Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 193 (2004) 357–397

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Review

Jacobian-free Newton–Krylov methods: a survey of approaches and applications

D.A. Knoll [a,*], D.E. Keyes [b,1]

[a] Theoretical Division, Fluid Dynamics Group (T-3), Los Alamos National Laboratory, MS B216, Los Alamos, NM 87545, USA

[b] Department of Applied Physics and Applied Mathematics, Columbia University, 500 W. 120th Street, New York, NY 10027, USA

Received 17 October 2002; received in revised form 15 August 2003; accepted 18 August 2003

Abstract

Jacobian-free Newton–Krylov (JFNK) methods are synergistic combinations of Newton-type methods for superlinearly convergent solution of nonlinear equations and Krylov subspace methods for solving the Newton correction equations. The link between the two methods is the Jacobian-vector product, which may be probed approximately without forming and storing the elements of the true Jacobian, through a variety of means. Various approximations to the Jacobian matrix may still be required for preconditioning the resulting Krylov iteration. As with Krylov methods for linear problems, successful application of the JFNK method to any given problem is dependent on adequate preconditioning. JFNK has potential for application throughout problems governed by nonlinear partial differential equations and integro-differential equations. In this survey paper, we place JFNK in context with other nonlinear solution algorithms for both boundary value problems (BVPs) and initial value problems (IVPs). We provide an overview of the mechanics of JFNK and attempt to illustrate the wide variety of preconditioning options available. It is emphasized that JFNK can be wrapped (as an accelerator) around another nonlinear fixed point method (interpreted as a preconditioning process, potentially with significant code reuse). The aim of this paper is not to trace fully the evolution of JFNK, nor to provide proofs of accuracy or optimal convergence for all of the constituent methods, but rather to present the reader with a perspective on how JFNK may be applicable to applications of interest and to provide sources of further practical information.

1. Introduction and Background

The need to solve nonlinear systems of algebraic equations is ubiquitous throughout computational physics. Such systems typically arise from the discretization of partial differential equations (PDEs), whether scalar (such as heat conduction) or a system of coupled equations (such as the Navier–Stokes

[*] Corresponding author. Tel.: +1-505-667-7467; fax: +1-505-665-5926.
E-mail addresses: nol@lanl.gov (D.A. Knoll), david.keyes@columbia.edu (D.E. Keyes).
[1] Tel.: +1-212-854-4458; fax: +1-212-854-8257.

0021-9991/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.jcp.2003.08.010

equations). One may be interested in the steady-state solution of these equations (a boundary value problem, BVP) or in their dynamical evolution (an initial value problem, IVP). For BVPs, nonlinear iterative methods are desirable. The same is true for multiple time-scale IVPs, when discretized implicitly at each time step. A particular unified solution algorithm for these two classes of nonlinear systems, the Jacobian-Free Newton–Krylov method (JFNK), is the focus of this survey paper. JFNK methods have been developed and applied in many areas of computational physics, but so far by a relatively small number of researchers. The aim of this paper is to review recent advances and help accelerate the development and application of JFNK methods by a broader community of computational physicists.

It is our observation that solution strategies for nonlinearly implicit PDEs have evolved along somewhat different trajectories in the applied mathematics community and the computational physics community. In discussing solution strategies for BVPs [84,137] the applied mathematics community has emphasized Newton-based methods. Outside of finite element practitioners, the computational fluid dynamics (CFD) community has emphasized Picard-type linearizations and splitting by equation or splitting by coordinate direction [2,138]. The difference in predominating approach (Newton versus Picard) seems stronger for implicit IVPs. Again, the applied mathematics community has focused on Newton-based methods and on converging the nonlinear residual within a time step. In the computational physics community, operator splitting (time splitting, fractional step methods) [20,136] has been the "bread and butter" approach, with little attention to monitoring or converging the nonlinear residual within a time step, often allowing a splitting error to remain in time. In both IVP and BVP contexts, the concept of splitting (a form of divide-and-conquer at the operator level) has been motivated by the desire to numerically integrate complicated problems with limited computer resources. This tension does not vanish with terascale hardware, since the hardware is justified by the need to do ever more refined simulations of more complex physics. On can argue that the stakes for effective methods become higher, not lower, with the availability of advanced hardware.

Recent emphasis on achieving predictive simulations (e.g., in the ASCI [115] and SciDAC [178] programs of the US Department of Energy) has caused computational scientists to take a deeper look at operator splitting methods for IVPs and the resulting errors. As a result, the computational physics community is now increasingly driven towards nonlinear multigrid methods [22,188] and Jacobian-Free Newton–Krylov methods (JFNK) [27,47,84]. These nonlinear iterative methods have grown out of advances in linear iterative methods [6,67], multigrid methods [23,72,174,188], and preconditioned Krylov methods [157].

The standard nonlinear multigrid method is called the "full approximation scheme" or FAS [22,188]. Whereas a linear multigrid scheme usually solves for a delta correction for the solution based on linearized equations on coarser grid levels, FAS performs relaxation on the full (nonlinear) problem on each successively coarsened grid. In the FAS approach, the nonlinear correction (either Newton or simpler Picard) is not global, but resides inside the cycle over levels and the sweep over blocks of unknowns at each level. As a result, asymptotic quadratic convergence of the overall nonlinear iteration is not guaranteed. The virtues of FAS include its low storage requirement (if one can use a simple smoother), optimal convergence on some problems, and a tendency for an enlarged domain of convergence, relative to a straight Newton method directly on the finest discretization. Disadvantages include the hurdle of forming hierarchical grids, the expertise required to develop coarse grid representations of a nonlinear operator, and the potential for many expensive nonlinear function evaluations. FAS has been used extensively and successfully in many computational fluid dynamics settings [117,119,184].

In JFNK methods [27,47,84], the nonlinear iterative method is on the outside, and a linear iterative method on the inside. Typically, the outer Newton iteration is "inexact" [53] and strict quadratic convergence is not achieved. Asymptotic quadratic convergence is achievable, but only with effort on the part of the inner, linear iterative method, which is usually unwarranted when overall time to solution is the metric. An advantage of JFNK is that the code development curve is not steep, given a subroutine that evaluates the discrete residual on the desired (output) grid. Furthermore, inexpensive linearized solvers can be used as preconditioners. Developing effective preconditioners may be a challenge, and the storage required for the preconditioner and Krylov vectors may be a limitation.

There have been limited comparisons between FAS and JFNK methods on identical problems. The authors regard both methods as important and complementary. It is not the purpose of this survey paper to provide such comparisons, but rather to hasten the applicability of JFNK to new applications, via "real world" examples. Additionally, we direct the reader's attention to ongoing interactions between these two approaches such as JFNK as a smoother for FAS and multigrid as a preconditioner for JFNK [118,139,141].

An important feature of JFNK is that the overall nonlinear convergence of the method is not directly affected by the approximations made in the preconditioning. The overall framework, making use of multiple discrete approximations of the Jacobian operator, has a polymorphic object-oriented flavor that lends itself well to modern trends in software design and software integration. In many cases, including some referenced as case studies herein, JFNK has been used to retrofit existing BVP and IVP codes while retaining the most important investments (in the physics routines) of the original code.

The remainder of this paper is organized as follows. In Section 2, we present the fundamentals of the JFNK approach. Section 3 is devoted to considerations of preconditioning. In Section 4, we survey examples of JFNK from a variety of applications. In Section 5 we illustrate a number of techniques and "tricks" associated with using JFNK in real problems, including many of the applications discussed in Section 4. Section 6 considers a novel application of JFNK to PDE-constrained optimization. We conclude in Section 7 with a discussion of future directions for JFNK methodology, as influenced by directions for scientific and engineering applications, computer architecture, mathematical software, and the on-going development of other numerical techniques.

2. Fundamentals of the JFNK method

The Jacobian-free Newton–Krylov (JFNK) method is a nested iteration method consisting of at least two, and usually four levels. The primary levels, which give the method its name, are the loop over the Newton corrections and the loop building up the Krylov subspace out of which each Newton correction is drawn. Interior to the Krylov loop, a preconditioner is usually required, which can itself be direct or iterative. Outside of the Newton loop, a globalization method is often required. This can be implicit time stepping, with time steps chosen to preserve a physically accurate transient or otherwise, or this can be some other form of parameter continuation such as mesh sequencing.

