INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2001; **35**: 721–741

A convergence accelerator of a linear system of equations based upon the power method

A. Dagan*

MOD Scientific Department, Division 39, Haifa, Israel

SUMMARY

This paper considers the convergence rate of an iterative numerical scheme as a method for accelerating at the post-processor stage. The methodology adapted here is: (1) residual eigenmodes included in the origin of the convex hull are eliminated; (2) remaining residual terms are smoothed away by the main convergence algorithm. For this purpose, the polynomial matrix approach is employed for deriving the characteristic equation by two different methods. The first method is based on vector scaling and the second is based on the normal equations approach. The input for both methods is the solution difference between two consecutive iteration/cycle levels obtained from the main program. The singular value decomposition was employed for both methods due to the ill-conditioned structure of the matrices. The use of the explicit form of the Richardson extrapolation in the present work overrules the need to employ the Richardson iteration with a Leja ordering. The performance of these methods was compared with the GMRES algorithm for three representative problems: two-dimensional boundary value problem using the Laplace equation, three-dimensional multi-grid, potential solution over a sphere and the one-dimensional steady state Burger equation. In all three examples both methods have the same rate of convergence, or better, as that of the GMRES method in terms of computer operational count. However, in terms of storage requirements, the method based upon vector scaling has a significant advantage over the normal equations approach as well as the GMRES method, in which only one vector of the N grid-points is required. Copyright © 2001 John Wiley & Sons, Ltd.

KEY WORDS: convergence accelerator; power method; generalized minimal residual; Richardson extrapolation method; singular value decomposition

1. INTRODUCTION

The rapid development of very efficient methods for solving a linear system of equations is one of the main characteristics of recent advances in computational fluid dynamics (CFD). However, these convergence algorithms are an integral part of the numerical codes in which implementation of these methods require substantial numerical effort. In some cases, the nature of the numerical codes does not permit any kind of modification. Therefore, there is a

* Correspondence to: MOD Scientific Department, Division 39, PO Box 2250, Haifa 31021, Israel.

Received July 1998
Revised June 2000

clear need for a post-processor to accelerate the convergence of the numerical codes. This post-processor must be able to use the available numerical output, such as residuals, without interfering with the code execution process. Development of such methods is the main motivation for the present work.

Traditionally, the classical methods such as Jacobi, Gauss–Seidel and successive overrelaxation (SOR)/successive line overrelaxation (SLOR) have played an important role in the elliptic solution of the full potential transonic problem. However, these methods are characterized by a slow rate of convergence and require substantial computational time. The need for an efficient and reliable algorithm has motivated research into the class of numerical algorithms, known as the alternating direction implicit (ADI) methods. Among these methods, the modified version of the Peacman–Rachford algorithm (approximate factorization (AF1 and AF2)) has because an important tool in full potential equation solvers. The Peacman–Rachford algorithm works well for a rectangular domain in two-dimensional linear elliptic problems, where for the proper pseudo-time step it converges at $\min\{N, M\}$ iterations to the exact solution (N and M are the number of grid-points at each direction). However, for general three-dimensional problems, such a method does not eliminate the error entirely. The choice of the pseudo-time step for general three-dimensional problems is the main drawback of this method. The strongly implicit procedure (SIP) method is also worth mentioning. Despite the fact that the SIP method is very promising, it is rarely used in CFD.

A very powerful convergence algorithm that is widely used in CFD is the Multi-grid method. The Multi-grid concept is similar to the Peacman–Rachford algorithm, in which each wavelength is dampened at each multi-grid grid level. The slowest (long waves) modes are efficiently decayed in the coarse grid level, while the short waves are dampened on the fine meshes. Therefore, such an algorithm does not exhibit the same slow rate of convergence that is typical to the long waves, as in the traditional convergence algorithms mentioned above.

A different approach that has been proven to be extremely effective for solving large sets of linear equations is the conjugate gradient (CG) method. This class of method is sometimes regarded as a convergence accelerator technique, and produces extremely efficient convergence, particularly when coupled to an appropriate preconditioning method. Eigenvalue clustering obtained from a preconditioner, such as Choleski factorization (incomplete lower–upper (ILU)), is the key for the rapid convergence [1,2]. The CG method is known to be efficient in dealing with symmetric positive definite matrices. However, for non-symmetric matrices, several generalizations of the CG method have been proposed in the literature. The Lanczos-based methods, such as the CG stabilized (CGS) method [3] and the bi-CG stabilized (Bi-CGSTAB) method [4] are widely employed, while the Generalized Minimal RESidual (GMRES) [5] belongs to the class of Arnoldi-based methods. For non-symmetric matrices the GMRES proved to be very effective for solving large sets of linear equations. The GMRES method is based on the minimal residual. It forms an orthogonal basis (search direction) spanning the Krylov subspace by a modified Gram–Schmidt method. An easy implementation of the GMRES method is based upon slaving the convergence algorithm of the main program, such as the symmetric successive overrelaxation (SSOR) or the ILU algorithm, as a preconditioner to accelerate the convergence rate of the main algorithm [6]. Therefore, such an approach does not require substantial programming effort. However, the storage requirement of the GMRES, as well as that of CG, seems to be the main limitation of this method. An alternative implementation of the GMRES using householder transformation is given by Walker [7], since the Gram–Schmidt process is a potential source of numerical error.

A different approach that has been widely employed as a convergence accelerator is the power method (PM). In the PM, the extreme eigenvalues are estimated. Based on the eigenvalues estimation, either the Richardson extrapolation or the Chebyshev iterative procedure are then employed to obtain the asymptotic solution [8–10]. In a similar fashion, the extreme eigenvalues in the hybrid methods are estimated either by the PM or by the Arnoldi/GMRES procedure. The iterative solution is then repeatedly accelerated by the use of the Richardson extrapolation algorithm or by Chebyshev iterative procedure [10,11]. This matter is elaborated upon in the forthcoming sections.

Traditionally, much effort was devoted to solving the discretized form of the Navier–Stokes equations on structure meshes. However, due to the convenience of the unstructured grid generation, Navier–Stokes solutions on unstructured grids are gaining popularity. However, this methodology has not reached the same maturity level as the solutions on structured grids. The semi-implicit ADI or residual smoothing (Multi-grid) solutions are efficiently obtained along the structured gridlines. On unstructured meshes an explicit Jacobi method is often employed, which may limit the allowable Courant–Friedrich–Lewy (CFL) number. Both structured and unstructured solutions exhibit a slow converge rate when a viscous solution is under consideration. In such a case, a highly stretched grid is needed in order to resolve the boundary layer. Large cell aspect ratios in the boundary layer can reduce the rate of convergence. Therefore, in structured grids semi-coarsening the multi-grid plays an important role in increasing the convergence rate. However, in unstructured grid topology, cell aspect ratio seems to have a pronounced effect on the accuracy and the convergence rate of the results. Therefore, hybrid meshes are often used to overcome this problem.

Numerical codes (executable commercial and source files) do not always include the numerical tools needed to cope with slow convergence rates resulting from skewness and large cell aspect ratio. Upgrading of such source files requires a substantial numerical effort, which is not always justifiable and affordable. Some of the executable commercial codes do not even include a convergence algorithm, such as the Multi-grid technique. Moreover, the level of executable codes does not permit any kind of modification, as they are supplied as source files. In view of this, there exists a clear need for an algorithm (post-processor) to accelerate the convergence rate of the numerical codes. However, post-processors based on the GMRES method in a fashion similar to that of Wigton *et al.*'s [6] require large computer storage capacity. Moreover, the convergence algorithm of the numerical program needs to be slaved as a preconditioner. Both requirements constitute a major drawback for small workstations in which slaving the main algorithm of an executable code is not an easy task. Therefore, a post-processor that can use the available numerical output such as the residual or the solution difference, without interfering in the process of the numerical codes execution, is the main motivation of the present work.

