Comp. & Maths. with Appls., Vol. 3, pp. 235–245. Pergamon Press 1977. Printed in Great Britain

ALGORITHMS FOR PROJECTION METHODS FOR SOLVING LINEAR SYSTEMS OF EQUATIONS

ROGER L. WAINWRIGHT
The University of Tulsa, Tulsa, Oklahoma 74104, U.S.A.

and

R. F. KELLER
Iowa State University, Ames Laboratory, USERDA, Ames, Iowa 50010, U.S.A.

Communicated by E. Y. Rodin

(*Received December*, 1976)

Abstract—The solution of linear systems of equations using various projection algorithms is considered. Since nonsingularity of the coefficient matrix is the only requirement for convergence, techniques for increasing the rate of convergence are presented. Various criteria for the selection of two and three-dimensional subspaces to project the residual vector at any step are reviewed in the literature. These algorithms are called quasi-optimal since the subspaces formed by the column vectors of the coefficient matrix are done *a priori*. They are shown to significantly reduce the number of cycles required for convergence and compare favorably with standard methods. A new class of projection algorithms is presented which is proven to be superior to conventional projection algorithms. It is shown that the new algorithms are equivalent to the conventional projection algorithms, but require less than half the number of arithmetic computations per iterative step and that the number of computations required per iterative step per component is independent of the dimension of the algorithm used.

1. INTRODUCTION

Projection methods for solving systems of linear equations have been known for some time. The initial development was done by A. de la Garza[1]. His work established what is now called n-dimensional methods, but, convergence conditions were not obtained. His work is geometrically described by Householder[2].

Keller[3] independently developed a 1-dimensional class of projection methods and proved that nonsingularity of the coefficient matrix was sufficient for convergence. The same type of argument as given by Keller establishes the same sufficient conditions for any n-dimensional projection method where $n \geq 0$ and less than or equal to the dimension of the linear system.

Projection methods are among the class of methods called gradient methods. For the development of the 1-dimensional projection method as a special case of the class of gradient methods, see Fox[4, p. 205–206].

Projection methods are so named because an m-dimensional method will change m components of the approximate solution vector at each iterative step by projecting the residual vector onto a subspace determined by m of the column vectors of A.

Since projection methods are guaranteed to converge, research is devoted totally to the rate of convergence which depends on properties of the linear system itself as well as:

(a) the number of iteration steps required. This is dependent on the dimension of the method used as well as which column(s) of the coefficient matrix to project the residual vector onto at each iterative step.

(b) the number of arithmetic computations required per iteration step.

In this paper, we first present the basic concept of projection methods (Section 2). All of the research to date has been with respect to (a) above. A summary of this research is presented in Section 3. In Section 4 we present a new class of projection algorithms. They are shown to be equivalent to the old (or conventional) projection algorithms presented in Section 2 and require less than half the number of arithmetic computations per iteration step. In addition it is shown that the number of operations per component per iterative step of the new projection algorithms is independent of the dimension of the method used. Test problems and comparisons

are summarized in Section 5. Concluding remarks are presented in Section 6 followed by the new 2-dimensional projection algorithm.

2. BASIC CONCEPTS OF PROJECTION METHODS

For a system of n linear equations defined by $Ax = b$ an m-dimensional projection method ($1 \leq m \leq n$) can be derived by minimizing the quadratic form (r^k, r^k) where $r^k = b - Ax^k$ is the residual vector of the kth iteration. The approximation to the solution vector is modified m components at a time at each iteration by the following scheme

$$x^{k+1} = x^k + d_1{}^k e_1 + d_2{}^k e_2 \cdots + d_m{}^k e_m \tag{2.1}$$

where $x_1, x_2, x_3, \ldots, x_m$ are m arbitrary components of the approximate solution vector. d_i^k is the change to the ith component of x at the kth iteration and e_i is the ith column vector of an $n \times n$ identity matrix. The next residual vector after each step is computed by $r^{k+1} = b - Ax^{k+1}$. Substituting (2.1) for x^{k+1} and expanding, the next residual vector becomes

$$r^{k+1} = r^k - d_1{}^k a_1 - d_2{}^k a_2 \cdots - d_m{}^k a_m \tag{2.2}$$

where a_i is the ith column of A.

The process involved in the development of an m-dimensional projection method is presented in the following theorem:

THEOREM 2.1. The changes to m components of the approximate solution vector at the kth step of an m-dimensional projection method used to solve a linear system of n equations with a nonsingular coefficient matrix are given as the solution to the following symmetric system of m linear equations.

$$\begin{aligned}
&a_{11}d_1{}^k + a_{12}d_2{}^k + a_{13}d_3{}^k \cdots + a_{1,m}d_m{}^k = (r^k, a_1)\\
&a_{21}d_1{}^k + a_{22}d_2{}^k + a_{23}d_3{}^k \cdots + a_{2,m}d_m{}^k = (r^k, a_2)\\
&\vdots\\
&a_{m,1}d_1{}^k + a_{m,2}d_2{}^k + a_{m,3}d_3{}^k \cdots + a_{m,m}d_m{}^k = (r^k, a_m)
\end{aligned}$$

where $x_1, x_2, x_3, \ldots, x_m$ are m arbitrary components of the approximate solution vector and $a_{ij} = (a_j, a_j)$. A proof of this can be found in Householder [2].

