References

[1] L.A. Hageman and D.M. Young, *Applied Iterative Methods*, Academic Press, Orlando Florida, (1981).

[2] Y. Saad, "On the Lanczos method for solving symmetric linear systems with several right-hand sides," *Math. Comp.* 48(178), 651-62 (1987).

[3] H.A. van der Vorst, "An iterative method for solving $f(A)x = b$ using Krylov subspace information obtained for the symmetric positive definite matrix A," *J. Comput. App. Math.* **18** 249-63 (1987)

[4] M.D. Gunzburger *Finite Element Methods for Viscous Incompressible Flows*, Academic Press, San Diego, (1989).

[5] J. Peterson, "The reduced basis method for incompressible viscous flow calculations," *SIAM J. Sci. Statist. Comput.* **10**, 777-786 (1989).

[6] G. Dahlquist and Å. Björk, *Numerical Methods*, Prentice Hall, Englewood Cliffs, New Jersey, (1974).

[7] G.H. Golub and C.F. van Loan, *Matrix Computations*, Johns Hopkins University Press, (1983).

[8] Y. Maday, and A.T. Patera, "Spectral element methods for the Navier-Stokes equations", in *State of the Art Surveys in Computational Mechanics*, edited by A.K. Noor, ASME, New York, pp. 71-143, 1989.

[9] Y. Maday, A.T. Patera, and E.M. Rønquist, "An operator-integration-factor splitting method for time-dependent problems: application to incompressible fluid flow." *J. Sci. Comput.*, **5**(4), pp. 310-37, (1990).

[10] E.M. Rønquist, "A Domain Decomposition Method for Elliptic Boundary Value Problems: Application to Unsteady Incompressible Fluid Flow," in *Fifth Conference on Domain Decomposition Methods for Partial Differential Equations* (D.E. Keyes, T.F. Chan, G.A. Meurant, J.S. Scroggs, and R.G. Voigt, eds.), SIAM, 1992.

[11] L. Mansfield, "On the use of deflation to improve the convergence of conjugate gradient iteration", *Comm. in Appl. Numer. Meth.*, **4**, 151-56 (1988).

[12] R.A. Nicolaides, Deflation of conjugate gradients with applications to boundary value problems", *SIAM J. Numer. Anal.*, **24**(2), 355-65 (1987).

[13] P.F. Fischer, "Parallel Domain Decomposition for Incompressible Fluid Dynamics", in *Proceedings of the Sixth Int. Conf. on Domain Decomposition Methods in Science and Engineering*, edited by A. Quarteroni, American Mathematical Soc., in press.

[14] S.E. Krist and T.A. Zang "Numerical simulation of channel flow transition," NASA Tech. Paper 2667, LaRC, Hampton, VA (1987).

