	5 × 10, different
Number of Blocks	200	200	200	200	180	180
Density of A	0,55%	0,55%	0,83%	0,83%	1,1%	1,1%
Bounds	$-100 \leq x \leq 100$	$-100 \leq x \leq 100$	$3 \leq x \leq 5$	$3 \leq x \leq 5$	$5 \leq x \leq 10$	$5 \leq x \leq 10$
Objective Function	LINEAR	LINEAR	QUADRATIC	QUADRATIC	ENTROPY	ENTROPY
Initial Point	0	0	4	4	8	8
Number of bands of R	15	15	10	10	15	15
JGAT	10	10	4	4	5	5
Iterations	55	7	74	378	3	67
Number of different B_k	56	8	28	64	3	13
Calls to the Preconditioning routine	2	1	15	52	1	5
Evaluations of f and ∇f	56	8	125	713	5	85
Mean Size of B	1000 × 1000	1000 × 1000	800 × 800	800 × 800	900 × 900	900 × 900
REAL*8 positions used	74348	74348	51613	51613	55089	55089
CPU Time	55.2'	6.75'	33.8'	195'	24.6'	545'

TABLE 2 - Second set of experiments with BUGRE.

Problem	13
m	4000
n	6002
Type of Blocks	2 × 5, different
Number of Blocks	2000
Density of A	0,083%
Bounds	$\underline{0} \leq x \leq \underline{10}$
Objective Function	LINEAR $(-x_1)$
Initial Point	$\underline{0}$
Number of bands of R	6
IGAT	5
Iterations	8
Number of different B_k	9
Calls to the Preconditioning routine	3
Evaluations of f and ∇f	9
Mean Size of B	100 × 100
REAL*8 positions used	156914
CPU Time	3.99'

TABLE 3 – Performance of BUGRE with problem 13.

The results presented in the tables above are typical. However they represent only 10% of our tests. In the whole, BUGRE failed in about 8% of our experiments, with a diagnostic of "loss of feasibility". On the other hand MINOS never found the solution of a problem where BUGRE failed.

8. FINAL REMARKS

The numerical experiments presented show that BUGRE is a reliable alternative to MINOS when either conditioning or storage is critical in the problem under consideration. So, additional research is merited in order to improve and to fully evaluate the potentialities of BUGRE. We are planning to continue our research along the following lines:

(a) Implementation of alternative,less expensive, preconditioning schemes.

(b) Incorporation of a clever Phase I algorithm (At the present stage, the user must code his/her own Phase I).

(c) Testing BUGRE with different structures of the matrix A.

(d) Symbolic apriori manipulation of the structure of A in order to minimize the fill-in in the preconditioning phase.

ACKNOWLEDGEMENTS

J. M. Martínez acknowledges the Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) and the Conselho de Desenvolvimento Científico e Tecnológico do Brasil (CNPq) for financial support.

H.S. Gomes acknowledges the Universidade Federal do Pará for supporting his graduate studies at UNICAMP.

Both authors are indebted to Renato B. Guerra for many stimulating discussions.

BIBLIOGRAFY

[1] L.S. AMARAL, T.L. LOPES, J.M. MARTÍNEZ and M. TAUBE: "A numerically Stable Multiblending System for Microcomputers" , contributed paper at SIAM Conference on Optimization, Houston, Texas, 1987.

[2] O. AXELSSON, G. LINDSKOG: On the Rate of Convergence of the Preconditioned Conjugate Gradient Method, Numer. Math. 48 (1986), 479-498.

[3] R.H. BARTELS: A Stabilization of the Simplex Method, Numerische Mathematik 16, (1971) , 414-434.

[4] R.H. BARTELS , G. H.: GOLUB: The Simplex Method using LU decomposition, Comm. ACM 12, (1969), 266-268.

[5] R.H. BARTELS, G.H. GOLUB, M.A. SAUNDERS: Numerical Techniques in Mathematical Programming, in *Nonlinear Programming* , J. B. Rosen, O. L. Mangasarian and K. Ritter editors, Academic Press, (1970), 123-176.

[6] M. BERTOCCHI, E. SPEDICATO: Computational Experience with Conjugate Gradient Algorithms, Calcolo, Vol. 15, 3 (1979), 255-269.

[7] D.P. BERTSEKAS: Projected Newton Methods for Optimization Problems with Simple Constraints, SIAM J. Control and Optimization, Vol. 20, 2 (1982), 221-246.

[8] Å. BJORCK: Methods for Sparse Linear Least Squares Problems, in *Sparse Matrix Computations*, J.R. Bunch and D.J. Rose editors, Academic Press Inc. (1976), 177-199.

[9] G.B. DANTZIG: Solving Staircase Linear Programs by a nested Block-Angular Method, TR 73-1, Dept. of Operations Research, Stanford University, 1973.