2.1. Newton methods

The Newton iteration for $\mathbf{F}(\mathbf{u}) = 0$ derives from a multivariate Taylor expansion about a current point $\mathbf{u}^k$:

$$\mathbf{F}(\mathbf{u}^{k+1}) = \mathbf{F}(\mathbf{u}^k) + \mathbf{F}'(\mathbf{u}^k)(\mathbf{u}^{k+1} - \mathbf{u}^k) + \text{higher-order terms}. \tag{1}$$

Setting the right-hand side to zero and neglecting the terms of higher-order curvature yields a strict Newton method, iteration over a sequence of linear systems

$$\mathbf{J}(\mathbf{u}^k)\delta\mathbf{u}^k = -\mathbf{F}(\mathbf{u}^k), \quad \mathbf{u}^{k+1} = \mathbf{u}^k + \delta\mathbf{u}^k, \qquad k = 0, 1, \ldots \tag{2}$$

given $\mathbf{u}^0$. Here, $\mathbf{F}(\mathbf{u})$ is the vector-valued function of nonlinear residuals, $\mathbf{J} \equiv \mathbf{F}'$ is its associated Jacobian matrix, $\mathbf{u}$ is the state vector to be found, and k is the nonlinear iteration index. The Newton iteration is terminated based on a required drop in the norm of the nonlinear residual

$$\frac{\|\mathbf{F}(\mathbf{u}^k)\|}{\|\mathbf{F}(\mathbf{u}^0)\|} < \mathrm{tol}_{\mathrm{res}}, \tag{3}$$

and/or a sufficiently small Newton update

$$\left\| \frac{\delta \mathbf{u}^k}{\mathbf{u}^k} \right\| < \mathrm{tol}_{\mathrm{update}}. \tag{4}$$

For a scalar problem, discretized into n equations and n unknowns, we have

$$\mathbf{F}(\mathbf{u}) = \{F_1, F_2, \ldots, F_i, \ldots, F_n\} \tag{5}$$

and

$$\mathbf{u} = \{u_1, u_2, \ldots, u_i, \ldots, u_n\}, \tag{6}$$

where i is the component index. In vector notation, the (i,j)th element (ith row, jth column) of the Jacobian matrix is

$$J_{ij} = \frac{\partial F_i(\mathbf{u})}{\partial u_j}. \tag{7}$$

In this scalar example there is a one-to-one mapping between grid points and rows in the Jacobian. Forming each element of $\mathbf{J}$ requires taking analytic or discrete derivatives of the system of equations with respect to $\mathbf{u}$. This can be both error-prone and time consuming for many problems in computational physics. Nevertheless, there are numerous examples of forming $\mathbf{J}$ numerically and solving Eq. (2) with a preconditioned Krylov method [79,93,102,121,164,165,167]. $\mathbf{J}$ can also be formed using automatic differentiation [77].

2.2. Krylov methods

Krylov subspace methods are approaches for solving large linear systems introduced as direct methods in the 1950s [75], whose popularity took off after Reid reintroduced them as iterative methods in 1971 [148] (see the interesting history in [66]). They are projection (Galerkin) or generalized projection (Petrov–Galerkin) methods [157] for solving $\mathbf{Ax} = \mathbf{b}$ using the Krylov subspace, $\mathbf{K}_j$,

$$\mathbf{K}_j = \mathrm{span}\left(\mathbf{r}_0, \mathbf{Ar}_0, \mathbf{A}^2\mathbf{r}_0, \ldots, \mathbf{A}^{j-1}\mathbf{r}_0\right),$$

where $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$. These methods require only matrix–vector products to carry out the iteration (not the individual elements of $\mathbf{A}$) and this is key to their use with Newton's method, as will be seen below.

A wide variety of iterative methods fall within the Krylov taxonomy [11,84,157] and there are numerous recent survey papers in this area [71,146,159]. A principal bifurcation in the family tree is applicability to non-symmetric systems. Since the vast majority of fully coupled nonlinear applications of primary interest result in Jacobian matrices that are non-symmetric, we focus the discussion on this side of the tree. A further point of discrimination is whether the method is derived from the long-recurrence Arnoldi orthogonalization procedure, which generates orthonormal bases of the Krylov subspace, or the short-recurrence Lanczos bi-orthogonalization procedure, which generates non-orthogonal bases.

The widely used Generalized Minimal RESidual method (GMRES) [158] is an Arnoldi-based method. In GMRES, the Arnoldi basis vectors form the trial subspace out of which the solution is constructed. One matrix–vector product is required per iteration to create each new trial vector, and the iterations are terminated based on a by-product estimate of the residual that does not require explicit construction of intermediate residual vectors or solutions – a major beneficial feature of the algorithm. GMRES has a residual minimization property in the Euclidean norm (easily adaptable to any inner-product norm) but requires the storage of all previous Arnoldi basis vectors. Full restarts, seeded restarts, and moving fixed-sized windows of Arnoldi basis vectors are all options for fixed-storage versions. Full restart is simple and historically the most popular, though seeded restarts show promise. The Bi-Conjugate Gradient STABilized (BiCGSTAB) [179] and Transpose-free Quasi Minimal Residual (TFQMR) [60] methods are Lanczos-based alternatives to GMRES for non-symmetric problems. In neither method are the Lanczos basis vectors normalized and two matrix-vector products are required per iteration. However, these methods enjoy a short recursion relation, so there is no requirement to store many Lanczos basis vectors. These methods do not guarantee monotonically decreasing residuals.

We refer to [11,13,84,157] for more details on Krylov methods, and for preconditioning for linear problems. We also call attention to the delightful paper [133], which shows that there is no universal ranking possible for iterative methods for non-symmetric linear problems. Each of the major candidate methods finishes first, last, and in the middle of the pack over the span of a few insight-provoking examples.

As a result of studies in [107,122], we tend to use GMRES (and its variants) almost exclusively with JFNK. The resulting pressure on memory has put an increased emphasis on quality preconditioning. We believe that it is only through effective preconditioning that JFNK is feasible on large-scale problems. It is in the preconditioner that one achieves algorithmic scaling and also in the preconditioner that one may stand to lose the natural excellent parallel scaling [163] enjoyed by all other components of the JFNK algorithm as applied to PDEs. For this reason we focus our main attention in this review on innovations in preconditioning.

2.3. Jacobian-free Newton–Krylov methods

The origins of the Jacobian-Free Newton–Krylov method can be traced back to publications motivated by the solution of ODEs [25,62] and publications motivated by the solution of PDEs [27,47]. The primary motivation in all cases appears to be the ability to perform a Newton iteration without forming the Jacobian. Within the ODE community these methods helped to promote the use of higher-order implicit integration. The studies on PDE problems focused on the use of nonlinear preconditioning, preconditioners constructed from linear parts of the PDEs, and the addition of globalization methods.

In the JFNK approach, a Krylov method is used to solve the linear system of equations given by Eq. (2). An initial linear residual, $\mathbf{r}_0$, is defined, given an initial guess, $\delta\mathbf{u}_0$, for the Newton correction,

$$\mathbf{r}_0 = -\mathbf{F}(\mathbf{u}) - \mathbf{J}\delta\mathbf{u}_0. \tag{8}$$

Note that the nonlinear iteration index, k, has been dropped. This is because the Krylov iteration is performed at a fixed k. Let j be the Krylov iteration index. Since the Krylov solution is a Newton correction, and since a locally optimal move was just made in the direction of the previous Newton correction, the initial iterate for the Krylov iteration for $\delta\mathbf{u}_0$ is typically zero. This is asymptotically a reasonable guess in the Newton context, as the converged value for $\delta\mathbf{u}$ should approach zero in late Newton iterations. The jth GMRES iteration minimizes $\|\mathbf{J}\delta\mathbf{u}_j + \mathbf{F}(\mathbf{u})\|_2$ within a subspace of small dimension, relative to n (the number of unknowns), in a least-squares sense. $\delta\mathbf{u}_j$ is drawn from the subspace spanned by the Krylov vectors, $\{\mathbf{r}_0, \mathbf{J}\mathbf{r}_0, (\mathbf{J})^2\mathbf{r}_0, \dots, (\mathbf{J})^{j-1}\mathbf{r}_0\}$, and can be written as

$$\delta\mathbf{u}_j = \delta\mathbf{u}_0 + \sum_{i=0}^{j-1} \beta_i (\mathbf{J})^i \mathbf{r}_0, \tag{9}$$

where the scalars β_i minimize the residual. (In practice, $\delta\mathbf{u}_j$ is determined as a linear combination of the orthonormal Arnoldi vectors produced by GMRES.)