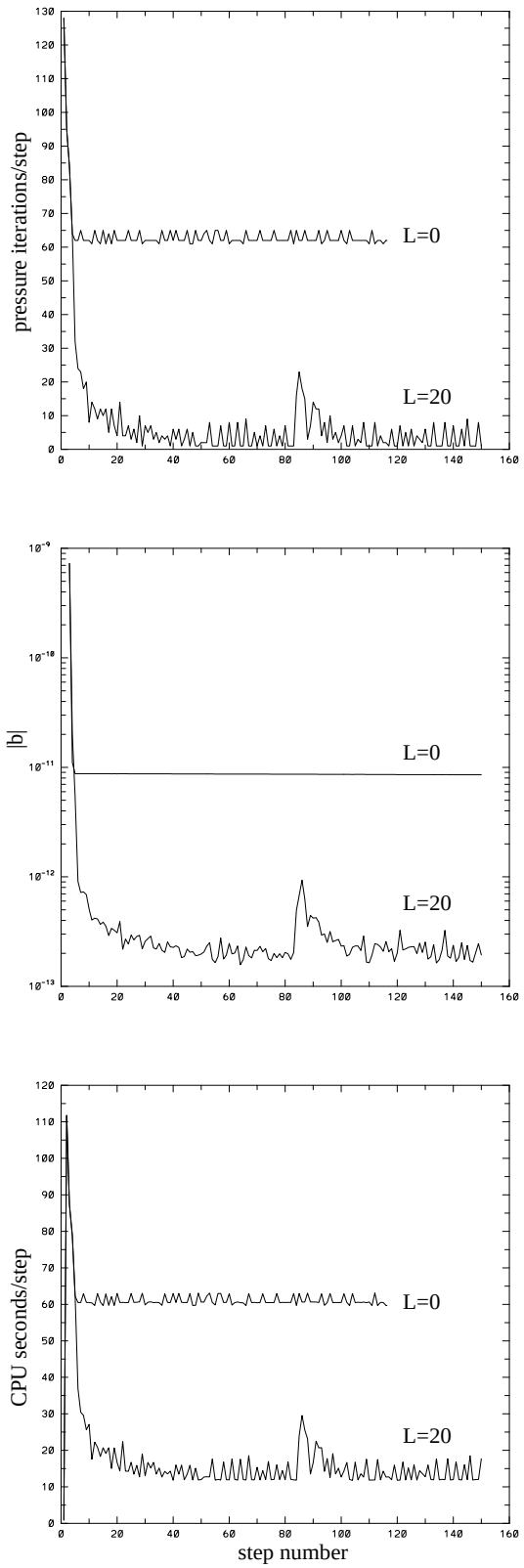

Figure 2: Iteration count (a), initial residual (b), and eight-processor iPSC/860 CPU time (c) for the A-conjugate projection technique applied to the three-dimensional Tollmien-Schlichting wave benchmark problem with $K = 54$ elements of order $N = 7$.

observed growth rate is $Im(\omega_{\rm SE}) = -.028273$, compared to $Im(\omega_{\rm LT}) = -.028230$ predicted by linear theory. The calculations were performed in 64-bit arithmetic and the pressure tolerance was set to 10^{-13} in order to observe high-order spatial and second-order temporal convergence rates.

In Fig. 2a we compare the required number of pressure iterations for the A-conjugate projection method ($L = 80$) to the standard case ($L = 0$). For this problem, the projection method reduces the number of iterations from roughly 60 to as few as *one* per time step, with an average of 3.3. A peak of roughly 20 iterations results when the basis set is restarted, e.g., at step 83. The addition of more basis vectors does not further reduce the iteration count other than by reducing the frequency of restart. The corresponding pre-solver residual and Navier-Stokes solution times shown in Figs. 2b and 2c indicate respective fifty- and four-fold reductions. The computations were carried out on an eight-node Intel iPSC/860. We note that the savings attained is typical for this particular class of problems. However, the performance of the projection techniques for these convergence benchmarks is exceptional and does not reflect the reduction attained in more general engineering flows.

Finally, we remark that the $O(mL)$ memory requirement for the projection methods may at first seem quite high. However, this must be examined in the context of the application. First, the present application is for a general geometry Navier-Stokes solver, rather than just a linear equation solver. Consequently, the *total* memory requirements are already quite high, as it is necessary to store the grid coordinates, metrics, Jacobians, etc., as well as several scalar and vector fields. In addition, efficient iterative solvers generally require significant storage for preconditioners - with memory costs scaling at least as m. Thus, the relative increase in memory demanded by saving a set of basis vectors may not be prohibitive. Secondly, on dedicated distributed memory machines, these algorithms provide a classic example of superlinear speedup; for a problem of fixed size, increasing the number of processors results in increased memory, thus allowing an increased value of L and corresponding decrease in $\mathcal{N}_E$. Our preference is to regard this fact as a flaw in the fixed-problem-size parallel performance metric, rather than to claim that projection techniques are a pathway to super-linear parallel algorithms. It is nonetheless a classic example of a space-time trade-off which can have a very real impact in many circumstances.

Acknowledgements

The author would like to thank Dr. Einar Rønquist and Professor Anthony Patera for many valuable discussions during the course of this work.

Table 1 compares the average iteration count per step in the von Karman street regime for several values of Re and Δt. In all cases, the Method 2 ($L = 20$) yields a slight improvement over Method 1 ($L = 20$) and roughly a fifty-percent reduction over the standard case ($L = 0$). This is typical of the performance observed in other two- and three-dimensional flows of similar complexity. In large three-dimensional flows, the savings in pressure iterations typically translates into a fifty-percent reduction in CPU time [13].

Table I: Iteration count for flow past a cylinder.