2. THE ASYMPTOTIC BEHAVIOR OF THE NUMERICAL ERROR

The asymptotic convergence of the numerical scheme is dominated by the module of the largest eigenvalue. It should be recalled that the spectral radius of the numerical error has to be smaller than unity for convergence. The closeness of the spectral radius to unity plays an important role in the efficiency of the numerical scheme. Typically, the main characteristic of the iterative process is rapid convergence at the early stages of the solution, which slows down gradually to the asymptotic value of the numerical error. For non-linear problems it is not uncommon to observe an increase in the residual before it drops to its asymptotic value. The rapid convergence at the early stages of the numerical solution is essentially due to the initial error distribution, in which the dominant error terms (with respect to the initial error distribution and not necessarily corresponding to the largest eigenvalue) are efficiently dampened away until they reach their asymptotic stage.

The iterative solution of the numerical scheme can be viewed as an eigenvalue problem, where at each stage of the iterative process the residual $\Re^{(k)}$ can be written as follows:

$$\epsilon^{(\kappa)} + \Re^{(\kappa)} = 0 \tag{1}$$

where $\Re^{(\kappa)} = \Re(u^{(\kappa)})$ and $u^{(\kappa+1)} = u^{(\kappa)} + \epsilon^{(\kappa)}$ are given at the iteration level k. Generally, Equation (1) is given in its implicit form, such as $L\epsilon^{(\kappa)} + G^{(\kappa)} = 0$, where L is some difference operator and $G = L\Re^{\kappa}$. For simplicity, the form of Equation (1) is given here. The Taylor series expansion of Equation (1) at the iteration level $(\kappa + 1)$ yields

$$\epsilon^{(\kappa+1)} + B^{(\kappa)}\epsilon^{(\kappa)} + O(\epsilon^2) = \epsilon^{(\kappa+1)} + B\epsilon^{(\kappa)} + O(\delta\epsilon) = 0 \tag{2}$$

where $B^{\kappa} = B + O(\delta)$ and $B = I - A$. The flux Jacobian matrices $A = \partial_u \Re$ are evaluated at u, where u is the asymptotic solution ($\Re(u) = 0$). The terms ϵ and δ are defined as $\epsilon^{\kappa} = O(\epsilon)$, $u - u^{\kappa} = O(\delta)$. Therefore, for an initial guess close enough to u, the residual at each stage of the iterative process can be regarded as an eigenvalue problem (Equation (2)), where the residual decays asymptotically with respect to the spectral radius of B. Consequently, the analysis and the algorithms in this work refer to the solution of a linear system $Au - b = 0$. This fact can be exploited to construct an asymptotic solution to the iterative process (Equation (1)) in a fashion similar to the Manteuffel [8], Saylor [9] and Saad [12] algorithms. The solution of the eigenvalue problem (2) can be written as follows:

$$\epsilon^{(k)} = \Phi\Lambda^{\kappa}\Phi^{-1}\epsilon^{(0)}$$

or alternatively as

$$\epsilon_i^{(\kappa)} = \sum_n C_n \lambda_n^k \phi_{n,i} \tag{3}$$

where Λ and Φ are the eigenvalues and the eigenvectors of matrix B, while $\epsilon^{(0)}$ is the initial residual (see Equation (1)). Some efforts have already been devoted [13] to the estimation of these eigenvalues, but the results are unsatisfactory. Convergence difficulties may arise if the first two eigenvalues are close in magnitude. Major advances in the PM were initiated by the work of several authors, including Manteuffel [8], Saylor [9], Elman *et al.* [10]. In the Manteuffel [8] approach, the PM is employed to derive estimates of the extreme eigenvalues of A. These estimates are then used to defined an ellipse, and the Chebyshev iterative procedure is then carried out with the parameters corresponding to that ellipse. Saylor [9] has employed the singular value decomposition (SVD) with respect to the least square problem in order to get a reliable estimate of the extreme eigenvalues of A. Based on these eigenvalues, the Chebyshev iteration method was employed to accelerate the convergence of the iterative algorithm. Elman *et al.* [10] have used the Arnoldi process and the GMRES method in order to get better accuracy of the predicted eigenvalues, which are needed for the hybrid method.

For a linear system of equations, the hybrid scheme has been found to be extremely effective, when a few steps of GMRES are followed by a Richardson iteration based on polynomial constructed, either by eigenvalues obtained by the PM or by the GMRES/Arnoldi methods [10,11]. However, when a non-linear set of equations is under consideration, it seems to be that the eigenvalues need to be updated after some amount of iteration.

The order of accuracy of the eigenvalues might have some influence on the accuracy of the method. However, Nachtigal *et al.* [11] have noticed that in the case of a linear system of equations, eigenvalue estimates are sometimes more reliable than exact eigenvalues. On the other hand, accurate methods to estimate the eigenvalues, such as the Arnoldi/GMRES technique, have recently replaced the PM [10].

The Richardson extrapolation method is usually employed in order to get the corrected solution of the hybrid methods, i.e.

$$u^{(\kappa+1)} = u^{(\kappa)} - \frac{1}{1-\lambda_{\kappa+1}} r^{(\kappa)}$$

$$r^{(\kappa+1)} = \left\{ I - \frac{1}{1-\lambda_{\kappa+1}} A \right\} r^{(\kappa)} = -\frac{\lambda_{\kappa+1}}{1-\lambda_{\kappa+1}} \left\{ I - \frac{1}{\lambda_{\kappa+1}} B \right\} r^{(\kappa)} \tag{4}$$

where $\lambda_{\kappa+1}$ are the eigenvalues of the matrix B ($k = 0, 1, 2, \ldots, N-1$) and $r^{(\kappa)}$ denotes the Richardson extrapolation residual. For a well-conditioned matrix A, whose eigenvalues are clustered around unity, the above residual decays asymptotically as $[\max(1-\lambda_i)]^M$, for a given M eigenvalue terms. However, difficulty in convergence may be encountered if some of the eigenvalues have a negative real part of the origin of the convex hull is included [9]. To clarify this point, assume that M ($M < N$) dominant eigenvalues of B ($B = I - A$) are being used in the hybrid scheme. In such a case, the residuals decay as

$$q = P(\Lambda) q^{(0)} \tag{5}$$

where $q = \Phi^{-1} r$, while $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_N)$ is a diagonal matrix containing the eigenvalues of the matrix B in the diagonal. The residual polynomial ($P(\lambda)$) is given as

$$P(\lambda) = \prod_{j=0}^{M-1} \frac{\lambda_{j+1}}{1-\lambda_{j+1}} \left\{ I - \frac{1}{\lambda_{j+1}} \lambda \right\} \tag{6}$$

For a well-conditioned matrix A all the eigenvalues are clustered around unity; therefore, $\|P(\Lambda)\|_2 < 1$ (see Equation (6)). However, in the present method, the residual polynomial is constructed with the largest eigenvalue of the matrix B. Particularly, it becomes a severe problem when some of the eigenvalues of the matrix A are in the vicinity of the origin, i.e. $|1-\lambda_{j+1}|\rightarrow 0$, in which case Richardson extrapolation eliminates the M dominant terms but amplifies the remaining modes, or in other words $\|P(\Lambda)\|_2 > 1$. This usually occurs when the matrix A is not well conditioned. Consequently, hybrid methods might lead in such a case to divergence rather then convergence. Therefore, it seems that a combination between the Richardson extrapolation method and a smoothing operator, such as the iterative convergence algorithm of the solver, may be the best way to operate the hybrid method. In such a case, the residual behaves after L iterations of the convergence algorithm (2) as

$$q = \prod_{j=0}^{M-1} \frac{\lambda_{j+1}}{1-\lambda_{j+1}} \left\{ I - \frac{1}{\lambda_{j+1}} \Lambda \right\} \Lambda^L q^{(0)} \tag{7}$$

Usually the convergence algorithm of the main solver is well conditioned, except for some errant modes, resulting probably from high aspect ratio or bad skewness of the cells. Therefore, an effective residual polynomial eliminates or removes the M leading eigenvalues, while the remaining eigenvalues are supposed to be damped away by the available smoothing operator Λ^L. Such a procedure can be employed as a post-processor to accelerate the convergence of the numerical codes at the source level or executable files, without interfering with the process of the main solver execution. A similar concept was presented by Webster [14] in connection with slow modes resulting from low-order prolongation operators in the algebraic multi-grid process.

