ABS METHODS AND ABSPACK, A REVIEW

Emilio Spedicato
Department of Mathematics, University of Bergamo

Abstract

We give a review of results obtained in the field of ABS methods, from both the point of view of theory and the numerical performance of ABSPACK.

1 Introduction

ABS algorithms were introduced by Abaffy, Broyden and Spedicato (1984), to solve linear equations first in the form of the *basic ABS class*, later generalized as the *scaled ABS class*. They were then applied to linear least squares, nonlinear equations and optimization problems, see e.g. the monographs by Abaffy and Spedicato (1989) and Zhang et al. (1999), or the bibliography by Spedicato et al. (2000) listing 350 ABS papers. In this paper we review some of the main results obtained in the field of ABS methods and we provide some results on the performance ABSPACK, a FORTRAN package based on ABS methods presently under development.

The main results obtained in almost twenty years of research in the ABS field can be summarized as follows:

- ABS methods provide a unification of the field of finitely terminating methods for linear systems and for feasible direction methods for linearly constrained optimization; due to the several alternative formulations of their algebra they lead to different computational implementations of the algorithms, each one with a special feature that may be of advantage

- ABS methods provide some new methods that are better than classical ones under several respects. For instance the implicit LX method requires the same number of multiplications as Gaussian elimination (which is optimal under very general conditions) but requires less memory and does not need pivoting in general; moreover its application to the simplex method has not only a lower memory requirement, but is cheaper in multiplications up to one order with respect to e.g. the Forrest-Goldfarb implementation of the simplex method via the LU factorization. Also there are ABS methods for nonlinear problems that are better than Newton method in term of convergence speed or in term of required information (about $n^2/2$ Jacobian components against the fulll n^2)

- on some classes of significant problems ABSPACK codes have a better performance in both accuracy and speed that the codes in LAPACK (usually considered the best package now on the market)
- ABS methods have allowed to solve open problems in the literature (e.g. the explicit determination of Quasi-Newton updates for the sparse symmetric case)
- for linear Diophantine equations ABS methods have led to a new solution method, that also provides the first generalization of the classical existence theorem of Euclides, Diophantus and Euler from a single n-dimensional equation to a general m-equations system in n variables.

2 The scaled ABS class: definition and main properties

Let us consider the following determined or underdetermined linear system, where $rank(A)$ is arbitrary and $A^T = (a_1, \ldots, a_m)$

$$Ax = b \qquad x \in R^n, \quad b \in R^m, \quad m \leq n \tag{1}$$

or

$$a_i^T x - b_i = 0, \quad i = 1, \ldots, m \tag{2}$$

The above system can be solved by the following *scaled ABS class* of algorithms, which can be shown to provide a complete realization of the general class of methods implementing the *Petrov-Galerkin* criterion (stating that the residual vector at the i-th iteration is orthogonal to a given arbitrary set of $i-1$ linearly independent vectors).

The scaled ABS algorithm

(A) Give $x_1 \in R^n$ arbitrary, $H_1 \in R^{n,n}$ nonsingular arbitrary, $v_1 \in R^m$ arbitrary nonzero. Set $i = 1$.

(B) Compute the residual $r_i = Ax_i - b$. If $r_i = 0$ stop (x_i solves the problem.) Otherwise compute $s_i = H_i A^T v_i$. If $s_i \neq 0$, then go to (C). If $s_i = 0$ and $\tau = v_i^T r_i = 0$, then set $x_{i+1} = x_i$, $H_{i+1} = H_i$ (the i-th equation is redundant) and go to (F). Otherwise stop (the system has no solution).

(C) Compute the search vector p_i by

$$p_i = H_i^T z_i \tag{3}$$

where $z_i \in R^n$ is arbitrary save for the condition

$$v_i^T A H_i^T z_i \neq 0 \tag{4}$$

(D) Update the estimate of the solution by

$$x_{i+1} = x_i - \alpha_i p_i, \quad \alpha_i = v_i^T r_i / v_i^T A p_i. \tag{5}$$

(E) Update the matrix H_i by

$$H_{i+1} = H_i - H_i A^T v_i w_i^T H_i / w_i^T H_i A^T v_i \tag{6}$$

where $w_i \in R^n$ is arbitrary save for the condition

$$w_i^T H_i A^T v_i \neq 0. \tag{7}$$

(F) If $i = m$, stop (x_{m+1} solves the system). Otherwise give $v_{i+1} \in R^m$ arbitrary linearly independent from $v_1, \ldots, v_i$. Increment i by one and go to (**B**).