Re	Δt	Standard	Meth. 1	(ratio)	Meth. 2	(ratio)
100	0.01	65	45	(.68)	39	(.59)
200	0.01	90	48	(.53)	43	(.48)
100	0.04	125	65	(.52)	61	(.49)
200	0.04	159	89	(.56)	85	(.53)

As a second example, we consider the benchmark problem of computing the growth rate of small amplitude three-dimensional Tollmien-Schlichting (TS) waves in plane Poiseuille flow at $Re = 1500$, e.g. [14]. The domain consists of two-flat plates separated by a distance $2h$, periodic boundary conditions in the streamwise and spanwise directions with periodicity lengths $2\pi h$. The initial condition is a parabolic profile with unit centerline velocity, with a superimposed three-dimensional TS wave corresponding to the least damped eigenmode having amplitude 10^{-4} and horizontal wave numbers α and β of unity. For the spectral element calculation with $K = 54$, $N = 7$, and non-dimensional time step $\Delta t = .00625$, the

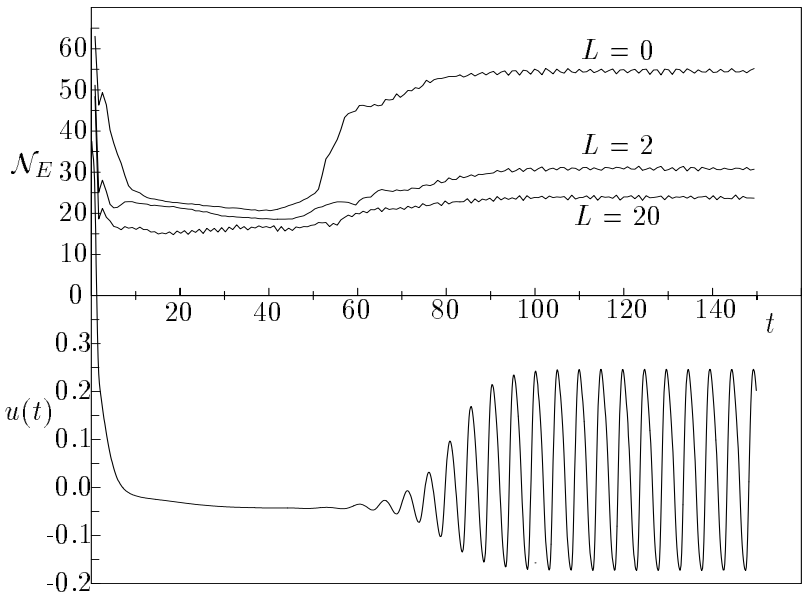

Figure 1: Pressure iteration count and time history of velocity for impulsively started flow past a cylinder at $Re = 200$.

and coarse subproblems,

$$E_f \underline{p}_f = \underline{g} - J E_c^{-1} J^T \underline{g} , \quad (16)$$

$$E_c \underline{p}_c = J^T \underline{g} - J^T E \underline{p}_f , \quad (17)$$

respectively. Here $E_f = E - E J E_c^{-1} J^T E$, and $E_c = J^T E J$. The fine system (16) is solved by conjugate-gradient iteration (with appropriate orthogonality conditions). Once $\underline{p}_f$ is established, the coarse-grid problem is solved (directly) for $\underline{p}_c$, and the procedure is complete. With appropriate application of a local, element-based preconditioner to E_f, the condition number of the fine system is significantly reduced relative to the originating E matrix.

The projection methods of Section 2 are implemented at the level of equation (14), rather than being applied directly to (16). At each time step, we solve for the change in pressure $\Delta \underline{p}^n \equiv \underline{p}^n - \underline{p}^{n-1}$. Thus, in the notation of Section 2, we take $A = E$, $\underline{x}^n = \Delta \underline{p}^n$, and $\underline{b}^n = \underline{g}^n - E \underline{p}^{n-1}$.

4 Results and Conclusion

We first consider the problem of two-dimensional start-up flow past a cylinder at $Re = \frac{DU_\infty}{\nu} = 200$. The discretization consists of $K = 116$ spectral elements of degree $N = 9$ ($m = 7424$), with time step $\Delta t = .0168$, non-dimensionalized with respect to U_∞ and D. The tolerance for the L_2 norm of the pressure residual was set to 3×10^{-6}, a value commensurate with the achievable discrete divergence of the resultant velocity field in 32-bit precision.

