Adaptive sparse linear solvers for implicit CFD using Newton–Krylov algorithms

L. McInnes [a], B. Norris [a,*], S. Bhowmick [b], P. Raghavan [b]

[a] *Mathematics and Computer Sciences Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439-4844, USA*

[b] *Department of Computer Science and Engineering, The Pennsylvania State University, 220 Pond Lab, University Park, PA 16802-6106, USA*

Abstract

We consider the simulation of three-dimensional transonic Euler flow using pseudo-transient Newton–Krylov methods [8,9]. The main computation involves solving a large, sparse linear system at each Newton (nonlinear) iteration. We develop a technique for adaptively selecting the linear solver method to match better the numeric properties of the linear systems as they evolve during the course of the nonlinear iterations. We show how such adaptive methods can be implemented using advanced software environments, leading to significant improvements in simulation time.

Keywords: Large-scale CFD simulations; Transonic Euler flow; Multi-method linear solvers; Sparse linear solution; Newton–Krylov methods; Pseudo-transient continuation

1. Introduction

Implicit solution methods play a critical role in computational dynamics (CFD) applications modeled by partial differential equations (PDEs) with different temporal and spatial scales. We consider the solution of the classical problem of three-dimensional transonic Euler about an ONERA M6 wing using pseudo-transient Newton–Krylov methods [8]. The majority of computation time in these simulations is spent on solving a large, sparse linear system at each nonlinear iteration, and the numeric properties of these linear systems evolve during the course of the nonlinear iterations. In this paper, we develop an approach for adaptively selecting linear solvers to match more closely the evolving numeric properties of the linear systems. We discuss the instantiation of such adaptive methods using advanced software environments, and we report on experiments that demonstrate significant improvements in overall simulation time through adaptive methods.

Section 2 contains a description of our transonic application. Section 3 describes the Newton–Krylov algorithmic framework for the solution of such PDE-based CFD problems and its implementation using advanced software environments. Section 4 presents an approach for adaptive linear solver selection for our application. This section contains empirical results that show the significant reductions in total simulation times achieved with adaptive solvers relative to the traditional approach of using a single linear solution method. Section 5 contains concluding remarks and future research directions.

2. Simulating three-dimensional transonic flow

We solve the steady-state, three-dimensional compressible Euler equations on mapped, structured meshes using a second-order, Roe-type, finite-volume discretization. The governing system of PDEs can be expressed in coordinate-invariant form by

$$\nabla \cdot (\rho \mathbf{u}) = 0, \tag{1}$$

$$\nabla \cdot (\rho \mathbf{u}\mathbf{u} + pI) = 0, \tag{2}$$

$$\nabla \cdot ((\rho e + p)\mathbf{u}) = 0, \tag{3}$$

where ρ, $\mathbf{u}$, and e represent the density, three-dimensional velocity, and energy density fields, respectively. The pressure field p is determined by an algebraic equation of state, $e = \frac{p}{\gamma - 1} + \frac{1}{2}\rho(u^2 + v^2 + w^2)$, where γ is the ratio of specific heats. As described in [13], this system and its discretiza-

* Corresponding author. Tel.: +1 630 252 7908; Fax: +1 630 252 5986; E-mail: norris@mcs.anl.gov

Computational Fluid and Solid Mechanics 2003
K.J. Bathe (Editor)

Fig. 1. Mach number contours (the local tangential velocity magnitude divided by the local sound speed) of the converged flowfield on the upper wing surface for a mesh with 224,264 nodes.

tion are standard and produce a nonlinear system of the form

$$f(u) = 0, \tag{4}$$

where u is a vector of unknowns representing the state of the system, and $f(u)$ is a vector-valued function of residuals of the governing equations.

The basis for our implementation, as discussed in [8], is a legacy sequential Fortran 77 code by Whitfield and Taylor [13] that uses a mapped, structured C-H mesh to compute steady-state residuals and finite-difference Jacobians. The residual evaluations are undertaken to second order, while the Jacobian matrices (used only for preconditioning in this work) are evaluated to first order. We replaced the explicit characteristic boundary conditions with fully implicit characteristic variants, which help to maintain stability in the pseudo-transient Newton–Krylov approach, which is discussed in Section 3.