Upon examining Eq. (9) we see that GMRES requires the action of the Jacobian only in the form of matrix–vector products, which may be approximated by [27,47]:

$$\mathbf{J}\mathbf{v} \approx [\mathbf{F}(\mathbf{u} + \epsilon\mathbf{v}) - \mathbf{F}(\mathbf{u})]/\epsilon, \tag{10}$$

where ϵ is a small perturbation.

Eq. (10) is simply a first-order Taylor series expansion approximation to the Jacobian, $\mathbf{J}$, times a vector, $\mathbf{v}$. For illustration, consider the two coupled nonlinear equations $F_1(u_1, u_2) = 0$, $F_2(u_1, u_2) = 0$. The Jacobian matrix is

$$\mathbf{J} = \begin{bmatrix} \frac{\partial F_1}{\partial u_1} & \frac{\partial F_1}{\partial u_2} \\ \frac{\partial F_2}{\partial u_1} & \frac{\partial F_2}{\partial u_2} \end{bmatrix}.$$

JFNK does not require the formation of this matrix; we instead form a result vector that approximates this matrix multiplied by a vector. Working backwards from Eq. (10), we have

$$\frac{\mathbf{F}(\mathbf{u} + \epsilon\mathbf{v}) - \mathbf{F}(\mathbf{u})}{\epsilon} = \begin{pmatrix} \frac{F_1(u_1+\epsilon v_1, u_2+\epsilon v_2) - F_1(u_1,u_2)}{\epsilon} \\ \frac{F_2(u_1+\epsilon v_1, u_2+\epsilon v_2) - F_2(u_1,u_2)}{\epsilon} \end{pmatrix}.$$

Approximating $\mathbf{F}(\mathbf{u} + \epsilon\mathbf{v})$ with a first-order Taylor series expansion about $\mathbf{u}$, we have

$$\frac{\mathbf{F}(\mathbf{u} + \epsilon\mathbf{v}) - \mathbf{F}(\mathbf{u})}{\epsilon} \approx \begin{pmatrix} \frac{F_1(u_1,u_2) + \epsilon v_1 \frac{\partial F_1}{\partial u_1} + \epsilon v_2 \frac{\partial F_1}{\partial u_2} - F_1(u_1,u_2)}{\epsilon} \\ \frac{F_2(u_1,u_2) + \epsilon v_1 \frac{\partial F_2}{\partial u_1} + \epsilon v_2 \frac{\partial F_2}{\partial u_2} - F_2(u_1,u_2)}{\epsilon} \end{pmatrix},$$

which simplifies

$$\begin{pmatrix} v_1 \frac{\partial F_1}{\partial u_1} + v_2 \frac{\partial F_1}{\partial u_2} \\ v_1 \frac{\partial F_2}{\partial u_1} + v_2 \frac{\partial F_2}{\partial u_2} \end{pmatrix} = \mathbf{J}\mathbf{v}.$$

The error in this approximation is proportional to ϵ. This matrix-free approach has many advantages. The most attractive is Newton-like nonlinear convergence without the costs of *forming* or *storing* the true Jacobian. In practice one forms a matrix (or set of matrices) for preconditioning purposes, so we eschew the common description of this family of methods as fully "matrix-free." However, the matrices employed in preconditioning can be simpler than the true Jacobian of the problem, so the algorithm is properly said to be "Jacobian-free." We briefly discuss options for matrix-free (or nearly matrix-free) preconditioning in Section 3.5. A convergence theory has been developed for JFNK in [24,28]. Conditions are provided on the size of ϵ that guarantee local convergence. Issues of global convergence are studied in [27]. For further mathematical discussion, one can consult this subject [84]. Issues regarding convergence are often raised with Newton-based methods and Jacobian-free Newton–Krylov methods are no exception. Two specific situations known to cause convergence problems for JFNK are sharp nonlinear solution structure such as a shock or a reaction front, and discontinuities in the nonlinear function such as one might see in higher-order monotone advection schemes. Issues of non-convergence tend to be seen more in boundary value problems and less in initial value problems.

2.3.1. The Jacobian-vector product approximation

As shown above, the Jacobian-vector product approximation is based on a Taylor series expansion. Here, we discuss various options for choosing the perturbation parameter, ϵ in Eq. (10), which is obviously sensitive to scaling, given **u** and **v**. If ϵ is too large, the derivative is poorly approximated and if it is too small the result of the finite difference is contaminated by floating-point roundoff error. The best ϵ to use for a scalar finite-difference of a single argument can be accurately optimized as a balance of these two quantifiable trade-offs. However, the choice of ϵ for the vector finite difference is as much of an art as a science. For well-scaled single-component PDEs, the choice of ϵ is not challenging. In [47] ϵ was set equal to something larger than the square root of machine epsilon (ϵ_{mach}).

Other approaches for choosing ϵ have their roots in algorithms used to compute individual elements of the Jacobian or columns of the Jacobian. A simple choice of ϵ is

$$\epsilon = \frac{1}{n\|\mathbf{v}\|_2} \sum_{i=1}^{n} b|u_i| + b, \tag{11}$$

where n is the linear system dimension and b is a constant whose magnitude is within a few orders of magnitude of the square root of machine roundoff (typically 10^{-6} for 64-bit double precision). This approach produces the "average" ϵ that one would get if each individual element of the Jacobian was computed as

$$J_{ij} = \frac{F_i(\mathbf{u} + \epsilon_j \mathbf{e}_j) - F_i(\mathbf{u})}{\epsilon_j}, \tag{12}$$

with $\epsilon_j = bu_j + b$. The vector $\mathbf{e}_j$ has all zeros and the value 1 in the jth location.

Another more sophisticated approach proposed by Brown and Saad [27] has its roots in an approach for computing columns of the Jacobian numerically as discussed in [54], and is given by

$$\epsilon = \frac{b}{\|\mathbf{v}\|_2} \max[|\mathbf{u}^{\mathrm{T}}\mathbf{v}|, \mathrm{typ}\,\mathbf{u}|\mathbf{v}|]\mathrm{sign}(\mathbf{u}^{\mathrm{T}}\mathbf{v}). \tag{13}$$

Here, typ**u** is a user-supplied "typical size" of **u**.

It is important to note that the choice of $b = \sqrt{\epsilon_{\text{mach}}}$ assumes that $\mathbf{F}(\mathbf{u})$ can be evaluated to the precision of ϵ_{mach}, which is often optimistic. When one knows that $\mathbf{F}(\mathbf{u})$ can only be evaluated to ϵ_{rel}, then $b = \sqrt{\epsilon_{\text{rel}}}$ is a good choice. When precision is known to be limited in the evaluation of $\mathbf{F}(\mathbf{u})$, then another effective formula for the evaluation of ϵ is [140]

$$\epsilon = \frac{\sqrt{(1 + \|\mathbf{u}\|)\epsilon_{\text{mach}}}}{\|\mathbf{v}\|}, \tag{14}$$

which is used in the NITSOL package [142].

We note in passing that Eq. (10) is a first-order approximation. It is straightforward to construct a second-order approximation,

$$\mathbf{Jv} \approx [\mathbf{F}(\mathbf{u} + \epsilon\mathbf{v}) - \mathbf{F}(\mathbf{u} - \epsilon\mathbf{v})]/\epsilon. \tag{15}$$

The goal of such an approach would be to reduce the number of required Newton iteration by making the Jacobian-vector product approximation more accurate. There has not been widespread use of higher-order approximations such as Eq. (10) on the types of problems covered in this survey paper. A possible disadvantage of second order is the cost of two fresh function evaluations per matrix–vector multiply, although creative ways of minimizing this cost have been developed [177].

2.3.2. Inexact Newton methods

Since the use of an iterative technique to solve Eq. (2) does not require the exact solution of the linear system, the resulting algorithm is categorized as an "inexact" Newton's method [53]. A simple inexact method results in the following convergence criteria on each linear iteration.

$$\|\mathbf{J}^k\delta\mathbf{u}^k + \mathbf{F}(\mathbf{u}^k)\|_2 < \gamma\|\mathbf{F}(\mathbf{u}^k)\|_2, \tag{16}$$

where γ, the forcing term, is a constant smaller than unity. For details of local convergence theory and the role played by the forcing term consult [53]. There may be a trade-off between the effort required to solve the linear system to a tight tolerance and the resulting required number of nonlinear iterations. Too large a value for γ results in less work for the Krylov method but more nonlinear iterations, whereas too small a value for γ results in more Krylov iterations per Newton iteration. Examples of this trade-off between total nonlinear iterations and execution time are given in [121,142]. Several strategies for optimizing the computational work with a variable "forcing term" γ are given in [57].