In Fig. 1 we plot the required number of pressure iterations per step, $\mathcal{N}_E$, for the cases $L = 0$, 2, and 20, using the A-conjugate projection technique of Section 2.2. For clarity, a 50 step windowed average is presented. Over the non-dimensional time simulated, $t = 0$ to 150, the flow passes through three transient regimes: symmetric wake formation, wake destabilization, and periodic (von Karman) vortex shedding. The first and third regimes are characterized by a high level of dynamic activity, while the second is relatively quiescent, as illustrated in the lower half of Fig. 1 by the time trace of u at a point in the near wake region of the cylinder. In flows devoid of dynamics, the pressure at time t^n is well represented by $\underline{p}^{n-1}$. Hence, little improvement results from incorporating information from more than one time step, as seen in the quiescent regime ($t \simeq 10 - 50$). However, for flows having a richer dynamical structure, the enriched basis of the projection method provides potential for significant savings, as seen in the von Karman street regime in which a two-fold reduction in iteration count is attained for $L = 20$. Increasing the number of basis functions to $L = 30$ brings no further significant reduction in this case.

explicitly via a characteristic/sub-cycling scheme, and the viscous and divergence operators are treated implicitly. The discretization leads to the following linear Stokes problem to be solved at each time step:

$$\begin{aligned} H\,\underline{u}_i - D_i^T\,\underline{p} &= B\,\underline{f}_i, \qquad i = 1, ..., d \quad , \\ D_i\,\underline{u}_i &= 0 \quad . \end{aligned} \tag{13}$$

Here, H is the discrete equivalent of the Helmholtz operator, $\{ -\frac{1}{Re}\nabla^2 + \frac{1}{\Delta t}\}$; B is the mass matrix associated with the velocity mesh; $\mathbf{D} = (D_1, ..., D_d)$ is the discrete gradient operator; and underscore refers to basis coefficients. Further details of spectral element discretizations for the Navier-Stokes equations may be found in [8].

The solution of (13) is simplified by a Stokes operator splitting which decouples the viscous and pressure/divergence constraint [9]. This splitting leads to the solution of a standard Helmholtz equation for each velocity component, while the resulting system for the pressure is similar to (13) save that H is replaced by $\frac{1}{\Delta t}B$. The resulting system can be efficiently treated by formally carrying out block Gaussian elimination (Uzawa decoupling) for $\underline{p}$, leading to:

$$E\underline{p} = \underline{g}\,, \tag{14}$$

where

$$E = -\sum_{i=1}^{d} D_i\,B^{-1}D_i^T\,, \tag{15}$$

and $\underline{g}$ is the inhomogeneity resulting from the time-split treatment of (12). E corresponds to a consistent Poisson operator for the pressure and, though symmetric-positive definite, is less well conditioned than the Helmholtz problems for the velocity components. Consequently, solution of (14) dominates the Navier-Stokes solution time. The advantage of the Stokes splitting is that no system solves are required when applying E, as B is diagonal.

The consistent Poisson problem (14) is solved via a two-level iteration scheme developed by Rønquist [10] in which a coarse-grid operator is folded into a global conjugate-gradient iteration through deflation [11,12]. The coarse (subscript c) and fine (subscript f) decomposition is effected through a subdomain-motivated prolongation operator $J \in \mathcal{R}^{m\times K}$, where $m = K(N-1)^d$ is the number of pressure degrees-of-freedom. The column space of the prolongation operator J is intended to approximate the span of the low eigenmodes of the E system; for this particular problem, J maps element-piecewise-constant functions to the m nodes of the underlying spectral element discretization. The pressure is then expressed as $\underline{p} = J\underline{p}_c + \underline{p}_f$, leading to an algebraic reformulation of the original problem as solvable fine

Notice that the storage for this procedure is roughly half that of Method 1 as it only requires X_l, and not B_l. However, one additional A-multiply is required prior to the *solve_A* stage.