Matrices H_i, which are generalizations of (oblique) projection matrices, have been named *Abaffians* at the First International Conference on ABS methods (Luoyang, China, 1991). However we must recall that these matrices were used in several little known papers by Egervary, predating the ABS algorithms. There are alternative formulations of the scaled ABS algorithms, e.g. using vectors instead of the square matrix H_i, with possible advantages in storage and number of operations. For details see Abaffy and Spedicato (1989) and on how these formulations can be used to imbed in an ABS approach even iterative methods (including the Kaczmarz, Gauss-Seidel, ORTHOMIN, ORTHODIR etc. methods), see Spedicato and Li (2000).

The choice of the parameters $H_1,\ v_i,\ ,z_i,\ w_i$ determines particular methods. The *basic ABS class* is obtained by taking $v_i = e_i$, as the i-th unit vector in R^m.

We recall some properties of the scaled ABS class, assuming that A has full rank.

- Define $V_i = (v_1, \ldots, v_i)$ and $W_i = (w_1, \ldots, w_i)$. Then

$$H_{i+1} A^T V_i = 0, \quad H_{i+1}^T W_i = 0 \tag{8}$$

.

- The vectors $H_i A^T v_i,\ H_i^T w_i$ are zero if and only if $a_i,\ w_i$ are respectively linearly dependent from $a_1, \ldots, a_{i-1},\ w_1, \ldots, w_{i-1}$.

- Define $P_i = (p_1, \ldots, p_i)$ and $A_i = (a_1, \ldots, a_i)$. Then the implicit factorization $V_i^T A_i^T P_i = L_i$ holds, where L_i is nonsingular lower triangular. Hence, if $m = n$, one obtains a semiexplicit factorization of the inverse, with $P = P_n,\ V = V_n,\ L = L_n$

$$A^{-1} = P L^{-1} V^T. \tag{9}$$

For several choices of V the matrix L is diagonal, hence formula (8) gives a fully explicit factorization of the inverse as a byproduct of the ABS solution of a linear system.

- The general solution of system (1) can be written as follows, with $q \in R^n$ arbitrary

$$x = x_{m+1} + H_{m+1}^T q \tag{10}$$

- The Abaffian can be written in term of the parameter matrices as

$$H_{i+1} = H_1 - H_1 A^T V_i (W_i^T H_1 A^T V_i)^{-1} W_i^T H_1. \tag{11}$$

Letting $V = V_m$, $W = W_m$, one can show that the parameter matrices H_1, V, W are admissible (i.e. condition (7) is satisfied) iff the matrix $Q = V^T A H_1^T W$ is strongly nonsingular (i.e. it is LU factorizable). This condition can be satisfied by exchanges of the columns of V or W. If Q is strongly nonsingular and we take, as is done in all algorithms so far considered, $z_i = w_i$, then condition (4) is also satisfied. Analysis of the conditions under which Q is not strongly nonsingular leads, when dealing with Krylov space methods in their ABS formulation, to a characterization of the topology of the starting points that can produce a breakdown (either a division by zero or a vanishing search direction) and to several ways of curing it, including those considered in the literature.

Three subclasses of the scaled ABS class and particular algorithms are now recalled.

(a) The *conjugate direction subclass*. This class is obtained by setting $v_i = p_i$. It is well defined under the condition (sufficient but not necessary) that A is symmetric and positive definite. It contains the ABS versions of the Choleski, the Hestenes-Stiefel and the Lanczos algorithms. This class generates all possible algorithms whose search directions are A-conjugate. If $x_1 = 0$, the vector x_{i+1} minimizes the energy (A-weighted Euclidean) norm of the error over $Span(p_1, \ldots, p_i)$ and the solution is approached monotonically from below in the energy norm.

(b) The *orthogonally scaled subclass*. This class is obtained by setting $v_i = Ap_i$. It is well defined if A has full column rank and remains well defined even if m is greater than n. It contains the ABS formulation of the QR algorithm (the so called *implicit QR algorithm*), the GMRES and the conjugate residual algorithms. The scaling vectors are orthogonal and the search vectors are $A^T A$-conjugate. If $x_1 = 0$, the vector x_{i+1} minimizes the Euclidean norm of the residual over $Span(p_1, \ldots, p_i)$ and the solution is monotonically approached from below in the residual norm. It can be shown that the methods in this class can be applied to overdetermined systems of $m > n$ equations, where in n steps they obtain the solution in the least squares sense.

(c) The *optimally stable subclass*. This class is obtained by setting $v_i = A^{-T} p_i$, the inverse disappearing in the actual recursions. The search vectors in this class are orthogonal. If $x_1 = 0$, then the vector x_{i+1} is the vector of least Euclidean

norm over $Span(p_1, \ldots, p_i)$ and the solution is approached monotonically from below in the Euclidean norm. The methods of Gram-Schmidt and of Craig belong to this subclass. The methods in this class have minimum error growth in the approximation to the solution according to a criterion by Broyden.