One normally chooses $w = \lfloor (n + m - 1)/m \rfloor$ groups of m columns of the coefficient matrix in such a way that each column is included in at least one group. This minimizes the number of iterations required per cycle. $\lfloor y \rfloor$ represents the floor function which is defined as the greatest integer less than or equal to y where y is any arithmetic expression. Hence, every w iterative steps constitutes a cycle of a stationary method, i.e. every element of the approximate solution vector will be changed at least once.

Each iterative step involves solving a symmetric system of m linear equations. Usually this system is solved by some direct method, such as Gaussian elimination where first the coefficient matrix is transformed into an upper triangular matrix and back substitution is performed to obtain an m element solution vector.

From cycle to cycle the same w linear systems are solved over and over with only the constant vector changing. Hence, triangularizing these matrices need only be done once.

Define B_i to be a back substitution function for Gaussian elimination to obtain d_i at any given iterative step. To emphasize that B_i depends only on the constant vector one can write

$$d_i^k = B_i[(r^k, a_i)] \tag{2.3}$$

where

$$(r^k, a_i)$$

is the ith component of the constant vector at the kth iterative step, and the coefficient matrix is that of Theorem 2.1.

The calculations that are performed only once are presented in Table 1. Substituting $(n+m-1)/m$ for w in Table 1, the total number of overhead operations is bounded by

$$n^3+n^2+\frac{2m^3}{3}-\frac{7m^2}{6}+\frac{2m^2n}{3}+\frac{m}{3}-\frac{mn}{2}-\frac{n}{6}+\frac{1}{6}. \tag{2.4}$$

Let $x_1, x_2, \ldots, x_m$ be m arbitrary components of the approximate solution vector modified by a given iterative step, then Table 2 depicts in the order given the calculations that are performed at each step. The total number of operations performed at each step is $4mn+2m^2$. Hence, the number of operations required per component per iterative step is $4n+2m$ which is dependent on the dimension of the method.

3. SELECTION OF PROJECTION SUBSPACES

In general one would like to be able to select *a priori* a sequence of projection subspaces which guarantee most rapid convergence for any given system of linear equations.

Keller[5], showed that for any given 1-dimensional method there is a 2-dimensional method which is essentially an acceleration of the 1-dimensional one. Pyron[6], Keller[5], and Georg and Keller[7] established criteria for selecting quasi-optimal and optimal 2-dimensional methods. Tokko and Keller[8] established criteria for selecting quasi-optimal 3-dimensional methods.

Although current work is extending the existing subspace selection algorithms significantly we are yet a long way from determining *a priori* the optimal sequence. Even so projection methods are now competitive with other well known iterative methods.

The crucial information needed for selecting an optimal projection is how the residue at any step is related to the subspaces determined by columns of the coefficient matrix. In particular, one wants the residue to lie in some projection subspace. This would give maximum reduction of the residual vector. As illustrated in [7] one can determine a subspace such that the residue 'most nearly' lies in this subspace using direction cosines of the residue and the column vectors. This requires 'overhead' computation which is not always justified. The quasi-optimal methods outlined below were developed to avoid this overhead and use the angles between column vectors instead of direction cosines.

In [6] a quasi-optimal 2-dimensional method is developed. This is obtained by the formula derived for (r^{k+1}, r^{k+1}),

$$(r^{k+1}, r^{k+1}) = (r^k, r^k) - C_j^i R_j^i \tag{3.1}$$

Table 1. Calculations that are performed only once (fixed overhead calculations)

	Number of additions	Number of multiplications
Calculate a_{ij} (note $a_{ij}=a_{ji}$)	$n^2(n+1)/2$	$n^2(n+1)/2$
Upper triangularizing w $m\times m$ coefficient matrices	$\left(\frac{m^3}{3}-\frac{m^2}{2}+\frac{m}{6}\right)w$	$\left(\frac{m^3}{3}-\frac{m}{3}\right)w$

Table 2. Calculations that are performed at each step in the order given

	Number of additions	Number of multiplications
Calculations to determine the constant vector: $((r^k, a_1), (r^k, a_2), \ldots, (r^k, a_m))$	mn	mn
Convert the constant vector as Gauss elimination would do	$\frac{m^2-m}{2}$	$\frac{m^2-m}{2}$
Determine $d_1^k, d_2^k, \ldots, d_m^k$ by back substitution, i.e. solving the B_i functions $(i=1\text{–}m)$	$\frac{m^2-m}{2}$	$\frac{m^2+m}{2}$
Calculate $x_1^{k+1}, x_2^{k+1}, \ldots, x_m^{k+1}$ by (2.1)	m	0
Calculate r^{k+1} by (2.2)	mn	mn

where

$$C_j^i = \frac{1}{1-\cos^2\theta_{ij}}$$

and

$$R_j^i = \frac{(r^k, a_i)^2}{(a_i, a_i)} + \frac{(r^k, a_j)^2}{(a_j, a_j)} - \frac{2(r^k, a_i)(r^k, a_j)(a_i, a_j)}{(a_i, a_i)(a_j, a_j)}.$$

To select a single step optimal method (i.e. one which would give a minimum value for (r^{k+1}, r^{k+1})) i, j are chosen such that $C_j^i R_j^i$ is a maximum for $i, j = 1, 2, \ldots, n$. Now $C_j^i \geq 1$ and $R_j^i \to 0$ as $r^k \to \emptyset$ ($\emptyset$ is the zero vector). Thus C_j^i dominates the product $C_j^i R_j^i$ as r^k becomes small. A quasi-optimal 2-dimensional algorithm is obtained by taking pairs of columns of A in decending order of the values of C_j^i $(i,j = 1, 2, \ldots, n)$. Each C_j^i is used at least once per cycle but none used more than twice. All C_j^i $(i,j = 1, 2, \ldots, n)$ are computed initially and a stationary algorithm is obtained. The largest C_j^i results from the two most parallel columns of A.