As before, we need a mechanism to update the set X_l. To satisfy (8) it is necessary project the most recent solution, $\underline{x}^n$, onto $X_l^\perp$ and normalize the result. If we do not insist upon a modified Gram-Schmidt procedure, this can be done with a single multiply by A as follows. If L is taken to be the maximum number of vector pairs to be stored, i.e., $l \leq L$, then at each time level:

$$
\begin{array}{lll}
\textit{If } (l = L) \textit{ then:} & \tilde{\underline{x}}_1 \longleftarrow \underline{x}^n / ||\underline{x}^n||_A & \\
& l \longleftarrow 1 & \\
\textit{else:} & \alpha_i = \tilde{\underline{x}}_i^T A \tilde{\underline{x}}, \qquad i = 1, \ldots, l & \\
& \tilde{\underline{x}}_{l+1} \longleftarrow (\tilde{\underline{x}} - \sum \alpha_i \tilde{\underline{x}}_i) / ||(\tilde{\underline{x}} - \sum \alpha_i \tilde{\underline{x}}_i)||_A & \\
& l \longleftarrow l + 1 & \\
\textit{endif} & &
\end{array}
\tag{11}
$$

Note that in (11) the required normalization satisfies $||(\tilde{\underline{x}} - \sum \alpha_i \tilde{\underline{x}}_i)||_A = (\tilde{\underline{x}}^T A \tilde{\underline{x}} - \sum \alpha_i^2)^{\frac{1}{2}}$ due to the the A-conjugate relationship (8) and can therefore be computed with no additional A multiplies.

3 Navier-Stokes Implementation

We have implemented the above projection techniques in spectral-element solution of the incompressible Navier-Stokes equations:

$$
\begin{aligned}
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p + \tfrac{1}{Re} \nabla^2 \mathbf{u} \qquad \text{in } \Omega, \\
\nabla \cdot \mathbf{u} &= 0 \qquad \text{in } \Omega,
\end{aligned}
\tag{12}
$$

where $\mathbf{u}$ is the velocity vector, p the pressure, and $Re = \frac{UL}{\nu}$ the Reynolds number based on a characteristic velocity and length scale, and kinematic viscosity.

Spatial discretization is based upon decomposition of the computational domain into K spectral elements which are locally mapped to $[-1, 1]^d$ in $\mathcal{R}^d$. Within each element, the geometry, solution, and data are expanded in terms of high-order tensor-product polynomial bases in each coordinate direction. Variational projection operators are used to discretize the elliptic equations arising from a semi-implicit treatment of (12) and a consistent variational formulation is used for the pressure/divergence treatment. The velocity is represented by Nth-order Lagrange polynomials on the Gauss-Lobatto-Legendre quadrature points, with C^0 continuity enforced at element interfaces. The pressure is represented by polynomials of degree $N - 2$ based upon the Gauss-Legendre quadrature points. Temporal discretization is based upon an operator splitting in which the nonlinear convective terms are treated

2.2 Method 2: A-conjugate Projection

The procedure of the preceding section follows the intuitive line of reasoning that if $\underline{b}^n$ is well approximated by $\bar{\underline{b}} = \sum \alpha_j \tilde{\underline{b}}_j$, then $\underline{x}^n$ will be well approximated by $\bar{\underline{x}} = \sum \alpha_j \tilde{\underline{x}}_j$. The degree of approximation can be quantified by noting that $\bar{\underline{b}}$ is the L_2 projection of $\underline{b}^n$ onto B_l, which implies that $\bar{\underline{x}}$ is the best approximation to $\underline{x}^n$ in X_l with respect to the A^2-norm: $||\underline{x}||_{A^2} \equiv < A\underline{x}, A\underline{x} >^{\frac{1}{2}}$. If A is symmetric positive definite and conjugate gradient iteration is employed, it is sensible to begin with a projection which minimizes the distance between $\underline{x}^n$ and X_l in the A-norm, $||\underline{x}||_A \equiv < \underline{x}, A\underline{x} >^{\frac{1}{2}}$, since the conjugate gradient method seeks approximations which successively minimize the error in the A-norm [7].