The forcing term and the issue of "oversolving" a Newton step has recently gained interest [164,176]. The concept of "oversolving" implies that at early Newton iterations γ is too small. Then one may obtain an accurate linear solution to an inaccurate Newton correction. This may result in a poor Newton update and degradation in the Newton convergence. In [164,176] it has been demonstrated that in some situations the Newton convergence may actually suffer if γ is too small in early Newton iterations.

2.4. Globalization

The lack of convergence robustness of Newton's method is frequently raised. In practice, globalization strategies leading from a convenient initial iterate into the ball of convergence of Newton's method around the desired root are required. For problems arising from differential equations, there are many choices. The issue of globalization is more vexing for BVPs than IVPs, where accurately following the physical transient often guarantees a good initial guess. Based on the robustness of IVP solvers, BVPs are often approached through a false time-stepping.

2.4.1. Standard approaches

Two of the more popular methods for globalization of Newton's method are the line search method and the trust region method [54,84]. Both have been used to globalize Newton–Krylov methods [27,56]. We briefly discuss both methods.

In the line search method the Newton update vector, $\delta\mathbf{u}$, is assumed to be a good direction in which to move. The question a line search algorithm asks is "how far should one move?". Thus a line search produces a scalar, s, (less than or equal to unity) which is used in

$$\mathbf{u}^{k+1} = \mathbf{u}^k + s\delta\mathbf{u}^k. \tag{17}$$

The simplest test used to select s is to require a decrease in the nonlinear residual

$$\mathbf{F}(\mathbf{u}^k + s\delta\mathbf{u}^k) < \mathbf{F}(\mathbf{u}^k). \tag{18}$$

A simple search that tries $s = 1, 0.5, 0.25, \ldots$ until the above criterion is met is shown to work in [84] where it is referred to as algorithm 8.1.1. More sophisticated line search methods replace the above simple decrease of the nonlinear residual with a required *sufficient* decrease. For more details and background the reader should consult [84].

The basic concept of the trust region approach is different from a line search in an important way. Here, one is no longer constrained to move along the original Newton direction, $\delta\mathbf{u}$. The philosophy is that if the Newton update is not acceptable then the Newton direction should be questioned. One is allowed to search for a solution inside a region where the linear model is "trusted" to represent the nonlinear residual well,

$$\mathbf{F}(\mathbf{u}^k) + \mathbf{J}(\mathbf{u}^k)\delta\mathbf{u}^k \approx \mathbf{F}(\mathbf{u}^k + \delta\mathbf{u}^k). \tag{19}$$

$\delta\mathbf{u}$ is typically selected as a linear combination of two or more candidate steps, one of which is an approximation to the Newton correction. We will not go into details here but such globalization methods have been used with JFNK in [27] and are presented in detail in [54].

Also it is interesting to note that the more crude, but often successful "damping on percentage change" [109,189] can be viewed as a physics-based line search.

2.4.2. Pseudo-transient continuation

Pseudo-transient continuation solves the steady-state problem $\mathbf{F}(\mathbf{u}) = 0$, for which a solution is presumed to exist, through a series of problems

$$\mathbf{f}_\ell(u) \equiv \frac{\mathbf{u} - \mathbf{u}^{\ell-1}}{\tau^\ell} + \mathbf{F}(\mathbf{u}) = 0, \quad \ell = 1, 2, \ldots, \tag{20}$$

which are derived from a method-of-lines model

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{F}(\mathbf{u}),$$

each of which is solved (in some cases approximately) for $\mathbf{u}^\ell$. The physical transient is followed when the timestep τ^ℓ is sufficiently small, and the problems at each timestep are well solved. Furthermore, the Jacobians associated with $\mathbf{f}_\ell(\mathbf{u}) = 0$ are well conditioned when τ^ℓ is small. See [59] for an analysis of this effect based on the spectrum of the preconditioned operator in the case of the constant coefficient heat equation.

τ^ℓ is advanced from $\tau^0 \ll 1$ to $\tau^\ell \to \infty$ as $\ell \to \infty$, so that $\mathbf{u}^\ell$ approaches the root of $\mathbf{F}(\mathbf{u}) = 0$. We emphasize that pseudo-transient continuation does *not* require reduction in $\|\mathbf{F}(\mathbf{u}^\ell)\|$ at each step, as do typical linesearch or trust region globalization strategies [54]; it can "climb hills." Thus the pseudo-transient method can escape local minima in a function while searching for its root.

A time-step selection scheme is required to complete the algorithm. One choice is successive evolution–relaxation (SER) [132], which lets the time step grow in inverse proportion to residual norm progress:

$$\tau^\ell = \tau^{\ell-1} \cdot \frac{\|\mathbf{F}(\mathbf{u}^{\ell-2})\|}{\|\mathbf{F}(\mathbf{u}^{\ell-1})\|}. \tag{21}$$

Alternatively, a temporal truncation error strategy bounds the maximum temporal truncation error in each individual component, based on a local estimate for the leading term of the the error. (The idea is not to control the error, per se, but to control the stepsize through its relationship to the error.) Another approach sets target maximum magnitudes for change in each component of the state vector and adjusts the time step so as to bring the change to the target. All such devices are "clipped" into a range about the current time step in practice. Typically, the time step is not allowed to more than double in a favorably converging situation, or to be reduced by more than an order of magnitude in an unfavorable one [85].

The theory for pseudo-transient continuation has recently been extended to index-1 differential-algebraic equations [48], in which not all of the equations possess a time derivative term. This is relevant for systems of PDEs in which temporal evolution takes place on a manifold of constraints, such as incompressibility in Navier–Stokes.

2.4.3. Continuation methods

In addition to pseudo-transient continuation, there are two other important types of continuation in the literature of numerical solutions for nonlinear BVPs, namely, continuation in a physical parameter of the problem, and mesh sequencing, which is continuation in a discretization parameter – namely a scale for the mesh spacing.

Physical parameters often provide "knobs" by which the nonlinearity in a problem can be varied. An easily understood example from computational fluid dynamics is the Reynolds number, which directly multiplies the convective terms of Navier–Stokes, but there are many other examples including body forcings and boundary forcings. The solution of $\mathbf{F}(\mathbf{u}, \pi^\ell) = 0$, where π^ℓ is such a parameter, can be implicitly defined as $\mathbf{u}(\pi^\ell)$.

We suppose that $\mathbf{F}(\mathbf{u}, \pi^0) = 0$ is "easy" to solve; for instance, it may be linear in $\mathbf{u}$, as when π is a Reynolds number and the governing equations reduce to the Stokes subset. Given $\mathbf{u}^{\ell-1}$ corresponding to $\pi^{\ell-1}$, we can posit a good initial guess for $\mathbf{u}^{\ell}$ at a nearby π^{ℓ} from the Taylor expansion

$$\mathbf{u}^{\ell,0} = \mathbf{u}(\pi^{\ell-1}) + \left(\frac{\partial \mathbf{u}}{\partial \pi}\right)^{\ell-1} (\pi^{\ell} - \pi^{\ell-1}). \tag{22}$$

Implicitly differentiating $\mathbf{F}(\mathbf{u}, \pi) = 0$ with respect to π gives

$$\left(\frac{\partial \mathbf{F}}{\partial \mathbf{u}}\right)\left(\frac{\partial \mathbf{u}}{\partial \pi}\right) + \left(\frac{\partial \mathbf{F}}{\partial \pi}\right) = 0 \tag{23}$$

or

$$\left(\frac{\partial \mathbf{u}}{\partial \pi}\right) = -\left(\frac{\partial \mathbf{F}}{\partial \mathbf{u}}\right)^{-1}\left(\frac{\partial \mathbf{F}}{\partial \pi}\right), \tag{24}$$

whence the right-hand side of (22) can be evaluated. This presumes that one is able to readily solve linear systems with the Jacobian, $\partial \mathbf{F}/\partial \mathbf{u}$; otherwise, poorer approximations are possible, including the simple "bootstrapping" procedure of using just $\mathbf{u}(\pi^{\ell-1})$, itself, for $\mathbf{u}^{\ell,0}$. The above approach becomes complicated near a bifurcation point, where $\partial \mathbf{F}/\partial \mathbf{u}$ can become ill-conditioned and more sophisticated approaches are required. A nice survey paper on continuation methods is [1].