[10] A. DRUD: CONOPT: A GRG, code for large sparse dynamic nonlinear optimization problems, Math. Prog. 31, (1985) 153-191.

[11] R. FLETCHER, C.M. REEVES: Function Minimization by Conjugate Gradients, Comput. J. 2 (1964), 149-153.

[12] R. FOURER: Staircase Matrices and Systems, SIAM Review 26, (1984) 1-70.

[13] A. FRIEDLANDER: Otimização com Estrutura Escada nas Restrições, Doctoral Dissertation, FEEC, UNICAMP, 1986.

[14] W.M. GENTLEMAN: Least Squares Computations by Givens Transformations without Square Roots, J. Inst. Maths. Applic. 12, (1973), 329-336.

[15] A. GEORGE, M.T. HEATH: Solution of Sparse Linear Least Squares using Givens Rotations. Linear Algebra and its Applics 34 (1980) 69-83.

[16] P.E. GILL, W. MURRAY, M. WRIGHT: Practical Optimization, Academic Press, 1981.

[17] M.R. HESTENES: Conjugate Direction Methods in Optimization , Springer-Verlag, 1980.

[18] M.R. HESTENES, E. STIEFEL: Methods of Conjugate Gradients for Solving Linear Systems, J. Res. Nat. Bur. Standards 49 (1952), 409-436.

[19] D.H. JACOBSON , W. OKSMAN: An algorithm that minimizes homogeneous functions of N variables in N+2 iterations and rapidly minimizes general functions , J. Math. Anal. Appl. 38 (1972) 535-552.

[20] J.S. KOWALIK , K. G.: RAMAKRISHMAN: A numerically stable optimization method based on an homogeneous functions , Math. Programming 11 (1976) 50-66.

[21] D.G. LUENBERGER: Linear and Nonlinear Programming, (2nd edition), Addison Wesley, 1984.

[22] J.M. MARTÍNEZ: An Algorithm for solving Sparse Nonlinear Least Squares Problems, Computing 39 (1987), 307-325.

[23] B.A. MURTAGH, M.A. SAUNDERS: Large-Scale Linearly Constrained Optimization, Mathematical Programming 14 (1978), 41-72.

[24] B.A. MURTAGH, M.A. SAUNDERS: MINOS User's Guide, Technical Report, Depatment of Oper. Res., Stanford, California, 1977.

[25] D.P. O'LEARY, A Generalized Conjugate Gradient Algorithm for Solving a Class of Quadratic Programming Problems, Linear Algebra and its Applics. 34 (1980), 371-399.

[26] C.C. PAIGE and M.A. SAUNDERS: LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares , ACM TOMS 8, (1982) 43-72.

[27] J.K. REID: A sparsity-exploiting variant of the Bartels – Golub decomposition for linear programming basis, Report CSS 20, AERE-Harwell, 1975.

[28] M. SAUNDERS: A fast stable implementation of the simplex method using Bartels-Golub updating, in *Sparse Matrix Computations* (Bunch J. and Rose D., ed.) Academic Press, New York, 213-226.

[29] R.B. SCHNABEL, TA-TUNG CHOW: Tensor Methods for Unconstrained Optimization, contributed paper at SIAM Conference on Optimization, Houston, Texas, 1987.

Hermínio Simões Gomes

Department of Mathematics,
Universidade Federal do Pará, Campus do Guamá
66.052 Belém, Pará, Brazil

José Mário Martínez
Applied Mathematics Laboratory,
IMECC - UNICAMP,
Caixa Postal 6065
13.081 - Campinas, SP, Brazil.

RELATÓRIOS TÉCNICOS — 1988

01/88 – A Linear Continuous Transportation Problem – *Enrique D. Andjel, Tarcisio L. Lopes* and *José Mario Martínez.*

02/88 – A Splitting Theorem for Complete Manifolds With Non-Negative Curvature Operator – *Maria Helena Noronha.*

03/88 – Mathematical Physics of the Generalized Monopole without String – *W. A. Rodrigues Jr., M. A. Faria-Rosa, A. Maia Jr.* and *E. Recami.*

04/88 – A Family of Quasi-Newton Methods with Direct Secant Updates of Matrix Factorizations – *José Mario Martínez.*

05/88 – Rotation Numbers of Differential Equations. A Framework in the Linear Case – *Luis San Martin.*

06/88 – A Geometrical Theory of Non Topological Magnetic Monopoles – *Marcio A. Faria-Rosa* and *Waldyr A. Rodrigues Jr.*

07/88 – Cosmic Walls and Axially Symmetric Sigma Models – *Patricio S. Letelier* and *Enric Verdaguer.*

08/88 – Verificação do Nível de Enlace do Protocolo X–25 – *Célio C. Guimarães* e *Edmundo R. M. Madeira.*