The derivation of the resultant projection method is based upon a straightforward minimization procedure. Assuming as before that we have a set of previous solution vectors $X_l = \{\tilde{\underline{x}}_i\}$, $i = 1, \ldots, l$, we seek coefficients α_i such that the approximation given by

$$\bar{\underline{x}} = \sum_{i=1}^{l} \alpha_i \tilde{\underline{x}}_i \tag{6}$$

minimizes the error in the A-norm:

$$||\underline{x}^n - \bar{\underline{x}}||_A^2 \quad = \quad (\underline{x}^n)^T A \underline{x}^n \; - \; 2 \sum_{i=1}^{l} \alpha_i (\tilde{\underline{x}}_i^T A \underline{x}^n) \; + \; \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j \, (\tilde{\underline{x}}_i^T A \tilde{\underline{x}}_j) \quad . \tag{7}$$

The minimization procedure is simplified if we insist that the $\tilde{\underline{x}}_i$'s are A-conjugate and normalized to satisfy:

$$\tilde{\underline{x}}_i^T A \tilde{\underline{x}}_j \; = \; \delta_{ij} \quad . \tag{8}$$

Requiring a vanishing first variation of (7) with respect to α_i leads to:

$$\alpha_i \; = \; \tilde{\underline{x}}_i^T A \underline{x}^n \; = \; \tilde{\underline{x}}_i^T \underline{b}^n \quad , \quad i = 1, \ldots, l \quad . \tag{9}$$

Thus, given a set of vectors $X_l = \{\tilde{\underline{x}}_i\}$ satisfying (8), the best approximation to $\underline{x}^n$ is found by simply projecting $\underline{b}^n$ onto X_l. This forms the kernel of Method 2:

$$\begin{aligned}
&\textit{At time level } n, \quad \textit{input } b^n\textit{:} \\
&\qquad \alpha_i = \tilde{\underline{x}}_i^T \underline{b}^n, \qquad i = 1, \ldots, l \\
&\qquad \bar{\underline{x}} \longleftarrow \textstyle\sum \alpha_i \tilde{\underline{x}}_i \\
&\qquad \tilde{\underline{b}} \longleftarrow \underline{b}^n - A\bar{\underline{x}} \\
&\qquad \textit{solve } \; A\tilde{\underline{x}} = \tilde{\underline{b}} \quad \text{to tolerance } \epsilon \\
&\qquad \underline{x}^n \longleftarrow \tilde{\underline{x}} + \bar{\underline{x}} \\
&\qquad \textit{update } \{\tilde{X}_l\} \\
&\qquad \textit{return } \; \underline{x}^n
\end{aligned} \tag{10}$$

The computation of $\underline{\tilde{b}}$ is simply standard Gram-Schmidt orthogonalization of $\underline{b}^n$ with respect to B_l. The α_i's are computed in a group prior to modifying $\underline{b}^n$. in order that, in a parallel computation in which $\underline{b}^n$ and $\underline{\tilde{b}}_k >$ are distributed across P processors, the global communication required for the inner-products may be carried out in a single $O(\log_2 P)$ data exchange of an l-vector, at a cost commensurate with that of a standard inner-product. If necessary, it is possible to employ a more stable modified Gram-Schmidt procedure [3,6] for the computation of $\underline{\tilde{b}}$, at the expense of l individual vector reductions. However, we have not found this to be necessary in any application to date, probably due to the fact that the orthogonality condition (3) results from a Gram-Schmidt procedure (below) rather than from recursion as in the case of Krylov methods.

To complete the procedure (4), we require a mechanism for updating the basis sets $\{B_l, X_l\}$. Initially, the sets are empty, and can be filled with the first l solutions and data. In fact, since $\underline{\tilde{b}} \perp \underline{\tilde{b}}_k$, $k = \{1, \ldots, l\}$ by construction, the vector pair $\{\underline{\tilde{b}}, \underline{\tilde{x}}\}$ seems like a likely candidate to add to the basis set. However, this will not, in general, be stable because $A\underline{\tilde{x}} = \underline{\tilde{b}}$ is not satisfied exactly. This situation can be corrected by re-computing the required inhomogeneity, i.e., setting $\underline{\hat{b}} = A\underline{\tilde{x}}$, and enforcing (3) via a second Gram-Schmidt procedure. Additionally, we need a strategy for deciding which vectors to keep when the size of the basis set exceeds available memory capacity. There are several possibilities, e.g., retaining those vectors which repeatedly capture most of the energy in $\underline{b}^n$. Initial trials indicate that a reasonable approach is to save just the solution to the current problem, $\underline{x}^n = \underline{\tilde{x}} + \sum \alpha_k \underline{\tilde{x}}_k$, which is a near optimal linear combination of elements in the current basis set.

