An Inexact Newton Method for Systems Arising from the Finite Element Method

Philip J. Capon* Peter K. Jimack†

Abstract

In this paper we introduce an efficient and robust technique for approximating the Jacobian matrix for a nonlinear system of algebraic equations which arises from the finite element discretization of a system of nonlinear partial differential equations. It is demonstrated that when an iterative solver, such as preconditioned GMRES, is used to solve the linear systems of equations that result from the application of Newton's method this approach is generally more efficient than using matrix-free techniques: the price paid being the extra memory requirement for storing the sparse Jacobian. The advantages of this approach over attempting to calculate the Jacobian exactly or of using other approximations are also discussed. A numerical example is included which is based upon the solution of a 2-d compressible viscous flow problem.

Key words. Nonlinear problems, finite element method, approximate Jacobian, iterative linear solver, preconditioning.

1 Introduction

The numerical approximation of a variety of scientific and engineering problems requires the solution of nonlinear systems of algebraic equations. A popular approach to solving these problems is via inexact Newton methods where the Newton equations are solved approximately by an iterative solver. In general the Jacobian matrix for the nonlinear system will be nonsymmetric and indefinite so a quite general linear iterative solver is required.

In this paper we concentrate on the nonlinear equations arising from a finite element discretization of the compressible Navier-Stokes equations and consider an inexact Newton method based upon the use of preconditioned GMRES at each step ([1]). In particular we suggest that there are practical advantages to be gained from actually calculating estimates of the Jacobian matrices used in the Newton algorithm, rather than using matrix-free methods (as in [2] for example). Moreover we suggest that, by using our knowledge of the finite element structure behind the nonlinear algebraic equations, it is possible to obtain an extremely straightforward and reliable estimate of the Jacobian for this particular class of nonlinear system.

*Tessella Support Services plc, Abingdon, OX14 3PX, UK

†School of Computer Studies, University of Leeds, Leeds, LS2 9JT, UK.

2 Nonlinear Equations

In [3] and [4] it is shown that the steady 2-d compressible navier-Stokes equations may be expressed in the general form

$$\mathcal{L}(\underline{U}) = \sum_{i=1}^{2} A_i(\underline{U}) \frac{\partial}{\partial x_i} \underline{U} - \sum_{i=1}^{2} \sum_{j=1}^{2} \frac{\partial}{\partial x_i} \left[K_{ij}(\underline{U}) \frac{\partial}{\partial x_j} \underline{U} \right] = \underline{F} \, , \tag{1}$$

where $\underline{U} = (\rho, u, v, T)^T$ is a vector of dependent variables and the domain, Ω, and boundary conditions are problem dependent.

Given a triangulation of Ω into E elements, Ω^e, we may look for a piecewise linear finite element solution of (1) by defining the following trial space:

$$\mathcal{U}^h = \{ \underline{U}^h \in (C^0(\Omega))^4 : \underline{U}^h|_{\Omega^e} \in (P_1)^4 \text{ and } \underline{U}^h \text{ satisfies the essential BCs} \} \, .$$

In order to ensure the stability of the discretization we use a variant of the Galerkin Least-Squares approach of Hughes and co-workers (see [5] for example). This requires that we find $\underline{U}^h \in \mathcal{U}^h$ such that

$$\begin{aligned} \int_\Omega \left[\sum_{i=1}^{2} A_i(\underline{U}^h) \frac{\partial}{\partial x_i} \underline{U}^h \cdot \underline{V}^h + \sum_{i=1}^{2} \sum_{j=1}^{2} \frac{\partial}{\partial x_i} \underline{V}^h \cdot K_{ij}(\underline{U}^h) \frac{\partial}{\partial x_j} \underline{U}^h - \underline{F} \cdot \underline{V}^h \right] d\Omega \; + \\ \sum_{e=1}^{E} \int_{\Omega^e} \mathcal{L}(\underline{V}^h) \cdot \tau(\underline{U}^h) \mathcal{L}(\underline{U}^h) \, d\Omega \; - \int_{\partial\Omega} \sum_{i=1}^{2} \sum_{j=1}^{2} \underline{V}^h \cdot K_{ij}(\underline{U}^h) \frac{\partial}{\partial x_j} \underline{U}^h n_i \, d\Gamma \;\; = \;\; 0 \end{aligned} \tag{2}$$

for all appropriate trial functions $\underline{V}^h$. Note that full discretization details may be found in [4] and a thorough discussion on the possible choices of the tensor τ may be found in [5]. It is clear however that the finite element problem (2) leads to a system of nonlinear algebraic equations of the form

$$\underline{G}(\underline{X}) = \underline{0} \, , \tag{3}$$

where the unknowns, $\underline{X}$, represent the nodal approximations to the dependent variables (ρ, u, v, T). (Note that if implicit time-stepping is used to reach steady-state, rather than attempting to solve the steady problem directly, then a nonlinear system of this form must be solved at each time-step.)