In the present work, the Richardson extrapolation in derived in a different form. The asymptotic correction of Equations (2) and (3) can be written as

$$u = u^{(0)} + \sum_{k=0}^{\infty} \epsilon^{(\kappa)} = u^{(0)} + A^{-1}\epsilon^{(0)} \tag{8}$$

$$A^{-1} = \Phi(I-\Lambda)^{-1}\Phi^{-1}$$

Upon substitution of Equation (3) into Equation (8), the following can be obtained:

$$u = u^{(0)} + E\Gamma^{-1}d \tag{9}$$

where $E = [\epsilon^{(0)}, \epsilon^{(1)}, \epsilon^{(2)}, \dots, \epsilon^{(M-1)}]$, the square Vandermonde matrix $(M \times M)$ $\Gamma = \{\lambda_i^{j-1}\}$ and the vector $d = [1/(1-\lambda_1), 1/(1-\lambda_2), 1/(1-\lambda_3), \dots, 1/(1-\lambda_M)]^T$. Equation (9) is the explicit form of the Richardson extrapolation (2) in terms of the Krylov subspace span by the matrix B. Note that the coefficients of the explicit form of Richardson extrapolation are given by $\Gamma^{-1}d$. Therefore, the SVD should be employed in order to avoid numerical error due to the ill-conditioned nature of the Vandermonde matrix Γ. Moreover, it seems that the ill-conditioned nature of Equation (9) is an indication of a possible inaccuracy problem in Equation (4). In other words, using Richardson extrapolation (Equation (4)), without taking care of the ill-conditioned nature of $\Gamma^{-1}d$ may produce unsatisfactory results. Therefore, in the present work Equation (9) was adopted for extrapolation. It would be worth mentioning that the same point was addressed by Nachtigal *et al.* [11], in which the weighted Leja ordering [15] was incorporated together with the Richardson method for the purpose of the extrapolation.

In practice, it would be rather difficult to find all the eigenvalues of Equation (3). Therefore, it seems that the key for accurate eigenvalue prediction is an iterative scheme that reduces efficiently most of the remaining modes for a given number of iterations. The adaptive Chebyshev method is probably a preferred way to handle this problem. However, it was not implemented here mainly due to the fact that it is highly sensitive to its parameter, and the eigenvalue estimates are sometimes more reliable than the exact eigenvalues [11]. In the present work we prefer to use the terminology 'smoother' for such an iterative convergence algorithm of the main solver.

To conclude this section, the following guidelines summarize the main ideas presented here. Note that the residual at each iteration stage is given by $\epsilon^{\kappa} = u^{\kappa+1} - u^{\kappa}$; therefore, the convergence of the numerical algorithm can be accelerated as follows:

1. Collect the residual $\epsilon^{\kappa} = u^{\kappa+1} - u^{\kappa}$.
2. Compute the M leading eigenvalues from the residual.
3. Use Richardson extrapolation (9) with the computed eigenvalues to predict a better restart solution.
4. Run the numerical code for L iterations to smooth out the results.
5. If $\|\epsilon\| >$ tolerance go to step 1.

The above procedure resembles in some respects the 'Unmodified' PM of Elman *et al.* [10]. However, the residual here is based upon the iterative scheme $\epsilon^{\kappa} = -B^{\kappa}\epsilon^{(0)}$ rather then on $r^{(\kappa)} = A^{\kappa}r^{(0)}$. The resultant smoothing is also another difference between the present method and that of Elman *et al.* [10]. In the present approach, *result smoothing is a must*, particularly when the matrix A is not well conditioned.

3. METHOD OF SOLUTION

In the previous section we examined and discussed the concept of the present research. Briefly, the main idea here is to eliminate the M leading terms and to smooth the remaining terms of the eigenvalues by a suitable smoother algorithm, such as the main convergence algorithm of the numerical code. The main concern is to adopt an efficient approach to compute the leading eigenvalues without interfering with the main numerical code execution. Therefore, methods based upon slaving the main numerical algorithm, such as Arnoldi/GMRES, are excluded.

In the following sections we will examine three methods of solution: the normal equations method, the SVD for the least squares problems (SVDLS) and the vector scaling method.

3.1. The normal equations

The leading terms of the eigenvalues can be determined by considering M terms of Equation (3). Therefore, the accuracy of the eigenvalues prediction is controlled by the remainder terms in Equation (3). One way to eliminate these terms is by application of a suitable smoother algorithm as already mentioned. However, the search for such an algorithm is beyond the scope of the present work. Moreover, since we are concerned with implementing this method as a post-processor to a main solver, controlling the remainder terms is not feasible in this way.

In a fashion similar to Manteuffel [8], Saylor [9] and Elman *et al.* [10], it seems that a more efficient way to compute the leading eigenvalues terms and to minimize the effect of the remainder terms can be devised by minimizing the norm of the residual given by Equation (6). However, the above method can also be viewed in a different manner by considering the residual polynomial equation $P_N^{(1)}(\lambda) = P_M(\lambda)Q_{N-M}(\lambda) = 0$, where P and Q are polynomials of degree M and $N - M$ respectively, while $\mathrm{P}_N^{(1)}$ is the characteristic polynomial of the matrix B, having degree N (N is the dimension of matrix B). In matrix representation it turns out that $P_M(B) = R_M(B)$, where $R_M(B)$ is a matrix whose first M eigenvalues terms are zero. Note that R_M is the remainder matrix, while $r \propto R_M(B)\epsilon^{(0)}$ is the Richardson extrapolation residual. In a fashion similar to Manteuffel [8], Saylor [9], Elman *et al.* [10] and Saad [16], the problem is then to find a set of coefficients λ_n (see Equation (6)) that makes the L_2-norm of the remainder terms $R_M(B)\epsilon^{(0)}$ as small as possible.

The L_2-norm of the remainder terms can be written as

$$\|R_M(B)\epsilon^{(0)}\|^2 = [\epsilon^{(0)}]^T[P_M(B)]^T P_M(B)\epsilon^{(0)}$$

Since $P_M(B) = a_0 + a_1 B + a_2 B^2 + \cdots + B^M$, the minimization of the remainder terms yields

$$\frac{\partial}{\partial a_j}\|R_M(B)\epsilon^{(0)}\|^2 = 2[\epsilon^{(0)}]^T[B^T]^j P_M(B)\epsilon^{(0)} = 0$$

Since $\epsilon^{(\kappa)} = B^\kappa \epsilon^{(0)}$ the last expression can be further simplified to the following form:

$$\sum_{m=0}^{M-1}[\epsilon^{(j)}]^T\varepsilon^{(m)}a_m = -[\epsilon^{(j)}]^T\epsilon^{(M)} \quad \text{for } j = 0, 1, 2, \ldots, M-1 \tag{10}$$

Equation (10) is sometimes employed in the least square approximation problem. It is also known as the normal equation and it is very useful for computation when M is small. When matrix B is symmetric, Equation (10) can be reduced to the Hankel matrix form, with entries $f_{i+j-1} = [\epsilon^{(0)}]^T\epsilon^{(i+j-2)}$. The solution of Equation (10) has two disadvantages. First, multiplication by both $[\epsilon^{(j)}]^T$ and $\epsilon^{(m)}$ are required at each step. Secondly, the condition number of $[\epsilon^{(j)}]^T\epsilon^{(m)}$ is the square of $\epsilon^{(m)}$. Nevertheless these types of the normal equation algorithms are sometimes employed in the CG method and are also known as Craig's method [1]. In spite of these drawbacks, the normal equations in the approach can exhibit the same convergence performances as the GMRES method, and sometimes even better as we shall see in the results section. Bearing in mind that some of the accuracy will be lost due to the less favorable condition number, the residual polynomial coefficients a_m can be obtained by solving Equation (10). However, since Equation (10) is also an ill-conditioned system, the SVD method was employed for this case.