We summarize the update procedure as follows. If L is taken to be the maximum number of vector pairs to be stored, i.e., $l \leq L$, then at each time step:

$$
\begin{array}{lll}
\textit{If } (l = L) \textit{ then:} & \underline{\tilde{b}}_1 \longleftarrow A\underline{x}^n / ||A\underline{x}^n|| & (5) \\
& \underline{\tilde{x}}_1 \longleftarrow \underline{x}^n / ||A\underline{x}^n|| & \\
& l = 1 & \\
\textit{else:} & \underline{\hat{b}} \longleftarrow A\underline{\tilde{x}} & \\
& \alpha_k = < \underline{\hat{b}}, \underline{\tilde{b}}_k >, \qquad k = 1, \ldots, l & \\
& \underline{\tilde{b}}_{l+1} \longleftarrow (\underline{\hat{b}} - \sum \alpha_k \underline{\tilde{b}}_k) / ||\underline{\hat{b}} - \sum \alpha_k \underline{\tilde{b}}_k|| & \\
& \underline{\tilde{x}}_{l+1} \longleftarrow (\underline{\tilde{x}} - \sum \alpha_k \underline{\tilde{x}}_k) / ||\underline{\hat{b}} - \sum \alpha_k \underline{\tilde{b}}_k|| & \\
& l = l + 1 & \\
\textit{endif} & &
\end{array}
$$

Here, $||.|| = < . >^{\frac{1}{2}}$. The procedure re-initializes $\{B_l, X_l\}$ with the most recent solution pair when the memory limits are exceeded, and then reconstructs a set which satisfies (2-3).

here, as the projection is guaranteed to reduce the error in a relevant norm provided that $\underline{x}^n$ has some component in $span\{\underline{x}^{n-l}, \ldots, \underline{x}^{n-1}\}$. In the case where $\underline{x}^n$ is not well represented in this space the residual will be unchanged.

The proposed projection techniques are similar to reduced basis methods used in non-linear finite element problems [4,5]. However, the current methods can be implemented as "black boxes", with calls to *any* iterative solver, "$solve\_A$" and, for reasons to be discussed, the forward operator application, "$multiply\_by\_A$".

The outline of the paper is as follows. In Section 2 we describe two projection techniques for generating initial guesses $\bar{\underline{x}} \simeq \underline{x}^n$ based on l previous solutions $\underline{x}^k$, $n - l \leq k \leq n - 1$. In Section 3 we describe an application of the technique to the incompressible Navier-Stokes equations and in Section 4 we present performance results for several fluid dynamics calculations.

2 Projection Methods

2.1 Method 1

We begin by assuming that we have stored a set of vectors $B_l = \{\tilde{\underline{b}}_1, \ldots, \tilde{\underline{b}}_l\}$ and solution vectors $X_l = \{\tilde{\underline{x}}_1, \ldots, \tilde{\underline{x}}_l\}$ satisfying:

$$A\tilde{\underline{x}}_k \quad = \quad \tilde{\underline{b}}_k \qquad k = \{1, \ldots, l\} \quad . \tag{2}$$

Though not requisite, B_l and X_l are assumed to be derived from the l most recent problems, $\mathcal{P}_k$, $k = n - l, \ldots, n - 1$, i.e., $span\{B_l\} = span\{\tilde{\underline{b}}_{n-l}, \ldots, \tilde{\underline{b}}_{n-1}\}$. To simplify the orthogonalization, B_l is assumed to be orthonormal:

$$< \tilde{\underline{b}}_i, \tilde{\underline{b}}_j > \quad = \quad \delta_{ij} \quad , \tag{3}$$

where δ_{ij} is the Kroenecker delta, and $< \, . \, >$ is an appropriately weighted inner-product. The algorithm is based upon the following Gram-Schmidt procedure:

$$\begin{aligned} &At\ time\ level\ n, \quad input\ \ \underline{b}^n\text{:} \\ &\qquad \alpha_k =< \underline{b}^n, \tilde{\underline{b}}_k >, \qquad k = 1, \ldots, l \\ &\qquad \tilde{\underline{b}} \longleftarrow \underline{b}^n - \textstyle\sum \alpha_k \tilde{\underline{b}}_k \\ &\qquad solve\ \ A\tilde{\underline{x}} = \tilde{\underline{b}} \quad \text{to tolerance } \epsilon \\ &\qquad \underline{x}^n \longleftarrow \tilde{\underline{x}} + \textstyle\sum \alpha_k \tilde{\underline{x}}_k \\ &\qquad update\ \{B_l, X_l\} \\ &\qquad return\ \ \underline{x}^n \end{aligned} \tag{4}$$