Mesh sequencing is useful when a nonlinear problem is easier to solve on a coarser grid than the one on which the solution is ultimately desired, either because the nonlinearity, itself, is milder or because the linear conditioning of the sequence of nonlinear correction problems is milder. An initial iterate for the next finer mesh is constructed by interpolation from the solution on the preceding coarser mesh. Asymptotically, under certain assumptions that are natural when the discretization ultimately becomes fine enough to accurately resolve the continuous statement of the BVP, it can be shown that the initial interpolant lies in the domain of convergence of Newton's method [165] on the finer grid. Unfortunately, it is usually not easy to determine when this asymptotic range is reached. Consequently, another continuation method, such as pseudo-transient, may be used to drive the initial interpolant towards the Newton domain on each mesh step. Such nested continuation methods are often required in practice on highly nonlinear problems, such as detailed kinetics combustion [105,166], or the tokamak edge plasma equations [104]. Since a decreasing number of inner continuation steps are required on the finer meshes, the nested approach can be economical.

3. Preconditioning of the JFNK method

The purpose of preconditioning the JFNK method is to reduce the number of GMRES (Krylov) iterations, as manifested (in the GMRES convergence theory; see [158]) by efficiently clustering eigenvalues of the iteration matrix. Traditionally, for linear problems, one chooses a few iterations of a simple iterative method (applied to the system matrix) as a preconditioner. A goal of the JFNK approach is to avoid forming the system matrix $\mathbf{J}$, and , as will be shown, an effective preconditioner for JFNK can typically be simpler than the strict Jacobian of the system.

A linear preconditioner can be applied on the left (rescaling the matrix rows and the right-hand side) or on the right (rescaling the matrix columns and the solution vector), or on both, if suitably factored. Since left preconditioning changes the norm of the residual by which convergence to a linear iterative method is generally measured, right preconditioning is often preferred in comparing the intrinsic merit of different preconditioning strategies. However, in the Newton context, left preconditioning is also used, since the preconditioned residual serves as a useful estimate of the size of the Newton correction, itself, when the preconditioning is of high quality. Either strategy, left or right preconditioning, may be employed in a Jacobian-free context, and there are pros and cons to both.

Using right preconditioning, one solves

$$(\mathbf{J}\mathbf{P}^{-1})(\mathbf{P}\delta\mathbf{u}) = -\mathbf{F}(\mathbf{u}). \tag{25}$$

$\mathbf{P}$ symbolically represents the preconditioning matrix (or process) and $\mathbf{P}^{-1}$ the inverse of preconditioning matrix. Right preconditioning is actually realized through a two-step process. First solve

$$(\mathbf{J}\mathbf{P}^{-1})\mathbf{w} = -\mathbf{F}(\mathbf{u}), \tag{26}$$

for $\mathbf{w}$. Then solve

$$\delta\mathbf{u} = \mathbf{P}^{-1}\mathbf{w}, \tag{27}$$

for $\delta\mathbf{u}$. Thus, while we may refer to the matrix $\mathbf{P}$, operationally the algorithm only requires the action of $\mathbf{P}^{-1}$ on a vector. Note that if a distributed or segregated approach is used for preconditioning, then $\mathbf{P}^{-1}$ may be formed as a linear combination of approximate inverses of submatrices. An example is the additive Schwarz method of Section 3.2. The right-preconditioned version of Eq. (10) is:

$$\mathbf{J}\mathbf{P}^{-1}\mathbf{v} \approx [\mathbf{F}(\mathbf{u} + \epsilon\mathbf{P}^{-1}\mathbf{v}) - \mathbf{F}(\mathbf{u})]/\epsilon. \tag{28}$$

This operation is done once per GMRES iteration, and is actually done in two steps:
1. Preconditioning: Solve (approximately) for $\mathbf{y}$ in $\mathbf{P}\mathbf{y} = \mathbf{v}$.
2. Perform matrix-free product $\mathbf{J}\mathbf{y} \approx [\mathbf{F}(\mathbf{u} + \epsilon\mathbf{y}) - \mathbf{F}(\mathbf{u})]/\epsilon$.

Only the matrix elements required for the action of $\mathbf{P}^{-1}$ are formed. There are two primary choices to be made:
1. What linearization should be used to form the matrices required in $\mathbf{P}^{-1}$? (A new decision facing the user of a Jacobian-free Newton–Krylov method.)
2. What linear iterative method should be used for $\mathbf{y} = \mathbf{P}^{-1}\mathbf{v}$? (A standard decision facing the user of a Krylov method.)

The following sections focus on specific issues. In practice, many preconditioning approaches use a combination of these ideas.

3.1. Standard approaches

In systems where forming the Jacobian is a dominant cost of the Newton step, one may employ a "stale" (or frozen) Jacobian from an earlier step in the preconditioner while obtaining the action of the current Jacobian in the JFNK matrix–vector multiply [27,103–105]. This is referred to as "MFNK" in [104]. This approach is not truly Jacobian-free since *some* true Jacobians are formed and stored. However, this usually expensive task is not done every Newton iteration. This is different from a traditional modified Newton–Krylov (MNK) method in which the actual matrix approximating the local tangent hyperplane in Newton's method (not just its preconditioner) is held constant over several Newton iterations. The MNK approach has much weaker nonlinear convergence properties. The Jacobian-free method "feels" the true Jacobian (to within finite difference truncation error) at each iteration.

In BVPs, incomplete lower-upper (ILU) factorizations [157] have been frequently employed when approximately inverting Jacobian matrices in the preconditioner. For systems of equations characterized by tight intra-equation coupling, a blocked ILU factorization may be more effective than a standard "point"

ILU factorization preconditioner [81,123]. Here the degrees of freedom defined at a common point are interlaced and a full factorization (usually dense for systems typical of Navier–Stokes equations or magnetohydrodynamics equations, with a dozen or fewer independent fields) is performed amongst them. The overall factorization is incomplete above the block level, with fill-in limited between degrees of freedom defined at different points.

In systems of conservation laws in which convection dominates, high-order convection schemes are desired for accuracy. Using the Jacobian-free method, one can construct the preconditioner from a low-order upwinded discretization that is more stable with respect to incomplete factorization, saving memory and often resulting in more effective preconditioning [88,95,123]. Convergence of the nonlinear system occurs to the higher-order discretization represented in the right-hand side residual. Operationally, this split-discretization Jacobian-free preconditioned product is

$$\mathbf{J}\mathbf{P}^{-1}\mathbf{v} \approx \frac{\mathbf{F}_{\text{high}}(\mathbf{u} + \epsilon\mathbf{P}_{\text{low}}^{-1}\mathbf{v}) - \mathbf{F}_{\text{high}}(\mathbf{u})}{\epsilon}. \tag{29}$$

Here, $\mathbf{F}_{\text{high}}(\mathbf{u})$ denotes the nonlinear function evaluated with a high-order discretization and $\mathbf{P}_{\text{low}}^{-1}$ denotes a preconditioning operator formed with a low-order discretization.

3.2. Newton–Krylov–Schwarz

Newton–Krylov–Schwarz (NKS) is a preconditioned Jacobian-free Newton–Krylov method in which the action of the preconditioner is composed from those of preconditioners defined on individual geometric subdomains. Historically, the primary motivation for NKS (first called by this name in [34]) is parallel processing through divide-and-conquer. Scalability studies based on dimensionless ratios of communication and computation parameters for message-passing aspects of Schwarz-type iterative methods appeared in [89,91]. Recently, a sequential (i.e., non-parallel) motivation for Schwarz methods has become apparent [187]: their localized working sets can be sized to fit in the Level-2 caches of contemporary microprocessors. Furthermore, multilevel Schwarz, algebraically similar to the AFAC form of multigrid [120], can have bounded condition number in the asymptotic limit of finely resolved meshes and is therefore an "optimal" method from the perspective of convergence rate.