3 Special algorithms in the basic ABS class

In this section we consider in more detail three specially important algorithms, that belong to the basic ABS class (i.e. $V = I$).

(A) The *Huang algorithm* is obtained by the choices $H_1 = I$, $z_i = w_i = a_i$. A mathematically equivalent, but numerically more stable, formulation is the so called *modified Huang algorithm* ($p_i = H_i(H_i a_i)$ and $H_{i+1} = H_i - p_i p_i^T / p_i^T p_i$). Huang algorithm generates search vectors that are orthogonal and identical with those obtained by the Gram-Schmidt procedure applied to the rows of A. If $x_1 = 0$, then x_{i+1} is the solution with least Euclidean norm of the first i equations. The solution x^+ with least Euclidean norm of the whole system is approached monotonically and from below by the sequence x_i. For arbitrary starting point the Huang algorithm generates that solution which is closest in Euclidean norm to the initial point.

(B) The *implicit LU algorithm* is given by the choices $H_1 = I$, $z_i = w_i = e_i$. It is well defined iff A is regular (i.e. all principal submatrices are nonsingular). Otherwise column pivoting has to be performed (or, if $m = n$, equation pivoting). The Abaffian has the following structure, with $K_i \in R^{n-i,i}$

$$H_{i+1} = \begin{bmatrix} 0 & 0 \\ \cdots & \cdots \\ 0 & 0 \\ K_i & I_{n-i} \end{bmatrix}. \tag{12}$$

implying that the matrix P_i is unit upper triangular, so that the implicit factorization $A = LP^{-1}$ is of the LU type, with units on the diagonal. The algorithm requires for $m = n$, $n^3/3$ multiplications plus lower order terms, the same cost of classical LU factorization or Gaussian elimination. Storage requirement for K_i requires at most $n^2/4$ positions, i.e. half the storage needed by Gaussian elimination and a fourth that needed by the LU factorization algorithm (assuming that A is not overwritten). Hence the implicit LU algorithm has same arithmetic cost but uses less memory than the most efficient classical methods.

(C) The *implicit LX algorithm*, see Spedicato, Xia and Zhang (1997), is defined by the choices $H_1 = I$, $z_i = w_i = e_{k_i}$, where k_i is an integer, $1 \leq k_i \leq n$, such that $e_{k_i}^T H_i a_i \neq 0$. By a general property of the ABS class for A with full

rank there is at least one index k_i such that $e_{k_i}^T H_i a_i \neq 0$. For stability reasons we select k_i such that $\mid e_{k_i}^T H_i a_i \mid$ is maximized. This algorithm has the same overhead and memory requirement as the implicit LU algorithm, but does not require pivoting. Its computational performance is also superior and generally better than the performance of the classical LU factorization algorithm with row pivoting, as available for instance in LAPACK or MATLAB, see Mirnia (1996). Therefore this algorithm can be considered as *the most efficient general purpose linear solver not of the Strassen type*.

The implicit LX algorithm has also very important applications in a reformulation of the simplex method for the LP problem, see Zhang, Xia and Feng (1999), where it leads to a reduction of storage up to a factor 8 and of multiplications up to one order for problems where there is a small number of degrees of freedom (m close to n), with respect to implementations based upon the classical LU factorizations, e.g. the one proposed by Forrest and Goldfarb.

4 Solution of linear Diophantine equations

One of the main results in the ABS field has been the derivation of ABS methods for linear Diophantine equations. The ABS algorithm determines if the Diophantine system has an integer solution, computes a particular solution and provides a representation of all integer solutions. It is a generalization of a method proposed by Egervary (1955) for the particular case of a homogeneous system.

Let Z be the set of all integers and consider the Diophantine linear system of equations

$$Ax = b, \qquad x \in Z^n,\ A \in Z^{m\times n},\ b \in Z^m,\ m \leq n. \tag{13}$$

While thousands of papers have been written concerning nonlinear, usually polynomial, Diophantine equations in few variables, the general linear system has attracted much less attention. The single linear equation in n variables was first solved by Bertrand and Betti (1850). Egervary was probably the first author dealing with a system (albeit only the homogeneous one). Several methods for the nonhomogeneous system have recently been proposed based mainly on reduction to canonical forms.