3.2. The singular value decomposition for the least squares problem

Some accuracy may be lost due to the large condition number of the normal equation. Therefore, a better approach is to consider, in a very similar way to Saylor [9], the L_2 norm of the residual in the following form:

$$\min\|R_M\epsilon^{(0)}\|_2^2 = \min\|P_M\epsilon^{(0)}\|_2^2 = \min\|\epsilon^{(M)} + Ea\|_2^2 = \min\|WV^T(g_M + a)\|_2^2$$

where $E = [\epsilon^{(0)}, \epsilon^{(1)}, \epsilon^{(2)}, \ldots, \epsilon^{(M-1)}] = UWV^T$ and $\epsilon^{(M)} = UWV^Tg_M$. Here the $N \times M$ matrix E, whose number of rows N is greater then the number columns M, is written as a product of a $N \times M$ column-orthogonal matrix U, an $M \times M$ diagonal matrix W and an $M \times M$ orthogonal matrix V. For further details see Appendix A. For such a representation, the minimization is given by $a = -g_M = -VW^{-1}U^T\epsilon^{(M)}$ (see Appendix A). Note that the norm of the a_n is not enforced to unity as it is in Saylor [9]. We shall examine the efficiency of this procedure in the forthcoming sections.

3.3. The vector scaling

Both the normal equation approach and the SVDLS algorithm are standard procedures for finding the residual polynomial coefficients. However, the storage requirements for both methods are a heavy burden on the computational facility, in which $O(M)$ vectors, with size $O(N)$, are to be stored. In this section we will adapt a different approach, in which only one vector scaling of size $O(N)$ in needed.

The main idea is to look for a vector scaling, say ω_0, in which $\omega_0^TR_M\epsilon^{(0)} = 0$. An obvious choice for ω_0 is a linear combination of the M eigenvectors inverse. Generally, T^{-1} has no unique representation, however, among all the possible ways to construct T^{-1}, the SVD ensures the smallest error. Therefore, the eigenvector matrix and its inverse can be obtained from Equation (3) as follows:

$$T = E\Gamma_M^{-1} \quad \text{and} \quad T^{-1} = \Gamma_MVW^{-1}U^T = \Gamma_M(E^TE)^{-1}E^T$$

where $T = \{\phi_1, \phi_2, \phi_3, \ldots, \phi_M\}$ is the eigenvector matrix, while $E = UVW^T$ is the residual matrix. Note that the M eigenvalues require M eigenvectors, and for that reason E was chosen as $E = \{\epsilon^{(1)}, \epsilon^{(2)}, \ldots, \epsilon^{(M)}\}$. The matrix Γ_M is the square Vandermonde matrix form of the M eigenvalues defined by $\{\lambda_j^i\}_{j=1,M,i=1,M}$. Any vector defined by the linear combination of the rows of T^{-1} can be written as

$$v = x_0T^{-1} = x_0\Gamma_M(E^TE)^{-1}E^T = \alpha E^T \tag{11}$$

where

$$\alpha = x_0\Gamma_M(E^TE)^{-1}$$

where x_0 is an arbitrary vector and $\alpha = \{\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(M)}\}$. Therefore, v can be written as $v = \{\alpha^{(1)}\epsilon^{(1)} + \alpha^{(2)}\epsilon^{(2)} + \cdots + \alpha^{(M)}\epsilon^{(M)}\}$. The dot product between the vector v^T and R_M(B)$\epsilon^{(0)}$ yields the orthogonality condition

$$|v^TR_M(B)\epsilon^{(0)}| = |\{\alpha^{(1)}\epsilon^{(1)} + \alpha^{(2)}\epsilon^{(2)} + \cdots + \alpha^{(M)}\epsilon^{(M)}\}^TP_M(B)\epsilon^{(0)}| \tag{12}$$

It turns out that the solution of Equation (10) imposes orthogonality for any vector α (Equation (12)). Since the vector scaling v is defined as a linear combination for any given vector α, it follows that $v = \epsilon^{(0)}$ is a particular solution for such a case. Moreover, $\epsilon^{(0)}$ is the most available vector that contains the relevant information regarding the dominant eigenvalue distribution. Therefore, the scaling vector $\omega_0 = \epsilon^{(0)}$ and the reminder term $R_M\epsilon^{(0)}$ yields

$$\sum_{n=1}^{M} a_n f_{n+m-2} + f_{M+m-1} = 0, \quad 1 \leq m \leq M \tag{13}$$

The term f_l is defined as $f_l = \omega_0^T \epsilon^{(l)} = [\epsilon^{(0)}]^T \epsilon^{(l)}$. In the case of a symmetric matrix, Equation (13) will yield the normal equations. The coefficients of the residual polynomial are given by the solution of the linear equations (13). Unfortunately, the Hankel matrix $\{f_{n+m-2}\}$ is ill conditioned. Therefore, the SVD method ([17]; Appendix A) is employed again to solve the linear system (13).

The form of Equations (9) and (13) has some advantages over the normal equation as well as the SVDLS method in terms of storage requirements. Here, the vector scaling approach requires only one vector $O(N)$. The performance assessment of the three methods with respect to the GMRES algorithm as reference method will be discussed in the following section.

4. THE EIGENVALUE COMPUTATION

The characteristics equations are generally not recommended for evaluating the eigenvalues, mainly due to the fact that for large M the accumulated round-off error can affect the accuracy of the results. In fact, for large M ($M \geq 20$) the solution of the type of polynomial equation became sparse with one exception: computer algebra, where the need for an efficient method for solving the polynomial equation has fuelled research on the effective algorithms in this area [18]. The Newton–Raphson method is known to be a reliable and efficient method for solving the polynomial equation for $M \geq 20$. For larger N, the QR algorithm applied on the associated Frobenius matrix may successfully compete with the Newton method for $M \leq 50$. In the present work, the QR algorithm (see ref. 13) was employed on the associated Frobenius matrix, i.e.

$$\begin{Bmatrix} 0 & 0 & \cdots & 0 & 0 & -a_0 \\ 1 & 0 & \cdots & 0 & 0 & -a_1 \\ & \ddots & & & & \\ & & \ddots & & & \\ 0 & 0 & \cdots & 1 & 0 & -a_{M-2} \\ 0 & 0 & \cdots & 0 & 1 & -a_{M-1} \end{Bmatrix}$$

It turned out that some of the M computed eigenvalues are spurious. The presence of the spurious eigenvalues is probably due to the effect of the remainder terms as well as numerical inaccuracy resulted from the SVD solution. However, it transpires numerically that those terms carry a low energy spectrum (x_0). In order to eliminate them from the given M set of eigenvalues, the following equation was solved:

$$x_0 = \{\epsilon^{(0)}\}^T E \Gamma_M^{-1} \tag{14}$$

The obtained x_0 was found numerically to be the best means to identify those spurious eigenvalues since their amplitude ($x_0^{(n)}$) is small. In general, the set of the M eigenvalues terms was truncated whenever $|x_0^{(n)}|/\|x_0\|_\infty \leq 10^{-9}$ to a set of M_t eigenvalues ($M_t \leq M$).

To reduce the computer storage in the vector scaling approach, a new set of M_t residuals $\{\epsilon^{(2M)}, \epsilon^{(2M+1)}, \ldots, \epsilon^{(2M+M_t-1)}\}$ where computed. The Richardson extrapolation method (9) was employed to find the asymptotic solution. In the normal equation and the SVDLS approaches, the latest M_t residual vectors were employed in the Richardson extrapolation (9).

In the case of vector scaling the smoother iteration count L (see Equation (7)) is determined by the $2M$ iterations needed to defined the Hankel matrix, plus another M_t iterations to generate the residual vectors for the Richardson extrapolation, thus $L = 2 * M + M_t$. For the SVDLS and the normal equation cases, $L = M - M_t$. Numerically, L was found to be large enough to ensure stability.