We explore the standard aerodynamics test case of transonic flow over an ONERA M6 wing using the frequently studied parameter combination of a freestream Mach number of 0.84 with an angle of attack of 3.06°. The robustness of solution strategies is particularly important for this model because of the so-called λ-shock that develops on the upper wing surface, as depicted in Fig. 1.

3. Algorithms and software

We use pseudo-transient continuation to solve the discretized steady-state nonlinear problem given by Eq. (4). As discussed in [9], we solve a sequence of problems derived from the model $\frac{\partial u}{\partial t} = -f(u)$, namely

$$g_\ell(u) \equiv \frac{1}{\tau^\ell}(u - u^{\ell-1}) + f(u) = 0, \quad \ell = 1, 2, \ldots. \tag{5}$$

Our time-stepping scheme is based on the SER technique [10] and advances the time step in inverse proportion to residual norm progress, $\tau^\ell = \tau^{\ell-1} \cdot \frac{\|f(u^{\ell-2})\|}{\|f(u^{\ell-1})\|}$, within bounds relative to the current step [8]. We employ a locally adapted variant, where for a given mesh point j, $\tau_j^\ell = \alpha_j^\ell N_{\mathrm{CFL}}^\ell$, where α_j is a ratio of signal transit time to cell volume, and N_{CFL} is the CFL number, which serves as a global dimensionless scaling factor. At each time step we apply a single inexact Newton iteration (see, e.g., [11]) to Eq. (5) through the two-step sequence of (approximately) solving the Newton correction equation

$$\left(\frac{I}{\tau^\ell} + f'(u^{k-1})\right) \delta u^k = -f(u^{k-1}), \tag{6}$$

where I is the identity matrix, and then updating the iterate via $u^k = u^{k-1} + \delta u^k$. We employ matrix-free Newton–Krylov methods (see, e.g., [7]), in which we compute the action of the Jacobian on a vector v via directional differencing of the form $f'(u)v \approx \frac{f(u+hv)-f(u)}{h}$, where h is a differencing parameter. We precondition the Newton–Krylov methods, whereby we increase the linear convergence rate at each nonlinear iteration by transforming the linear system (6) into the equivalent form

$$B^{-1}\left(\frac{I}{\tau^\ell} + f'(u^{k-1})\right) \delta u^k = -B^{-1} f(u^{k-1}), \tag{7}$$

through the action of a preconditioner, B, whose inverse action approximates that of the Jacobian (see, e.g., [2]).

We implement the pseudo-transient Newton–Krylov solvers using PETSc [3,4], a suite of software for the scalable solution of PDE-based applications. The software design, which incorporates a hierarchy of data and algorithmic components, facilitates experimentation and the development of novel algorithms. For example, we implemented the adaptive linear solvers discussed in Section 4 by simply introducing a new routine that monitors the nonlinear convergence and then activates different linear solvers depending on how the simulation is progressing. No changes in the application code or library were required for these experiments.

4. Adaptive linear solvers

We report on experiments that compare the traditional approach of using a single linear solver for the entire pseudo-transient Newton process with an adaptive scheme that employs different preconditioners during different simulation phases. We consider two problem sizes, each having five degrees of freedom per node: a mesh of dimension $50 \times 10 \times 10$ (labeled problem 1) and a mesh of $98 \times 18 \times 18$ (labeled problem 2). As a preconditioning operator we use a first-order finite difference approximation of the Jacobian, which is held fixed over ten nonlinear iterations. All runs plotted in Figs. 2–4 use matrix-free GMRES(10) and a relative linear convergence tolerance that holds fixed for most of the nonlinear solve at 10^2 but then decreases during the