In [7] further criteria for ordering columns for 2-dimensional projection methods is established and studied. In particular the equation for R_j^i above is rewritten in terms of cosines of angles involved. We obtain

$$R_j^i = (r^k, r^k)[\cos^2\theta_{i,r}k + \cos^2\theta_{j,r}k - 2\cos\theta_{i,j}\cos\theta_{i,r}k\cos\theta_{j,r}k]$$

(where $\theta_{i,r}k$ is the angle between a_i and r^k). Using this in (3.1) we get

$$(r^{k+1}, r^{k+1}) = (r^k, r^k)(1 - Q_j^i) \tag{3.2}$$

where

$$Q_j^i = \frac{\cos^2\theta_{i,r}k + \cos^2\theta_{j,r}k - 2\cos\theta_{i,j}\cos\theta_{i,r}k\cos\theta_{j,r}k}{1-\cos^2\theta_{i,j}}.$$

From (3.2) we write (r^{k+1}, r^{k+1}) in terms of (r^0, r^0)

$$(r^{k+1}, r^{k+1}) = (r^0, r^0)\prod_{g=1}^{k}(1 - Q_{ig}^i).$$

This makes it possible to define an n-step optimal 2-dimensional method for any finite n as one which minimizes

$$\prod_{g=t}^{t+n}(1 - Q_{ig}^i).$$

A 1-step optimal 2-dimensional method clearly from (3.2) should maximize Q_j^i (Q_j^i is always less than or equal to 1) where now

$$Q_j^i = C_j^i(\cos^2\theta_{i,r}k + \cos^2\theta_{j,r}k - 2\cos\theta_{i,j}\cos\theta_{i,r}k\cos\theta_{j,r}k).$$

If C_j^i for all i, j are computed initially, then obtaining the maximum Q_j^i requires computing (r^k, a_i) for $i = 1, 2, \ldots, n$. This is a great amount of computing but as examples show in [7] it sometimes is worth doing. A more practical algorithm is developed by Georg[7] which does not maximize Q_j^i. He uses direction cosines to maximize $\cos^2\theta_{i,r}k$ and $\cos^2\theta_{j,r}k$ and selects i, j based on these values and C_j^i. Direction cosines are computed initially for all a_i, $i = 1, 2, \ldots, n$, and then at each step the direction cosines of r^k only need to be computed.

In [8] 3-dimensional optimal projection methods are considered and 1-step quasi-optimal algorithms are developed. As for the 2-dimensional methods the basic equation to be maximized has the form:

$$(r^k, r^k) - (r^{k+1}, r^{k+1}) = \bar{C}\bar{R}$$

but now $\bar{C}$ and $\bar{R}$ are similar to C_j^i and R_j^i but somewhat more complicated. Here we wish to

select three column vectors of A such that $\bar{C}\bar{R}$ is a maximum. The triple is denoted by $(1, 2, 3)$ instead of say (i, j, k) to simplify notation. The equations for $\bar{C}$ and $\bar{R}$ are:

$$\bar{C} = \frac{1}{1 + 2\cos\theta_{1,2}\cos\theta_{1,3}\cos\theta_{2,3} - \cos^2\theta_{1,2} - \cos^2\theta_{1,3} - \cos^2\theta_{2,3}}$$

$$\begin{aligned}\bar{R} = {} & \frac{(r^k, a_1)^2}{(a_1, a_1)}(1 - \cos^2\theta_{2,3}) + \frac{(r^k, a_2)^2}{(a_2, a_2)}(1 - \cos^2\theta_{1,3}) + \frac{(r^k, a_3)^2}{(a_3, a_3)}(1 - \cos^2\theta_{1,2}) \\ & + 2\frac{(r^k, a_1)(r^k, a_2)}{[(a_1, a_1)(a_2, a_2)]^{1/2}}(\cos\theta_{1,3}\cos\theta_{2,3} - \cos\theta_{1,2}) \\ & + 2\frac{(r^k, a_1)(r^k, a_3)}{[(a_1, a_1)(a_3, a_3)]^{1/2}}(\cos\theta_{1,2}\cos\theta_{2,3} - \cos\theta_{1,3}) \\ & + 2\frac{(r^k, a_2)(r^k, a_3)}{[(a_2, a_2)(a_3, a_3)]^{1/2}}(\cos\theta_{1,3}\cos\theta_{1,2} - \cos\theta_{2,3}).\end{aligned}$$