If we decompose the domain of a PDE problem into a set of possibly overlapping subdomains Ω_i, the standard additive Schwarz preconditioner can be expressed as

$$\mathbf{P}_{\text{ASM}}^{-1} = \sum_i R_i^{\text{T}}\mathbf{J}_i^{-1}R_i, \tag{30}$$

where the three-phase solution process (reading operators from right to left) consists of first collecting data from the local and neighboring subdomains via global-to-local restriction operators R_i, then performing a local linear solve on each subdomain $\mathbf{J}_i^{-1}$, and finally sending partial solutions to the local and neighboring subdomains via the local-to-global prolongation operators R_i^{T}. The solve with the local Jacobian can be replaced with an approximate solve, such as a local incomplete factorization or a multigrid sweep.

While the three phases are sequential and synchronized by communication requirements, each term in the sum can be computed concurrently, leading to parallelism proportional to the number of subdomains. This is in contrast with a global incomplete factorization, whose concurrency is determined by the discretization stencil and the matrix ordering and cannot be scaled to an arbitrary number of processors.

Parallel experience with NKS methods is growing. We mention the shared-memory implementation of [124] and the distributed-memory implementations of [33,37,83]. Domain-based parallelism is recognized as the form of data parallelism that most effectively exploits contemporary microprocessors with multi-level memory hierarchy [49,187]. Schwarz-type domain decomposition methods have been extensively developed for finite difference/element/volume PDE discretizations over the past decade, as reported in the annual proceedings of the international conferences on domain decomposition methods, of which the most recent volume is [61].

In practice, we advocate the *restricted* additive Schwarz method (RASM), which eliminates interprocess communication during either the restriction or prolongation phase of the additive Schwarz technique [39]. One version of the RASM preconditioner can be expressed in operator notation as

$$\mathbf{P}_{\mathrm{RASM}}^{-1} = \sum_i R_i'^{\mathrm{T}} \mathbf{J}_i^{-1} R_i. \tag{31}$$

It performs a complete restriction operation but does not use any communication during the interpolation phase, $R_i'^{\mathrm{T}}$. This provides the obvious benefit of a 50% reduction in nearest-neighbor communication overhead. In addition, experimentally, it preconditions better than the original additive Schwarz method over a broad class of problems [39], for reasons that are beginning to be understood in the function space theory that underlies Schwarz methodology [30].

As originally introduced in [55], additive Schwarz preconditioning includes a coarse grid term in the sum (30). Indeed, the coarse grid is essential for optimal conditioning in the scalar elliptic case. Table 1 shows the successive improvements towards optimality of a hierarchy of methods, all of which fit within the additive Schwarz algebraic framework, Eq. (30). The most primitive is point Jacobi, in which each subdomain is one point and there is no overlap. Subdomain Jacobi clusters all the points in one subdomain into a single subdomain solve, which is performed concurrently within each subdomain, with no overlap. One-level additive Schwarz has the same concurrency as Jacobi, except that the subdomains overlap, and nontrivial communication is required to set up the subproblems. To achieve the mesh-independent estimates shown (iterations depending only upon the number of processors or subdomains), some operator-dependent, not very severe in practice, assumptions need to be made about the extent of overlap. Finally, the Schwarz preconditioner supplemented with a low-dimensional but global coarse grid problem (two-level) achieves independence of the number of subdomains, at the price of an increasingly complex problem linking the subdomains.

NKS methods have been developed and studied by Cai and collaborators [31–33,37], Pernice and collaborators [143], Tidriri [169,171], and Knoll and collaborators [105,107,123], among many others. The combination of pseudo-transient continuation and NKS has been called ΨNKS, and is discussed in [69].

3.3. Multigrid approaches

There has been considerable success in applying the multigrid method as a preconditioner to Krylov methods on linear problems [5,100,125,135]. As a result of this success, JFNK researchers have begun to consider the performance of linear multigrid as a preconditioner to a Jacobian-free Newton–Krylov

Table 1
Iteration count scaling of Schwarz-preconditioned Krylov methods, translated from the theory in terms of mesh spacing and subdomain diameter into the corresponding quantities of discrete problem size N and processor number P, assuming quasi-uniform grid, quasi-unit aspect ratio grid and decomposition, and quasi-isotropic elliptic operator

Preconditioning	Iteration count	
	2D	3D
Point Jacobi	$\mathcal{O}(N^{1/2})$	$\mathcal{O}(N^{1/3})$
Subdomain Jacobi	$\mathcal{O}((NP)^{1/4})$	$\mathcal{O}((NP)^{1/6})$
One-level additive Schwarz	$\mathcal{O}(P^{1/2})$	$\mathcal{O}(P^{1/3})$
Two-level additive Schwarz	$\mathcal{O}(1)$	$\mathcal{O}(1)$

method [29,41,82,108,110,141,153]. In the multigrid preconditioned Newton–Krylov method (NKMG), the system $\mathbf{y} = \mathbf{P}^{-1}\mathbf{v}$, in Eq. (28), is approximately solved for $\mathbf{y}$ using a linear multigrid algorithm.

Whereas the primary motivation for Schwarz-type preconditioning is concurrency, the primary motivation in NKMG is optimal operation complexity. By this we mean a preconditioner that not only renders the number of preconditioned Krylov iterations per Newton iteration independent of grid resolution, but imposes a cost per iteration that grows only linearly in the number of discrete unknowns. NKMG has also been quite effectively implemented in parallel [29,141].

The basic building blocks of a multigrid algorithm are the mesh interpolation operators, restriction $\mathscr{R}$ and prolongation $\mathscr{P}$, and a method of constructing the coarse grid operators. In the limit (for nested multilevel methods), the "perfect" coarse grid operator is the Schur complement of the fine grid operator with the degrees of freedom not represented on the coarse grid eliminated. The corresponding perfect restriction is the elimination step and the corresponding prolongation the backsolve. In practice, one uses much less expensive grid transfer operators, such as multilinear interpolation, or even simple injection restriction and piecewise constant prolongation. In [5,135] it is shown on some challenging problems that multigrid as a preconditioner may outperform multigrid as a solver, and in general it is also more robust, due to the outer Krylov method. In [100] it is shown that a suboptimal multigrid method, which is not a scalable solver as a result of overly simplified restriction and prolongation operators, can produce a scalable method when used as a preconditioner.

In [110] the restriction and prolongation operators are piecewise constant and piecewise linear. Since the systems considered there contain second-order operators, the choice of $\mathscr{R}$ and $\mathscr{P}$ as piecewise constant violates the level transfer "order rule", $m_{\mathscr{P}} + m_{\mathscr{R}} > 2$ [188]. Here $m_{\mathscr{P}}$ and $m_{\mathscr{R}}$ are the order of interpolation plus one for the prolongation and restriction operators, respectively. Thus, this approach cannot be considered an optimal multigrid method. In [110] it is demonstrated that multigrid methods make excellent preconditioners for JFNK, superior to comparably complex SGS and ILU, with a typical approximate inverse being only one V-cycle. It is also demonstrated that the algorithmic simplifications which may result in loss of convergence for multigrid as a solver (such as piecewise constant prolongation in place of piecewise linear prolongation) have a much weaker effect when multigrid is the preconditioner.

In [108] two different approaches for defining the coarse grid approximations to the preconditioner are considered. The first approach is to restrict the dependent variables ($\mathbf{u}$) down through a series of grids, rediscretize the equations $\mathbf{F}(\mathbf{u})$, and then form each of the preconditioner elements independently. This may be troublesome for multi-scale nonlinear physics and/or nonlinear discretizations, in addition to the fact that theoretically this is not an optimal multigrid method. The second method is to build the coarse grid operators using an additive correction [78] procedure, which can also be viewed as a Galerkin, or variational, approach [188]. Here, the coarse grid operator, $\mathbf{P}_\mathrm{c}$ is constructed from the fine grid operator, $\mathbf{P}_\mathrm{f}$ as:

$$\mathbf{P}_\mathrm{c} = \mathscr{R} * \mathbf{P}_\mathrm{f} * \mathscr{P}. \tag{32}$$

When $\mathscr{R}$ and $\mathscr{P}$ are piecewise constant, this should be viewed as an additive correction multigrid method [78,161]. This is attractive for complex physics codes and/or unstructured grids since no discretizations on the coarse grids are required. Details of this specific multigrid approach, used as a preconditioner to GMRES, on a scalar problem can be found in [100]. In [108] both approaches of forming the coarse grid operators were found to give good algorithmic performance.