We recall some results from number theory. Let a and b be integers. If there is integer γ so that $b = \gamma a$ then we say that a divides b and write $a|b$, otherwise we write $a \not| b$. If $a_1, \ldots, a_n$ are integers, not all being zero, then the greatest common divisor (gcd) of these numbers is the greatest positive integer δ which divides all a_i, $i = 1, \ldots, n$ and we write $\delta = gcd(a_1, \ldots, a_n)$. We note that $\delta \geq 1$ and that δ can be written as an integer linear combination of the a_i, i.e. $\delta = z^T a$ for some $z \in R^n$. One can show that δ is the least positive integer for which the equation $a_1 x_1 + \ldots + a_n x_n = \delta$ has an integer solution. Now δ plays a main role in the following

Fundamental Theorem of the Linear Diophantine Equation

Let $a_1, \ldots, a_n$ and b be integer numbers. Then the Diophantine linear equation $a_1x_1 + \ldots + a_nx_n = b$ has integer solutions if and only if $gcd(a_1, \ldots, a_n) \mid b$. In such a case if $n > 1$ then there is an infinite number of integer solutions.

In order to find the general integer solution of the Diophantine equation $a_1x_1 + \ldots + a_nx_n = b$, the main step is to solve $a_1x_1 + \ldots + a_nx_n = \delta$, where $\delta = gcd(a_1, \ldots, a_n)$, for a special integer solution. There exist several algorithms for this problem. The basic step is the computation of δ and z, often done using the algorithm of Rosser (1941), which avoids a too rapid growth of the intermediate integers, and which terminates in polynomial time, as shown by Schrijver (1986). The scaled ABS algorithm can be applied to Diophantine equations via a special choice of its parameters, originating from the following considerations and Theorems.

Suppose x_i is an integer vector. Since $x_{i+1} = x_i - \alpha_i p_i$, then x_{i+1} is integer if α_i and p_i are integers. If $v_i^T A p_i \mid (v_i^T r_i)$, then α_i is an integer. If H_i and z_i are respectively an integer matrix and an integer vector, then $p_i = H_i^T z_i$ is also an integer vector. Assume H_i is an integer matrix. From (6), if $v_i^T A H_i^T w_i$ divides all the components of $H_i A^T v_i$, then H_{i+1} is an integer matrix.

Conditions for the existence of an integer solution and determination of all integer solutions of the Diophantine system are given in the following theorems, generalizing the Fundamental Theorem, see Esmaeili, Mahdavi-Amiri and Spedicato (1999), or Fodor (2000) for a different proof under somewhat less general conditions.

Theorem 1 *Let A be full rank and suppose that the Diophantine system (13) is integerly solvable. Consider the Abaffians generated by the scaled ABS algorithm with the parameter choices: H_1 is unimodular (i.e. both H_1 and H_1^{-1} are integer matrices); for $i = 1, \ldots, m$, w_i is such that $w_i^T H_i A^T v_i = \delta_i$, $\delta_i = gcd(H_i A^T v_i)$. Then the following properties are true:*

(a) *the Abaffians generated by the algorithm are well-defined and are integer matrices*

(b) *if y is a special integer solution of the first i equations, then any integer solution x of such equations can be written as $x = y + H_{i+1}^T q$ for some integer vector q.*

Theorem 2 *Let A be full rank and consider the sequence of matrices H_i generated by the scaled ABS algorithm with parameter choices as in Theorem 1. Let x_1 in the scaled ABS algorithm be an arbitrary integer vector and let z_i be such that $z_i^T H_i A^T v_i = gcd(H_i A^T v_i)$. Then system (13) has integer solutions iff $gcd(H_i A^T v_i)$ divides $v_i^T r_i$ for $i = 1, \ldots, m$.*

From the above theorems we obtain the following scaled ABS algorithm for Diophantine equations.

The ABS Algorithm for Diophantine Linear Equations

(1) Choose $x_1 \in Z^n$, arbitrary, $H_1 \in Z^{n\times n}$, arbitrary unimodular. Let $i = 1$.

(2) Compute $\tau_i = v_i^T r_i$ and $s_i = H_i A^T v_i$.

(3) *If* ($s_i = 0$ and $\tau_i = 0$) *then* let $x_{i+1} = x_i$, $H_{i+1} = H_i$, $r_{i+1} = r_i$ and *go to* step (5) (the ith equation is redundant). *If* ($s_i = 0$ and $\tau_i \neq 0$) *then* Stop (the ith equation and hence the system is incompatible).

(4) $\{s_i \neq 0\}$ Compute $\delta_i = gcd(s_i)$ and $p_i = H_i^T z_i$, where $z_i \in Z^n$ is an arbitrary integer vector satisfying $z_i^T s_i = \delta_i$. *If* $\delta_i \not| \tau_i$ *then* Stop (the system is integerly inconsistent), *else* Compute $\alpha_i = \tau_i/\delta_i$, let $x_{i+1} = x_i - \alpha_i p_i$ and update H_i by $H_{i+1} = H_i - \frac{H_i A^T v_i w_i^T H_i}{w_i^T H_i A^T v_i}$ where $w_i \in R^n$ is an arbitrary integer vector satisfying $w_i^T s_i = \delta_i$.