As for the 2-dimensional methods $\bar{R} \to 0$ as $r^k \to \emptyset$ and $\bar{C} \geq 1$ so $\bar{C}$ dominates the product, $\bar{C}\bar{R}$, as r^k becomes small. Quasi-optimal algorithms then can be based on maximizing $\bar{C}$. Tokko[8] does this by choosing the triple $(1, 2, 3)$ in such a way to maximize $\cos^2\theta_{1,2} + \cos^2\theta_{1,3} + \cos^2\theta_{2,3}$ and shows how the magnitude of $\bar{C}$ is affected. The triples are selected on the basis of being the most co-planar which is what maximizes $\cos^2\theta_{1,2} + \cos^2\theta_{1,3} + \cos^2\theta_{2,3}$. The most co-planar triple is used first, the second most co-planar next, etc., with an added criteria that each column vector of A be used at least once. The result is a stationary algorithm with triples determined from initial calculations based on angles between column vectors of A.

4. NEW PROJECTION ALGORITHMS

Define

$$\bar{d}_i^k = \sum_{j=1}^{k} d_i^j = \bar{d}_i^{k-1} + d_i^k \tag{4.1}$$

to be the sum of all changes to the ith component of x accumulated through the kth step. A few observations concerning the B_i notation are now in order.

THEOREM 4.1. Let $x_1, x_2, \ldots, x_m$ be m arbitrary components of the approximate solution vector to be modified during an iteration step of an m-dimensional projection method. Then for any $1 \leq i \leq m$ the following identity holds

$$B_i[(-\bar{d}_1^k a_1 - \bar{d}_2^k a_2 \cdots - \bar{d}_i^k a_i \cdots - \bar{d}_m^k a_m, a_i)] = -\bar{d}_i^k$$

Proof. Applying the result of Theorem 2.1 and (2.3)

$$B_i[(-\bar{d}_1^k a_1 - \bar{d}_2^k a_2 \cdots - \bar{d}_i^k a_i \cdots - \bar{d}_m^k a_m, a_i)]$$

(for $i = 1 - m$) is the solution vector, y, of the following system of m linear equations

$$\begin{aligned}
a_{11}y_1 + a_{12}y_2 \cdots + a_{1,n}y_n &= (-\bar{d}_1^k a_1 - \bar{d}_2^k a_2 \cdots - \bar{d}_m^k a_m, a_1) \\
a_{21}y_1 + a_{22}y_2 \cdots + a_{2,n}y_n &= (-\bar{d}_1^k a_1 - \bar{d}_2^k a_2 \cdots - \bar{d}_m^k a_m, a_2) \\
&\vdots \\
a_{m,1}y_1 + a_{m,2}y_2 \cdots + a_{m,n}y_n &= (-\bar{d}_1^k a_1 - \bar{d}_2^k a_2 \cdots - \bar{d}_m^k a_m, a_m).
\end{aligned}$$

The solution to the above system is

$$y_i = -\bar{d}_i^k, \quad i = 1, 2, \ldots, m$$

which is the desired result.

A second observation is that B_i is a distributive operator, i.e.,

$$B_i[p+q] = B_i[p] + B_i[q]. \tag{4.2}$$

A simple proof can be found in Wainwright[9].

The change to the ith component of the approximate solution vector at the kth step for the old projection algorithm is given by (2.3). After substituting (2.2) and (4.1) into (2.3) it follows that

$$d_i^k = B_i[(r^0 - \bar{d}_1^k a_1 - \bar{d}_2^k a_2 \cdots - \bar{d}_i^k a_i \cdots - \bar{d}_n^k a_n, a_i)]. \tag{4.3}$$

Now an application of (4.2) yields

$$\begin{aligned} d_i^k = {} & B_i[(r^0 - \bar{d}_{m+1}^k a_{m+1} - \bar{d}_{m+2}^k a_{m+2} \cdots - \bar{d}_n^k a_n, a_i)] \\ & + B_i[(-\bar{d}_1^k a_1 - \bar{d}_2^k a_2 \cdots - \bar{d}_m^k a_m, a_i)] \end{aligned}$$

and finally applying (4.1) and Theorem 4.1 the final result is obtained:

$$\bar{d}_i^k = B_i[(r^0 - \bar{d}_{m+1}^k a_{m+1} - \bar{d}_{m+2}^k a_{m+2} \cdots - \bar{d}_n^k a_n, a_i)]. \tag{4.4}$$

Let the new m-dimensional projection algorithm be defined by (4.4) then since (4.4) is (2.3) with r^k replaced by an expression equal to it the new and old projection algorithms are equivalent.

The steps involved in using the new m-dimensional projection algorithm are summarized below.

(1) Choose the dimension of the algorithm to be used, m.

(2) Choose the groupings of the columns of the coefficient matrix m at a time such that every column is in at least one group. Let $w = \lfloor (m+n-1)/m \rfloor$ be the number of groups chosen.

(3) The following fixed overhead operations are performed.

	Number of additions	Number of multiplications
(a) a_{ij}	$n^2(n+1)/2$	$n^2(n+1)/2$
(b) Upper triangularize w $m \times m$ coefficient matrices (B_i functions)	$\left(\frac{m^3}{3} - \frac{m^2}{2} + \frac{m}{6}\right)w$	$\left(\frac{m^3}{3} - \frac{m}{3}\right)w$
(c) (r^0, a_i)	n^2	n^2

The total number of fixed overhead operations (additions plus multiplications) is

$$n^3 + 3n^2 + \frac{2m^3}{3} - \frac{7m^2}{6} + \frac{2m^2 n}{3} + \frac{m}{3} - \frac{mn}{2} - \frac{n}{6} + \frac{1}{6}. \tag{4.5}$$

(4) At each iterative step the following calculations are performed.