Recently, Mavripilis [118] has considered nonlinear multigrid as a preconditioner to JFNK with encouraging results. The steady-state Navier–Stokes equations and a time-dependent reaction diffusion problem were considered, both on unstructured grids. While FAS as a preconditioner generally outperformed FAS as a solver in terms of CPU time, the overall winner was consistently linear multigrid as a preconditioner to JFNK.

3.4. Physics-based preconditioning

An important new class of preconditioners for the Jacobian-free Newton–Krylov method is referred to as physics-based or PDE-based. The motivation behind this approach is that there exist numerous, legacy algorithms to solve nonlinear systems, both IVPs and BVPs. These algorithms typically were developed with some insight into the time scales or physical behavior of the problem. As a benefit of this insight, a reduced implicit system, or a sequence of segregated explicit or implicit systems may be solved in place of the fully coupled system. Examples include the semi-implicit method for low-speed flow [74], the SIMPLE algorithm for incompressible flow [138], Gummel's method for the semiconductor drift-diffusion equations [70], and numerous other structure-based operator splitting methods for reaction-diffusion systems [20,136].

The SIMPLE algorithm [138] is a classical segregated solution method in computational fluid dynamics. Its use as a preconditioner to a Jacobian-free Newton–Krylov method is demonstrated in [141]. One employs multiple iterations of the SIMPLE algorithm in "delta form" as the preconditioner. By "delta form" we mean that the linear system $\mathbf{Pu} = \mathbf{b}$ is solved for $\delta\mathbf{u}$, i.e., $\mathbf{P}\delta\mathbf{u} = \mathbf{b} - \mathbf{Pu}_0$ ($\mathbf{y} = \mathbf{P}^{-1}\mathbf{v}$ in Eq. (28)) and $\mathbf{u} = \mathbf{u}_0 + \delta\mathbf{u}$. This is frequently refered to as residual form in the numerical linear algebra community. In this paper we attempt to reserve the use of the word residual to describe the right-hand side of the Newton problem. It is necessary to cast the preconditioner in "delta form" since this is the form of the problem upon which the outer Newton–Krylov iteration operates. Split, or segregated, methods are employed as preconditioners for Newton–Krylov on a system of time-dependent reaction diffusion equations [130], time-dependent MHD equations [43,44], and steady-state incompressible Navier–Stokes equations [108,141], and time-dependent incompressible Navier–Stokes equations [113,141]. Also in [113], a standard approximate linearization method used for phase-change heat conduction problems, is employed as a preconditioner for a JFNK solution of phase-change heat conduction problems. In this section we present detail on constructing preconditioners for stiff-wave systems using the semi-implicit method and constructing preconditioners using structure-based operator splitting.

3.4.1. Stiff wave systems

To demonstrate how to construct a physics-based preconditioner for a stiff wave system, consider the 1D shallow water wave equations with a stiff gravity wave (a hyperbolic system):

$$\frac{\partial h}{\partial t} + \frac{\partial uh}{\partial x} = 0, \tag{33}$$

$$\frac{\partial uh}{\partial t} + \frac{\partial u^2 h}{\partial x} = -gh\frac{\partial h}{\partial x}. \tag{34}$$

Here u is the fluid velocity, h is the hydrostatic pressure, x is the spatial coordinate, t is time, g is gravity, and $\sqrt{gh}$ is the fast wave speed. The fast wave time scale is the time scale we wish to "step over" in evolving the mesoscale dynamics of interest. A semi-implicit method is constructed by linearizing and implicitly discretizing only those terms which contribute to the stiff gravity wave. Here, advection in the momentum equation is time split and done explicitly. Thus, some physics insight is required to produce the implicit system. With $n+1$ as new time and n as old time, and suppressing spatial discretization, we have

$$\frac{h^{n+1} - h^n}{\Delta t} + \frac{\partial (uh)^{n+1}}{\partial x} = 0, \tag{35}$$

$$\frac{(uh)^{n+1} - (uh)^n}{\Delta t} + \frac{\partial (u^2 h)^n}{\partial x} + gh^n\frac{\partial h^{n+1}}{\partial x} = 0. \tag{36}$$

Note that the nonlinear term in h is linearized by evaluating the square of the wave speed (gh) at old time. We evaluate $\partial(u^2h)/\partial x$ at old time since it does not contribute to the *linearized* gravity wave.

We rearrange the momentum equation as

$$(uh)^{n+1} = -\Delta t g h^n \frac{\partial h^{n+1}}{\partial x} + S^n \quad \left(S^n = (uh)^n - \Delta t \frac{\partial (u^2h)^n}{\partial x} \right). \tag{37}$$

Eq. (37) is then substituted into Eq. (35) to give the following scalar parabolic equation:

$$\frac{h^{n+1} - h^n}{\Delta t} - \frac{\partial}{\partial x}\left(\Delta t g h^n \frac{\partial (h)^{n+1}}{\partial x} \right) = \frac{\partial S^n}{\partial x}. \tag{38}$$

Eq. (38) is solved for h^{n+1}, and then one can easily solve for $(uh)^{n+1}$ using Eq. (37). Again, for this simple problem the source of the linearization and time splitting is the linearized wave speed (a time discretization error) and the fact that advection in Eq. (36) is at a different time level. The innovation of physics-based preconditioning is realizing that the linearized time split solution of Eqs. (37) and (38) can be used as the preconditioner to an implicitly balanced [98] Newton–Krylov solution of Eqs. (33) and (34). This is possible since JFNK does not require the formation of the Jacobian, and thus time-splitting approaches, such as the semi-implicit method, can be used as preconditioners. Since the action of the true operator is maintained in the evaluation of the full nonlinear residual and the forward Jacobian used in the Newton–Krylov iteration, the inverse Jacobian used in the preconditioner can be further weakened without compromise to the solution in the interest of minimizing execution time. For instance, a few cycles of a multigrid method, which is ideal for diffusion problems, can be used to approximate the solution of Eq. (38) in the preconditioner.

To be more specific, the function of a preconditioner is to map $[\mathrm{res}_h, \mathrm{res}_{uh}]$, or "$\mathbf{y}$", to $[\delta h, \delta uh]$, or "$\mathbf{v}$". Using the semi-implicit method in delta form (suppressing spatial discretization) the linearized equations are:

$$\frac{\delta h}{\Delta t} + \frac{\partial \delta uh}{\partial x} = -\mathrm{res}_h, \tag{39}$$

$$\frac{\delta uh}{\Delta t} + g h^n \frac{\partial \delta h}{\partial x} = -\mathrm{res}_{uh}. \tag{40}$$

Substituting Eq. (40) into Eq. (39), and eliminating δuh, produces

$$\frac{\delta h}{\Delta t} + \frac{\partial}{\partial x}\left(\Delta t g h^n \frac{\partial \delta h}{\partial x} \right) = -\mathrm{res}_h + \frac{\partial}{\partial x}(\Delta t \mathrm{res}_{uh}). \tag{41}$$

This parabolic equation can be approximately solved for δh. Then δuh can be evaluated:

$$\delta uh = -\Delta t g h^n \frac{\partial \delta h}{\partial x} - \mathrm{res}_{uh}. \tag{42}$$

To summarize, we use a classical semi-implicit method, to map $(\mathrm{res}_h, \mathrm{res}_{uh})$ to $(\delta h, \delta uh)$ with one approximate parabolic solve. The utility of this preconditioning approach is verified on the 2D shallow water equations including the Coriolis force in [126]. In addition, this framework has been used to develop preconditioners for MHD problems [43,44] and the compressible Euler equations [151]. In [43] a connection is made between the concept of the semi-implicit method as a preconditioner and the Schur complement of the Jacobian.

3.4.2. Structure-based preconditioning

Traditional techniques based on operator structure, which may be only marginally appealing as solvers or stationary operator splittings, may be effective and efficient preconditioners. Two structure-based techniques of particular interest are direction-based splitting and phenomenon-based splitting. To illustrate, consider a canonical system of unsteady convection–diffusion–reaction problems, symbolized by

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{R}(\mathbf{u})\mathbf{u} + \mathbf{S}(\mathbf{u})\mathbf{u} = 0.$$

$\mathbf{u}$ is a discrete gridfunction of p components per point. The following discussion assumes a standard finite volume discretization.