3 Inexact Newton's Method

Newton's method requires the solution of the linear system

$$J(\underline{X}^{(n)}) \underline{\delta}^{(n)} = -\underline{G}(\underline{X}^{(n)}) \tag{4}$$

at each step $(n = 0, 1, 2, ...)$ where the Jacobian matrix, J, is such that $J_{ij} = \frac{\partial G_i}{\partial X_j}$.

When an iterative method, such as GMRES [1], is used to solve (4) the main step in each iteration is to find a matrix-vector product

$$\underline{w} = J(\underline{X}^{(n)}) \underline{p} \, . \tag{5}$$

It may be observed however that

$$J(\underline{X}^{(n)})\underline{p} \approx \frac{\underline{G}(\underline{X}^{(n)} + \delta\underline{p}) - \underline{G}(\underline{X}^{(n)})}{\delta} \tag{6}$$

and so the product (5) may be estimated without actually knowing $J(\underline{X}^{(n)})$, at the cost of an additional evaluation of $\underline{G}$. The advantages of this and similar matrix-free methods ([2]) are clear: no time is spent at the start of the n^{th} Newton step calculating $J(\underline{X}^{(n)})$, and no memory is required to store this matrix.

These advantages are perhaps not as overwhelming as one might immediately think however. Firstly, there are very few preconditioners that one can apply when J is not known explicitly. In addition, for finite element problems such as that being considered here, J is very sparse and need not require an inordinate amount of memory. Hence, if a sufficiently efficient way of calculating (or estimating) J can be found then it is clear that the potential for fewer iterations (due to preconditioning) and the lower cost of each iteration (since no function evaluations are needed) can ensure that using J explicitly is more efficient than a matrix-free method.

4 Calculation of Jacobians

Because our nonlinear system of equations, (2), has been derived from a finite element method it is possible to assemble the Jacobian matrix by looping through each element, Ω^e, in turn and calculating the *small* number of contributions to J from that element. In theory it is possible to calculate these contributions to $J(\underline{X}^{(n)})$ exactly be restricting equations (2) to the element in question and differentiating with respect to each of the 12 nodal degrees of freedom associated with this element (4 for each vertex). Clearly this is likely to be extremely complicated to do in practice, especially when $\tau(\underline{U}^h)$ is quite complex.

An alternative is to approximate the 12×12 element Jacobian on each triangle using a finite difference formula similar to (6) and then to assemble these into a sparse global matrix in the usual finite element manner. The cost of assembling an approximate Jacobian in this way is about the same as the cost of 13 evaluations of the function $\underline{G}$. This is because the j^{th} column of the element Jacobian, $J^{(e)}$ say, on Ω^e is found using

$$J^{(e)}(\underline{X}^{(n)})\underline{e}_j \approx \frac{\underline{G}^{(e)}(\underline{X}^{(n)} + \delta\underline{e}_j) - \underline{G}^{(e)}(\underline{X}^{(n)})}{\delta} ,$$

where $\underline{G}^{(e)}$ is the contribution to $\underline{G}$ from Ω^e and $\underline{e}_j \in R^{12}$ is such that each entry is zero apart from the j^{th} which is 1.

We now assume that the time required to take a single iteration of the linear solver is almost exclusively taken up with the matrix vector multiply. It then follows that the cost of each iteration using the matrix-free method (6) is approximately equal to the cost of a single evaluation of $\underline{G}$, T_G say. Hence, N iterations of the matrix-free method require approximately $(1 + N)T_G$ units of time.