	Number of additions	Number of multiplications
(a) Calculate the constant vector in (4.4)	$m(n-m)$	$m(n-m)$
(b) Convert the constant vector as Gaussian elimination would do	$(m^2-m)/2$	$(m^2-m)/2$
(c) Determine $x_1^k, x_2^k, \ldots, x_m^k$ by evaluating $B_1, \ldots, B_m$	$(m^2-m)/2$	$(m^2+m)/2$

Thus, the total number of operations to be performed at each step is $2mn - m$ which is

$$2n - 1 \tag{4.6}$$

operations per component.

(5) The new projection algorithm bypasses the use of the residual vector, thus termination of the iterative process is done by looking at successive approximate solution vectors after each cycle. If every corresponding component of two successive approximate solution vectors is within a predetermined tolerance then the iterative process terminates. The cost is insignificant when done once a cycle. However, as in the old projection algorithms one could calculate (r^k, r^k) once a cycle to see if it is less than some tolerance, but the total cost in operations is $2n^2+3n$, which is defeating since it is more than the cost of the cycle itself!

In the preceding analysis w was chosen to minimize the number of groups of size m among n columns. This minimizes the fixed overhead operational costs and also the number of iterations per cycle. Also m was assumed to be a fixed integer. This need not be the case. It may be that the rate of convergence may be faster if some of the columns are grouped in pairs, some in triples, sextets or whatever. At present there is no evidence that varying m at each step increases the rate of convergence nor is there any evidence that w should be chosen by any other criteria. The groupings must be chosen *a priori* and once chosen cannot change from cycle to cycle.

THEOREM 4.2. For any system of n linear algebraic equations with a nonsingular coefficient matrix and for any dimensional projection method, m, and for any initial starting vector, x^0, the new projection algorithm will obtain the solution in fewer arithmetic operations than the old projection algorithm.

Proof. From (4.5) and (2.4) the following is obtained.

	New projection algorithm	Old projection algorithm
Fixed overhead operations	$n^3+3n^2+2m^3/3$	$n^3+n^2+2m^3/3$
	$-7m^2/6+2m^2n/3+m/3$	$-7m^2/6+2m^2n/3+m/3$
	$-mn/2-n/6+1/6$	$-mn/2-n/6+1/6$
Step operations	$2mn-m$	$4mn+2m^2$

Let z be the step number at which point the new and the old projection algorithms cost the same (i.e. the breakeven point). z is determined by the following formula:

$$Fo+zSo = Fn+zSn$$

where Fo, Fn are the fixed overhead calculations for the old and the new projection methods respectively, and So, Sn are the step calculations for the old and new projection methods respectively. Solving for z one obtains $z = 2n^2/(2m^2+2mn+m)$. Finally substituting $(m+n-1)/m$ for w one obtains the following relationships:

$$2mn^2 < 2mn^2+2m^2n+mn$$

$$\frac{2n^2}{2m^2+2mn+m} < \frac{n}{m} < w.$$

Therefore, z is less than w and thus the breakeven point is somewhere in the first cycle. Since the iterative process must go at least one cycle it is clear that the new projection algorithm will always take fewer arithmetic operations than the old projection algorithm to obtain the solution.

A few observations should be mentioned at this point. The new projection algorithm solves for the new components of the approximate solution vector directly rather than just the changes to the components at each step. The step costs of the new algorithm are less than half that of the old algorithm and are independent of the dimension of the method.

COROLLARY 4.1. The rate of convergence of the new projection method does not depend on the number of arithmetic operations per step.

Proof. The proof follows immediately since (4.6) is independent of m. Note that the additional overhead calculation incurred for the new method, $2n^2$, is also independent of the method used.

An exception to Corollary 4.1 is when n is not an even multiple of m. For example, let $n = 10$ then for $m = 2$, $w = 5$ and each component is changed once per cycle. For $n = 10$ with $m = 3$, $w = 4$ and two of the components are changed twice per cycle. Hence, more operations are performed per cycle with $m = 3$ than $m = 2$. If one allows m to vary in the last step of the cycle in order that no component be changed twice in the same cycle then the number of operations performed per cycle is independent of m.

One advantage in general of an iterative method over a direct method is that rounding errors of an iterative method are not accumulative. This is certainly true of the old projection algorithms. The new projection algorithms, however, may be subject to rounding error propagation because the residual vector is not calculated directly after each step as in (2.2), but indirectly as in (4.3). Wainwright[9] presents a rounding error propagation prevention mechanism to be used with the new projection algorithms if needed. It requires $3n + m$ calculation per component per iterative step and $2n^2$ additional overhead calculations. He also indicates that rounding errors rarely propagate to a point where this mechanism need be used.