5. RESULTS AND CONCLUSION

In order to assess the efficiency of the PM, the Gauss–Seidel and V-cycle multi-grid algorithms of the Laplace equation were chosen as a smoother/preconditioner. The one-dimensional, steady state Burgers equation was taken as a non-linear model equation to examine the performances of the PM algorithms. The GMRES method was chosen as a reference algorithm, in which the performances of the normal equation and the vector scaling method are compared.

In order to check the accuracy of the predicted eigenvalues, the Dirichlet boundary value problem in a square computational domain was chosen as the first test case. For the Gauss–Seidel algorithm, the eigenvalues are given as

$$\lambda_{n,m} = \left[\frac{1}{1+\beta^2}\left\{\cos\left(\frac{n\pi}{(N-1)}\right) + \beta^2 \cos\left(\frac{m\pi}{(M-1)}\right)\right\}\right]^2 \tag{15}$$

Table I ($\beta = 1$) and Table II ($\beta = 0.5$) compare the eigenvalues of the normal equation and the vector scaling methods with respect to the GMRES and the analytic eigenvalues expression (Equation (15)). The computed eigenvalues using the normal equation method have better accuracy then the GMRES and the vector scaling (see Tables I and II). For larger matrices ($M \geq 20$), the accuracy of the PM declines, while the accuracy of the GMRES method improves. However, due to a storage limitation and computer operation counts, it would be impractical to employ large matrices. Therefore, it seems that the vector scaling method has some advantages over the GMRES method.

Table I. Comparison between the eigenvalues predicted by the GMRES and the PM with the analytical solution of the Laplace equation in a square domain ($100 \times 100 \times 1$, $\beta = 1.0$, monitored = 10).

GMRES (20)	NE (20)	VS (20)	Analytical
0.989973	0.989980	0.990130	0.98997886
0.950905	0.950939	0.950935	0.95086921
0.904828			0.90407868
	0.913828		0.91323028
0.875271	0.877127	0.880511	0.87715263
	0.799664		0.79669741
	0.778659	0.786799	0.77699475
0.764670	0.744310		0.74593230

NE, normal equation; VS, vector scaling.

Table II. Comparison between the eigenvalues predicted by the GMRES and the PM with the analytical solution of the Laplace equation in a square domain ($100 \times 100 \times 1$, $\beta = 0.5$, monitored = 50).

GMRES (20)	NE (20)	VS (20)	Analytical
0.9508885	0.95088321	0.95088870	0.95088853
0.9041679	0.90416715	0.89826848	0.90418134
0.8594388	0.85977071	0.83136473	0.85974657
	0.76009554		0.75158961
	0.74382409		0.73919547
0.7199936		0.72668516	0.72675742
	0.69209352		0.69096574
0.6341560		0.63754644	0.63518688
	0.57431210	0.54016109	0.57438259
0.4868725	0.48482647	0.43784185	0.48520479

NE, normal equation; VS, vector scaling.

Table III summarizes the computing time performances of the SVDLS compared with the vector scaling and normal equations for the case of the Laplace equation on a rectangular domain. Gauss–Seidel was chosen here as the smoother algorithm. The SVDLS is the worst case in terms of computing time (ALPHA500, 400 MHz), among the three cases of the PM under consideration. Therefore, in the forthcoming sections, only the normal equations and the vector scaling are considered for the purpose of method evaluation.

Since we are dealing with different methods which require varying amounts of computational work at each time step, we believe that the CPU time is the only true measure for comparing them [19]. As already mentioned, the convergence algorithms can exhibit rapid convergence at the early stage of the iterative process with subsequent gradual decay to the asymptotic level of the convergence scheme. The slow rate of convergence can result from cell aspect ratio, skewness and also from numerical problems with the convergence algorithm.

Table III. Comparison between the performance of the SVDLS to the normal equations and the vector scaling for a residual ratio of 10^{-14} (in seconds; ALPHA500, 400 MHz).

Grid size	M	SVDLS	NE	VS
100×100	10	4.5	2.8	2.8
	20	6.55	2.2	2.47
	30	10.1	3.2	2.52
200×200	10	80.0	39.0	36.0
	20	84.0	39.0	34.0
	30	191.0	59.0	36.0

Since the slow rate of convergence is typically due to some error mode distribution, the uniform initial guess as well as the long range of convergence history will eventually reveal such a mode if it exists. In view of this, both the PM algorithm and the GMRES method were compiled on a 64-bit machine (ALPHA500, 400 MHz) using double-precision. The initial solution for both algorithms was given as a uniform distribution of $O(10^{28})$. In some cases the average convergence error was reduced to $O(10^{-18})$.

The convergence histories for the PM and GMRES methods are shown in Figures 1 and 2 for the Dirichlet boundary value problem of the Laplace equation in a square domain. The Gauss–Seidel algorithm is not considered to be an efficient preconditioner (see the distribution of the eigenvalues in Equation (14)). Therefore, it was chosen as a test case when the matrix A is not well conditioned. The Gauss–Seidel method was chosen as the PM smoother and the GMRES preconditioner for this case. The solution was monitored every five (Figure 1) and ten iterations (Figure 2) in which the PM asymptotic solution ($\mu^{(0)}$) was predicted at each $M = 14$. For the GMRES method, the asymptotic solution was predicted at ten search directions, which were found to be optimal for this case. In each figure, only the predicted values are illustrated. The results indicate that the PM methods have the same or better rate of convergence in terms of computer operations as the GMRES method. Both vector scaling and the normal equation have almost the same rate of convergence in terms of computer operational counts (Figures 1 and 2). However, in terms of computer storage requirements, vector scaling has some advantages over the normal equation as well as the GMRES method. Only one vector needs to be stored, while the normal equation as well as the GMRES method require additional memory storage of $O(MN)$ for the orthogonalization process, and for the search directions.

The next example illustrates the efficiency of the PM algorithms compared with the GMRES method for three-dimensional potential flow over a sphere. The grid was clustered near the sphere's wall and near the sphere's poles. Exponential clustering was used in the radial and in the tangential direction. An evenly spaced grid was implemented in the circumferential direction. The $49 \times 21 \times 49$ grid-points in the radial, tangential and circumferential directions respectively were employed. As for a test case of the effectiveness of the preconditioned algorithms, the five level V-cycle multi-grid was implemented as a preconditioner/smoother in which both the SLOR and a residual smoothing algorithms were employed as the multi-grid smoother. About 10 work units of the multi-grid smoother were required at each cycle. The results are illustrated in Figure 3 for the double-precision algorithm. In order to remove any