(5) *If* $i = m$ *then* Stop (x_{m+1} is a solution) *else* let $i = i + 1$ and *go to* step (2).

It follows from Theorem 1 that if there exists a solution for the system (13), then $x = x_{m+1} + H_{m+1}^T q$, with arbitrary $q \in Z^n$, provides all solutions of (13).

Egervary's algorithm for homogeneous Diophantine systems corresponds to the choices $H_1 = I$, $x_1 = 0$ and $w_i = z_i$, for all i. Egervary claimed, without proof, that any set of $n - m$ linearly independent rows of H_{m+1} form an integer basis for the general solution of the system. We have shown by a counterexample that Egervary's claim is not true in general; we have also provided an analysis of conditions under which m rows in H_{m+1} can be eliminated.

One can show that by special choices of the scaling parameters and by an inessential modification of the update of the Abaffian it is not necessary to solve in general $2m$ single n-dimensional Diophantine equations (to determine z_i, w_i), but just one single Diophantine equation, at the last step.

The ABS algorithm for linear Diophantine systems has been extended also to systems of m linear inequalities ($m \leq n$), where it provides an almost explicit representation of all solutions, see Esmaeili, Mahdavi and Spedicato (2000a). The used technique allows, in the case of $m \leq n$ real inequalities to find a fully explicit representation of all solutions, see Esmaeili, Mahdavi and Spedicato (2000b), thereby obtaining a generalization of formula (10).

5 ABS methods for KT equations.

When A has a special form, ABS methods can often be modified to exploit the structure of A with a reduction in storage and in number of operations. This has been done e.g. for banded type matrices, block angular matrices and ND type

matrices, see Abaffy and Spedicato (1989). An important case for applications in optimization is that of the so called KT (Kuhn-Tucker) equations. These equations can be obtained considering the optimality conditions for minimizing a quadratic function with Hessian $G \in R^{n,n}$ subject to the linear equality constraint $Cp = c,\ C \in R^{m,n}$. Letting $p, g \in R^n,\ c, z \in R^m$, the KT equations read

$$\begin{bmatrix} G & C^T \\ C & 0 \end{bmatrix} \begin{pmatrix} p \\ z \end{pmatrix} = \begin{pmatrix} g \\ c \end{pmatrix} \tag{14}$$

Here we consider in some detail the derivation of ABS methods that use the structure of system (14), assuming for simplicity that A is nonsingular. Observe that (14) is equivalent to the two subsystems

$$Gp + C^T z = g \tag{15}$$

$$Cp = c. \tag{16}$$

Consider the general solution of $Cp = c$ in the ABS form, with $q \in R^n$ arbitrary

$$p = p_{m+1} + H_{m+1}^T q \tag{17}$$

The parameters used to construct p_{m+1} and H_{m+1} are arbitrary, hence (17) defines a class of algorithms.

Since the KT equations have a unique solution, there is a q which makes p the unique n-dimensional subvector defined by the first n components of the solution of (14). By multiplying $Gp + C^T z = g$ on the left by H_{m+1} we obtain the equation

$$H_{m+1} G p = H_{m+1} g \tag{18}$$

which does not contain z. Now there are two possibilities for determining p:

(A1) Consider the system formed by (18) and (16). Such a system is solvable but overdetermined. Since $\text{rank}(H_{m+1}) = n - m$, m equations are recognized as dependent and are eliminated in step (B) of any ABS algorithm applied to this system, which then computes the unique solution.

(A2) In equation (18) replace p by the general solution (17) to give

$$H_{m+1} G H_{m+1}^T q = H_{m+1} g - H_{m+1} G p_{m+1}. \tag{19}$$

The above system can be solved by any ABS method for a particular solution q, m equations being again removed at step **(B)** of the ABS algorithm as linearly dependent.

Once p is determined, one can determine z in two ways, namely:

(B1) Solve by any ABS method the overdetermined compatible system

$$C^T z = g - Gp \tag{20}$$

by removing at step (B) of the ABS algorithm the $n-m$ dependent equations.

(B2) Let $P = (p_1, \ldots, p_m)$ be the matrix whose columns are the search vectors generated on the system $Cp = c$. Now $CP = L$, with L nonsingular lower diagonal. Multiplying equation (20) on the left by P^T we obtain a triangular system, defining z uniquely

$$L^T z = P^T g - P^T G p. \tag{21}$$

Extensive numerical testing has evaluated the accuracy of the above ABS algorithms for KT equations for certain choices of the ABS parameters (corresponding to the implicit LU algorithm with row pivoting and the modified Huang algorithm). The methods have been tested against the Bunch-Parlett method applied to the (indefinite) original system and the so called range space and null space methods, see Spedicato, Bodon and Luksan (2000c). The experiments have shown that some ABS methods are the most accurate, in both residual and solution error, and faster in the case when the number of degrees of freedom is small (m close to n).