Equation (4.4) defines the new projection algorithm. When this process converges one must add the resulting vector of accumulated changes, $\bar{d}$, to x^0 to obtain the solution. There are two ways to choose x^0:

(1) Choose $x^0 = \emptyset$ then r^0 becomes b and the resulting vector of changes, $\bar{d}$, becomes the solution itself, x. In this case one can depict (4.4) as

$$x_i^k = B_i[(r^0 - x_{m+1}^k a_{m+1} - x_{m+2}^k a_{m+2} \cdots - x_n{}^k a_n, a_i)] \tag{4.7}$$

since the vector of total changes, $\bar{d}^k$, is the approximate solution vector, x^k.

(2) Choose $x^0 \neq \emptyset$ then at the first iteration when $k = 1$ no changes to the approximate solution vector have been calculated, hence (4.4) becomes

$$d_i^1 = B_i[(r^0, a_i)] \text{ where } r^0 = b - Ax^0. \tag{4.8}$$

Assume that x^0 is some non-zero vector of changes resulting from applying a new projection algorithm for k steps from an initial vector of zero. In this way (4.8) using some $x^0 \neq \emptyset$ is equivalent to (4.4) and (4.7) using an initial vector of zero for some iterative process. Thus, regardless of the choice of x^0, (4.7) defines the new m-dimensional projection algorithm. Furthermore, regardless of x^0, r^0 is chosen as if x^0 were identically zero, i.e. $r^0 = b$.

5. TEST PROBLEMS AND COMPARISONS

Wainwright[9] establishes the clear superiority of the new projection algorithms over the old projection algorithms giving a number of test problems. A portion of his results is summarized below.

Matrix		Methods G–S	OLD2	NEW2	OLD3	NEW3
1.	#Steps	Failed	436	436	399	397
	C.P.U.	—	1.47	0.99	1.57	0.87
2.	#Steps	198	255	255	102	102
	C.P.U.	0.74	1.27	0.99	0.99	0.64
3.	#Steps	Failed	6552	6552	7556	7556
	C.P.U.	—	7.94	4.25	11.45	6.12
4.	#Steps	228	2400	2400	464	464
	C.P.U.	0.95	3.62	1.90	1.54	0.95
5.	#Steps	30	15	15	10	10
	C.P.U.	0.59	0.59	0.65	0.50	0.62
6.	#Steps	816	2088	2088	1131	1131
	C.P.U.	1.10	3.47	2.14	2.24	1.62
7.	#Steps	108	135	135	24	24
	C.P.U.	0.64	0.84	0.74	0.70	0.52
8.	#Steps	Failed	3030	3030	5001*	5001*
	C.P.U.	—	5.12	3.27	10.22	5.65
9.	#Steps	20	6470	6470	7680	7680
	C.P.U.	0.59	10.64	6.64	15.52	9.15
10.	#Steps	Failed	3236	3236	2052	2052
	C.P.U.	—	4.20	2.45	3.80	2.15

*Step limit was reached.

G–S is the single step algorithm of Gauss–Seidel. OLD2, NEW2, OLD3, NEW3 are the old two-, new two-, old three- and new three-dimensional projection algorithms respectively. Each program is written in PL/1 compiled with the optimizing compiler and executed from a load module on an I.B.M. 360/65 under Hasp with nothing else in the system. All C.P.U. times are in seconds and all computations are double precision. No algorithm for subspace selections is used. The columns of the coefficient matrix in all cases are grouped consecutively. For example, for $n = 8$ and $m = 2$ all groupings are (1, 2), (3, 4), (5, 6), (7, 8) and for $m = 3$ the groupings are (1, 2, 3), (4, 5, 6), (6, 7, 8). Rounding error propagation correction is not used and in no case did this seem to have any effect whatsoever. Matrices #2, #6 and #7 are sparse and diagonally dominant. Matrix #8 is sparse and #5 is diagonally dominant. Matrix #3 is nearly singular and #9 is lower triangular. Matrices #1, #4 and #10 have no special characteristics. Since no subspace selection algorithm was used it is not surprising that the projection methods do not compare well with G–S.

6. CONCLUSION

The single most important attribute of projection algorithms is that for any dimension used for any nonsingular coefficient matrix the iterative process is guaranteed to converge. The conventional projection algorithms used with the various *a priori* subspace selection algorithms prove to be competitive with other known iterative algorithms [5–8]. The new projection algorithms presented are equivalent to the old projection algorithms, that is, they generate the same sequence of approximate solution vectors, hence, have the same rate of convergence. The only difference is that the new projection algorithms require less than half the number of arithmetic computations per iterative step. As a result of this improvement to the conventional projection algorithms, projection algorithms are being used more and more. For best results one should use a new projection algorithm with some *a priori* subspace selection algorithm.