1 Introduction

We consider iterative solution of a sequence of linear problems having the form:

$$\mathcal{P}_n : \qquad A\underline{x}^n = \underline{b}^n \quad , \qquad n = \{1, 2, \ldots\} \tag{1}$$

where A is an $m \times m$ matrix, and $\underline{x}^n$ is assumed to be a solution which is evolving with some parameter, e.g., time, in the case where (1) represents an implicit substep in numerical solution of a time dependent partial differential equation. When A is sufficiently sparse and not amenable to eigenfunction decomposition techniques (e.g., fast Poisson solvers), iterative methods are generally preferable to direct factorizations, both from the standpoint of storage and operation count. For the particular class of problems defined by (1), direct methods benefit from amortization of the one-time cost of matrix-factorization. However, they derive no benefit from the fact that successive problems might have very similar solutions, whereas iterative methods can exploit this possibilty as a good iniitial guess can lead to a significant reduction in the number of iterations required to bring the residual to within the specified tolerance.

The idea of using information generated from previous right-hand sides to speed iterative solution processes is not new. It is of course standard to solve only for the *change* in the solution, $\Delta\underline{x}^n \equiv \underline{x}^n - \underline{x}^{n-1}$ (e.g., [1]). For Krylov based methods, more substantial gains can potentially be attained if the residual vectors spanning $K_i^{n-1} = \{\underline{b}^{n-1}, A\underline{b}^{n-1}, \ldots, A^i\underline{b}^{n-1}\}$ are retained, and $\underline{b}^n$ is projected onto this space, e.g., as presented by Saad [2] for a Lanczos method and Van der Vorst [3] for the conjugate gradient method. The drawback of these techniques is that the required basis set, $K_i^n \in \mathcal{R}^{m\times i+1}$, might be quite large, and that there is no clear way to continue or update the set of saved vectors for a continuing sequence of right-hand sides.

In this paper we present two techniques for extracting information from previous problems, $\mathcal{P}_k$, $n-l \leq k \leq n-1$, to generate initial guesses to the current problem, $\mathcal{P}_n$. The first approach is to simply remove any component of $\underline{b}^n$ for which the solution is already known, by projecting $\underline{b}^n$ onto the set of vectors $\{\underline{b}^{n-l}, \ldots, \underline{b}^{n-1}\}$ having associated solutions $\{\underline{x}^{n-l}, \ldots, \underline{x}^{n-1}\}$, and to solve the problem corresponding to the component of $\underline{b}^n$ orthogonal to $span\{\underline{b}^{n-l}, \ldots, \underline{b}^{n-1}\}$. The second approach is a refinement of the first, which seeks the best approximation to $\underline{x}^n$ in $span\{\underline{x}^{n-l}, \ldots, \underline{x}^{n-1}\}$ with respect to a norm tailored to the convergence properties of the conjugate gradient method for the case when A is symmetric positive definite. These procedures are superior to those derived from extrapolation (e.g., based on high-order interpolants in time) in that projection techniques yield the best possible approximation within a given basis set. Moreover, while extrapolation techniques run the risk of generating a poor initial guess, this is not possible with the methods proposed

PROJECTION TECHNIQUES FOR ITERATIVE SOLUTION OF $A\underline{x} = \underline{b}$ WITH SUCCESSIVE RIGHT-HAND SIDES

Paul F. Fischer[1]
Division of Applied Mathematics
Brown University
Providence, RI 02912

ABSTRACT

We present two projection techniques for computing approximate solutions to linear systems of the form $A\underline{x}^n = \underline{b}^n$, for a sequence $n = 1, 2, ...$, e.g., such as arises from time discretization of a partial differential equation. The inexpensive approximate solutions can be used as initial guesses for iterative solution of the system, resulting in significantly reduced computational expense. Examples of two- and three-dimensional incompressible Navier-Stokes calculations are presented in which $\underline{x}$ represents the pressure, and A is a discrete Poisson operator. In flows containing significant dynamic activity, these projection techniques lead to as much as a two-fold reduction in solution time.

[1]This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681, and in part by the NSF under Grant # ASC-9107674.