6 ABS unification of feasible direction methods for linearly constrained minimization

ABS methods allow a unification of feasible direction methods for linearly constrained minimization, of which the LP problem is a special case. Let us first consider the problem with only equality constraints

$$min\ f(x), \quad x \in R^n$$

subject to

$$Ax = b, \quad A \in R^{m,n}, \quad m \leq n, \quad rank(A) = m.$$

Let x_1 be a feasible initial point. If we consider an iteration of the form $x_{i+1} = x_i - \alpha_i d_i$, then the sequence x_i consists of feasible points iff

$$Ad_i = 0 \tag{22}$$

The general solution of (22) can be written using the ABS formula (10)

$$d_i = H_{m+1}^T q \tag{23}$$

In (23) matrix H_{m+1} depends on parameters H_1, W and V and $q \in R^n$ can also be seen as a parameter. Hence the general iteration generating feasible points is

$$x_{i+1} = x_i - \alpha_i H_{m+1}^T q. \tag{24}$$

The search vector is a descent direction if $d^T \nabla f(x) = q^T H_{m+1} \nabla f(x) > 0$. This condition can always be satisfied by a choice of q unless $H_{m+1} \nabla f(x) = 0$, implying, from the structure of $Null(H_{m+1})$, that $\nabla f(x) = A^T \lambda$ for some λ, hence that x_{i+1} is a KT point with λ the vector of Lagrange multipliers. If x_{i+1} is not a KT point, then we can generate descent directions by taking

$$q = Q H_{m+1} \nabla f(x) \tag{25}$$

with Q symmetric positive definite. We obtain therefore a large class of methods with four parameter matrices (H_1, W, V, Q).

Some well-known methods in the literature correspond to taking in (25) $W = I$ and building the Abaffian as follows.

(a) The *reduced gradient method* of Wolfe. H_{m+1} is built via the implicit LU method.

(b) The *projection method* of Rosen. H_{m+1} is built via the Huang method.

(c) The *Goldfarb and Idnani method.* H_{m+1} is built via a modification of Huang method where H_1 is a symmetric positive definite approximation of the inverse Hessian of $f(x)$.

To deal with linear inequality constraints there are two approaches in literature.

- The *active set* method. Here the set of equality constraints is augmented with some inequality constraints, whose selection varies in the course of the process till the final set of active constraints is determined. Adding or cancelling a single constraint corresponds to a rank-one correction to the matrix defining the active set. The corresponding change in the Abaffian can be performed in order two operations, see Zhang (1995).

- The *standard form* approach. Here one uses slack variables to put the problem in the following equivalent *standard form*

  $$min\ f(x),$$

  with the constraints

  $$Ax = b, \quad x \geq 0.$$

If x_1 satisfies the above constraints, then a sequence of feasible points is generated by the t iteration

$$x_{i+1} = x_i - \alpha_i \beta_i H_{m+1} \nabla f(x) \tag{26}$$

where α_i may be chosen by a line search along vector $H_{m+1}\nabla f(x)$, while $\beta_i > 0$ is chosen to avoid violation of the nonnegativity constraints.

If $f(x)$ is nonlinear, then H_{m+1} can usually be determined once for all at the first iteration, since generally $\nabla f(x)$ changes from point to point, allowing the determination of a new search direction. However if $f(x) = c^T x$ is linear, in which case we obtain the LP problem, then to get a new search direction we have to change H_{m+1}. We already observed in section 3 that the simplex method in the ABS formulation is obtained by constructing the matrix H_{m+1} via the implicit LX method and at each step modifying one of the unit vectors used to build the Abaffian. One can show that the Karmarkar method corresponds to Abaffians built via a variation of the Huang algorithm, where the initial matrix is $H_1 = Diag(x_i)$ and is changed at every iteration (whether the update of the Abaffian can be performed in order two operations is still an open question). We expect that better methods may be obtained by exploiting all parameters available in the ABS class.

7 ABSPACK and its numerical performance

The ABSPACK project (based upon a collaboration between the University of Bergamo, the Dalian University of Technology and the Czech Academy of Sciences) aims at producing a mathematical package for solving linear and nonlinear systems and optimization problems using the ABS algorithms. The project will take several years for completion, in view of the substantial work needed to test the alternative ways ABS methods can be implemented (via different linear algebra formulations of the process, different possibilities of reprojections, different possible block formulations etc.) and of the necessity of comparing the produced software with the established packages in the market (e.g. MATLAB, LINPACK, LAPACK, UFO ...). It is expected that the software will be documented in a forthcoming monograph and willl be made available to general users.