We feel that projection algorithms are not just another competitive algorithm to choose from but one that will through further research become the dominant iterative algorithm for solving linear systems.

```
NEW2:  PROCEDURE(TOLER,PRT,STEPLT,N,A,B);
/* THIS ALGORITHM SOLVES A LINEAR SYSTEM USING THE NEW          */
/* TWO-DIMENSIONAL METHOD                                       */
/* AA      — MATRIX OF INNER PRODUCTS OF THE COLUMN             */
/*           VECTORS OF A.                                      */
/* NORMR— NORM SQUARED OF THE RESIDUAL VECTOR.                  */
/* QUIT    — AN INDICATOR IF THE TOLERANCE LIMIT HAS BEEN       */
/*           REACHED.                                           */
/*R       — RESIDUAL VECTOR                                    */
/* RA      — INNER PRODUCTS OF THE INITIAL RESIDUAL VECTOR      */
/*           WITH THE COLUMN VECTORS OF A.                      */
/* STEP    — STEP COUNTER                                       */
/* X       — THE APPROXIMATE SOLUTION VECTOR                    */
/* THE PARAMETERS TO THE ALGORITHM ARE:                         */
/* TOLER — TOLERANCE LIMIT FOR DETERMINING CONVERGENCE.         */
/*           THE ITERATIVE PROCESS TERMINATES WHEN EVERY         */
/*           CORRESPONDING ELEMENT OF THE APPROXIMATE           */
/*           SOLUTION VECTOR FROM TWO SUCCESSIVE CYCLES         */
/*           IS WITHIN THIS VALUE. RECOMMENDED VALUES           */
/*           RANGE FROM 0.00001 TO 0.0000001                    */
/* PRT     — THIS IS AN INDICATOR FOR THE AMOUNT OF OUTPUT      */
/*           DESIRED. A VALUE OF ZERO INDICATES MINIMUM         */
/*           OUTPUT. ONLY THE NUMBER OF STEPS REQUIRED AND      */
/*           THE SOLUTION ARE GIVEN. A VALUE NOT EQUAL TO       */
/*           ZERO WILL CAUSE THE FOLLOWING VARIABLES TO BE      */
/*           PRINTED: TOLER,STEPLT,X,B,R,C,A,AA,G,RA,T.         */
/*           IN ADDITION AFTER EACH ITERATIVE STEP X, STEP,     */
/*           NORMR AND AN INDICATOR OF WHAT COLUMNS OF A        */
/*           WERE USED ARE PRINTED.                             */
/* STEPLT— THE MAXIMUM NUMBER OF ITERATIVE STEPS TO BE          */
/*           ALLOWED                                            */
/* N       — THE DIMENSION OF THE LINEAR SYSTEM                 */
/* A       — THE COEFFICIENT MATRIX OF THE LINEAR SYSTEM        */
/* B       — THE CONSTANT VECTOR OF THE LINEAR SYSTEM           */
DECLARE (K,L,W,P,PRT,STEPLT,QUIT,I,J,N)
FIXED BINARY, ((X(*),C(*,*),
```

```
R(*),AA(*,*),G(*),T(*),RA(*)) CONTROLLED,TOLER,
XX,D1,D2,STEP,NORMR) FLOAT DECIMAL (16);
DECLARE (B(*),A(*,*)) FLOAT DECIMAL (16);
QUIT = 1;
P = -1;
STEP = 0;
ALLOCATE X(N),C(N + 1,N),R(N),AA(N,N),
RA(N),G(N + 1),T(N + 1);
X = 0;
/* CALCULATE R,RA,AA,T,G AND C                                    */
R = B;
DO I = 1 TO N; RA(I) = SUM(R*A(*,I));
    DO J = I TO N;
        AA(I,J) = SUM(A(*,I) * A(*,J));
        AA(J,I) = AA(I,J);
    END;
END;
IF PRT ≠ 0 THEN
DO; T(N + 1) = 0; G(N + 1) = 0; C(N + 1,*) = 0; END;
    /*              CALCULATIONS FOR T                             */
    DO I = 1 TO N BY 2;
    IF I = N THEN W = N - 1; ELSE W = I;
    T(I) = AA(W,W + 1) *AA(W,W + 1) - AA(W,W)*AA(W + 1,W + 1);
    T(I + 1) = T(I);
    END;
    /*              CALCULATIONS FOR G                             */
    DO I = 1 TO N BY 2;
    IF I = N THEN W = N - 1; ELSE W = I;
    G(I) = (RA(W + 1)*AA(W,W + 1) - RA(W)*AA(W + 1,W + 1))/T(I);
    G(I + 1) = (RA(W)*AA(W + 1,W) - RA(W + 1)*AA(W,W)/T(I);
    END;
    /*              CALCULATIONS FOR C                             */
    DO I = 1 TO N BY 2; IF I = N THEN W = N - 1; ELSE W = I;
    DO K = 1 TO N;
    C(I,K) = (AA(W,K)*AA(W + 1,W + 1)
            - AA(W + 1,K)*AA(W,W + 1))/T(I);
    C(I + 1,K) = (AA(W + 1,K)*AA(W,W)
                - AA(W,K)*AA(W + 1,W))/T(I);
    END;
    END;
IF PRT = 0 THEN GO TO LOOP;
PUT PAGE EDIT
('THE SUCCESSIVE APPROXIMATION VECTOR TOLERANCE = ',
TOLER,'THE STEP LIMIT = ',STEPLT) (A,F(15,10),A,F(5));
PUT SKIP(3) EDIT
('X','B','R',(X(I),B(I),R(I) DO I = 1 TO N))
(COL(10),A,COL(30),A,COL(50),A,
(N)(SKIP,F(15,6),COL(20),F(15,6),COL(40),F(15,6)));
PUT SKIP(2) EDIT('C MATRIX') (COL(50),A);
DO I = 1 TO N + 1;
    PUT SKIP EDIT((C(I,J) DO J = 1 TO N)) (R(LAB1));
END;
PUT SKIP(2) EDIT('A MATRIX')(COL(50),A);
DO I = 1 TO N;
    PUT SKIP EDIT((A(I,J) DO J = 1 TO N)) (R(LAB1));
END;
PUT SKIP(2) EDIT('AA MATRIX') (COL(50),A);
DO I = 1 TO N;
    PUT SKIP EDIT((AA(I,J) DO J = 1 TO N)) (R(LAB1));
END;
PUT SKIP(2) EDIT('G VECTOR',G) (R(LAB2));
PUT SKIP(2) EDIT('RA VECTOR',RA) (R(LAB2));
PUT SKIP(2) EDIT('T VECTOR',T) (R(LAB2));
NORMR = SUM(R*R);
PUT PAGE EDIT('STEP','X','NORM R','PAIR')
(A,COL(40),A,COL(119),A,COL(128),A);
PUT SKIP(2) EDIT(0,(X(I) DO I = 1 TO N))
(F(5),(N)(COL(6),(8)F(13,6)));
PUT EDIT(NORMR)(COL(113),F(15,6));
LAB1: FORMAT((N)(SKIP,(8)(F(15,6))));
LAB2: FORMAT(COL(50),A,SKIP,(N)(SKIP,(8)F(15,6)));
LOOP:
IF STEP > STEPLT THEN
    DO; PUT SKIP EDIT('STEP LIMIT WAS REACHED.')(A);
```

```
            GO TO EXIT;
        END;
    P = P + 2;
    IF P > N THEN
        DO; IF QUIT = 1 THEN
            DO; PUT SKIP EDIT('TOLERANCE LIMIT WAS REACHED')
                (A); GO TO EXIT;
            END;
            /* THE END OF A CYCLE HAS OCCURRED BUT          */
            /* MORE PROCESSING REMAINS                      */
        QUIT = 1; P = -1; GO TO LOOP;
        END;
    STEP = STEP + 1;
    IF P = N THEN W = P - 1; ELSE W = P;
    D1 = X(W); D2 = X(W + 1);
    X(W) = G(P); X(W + 1) = G(P + 1);
    DO K = 1 TO N;  IF K = W | K = W + 1 THEN GO TO BOT;
        XX = X(K);  IF XX = 0            THEN GO TO BOT;
        X(W) = X(W) + XX*C(P,K);
        X(W + 1) = X(W + 1) + XX*C(P + 1,K);
    BOT: END;
    IF QUIT = 1 THEN IF ABS(D1 - X(W)) > TOLER
                     THEN QUIT = 0;
                     ELSE IF ABS(D2 - X(W + 1)) > TOLER THEN QUIT = 0;
    IF PRT = 0 THEN GO TO LOOP;

    DO I = 1 TO N; R(I) = B(I) - SUM(A(I,*)*X); END;
    NORMR = SUM(R*R);
    PUT SKIP EDIT(STEP,(X(I) DO I = 1 TO N))
    (F(5),(N)(COL(6),(8)F(13,6)));
    PUT EDIT(NORMR,W) (COL(114),F(15,6),COL(130),F(3));
    GO TO LOOP;
    EXIT:
    PUT SKIP(2) EDIT('NUMBER OF STEPS = ',STEP,
    'THE SOLUTION FOLLOWS',(X(I) DO I = 1 TO N))
    (A,F(6),SKIP,COL(50),A,(N)(SKIP,(10)F(12,6)));
    END NEW2;
```