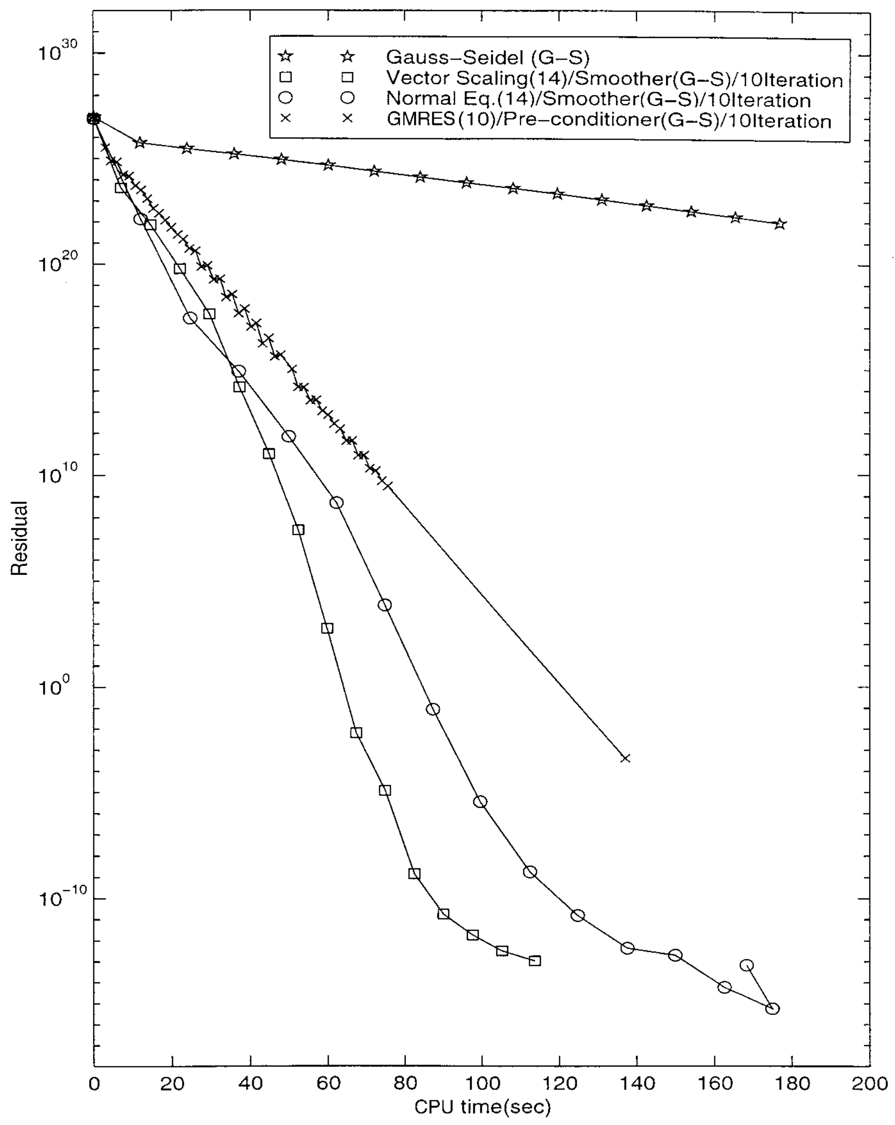

Figure 1. Comparison between the GMRES method and the normal with Gauss–Seidel as a preconditioner/smoother. Solution is monitored after five iterations, 200 × 200.

doubt about the sensitivity of the results to the machine accuracy (mainly because we are dealing with an ill-conditioned systems), the same computation (Figure 4) was carried out using a quadratic precision on the same machine (-r16 compiler). For the first test case, the multi-grid smoother was chosen as the PM smoother and the GMRES preconditioner. This

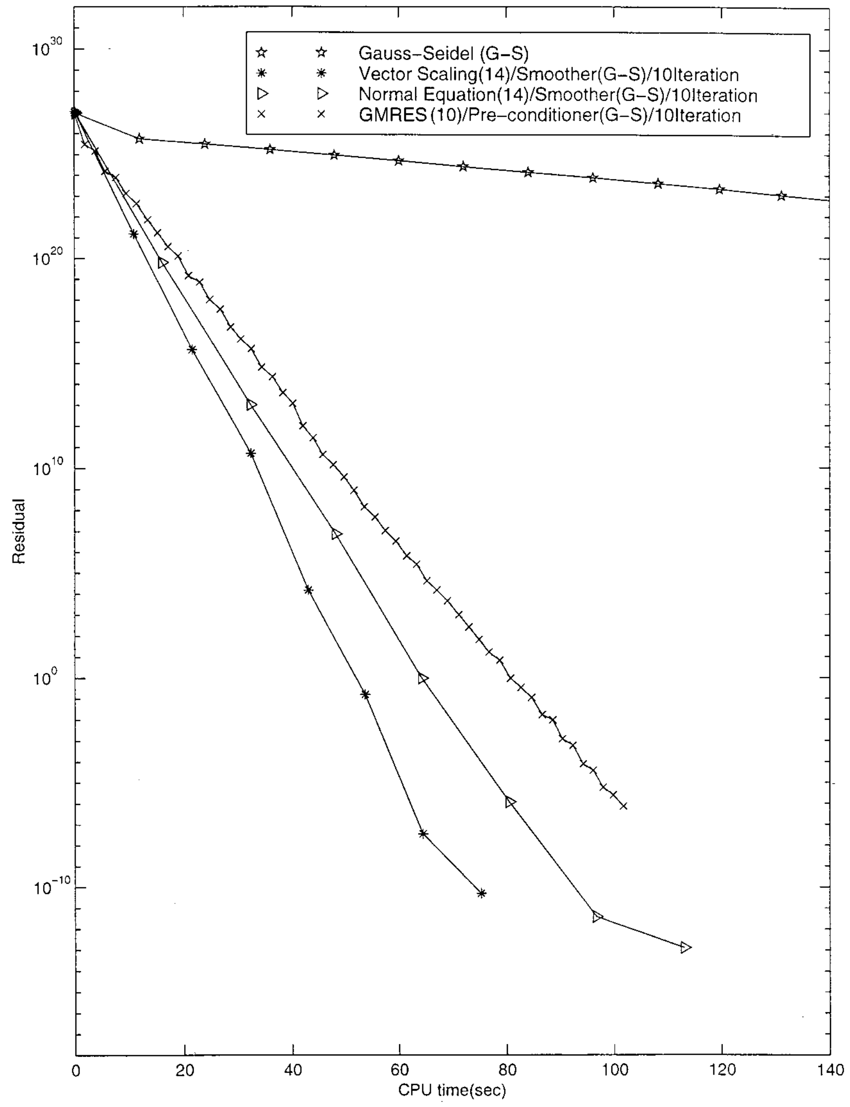

Figure 2. Comparison between the GMRES method and normal equation with Gauss–Seidel as a preconditioner/smoother. Solution is monitored after ten iterations, 200 × 200.

was done by turning off the multi-grid level and executing the Multi-grid algorithm on the fine grid only. In this case, the vector scaling method works as well as the GMRES method. However, neither the PM algorithm nor the GMRES method had the same efficiency as the Multi-grid technique. We need to bear in mind that the PM and GMRES are essentially

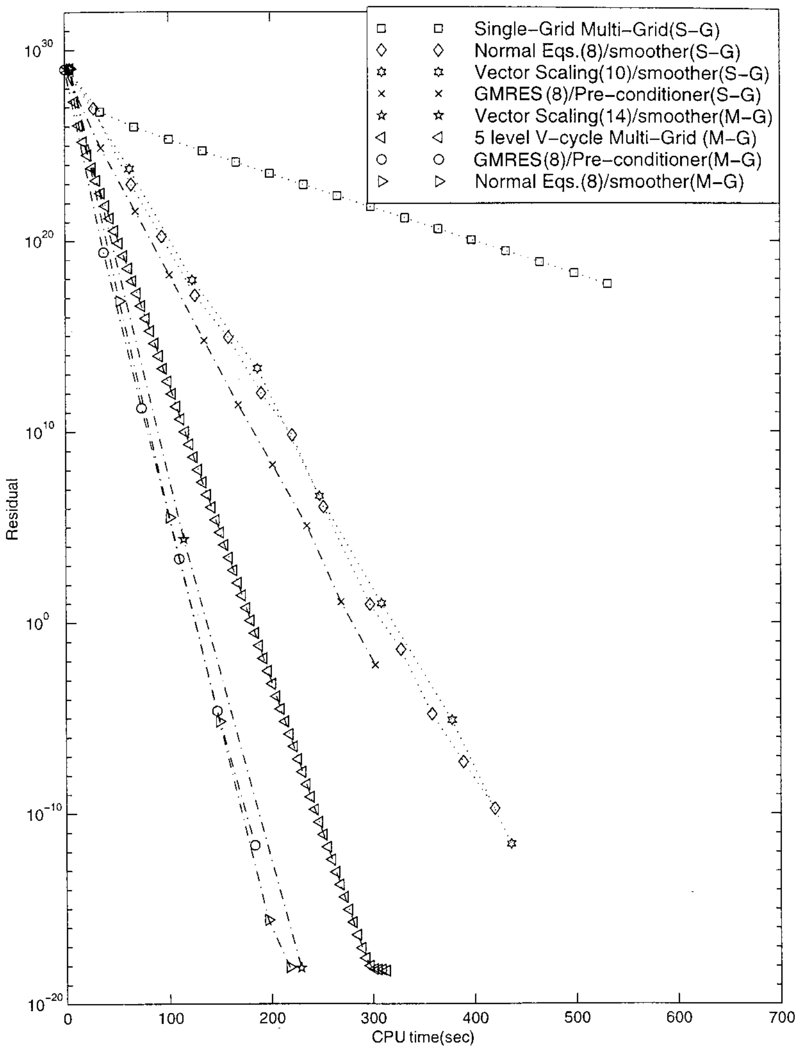

Figure 3. Comparison between the GMRES method and the normal equation for iterations with a five-level, V-cycle multi-grid as a preconditioner/smoother, 49 × 21 × 49.

convergence accelerator algorithms and their best performance is highly dependent on their smoother/preconditioner. Therefore, as a test case for a well-conditioned matrix A, the five-level, V-cycle multi-grid was employed as a preconditioner/smoother for both the PM algorithms and the GMRES method. The rate of convergence of the PM is essentially the same as that of the GMRES method. Therefore, it appears that vector scaling has some advantage over the normal equation as was explained before.