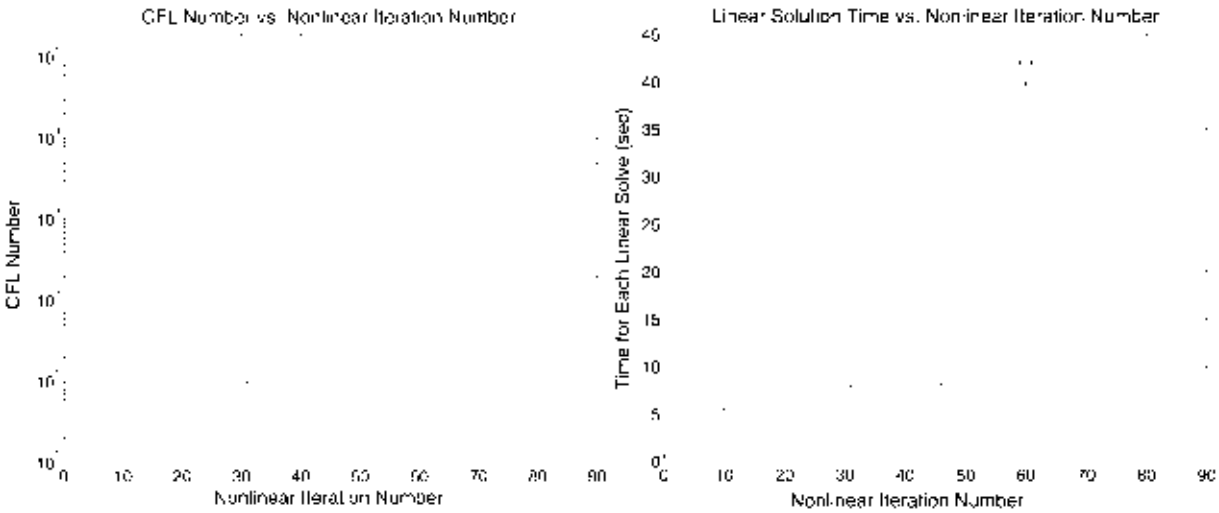

Fig. 2. Left-hand graph: Plot of CFL number vs. nonlinear iteration number, which illustrates the CFL transition from low to high values during the pseudo-transient Newton algorithm. Right-hand graph: Plot of time for each linear solve vs. nonlinear iteration number, which illustrates that different amounts of work are needed to solve the linear systems during various phases of the overall simulation. Both plots are for problem 2 using matrix-free GMRES(10) with a point-block ILU(1) preconditioner.

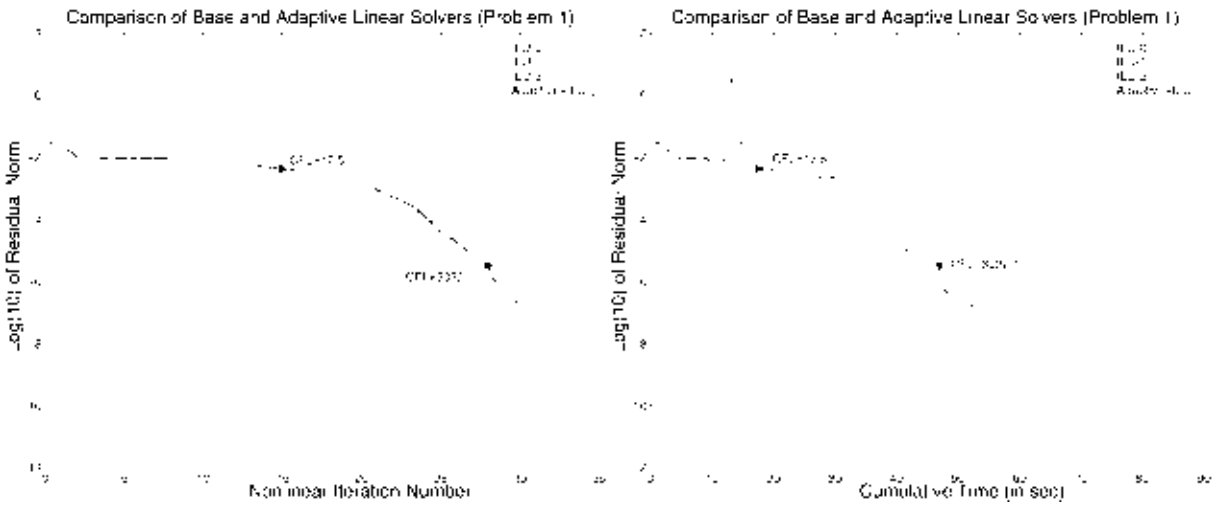

Fig. 3. Comparison of traditional linear solvers, which use a single preconditioner throughout the entire simulation, and an adaptive scheme, which uses a different preconditioner during each of the three phases of the pseudo-transient Newton–Krylov algorithm. We plot convergence rate (in terms of residual norm) versus both nonlinear iteration number (left-hand graph) and time (right-hand graph) for problem 1 using matrix-free GMRES(10).

final Newton phase. We performed these experiments on a dedicated workstation with a dual-CPU 500 MHz Pentium III with 512 MB of RAM.