Now suppose that when the sparse matrix $J(\underline{X}^{(n)})$ is known explicitly a single matrix-vector multiply can be achieved in time αT_G $(0 < \alpha < 1)$. It therefore follows that, in this case, N linear iterations will take approximately $(13 + \alpha N)T_G$ units of time. Hence, provided at least $\frac{12}{1-\alpha}$ linear iterations are required at each Newton step the second algorithm will be faster (even without taking into account the fact that this algorithm is far easier to precondition).

5 Numerical Example

For our test problem we consider one of the examples used in [2]. This requires the calculation of a two dimensional steady viscous flow around a NACA0012 aerofoil at an angle of incidence of 3^0. The free stream Mach number is 0.8 and the Reynold's number, Re, is 5000. When the problem is solved on meshes of 1617, 4146 and 8505 elements in turn we estimate α to be approximately 0.5 in each case. This means that provided each step of Newton iteration requires more than about 24 linear iterations our method, which explicitly estimates the Jacobian, will be superior. In our experience this always turns out to be the case for all but the most trivial of problems (and is also the case in [2] where over 80 function evaluations per step are reported).

In practice the nonlinear system (3) is actually solved using the software described in [6] which combines the inexact Newton approach with a linesearch backtracking algorithm to improve the convergence properties of the solver. For many problems this convergence to steady-state is obtained most efficiently through the use of time-stepping. Finally, the fact that an approximation to the Jacobian, $J(\underline{X}^{(n)})$, has been computed means that standard preconditioning techniques such as incomplete LU factorization, [7], may be utilized. Such preconditioners have a significant effect on the rate of convergence of the inner iterations.

For the problem described above it is possible to obtain convergence in a total of 35 nonlinear iterations and 1385 linear iterations on the finest of the three grids. This is based upon the use of local time-stepping with at most 6 Newton iterations per time step and at most 40 inner iterations per Newton step (using an ILU(0) preconditioner: i.e. no fill-in is permitted). The initial guess to the solution here is arbitrary and gives an initial residual of 8.5×10^{-2} in the nonlinear system (3). The final value of this residual is 1.5×10^{-9}.

6 Summary

We have described an approach for estimating the Jacobian of a large nonlinear system which arises from the finite element discretization of a nonlinear system of partial differential equations. Since this estimate of the Jacobian is built in an element-by-element manner it is extremely computationally efficient: costing about the same as just 13 nonlinear residual evaluations for 2-d compressible Navier-Stokes problems. (In 3-d, using a tetrahedral mesh, the cost would be about the same as 21 residual

evaluations.) Because of the low computational overhead associated with building this Jacobian we claim that it is more efficient to do this than to use matrix-free methods, such as those described in [2] for example. The price to be paid is the extra memory requirement of storing the sparse Jacobian at each Newton step.

In this paper we have not explicitly contrasted our approximation to the Jacobian with the use of an exact Jacobian matrix. Whilst the latter approach is theoretically possible it is worth noting that it is dramatically more complex to program than the approximate approach (and that an alteration to the equation being solved or to the choice of $\tau(\underline{U}^h)$ in (2) will mean significant additional programming). Moreover, in [4], numerical results indicate that there is no significant advantage in taking the exact approach since, even if an exact expression for the Jacobian can be encoded without error, it never appears to cause fewer Newton iterations to be taken in practice.

Acknowledgements

PJC would like to thank EPSRC and British Aerospace for funding in the form of a CASE studentship.

References

[1] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 7 (1986), pp. 856–869.

[2] R. Choquet and J. Erhel, *Newton-GMRES algorithm applied to compressible flow*, Int. J. Num. Meth. Fluids, 23 (1996), pp. 177–190.

[3] G. Hauke and T. J. R. Hughes, *A unified approach to compressible and incompressible flows*, Comp. Meth. in Appl. Mech. and Eng., 113 (1994), pp. 389–395.

[4] P. J. Capon, *Adaptive stable finite element methods for the compressible Navier-Stokes equations*, Ph.D. thesis, University of Leeds (1995).

[5] F. Shakib, T. J. R. Hughes and Z. Johan, *A new finite element formulation for computational fluid dynamics: X. The compressible Euler and Navier-Stokes equations*, Comp. Meth. in Appl. Mech. and Eng., 89 (1991), pp. 141–219.

[6] P. N. Brown and Y. Saad, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 450–481.

[7] J. A. Meijerink and H. A. van der Vorst, *Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems*, J. Comp. Phys., 44 (1981), pp. 134–155.