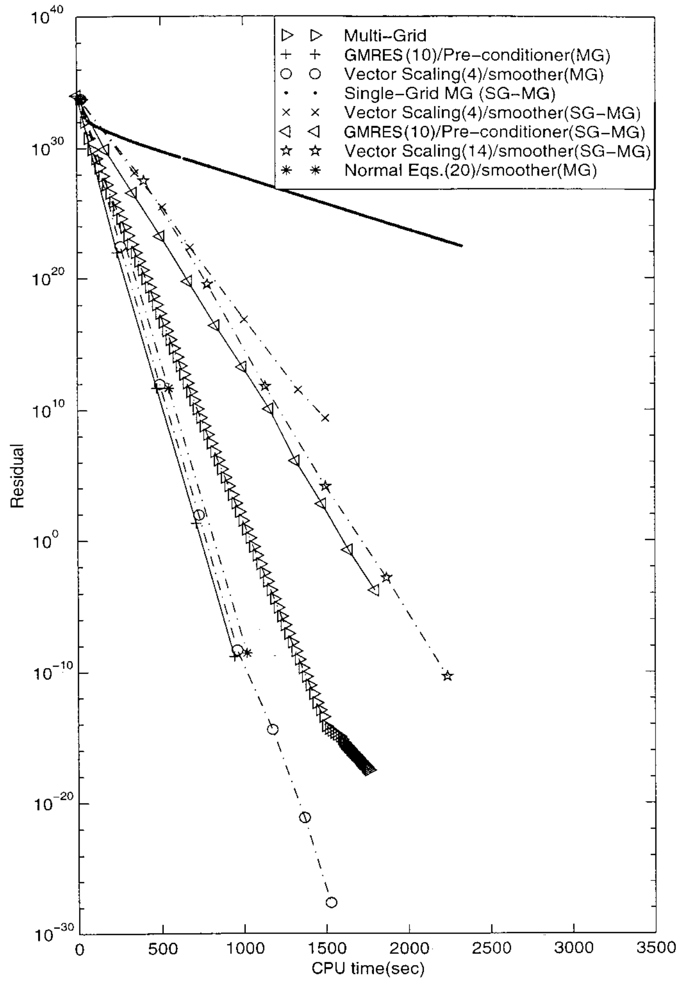

Figure 4. Comparison between the GMRES method and the normal equation with a five-level, V-cycle multi-grid as a preconditioner/smoother, $49 \times 21 \times 49$ (quadratic precision).

The complete non-linear Burger equation is a parabolic partial differential equation (PDE), which can serve as a model equation for the boundary layer equations for the 'parabolized Navier–Stokes' equations and for the complete Navier–Stokes equations [20]. Therefore, the last example chosen in the present work is the steady state, one-dimensional, non-linear Burger equation, written below

$$u\frac{\partial u}{\partial x}=\frac{1}{Re}\frac{\partial^2 u}{\partial x^2}; \quad u(0)=u_0, \quad u(L)=0$$

An upwinding first-order accurate scheme was used for the advection part, while the standard stencil of second-order central difference was employed for the diffusion term. It is known that the stencil of the first-order upwind scheme of the advection term contributes a numerical diffusion term. However, for the number of grid-points employed in this case ($N = 1000$), its effect on the solution is negligible. The Gauss–Seidel algorithm was employed on the linearized equation. In Figures 5 and 6, the rates of convergence of the Burger solution are illustrated in comparison with the PM algorithms and the GMRES method. The solution was monitored after 100 Newton iterations (Figure 5) and 500 iterations (Figure 6). The normal equation has a better convergence rate in terms of computer operations compared with

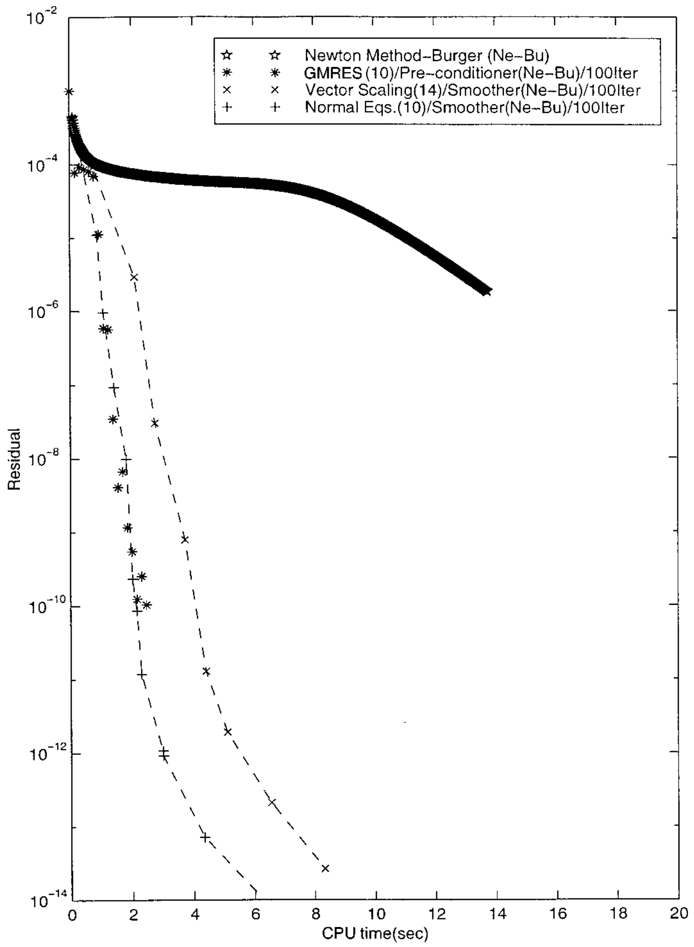

Figure 5. Comparison between the GNRES and the normal equations for the Burger equation (solution is monitored after 100 iterations, number of grid-points = 1000).