Approximately eighty percent of overall time is devoted to solving the linearized Newton systems, and the choice of preconditioners largely determines whether low overall computational cost can be achieved [8]. Thus, we have developed an adaptive strategy based on the changing numeric properties of the Newton systems (6) during the course of the pseudo-transient iterations. As Kelley and Keyes discuss in their convergence analysis of pseudo-transient techniques [9], we can view these schemes as composed of three phases. During the first phase, when the time step τ^ℓ remains relatively small, the Jacobians associated with Eq. (5) are well conditioned. During the second phase the time step τ^ℓ advances to moderate values, and in the final phase τ^ℓ transitions toward infinity, so that the iterate u^ℓ approaches the root of Eq. (4). The left-hand graph of Fig. 2 plots CFL number versus time for problem 2, while the right-hand graph depicts the time spent during

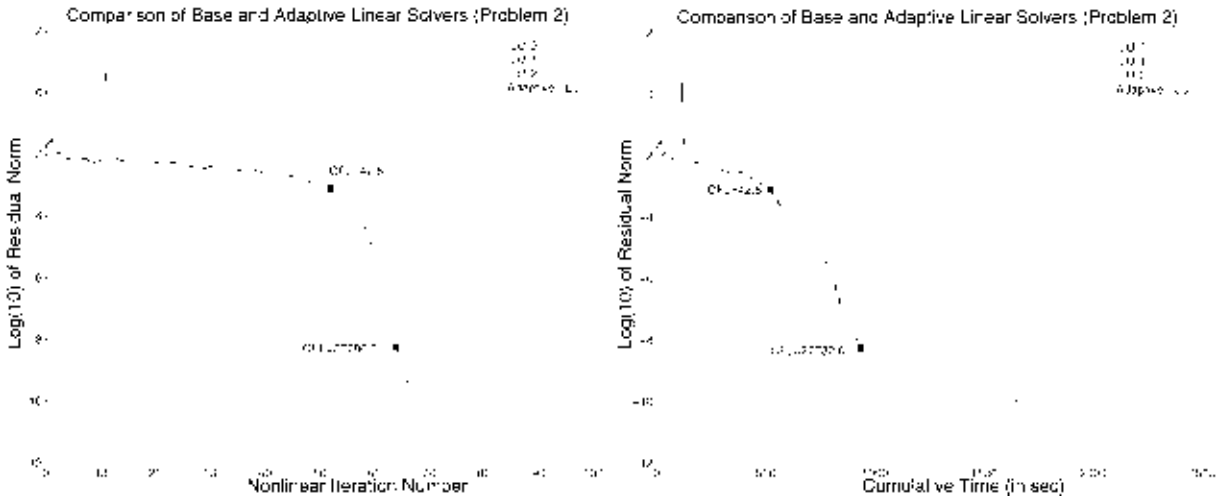

Fig. 4. Comparison of traditional linear solvers, which use a single preconditioner throughout the entire simulation, and an adaptive scheme, which uses a different preconditioner during each of the three phases of the pseudo-transient Newton–Krylov algorithm. We plot convergence rate (in terms of residual norm) versus both nonlinear iteration number (left-hand graph) and time (right-hand graph) for problem 2 using matrix-free GMRES(10).

each linear solve, using a fixed preconditioner of point-block ILU(1) with block size five. We observe that varying amounts of work are done during the linear solves in each phase of the pseudo-transient algorithm.

In response to this observation, we developed adaptive linear solvers that employ a different preconditioner for each of the three phases of the simulation, as indicated by CFL number. Our goal was to combine more robust (but more costly) methods when needed in a particular phase, with faster (though less powerful) methods in other phases, with the aim of reducing overall time to solution.

The graphs in Figs. 3 and 4 compare several base preconditioner methods with adaptive schemes for problems 1 and 2, respectively. We plot convergence rate (in terms of residual norm) versus both nonlinear iteration number (left-hand graph) and time (right-hand graph). We note that the nonlinear model employs a subtle form of continuation in boundary conditions by activating the full characteristic boundary conditions only after the tenth nonlinear iteration; this accounts for the spikes seen in the residual norm histories.