Presently FORTRAN 77 implementations have been made of several versions of the following ABS algorithms for solving linear systems:

1. The Huang and the modified Huang algorithms in two different linear algebra versions of the process, for solving determined, underdetermined and overdetermined systems, for a solution of least Euclidean norm

2. The implicit LU and implicit LX algorithm for determined, underdetermined and overdetermined linear systems, for a solution of basic type

3. The implicit QR algorithm for determined, underdetermined and overdetermined linear systems, for a solution of basic type

4. The above algorithms for some structured problems, namely KT equations and banded matrices.

For a full presentation of the above methods and their comparison with NAG, LAPACK, LINPACK and UFO codes see Bodon, Luksan and Spedicato (2000a,b,c). Some results are presented in the Appendix. There the columns refer respectively to: the problem, the dimension, the algorithm, the relative solution error (in Euclidean norm), the relative residual error in Euclidean norm (i.e. ratio of residual norm over norm of right hand side), the computed rank and the time in seconds. Computations have been performed in double precision on a Digital Alpha workstation with machine zero about 10^{-17}. All test problems have been generated with integer entries or powers of two such that all entries are exactly represented in the machine and the right hand side can be computed exactly, so that the given solution is an exact solution of the problem as it is represented in the machine. Comparison is given with some LAPACK codes, including those based upon singular value decomposition (*svd*) and rank revealing QR factorization (*gqr*).

Analysis of all obtained results indicates:

1. Modified Huang is generally the most accurate ABS algorithm and compares in accuracy with the best LAPACK solvers based upon singular value factorization and rank revealing QR factorization; also the estimated ranks are usually the same.

2. On problems where the numerical estimated rank is much less than the dimension, one of the versions of modified Huang is much faster than the LAPACK codes using SVD or rank revealing QR factorization, by even a factor more than 100. This is due to the fact that once an equation is recognized as dependent it does not contribute to the general overhead in ABS algorithms.

3. Modified Huang is more accurate than other ABS methods and faster than classical methods on KT equations.

It should be noted that the performance of the considered ABS algorithms in term of times could be improved by developing block versions. ABS methods are moreover expected to be faster than many classical methods on vector and parallel computers, see Bodon (1993).

References

[1] Abaffy J. and Spedicato E., ABS Projection Algorithms: Mathematical Techniques for Linear and Nonlinear Equations, Ellis Horwood, Chichester, 1989

[2] Abaffy J., Broyden C.G. and Spedicato E., A class of direct methods for linear systems, Numerische Mathematik 45, 361-376, 1984

[3] Bertrand I. and Betti G., Traité élémentaire d' algebre, 1850

[4] Bodon E., Numerical experiments with ABS algorithms on banded systems of linear equations, Report CERFACS TR/PA/93/13, Toulouse, 1993

[5] Egervary E., Aufloesung eines homogenen linearen diophantischen Gleichungsystems mit Hilf von Projektormatrizen, Publ. Math. Debrecen 4, 481-483, 1955

[6] Esmaeili H., Mahdavi-Amiri and Spedicato E., Solution of Diophantine linear systems via the ABS methods, Report DMSIA 99/29, University of Bergamo, 1999 (to appear in Numerische Mathematik)

[7] Esmaeili H., Mahdavi-Amiri and Spedicato E., ABS solution of a class of integer inequalities and integer LP problems, preprint, University of Bergamo, 2000a (submitted)

[8] Esmaeili H., Mahdavi-Amiri and Spedicato E., Explicit ABS solution of a class of linear inequality systems and LP problems, University of Bergamo, 2000b (submitted)

[9] Fodor S., A class of ABS methods for Diophantine systems of equations, Part I, Report DMSIA 15/00, University of Bergamo

[10] Mirnia K.: Numerical experiments with iterative refinement of solutions of linear equations by ABS methods, Report DMSIA 32/96, University of Bergamo, 1996

[11] Rosser J.B., A note on the linear Diophantine equation, Amer. Math. Monthly 48, 662-666, 1941

[12] Schrijver A., Theory of Linear and Integer Programming, John Wiley and Sons, 1986

[13] Spedicato E., Bodon E. and Luksan L., Computational experience with ABS algorithms for determined and underdetermined linear systems, Report DMSIA 00/21, University of Bergamo

[14] Spedicato E., Bodon E. and Luksan L., Computational experience with ABS algorithms for overdetermined linear systems, Report DMSIA 00/19, University of Bergamo

[15] Spedicato E., Bodon E. and Luksan L., Computational experience with ABS algorithms for KKT linear systems, Report DMSIA 00/20, University of Bergamo

[16] Spedicato E. and Li Z., On some classes of iterative ABS-like methods for large scale linear systems, 1999 (to appear)

[17] Spedicato E., Xia Z. and Zhang L., The implicit LX method of the ABS class, Optimization Methods and Software 8, 99-110, 1997