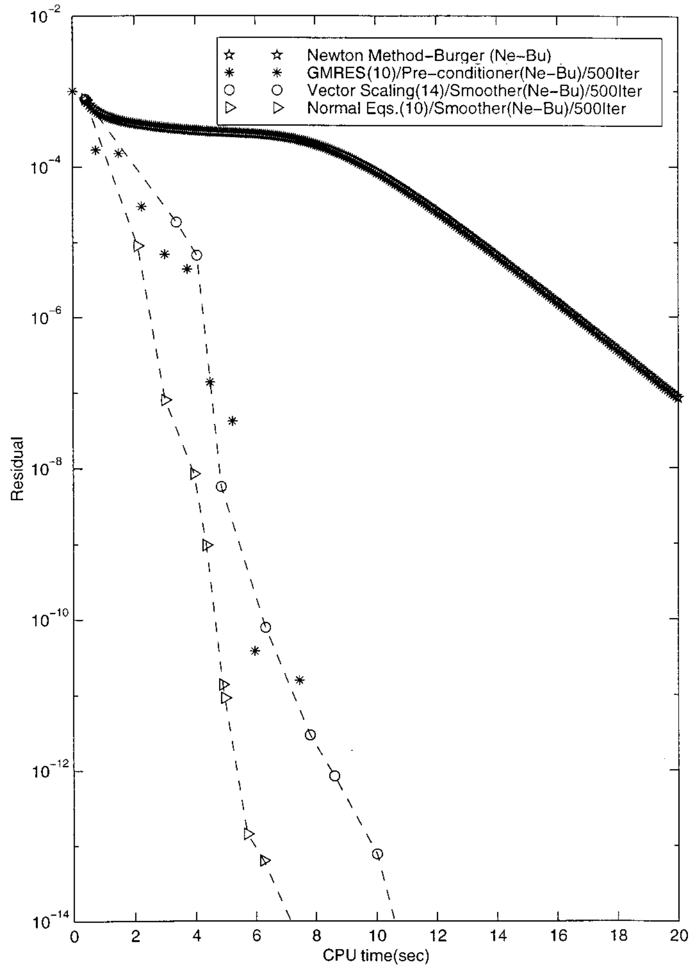

Figure 6. Comparison between the GMRES and the normal equations for the Burger equation (solution is monitored after 500 iterations, number of grid-points = 1000).

the vector scaling algorithm and with the GMRES method, as can be verified from Figures 5 and 6. However, in both cases the vector scaling has almost the same rate of convergence as the normal equation as well as the GMRES method. It is worth mentioning that the rate of convergence of both the PM algorithms and the GMRES method are very impressive compared with the rate of convergence of the Newton smoother/preconditioner. However, in terms of computer storage requirements, the vector scaling algorithm has some advantage over the normal equation as well as the GMRES methods.

In view of the present numerical results, it seems that the PM can be employed as a post-processor to accelerate convergence of executable numerical codes. Although the GMRES method is well proven to be an efficient means as a convergence accelerator, the concept of slaving the main convergence algorithm as a preconditioner limits the applicability of such a method as a post-processor. On the other hand, the PM does not slave the main algorithm and uses the available numerical output without interfering with the process of the numerical code execution. Therefore, the PM offers an option to generate the restart asymptotic solution. Such a post-processor uses the available numerical residual and runs parallel to the main solver. However, as was shown, the normal equation requires large computer storage. On the other hand, vector scaling has almost the same efficiency as the normal equation algorithm and requires only one vector of N grid points for computer storage. Therefore, vector scaling seems to be well suited as a convergence accelerator.

APPENDIX A. THE SINGULAR VALUE DECOMPOSITION TECHNIQUE FOR SOLVING THE LINEAR SYSTEM

The SVD method is the most powerful method to deal with ill-conditioned matrices. Any matrix E of dimension $N \times M$ can be written as $E = UWV^T$, where U and V are orthonormal matrices while $W = \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3, \ldots, \sigma_M)$ is a diagonal matrix with non-negative entries. In the particular case when E is a symmetric positive definite matrix, $U = V = Q$, where Q is the set of the eigenvectors of E. For symmetric, non-positive definite matrix, $V = \mathrm{sign}(W)Q$. The inverse of the matrix E is given by

$$E^{-1} = VW^{-1}U^T, \qquad W^{-1} = \begin{cases} \sigma^{-1} & \dfrac{1}{\sigma_{\max}} \sigma > \delta \\ 0 & \text{otherwise} \end{cases} \tag{A.1}$$

The SVD method is based on the minimum of error [17]. In the present work, δ for double precision calculation was chosen to be $\tau = 10^{-12}$.

A better approach to deal with ill-conditioning matrices is the Tichonov regularization method [21]. The minimization of the residual and the error yields

$$\min(\|x\|_2^2 + \tau \|Ex - f^{(1)}\|_2^2) = (\alpha^2 + E^T E)x = E^T f^{(1)}$$

where τ is the Lagrange multiplier, while $\alpha^2 = 1/\tau$ is the Tichonov regularization parameter. Upon substitution of $E = U\Lambda V^T$, the inverse form can be written in a fashion similar to Equation (A.1), where the diagonal matrix W takes the form

$$W^{-1} = \mathrm{diag}\{\sigma_1^{-1}, \sigma_2^{-1}, \sigma_3^{-1}, \ldots, \sigma_M^{-1}\}; \qquad \sigma_i^{-1} = \frac{1}{\dfrac{\alpha^2}{\sigma_i} + \sigma_i} \tag{A.2}$$

Equations (A.1) and (A.2) were employed in this work for ill-conditioned matrices and for the Vandermonde matrix form. The determination of an optimal value for the regularization parameter α is generally not easy [21]. Therefore, α was set for a double precision calculation to $\alpha = 10^{-12}$.

REFERENCES

1. Golub G, Ortega JM. *Scientific Computing an Introduction with Parallel Computing*. Academic Press: San Diego, CA, 1993.
2. Hirsch C. *Numerical Computation of Internal and External Flows*. Wiley: New York, 1994.
3. Sonneveld P. CGS, a fast Lanczos-type solver for non symmetric linear systems. *SIAM Journal of Scientific and Statistical Computing* 1989; **10**: 36–52.
4. Van der Vorst H. BiCGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of non symmetric linear systems. *SIAM Journal of Scientific and Statistical Computing* 1992; **13**: 631–644.
5. Saad Y, Schultz MH. GMRES: a generalized minimal residual algorithm for solving non-symmetric linear systems. *SIAM Journal of Scientific and Statistical Computing* 1986; **7**: 856–869.
6. Wigton LB, Yu NJ, Young DP. GMRES acceleration for fluid dynamics codes. AIAA Paper 85-1494CP, 1985.
7. Walker HF. Implementation of the GMRES method using Householder transformations. *SIAM Journal of Scientific and Statistical Computing* 1988; **9**: 152–163.
8. Manteuffel TA. Adaptive procedure for estimating parameters for the non-symmetric Chebyshev iteration. *Numerische Mathematik* 1978; **31**: 183–208.
9. Saylor PE. Use of the singular value decomposition with the Manteuffel algorithm for non-symmetric linear systems. *SIAM Journal of Scientific and Statistical Computing* 1980; **1**: 210–222.
10. Elman HC, Saad Y, Saylor PE. A hybrid Chebyshev–Krylov subspace algorithm for solving non-symmetric systems of linear equations. *SIAM Journal of Scientific and Statistical Computing* 1986; **7**: 840–855.
11. Nachtigal NM, Reichel L, Trefethen LN. A hybrid GMRES algorithm for nonsymmetric linear system. *SIAM Journal of Matrix Analysis and Applications* 1992; **13**(3): 796–825.
12. Saad Y. Least-squares polynomials in the complex plane and their use for solving non-symmetric linear systems. *SIAM Journal of Numerical Analysis* 1987; **24**: 155–169.
13. Isaacson E, Keller HB. *Analysis of Numerical Methods*. Wiley: New York, 1966.
14. Webster R. Efficient algebraic multigrid solvers with elementary restriction and prolongation. *International Journal for Numerical Methods in Fluids* 1998; **28**: 317–336.
15. Reichel L. The application of Leja points to Richardson iteration and polynomial preconditioning. *Linear Algebra Applications* 1991; **154–156**: 389–414.
16. Saad Y. Iterative solution of indefinite symmetric linear systems by methods using orthogonal polynomials over two disjoint intervals. *SIAM Journal of Numerical Analysis* 1983; **20**(4): 784–811.
17. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. *Numerical Recipes in Fortran*. Cambridge University Press: Cambridge, 1992.
18. Pan VY. Solving a polynomial equation: some history and recent progress. *SIAM Reviews* 1997; **39**(2): 187–220.
19. Venkatakrishnan V. Implicit schemes and parallel computing in unstructured grid CFD. In *Computational Fluid Dynamics*, Von Karman Institute for Fluid Dynamics, Lecture Series *1995-02*, 1995.
20. Anderson JD. *Modern Compressible Flow—with Historical Perspective*. McGraw-Hill: New York, 1982.
21. Hammerlin G, Hoffmann KH. *Numerical Mathematics*. Springer: New York, 1991.