The convergence of the adaptive ILU methods, which change the level of fill in an incomplete factorization preconditioner during each phase of the pseudo-transient method, reduced the overall time to solution for these cases by 20–40 percent relative to traditional fixed linear solvers. While these preliminary experiments show the promise of adaptive linear solvers, much research remains to determine the most effective combinations of methods for particular problems.

5. Conclusions

We have provided a framework for adaptively selecting the linear solvers that are at the heart of implicit CFD codes using pseudo-transient Newton–Krylov algorithms. We demonstrated how simulation time can be reduced significantly by adaptively selecting linear solvers to fit the evolving numeric properties of linear systems generated at each nonlinear iteration.

Such adaptive schemes can also be applied to other computational steps of PDE-based CFD simulations. Thus, using multi-method strategies for dynamic matching of method attributes to problem characteristics can result in dramatic performance improvements. Multi-methods can also be used in an alternate form; for example, when several methods with different computational costs and failure rates are available for solving the same problem, a composite of these methods can improve reliability while minimizing worst-case average execution time [5,6]. We plan to continue our work on multi-method techniques and their instantiation through the design of advanced software architectures and common abstract interfaces [12]. We conjecture that these approaches can provide significant performance improvements and lower development costs for large-scale applications over traditional techniques.

Acknowledgments

We gratefully acknowledge Barry Smith, who enhanced PETSc functionality to facilitate our work.

L.M. and B.N. acknowledge the support by the Mathematical, Information, and Computational Sciences Division

subprogram of the Office of Computational Technology Research, U.S. Department of Energy under Contract W-31-109-Eng-38.

The work of S.B. and P.R. was supported in part by the Maria Goeppert Mayer Award from the U. S. Department of Energy and Argonne National Laboratory, and, by the National Science Foundation through grants ACI-0102537, DMR-0205232, and EIA-0221916.

References

[1] Anderson WK, Gropp WD, Kaushik DK, Keyes DE, Smith BF. Achieving high sustained performance in an unstructured mesh CFD application. Proceedings of SC99, 1999.

[2] Axelsson O. A survey of preconditioned iterative methods for linear systems of equations. BIT 1987;25:166–187.

[3] Balay S, Gropp W, McInnes LC, Smith B. Efficient management of parallelism in object oriented numerical software libraries. In: Arge E, Bruaset AM, Langtangen HP (Eds), Modern Software Tools in Scientific Computing. Birkhauser Press, 1997.

[4] Balay S, Buschelman K, Gropp W, Kaushik D, Knepley M, McInnes LC, Smith B, Zhang H. PETSc users manual. Tech. Rep. ANL-95/11 – Revision 2.1.3, Argonne National Laboratory, 2002 (see www.mcs.anl.gov/petsc).

[5] Bhowmick S, Raghavan P, Teranishi K. A combinatorial scheme for developing efficient composite solvers. In: Sloot P, Tan C, Dongarra J, Hoekstra A (Eds), Lecture Notes in Computer Science. Springer Verlag, 2002.

[6] Bhowmick S, Raghavan P, McInnes L, Norris B. Faster PDE-based simulations using robust composite linear solvers. Argonne National Laboratory preprint ANL/MCS-P993-0902, 2002.

[7] Brown PN, Saad Y. Hybrid Krylov methods for nonlinear systems of equations. SIAM J Sci Statistical Comput 1990;11:450–481.

[8] Gropp W, Keyes D, McInnes LC, Tidriri MD. Globalized Newton–Krylov–Schwarz algorithms and software for parallel implicit CFD. Int J High Performance Computing Applications 2000;14:102–136.

[9] Kelley CT, Keyes DE. Convergence analysis of pseudo-transient continuation. SIAM J Numer Anal 1998;35:508–523.

[10] Mulder W, Van Leer B. Experiments with implicit upwind methods for the Euler equations. J Comput Phys 1985;59:232–246.

[11] Nocedal J, Wright SJ, Numerical Optimization, Springer-Verlag, New York, 1999.

[12] Norris B, Balay S, Benson S, Freitag L, Hovland P, McInnes LC, Smith B. Parallel components for PDEs and optimization: some issues and experiences. Parallel Computing 2003; to appear.

[13] Whitfield DL, Taylor LK. Discretized Newton-relaxation solution of high resolution flux-difference split schemes. Proceedings of the AIAA Tenth Annual Computational Fluid Dynamics Conference, AIAA-91-1539, 1991.