[18] Zhang L., Updating of Abaffian under perturbation in W and A, Report DMSIA 95/15, University of Bergamo

APPENDIX

RESULTS ON DETERMINED LINEAR SYSTEMS

Condition number: 0.21D+20

IDF2	2000	huang2	0.10D+01	0.69D-11	2000	262.00
IDF2	2000	mod.huang2	0.14D+01	0.96D-12	4	7.00
IDF2	2000	lu lapack	0.67D+04	0.18D-11	2000	53.00
IDF2	2000	qr lapack	0.34D+04	0.92D-12	2000	137.00
IDF2	2000	gqr lapack	0.10D+01	0.20D-14	3	226.00
IDF2	2000	lu linpack	0.67D+04	0.18D-11	2000	136.00

Condition number: 0.10D+61

IR50	1000	huang2	0.46D+00	0.33D-09	1000	36.00
IR50	1000	mod.huang2	0.46D+00	0.27D-14	772	61.00
IR50	1000	lu lapack	0.12D+04	0.12D+04	972	7.00
IR50	1000	qr lapack	0.63D+02	0.17D-12	1000	17.00
IR50	1000	gqr lapack	0.46D+00	0.42D-14	772	29.00
IR50	1000	lu linpack	--- break-down ---			

RESULTS ON OVERDETERMINED SYSTEMS

Condition number: 0.16D+21

IDF3	1050	950	huang7	0.32D+04	0.52D-13	950	31.00
IDF3	1050	950	mod.huang7	0.14D+04	0.20D-09	2	0.00
IDF3	1050	950	qr lapack	0.37D+13	0.83D-02	950	17.00
IDF3	1050	950	svd lapack	0.10D+01	0.24D-14	2	145.00
IDF3	1050	950	gqr lapack	0.10D+01	0.22D-14	2	27.00

Condition number: 0.63D+19

IDF3	2000	400	huang7	0.38D+04	0.35D-12	400	9.00
IDF3	2000	400	mod.huang7	0.44D+03	0.67D-12	2	0.00
IDF3	2000	400	impl.qr5	0.44D+03	0.62D-16	2	0.00
IDF3	2000	400	expl.qr	0.10D+01	0.62D-03	2	0.00
IDF3	2000	400	qr lapack	0.45D+12	0.24D-02	400	8.00
IDF3	2000	400	svd lapack	0.10D+01	0.65D-15	2	17.00
IDF3	2000	400	gqr lapack	0.10D+01	0.19D-14	2	12.00

RESULTS ON UNDERDETERMINED LINEAR SYSTEMS

Condition number:			0.29D+18				
IDF2	400	2000	huang2	0.12D-10	0.10D-12	400	12.00
IDF2	400	2000	mod.huang2	0.36D-08	0.61D-10	3	1.00
IDF2	400	2000	qr lapack	0.29D+03	0.37D-14	400	9.00
IDF2	400	2000	svd lapack	0.43D-13	0.22D-14	3	68.00
IDF2	400	2000	gqr lapack	0.18D-13	0.24D-14	3	12.00
Condition number:			0.24D+19				
IDF3	950	1050	huang2	0.00D+00	0.00D+00	950	33.00
IDF3	950	1050	mod.huang2	0.00D+00	0.00D+00	2	1.00
IDF3	950	1050	qr lapack	0.24D+03	0.56D-14	950	17.00
IDF3	950	1050	svd lapack	0.17D-14	0.92D-16	2	178.00
IDF3	950	1050	gqr lapack	0.21D-14	0.55D-15	2	26.00

RESULTS ON KT SYSTEMS

Condition number:			0.26D+21				
IDF2	1000	900	mod.huang	0.55D+01	0.23D-14	16	24.00
IDF2	1000	900	impl.lu8	0.44D+13	0.21D-03	1900	18.00
IDF2	1000	900	impl.lu9	0.12D+15	0.80D-02	1900	21.00
IDF2	1000	900	lu lapack	0.25D+03	0.31D-13	1900	62.00
IDF2	1000	900	range space	0.16D+05	0.14D-11	1900	87.00
IDF2	1000	900	null space	0.89D+03	0.15D-12	1900	93.00
Condition number:			0.70D+20				
IDF2	1200	600	mod.huang	0.62D+01	0.20D-14	17	36.00
IDF2	1200	600	impl.lu8	0.22D+07	0.10D-08	1800	44.00
IDF2	1200	600	impl.lu9	0.21D+06	0.56D-09	1800	33.00
IDF2	1200	600	lu lapack	0.10D+03	0.79D-14	1800	47.00
IDF2	1200	600	range space	0.11D+05	0.15D-11	1800	63.00
IDF2	1200	600	null space	0.38D+04	0.13D-12	1800	105.00