REFERENCES

1. A. de la Garza, An Iterative Method for Solving Systems of Linear Equations. Union Carbide and Carbon Corp. K-25 Plant, Oak Ridge, Tennessee, Report K-731 (1951).
2. A. S. Householder, *The Theory of Matrices in Numerical Analysis*. Blaisdell, New York (1964).
3. R. F. Keller, A Single-Step Gradient Method for Solving Systems of Linear Equations, Technical Report 8, Mathematical Sciences, University of Missouri, Columbia, Missouri (1964).
4. L. Fox, *An Introduction to Numerical Linear Algebra*. Oxford University Press, New York (1964).
5. R. F. Keller, H. D. Pyron and I-Ming Shen, On Determining an Optimal 2-Dimensional Projection Method, Ames Laboratory Report IS-2755, National Technical Information Service, Springfield, Virginia (1968).
6. H. D. Pyron, A Non-stationary Optimizing of Projection Methods. Unpublished Ph.D. Dissertation, Iowa State University, Ames, Iowa, Report IS-T-452, National Technical Information Service, Springfield, Virginia (1971).
7. D. Georg and R. F. Keller, Criteria for Ordering Columns in Two-Dimensional Projection Methods, Ames Laboratory Report IS-3147, National Technical Information Service, Springfield, Virginia (1973).
8. M. Tokko and R. F. Keller, Optimal Three-Dimensional Projection Method for Solving Linear Algebraic Equations, Ames Laboratory Report IS-3039, National Technical Information Service, Springfield, Virginia (1972).
9. R. L. Wainwright, Algorithms for Projection Methods for Solving Linear Systems of Equations, Unpublished Ph.D. Dissertation, Iowa State University, Ames, Iowa, Report IS-3397, National Technical Information Service, Springfield, Virginia (1974).