The operator $\mathbf{S}$, representing convection and diffusion, typically has discretization stencils that strongly couple near neighbors of the same gridfunction component but typically only weakly couple different components, except through convective velocities and perhaps through composition-dependent constitutive laws. $\mathbf{R}$, representing reaction (volumetric terms), may strongly couple different components, but typically involves only the unknowns defined at a single gridpoint. The remaining transient term is typically diagonal (or narrow banded in an unlumped finite element discretization).

The combination of the transient term and $\mathbf{R}$ is well preconditioned with a block-diagonal operator, with no spatial coupling. The combination of the transient term with $\mathbf{S}$ is well preconditioned with independent multigrid solves for each component, with no intercomponent coupling. Each of these two preconditionings is substantially less computationally complex than a block ILU preconditioning for the entire discrete operator, and it is natural to consider multiplicative or additive forms of operator splitting in which each is applied independently.

Consider a simple backward difference for the transient term in advancing through timestep δt from $\mathbf{u}$ to $\mathbf{u} + \delta\mathbf{u}$. Discretized implicitly at the advanced level and linearized about $\mathbf{u}$, we have

$$\frac{\delta \mathbf{u}}{\delta t} + \mathbf{S}(\mathbf{u})(\mathbf{u} + \delta\mathbf{u}) + \mathbf{R}(\mathbf{u})(\mathbf{u} + \delta\mathbf{u}) = 0,$$

or, in delta-form,

$$\frac{\delta \mathbf{u}}{\delta t} + \mathbf{S}(\mathbf{u})\delta\mathbf{u} + \mathbf{R}(\mathbf{u})\delta\mathbf{u} = -\mathbf{F}(\mathbf{u}).$$

where $\mathbf{F}(\mathbf{u}) \equiv \mathbf{S}(\mathbf{u})\mathbf{u} + \mathbf{R}(\mathbf{u})\mathbf{u} = 0$.

Let the linear system to be solved by Krylov iteration at a given Newton iteration be written $\mathbf{J}\delta\mathbf{u} = -\mathbf{F}$, where $\mathbf{J} = \alpha\mathbf{I} + \mathbf{S} + \mathbf{R}$ and α is the reciprocal of the timestep. Apply an operator-split preconditioner $\mathbf{P}$ to Krylov vector $\mathbf{v}$ with preconditioned output $\mathbf{v}'$ as follows:

- Phase 1, block-diagonal (reaction) coupling

$$\mathbf{v}^* \leftarrow (\alpha\mathbf{I} + \mathbf{S})^{-1}\mathbf{v}.$$

- Phase 2, segregated scalar (spatial) coupling

$$\mathbf{v}' \leftarrow (\alpha\mathbf{I} + \mathbf{R})^{-1} \cdot \alpha \cdot \mathbf{v}^*.$$

Thus, we have approximated the Jacobian inverse with

$$(\alpha\mathbf{I} + \mathbf{R})^{-1} \cdot \alpha \cdot (\alpha\mathbf{I} + \mathbf{S})^{-1},$$

which is equivalent to approximating $\mathbf{J}$ with an operator that has first-order temporal splitting error, namely with $\alpha\mathbf{I} + \mathbf{S} + \mathbf{R} + \alpha^{-1}\mathbf{SR}$. This differs from the unsplit original $\mathbf{J}$ only in the last term. When the time step is small, so is this difference.

Structure-based operator-split preconditioning has been employed in radiation transport [29,130], charge transport in semiconductors [10,86], and fluid flow [52,113,141].

Due to architectural factors in high-end computers, the operator-split preconditioners discussed herein – and perhaps several other varieties – are natural to try, to replace block preconditioners that have heavier storage and memory traffic costs. Where operator-splitting is already used as a solver, it can easily be converted into a preconditioner by putting it into delta-form and wrapping a matrix-free Newton–Krylov acceleration around it. A small number of Newton steps (two or three) cleans up the splitting error. In cases in which there are strong couplings across different components stored at different mesh points, the type of phenomenon-based operator splitting described above is not expected to be successful; hence it is probably best exploited in an adaptive way in a polyalgorithmic preconditioner.

The salient point of this subsection is that a JFNK "wrapper" can provide implicit balance [98] to an inner operator-split solver. This is true even if the operator split preconditioner does not allow one to use the large time step size that is otherwise achievable by the outer JFNK solver.

Structured grids are often still used in practice, being natural for many problems in regular geometries such as rectangular parallelipipeds or spherical annuli. In such contexts, directional splitting, of which the Alternating Direction Implicit method (or "ADI") is paradigmatic, remains a popular solver. The operator algebra is identical to that above, except that $\mathbf{R}$ may represent, for instance, x-directional splitting and $\mathbf{S}$ y-directional. The complexity advantages of ADI are obvious, application of each inverse reduces to a set of independent one-dimensional (e.g., block tridiagonal) problems. For all of the same qualitative reasons as for physics-based methods, and with only slight qualitative differences, such direction-based operator splittings may make excellent preconditioners, at least in serial computing environments. An outer Newton–Krylov iteration on each time step should quickly remove the splitting error.

Though the results of this section are developed for first-order implicit time-discretization, the real benefits of JFNK come from its ability to make higher-order implicit time discretizations worthwhile. Operationally, the only changes are in the presence of linear combinations of earlier time values of $\mathbf{u}$ on the right-hand side. As stated earlier, this was a motivation for some of the early development of JFNK.

The work of Dawson et. al. [52] on two-phase subsurface flow and that of Kerkhoven and Saad [86] on the drift-diffusion semiconductor equations are excellent examples of the use of split-based preconditioners, along with the work of Brown and Hindmarsh [26]. In [86], the outer Newton–Krylov method is regarded as the accelerator to the inner fixed point method. Understanding and quantifying the limitations of split-based preconditioners is an active area of research.

3.5. Matrix-free preconditioning approaches

As shown in the previous sections, there are numerous ways to take advantage of the matrix-free matrix–vector multiply while still forming matrices that are reduced in complexity as compared to the full Jacobian. In the "physics-based" example (Section 3.4.1), the approximate preconditioning matrix is derived from a scalar parabolic problem, while the Jacobian matrix is derived from a multi-component hyperbolic system. In the "structure-based" paradigm (Section 3.4.2), several independent systems replace a coupled system of greater overall complexity. However, preconditioner matrix objects are still formed. Storage and memory bandwidth limitations always provide a motive to investigate preconditioning approaches that do not require the formation of any matrix.

There is a continuum of choices ranging from forming no preconditioner to forming the complete Jacobian. In this section, we briefly outline a few ideas that lie closest to true matrix-free preconditioning. The only iterative method that can be implemented in a fashion that is literally matrix-free is a Krylov method. Since Krylov methods may present a different matrix polynomial approximation to the matrix inverse for every initial iterate, they have not always enjoyed a reputation as preconditioners. It is now well known, however, how to employ one Krylov method to precondition an outer Krylov method with $\mathbf{P}^{-1}$ "changing" on each outer Krylov iteration. The price is generally some extra dense vector storage, which must be traded against the cost of a sparse matrix. The Flexible GMRES [156] (FGMRES) and GMRES-R [180] methods were developed to address the issue of using a preconditioner (Krylov or otherwise) that may vary within the GMRES iteration.

These flexible accelerators open up a number of preconditioning options such as using a relaxation method preconditioner with variable termination from outer iteration to outer iteration. On some convection–diffusion problems in [156], FGMRES with GMRES as a preconditioner (a fully matrix-free option) outperformed GMRES with an ILU preconditioner. In the 1D shallow water problem previously discussed, the preconditioning matrix is symmetric and positive definite, thus we could use conjugate gradient as the preconditioner iterative method and be truly matrix-free. However, in this case, we should use FGMRES on the outside. In the JFNK context, however, where the Krylov solve is often performed inexactly throughout all but the endgame of the Newton cycle, plain GMRES is surprisingly forgiving of mildly inconsistent preconditioning.

The next step up in matrix formation is to implement a relaxation method in such a manner that only the main diagonal (or main block diagonal) need be formed. This is done in [47,149]. In [47] a nonlinear SOR method is used as a preconditioner, requiring only the diagonal. Another step up is represented by [145] where an approximate factorization of ADI type is implemented. There storage is retained for only one of the required block inverses. This matrix is re-populated and inverted several times to approximate the preconditioner inverse. In both [145,149] it is demonstrated that the price paid for this reduced storage method in the preconditioner is an increase in the execution time, as compared to the matrix counterparts.

As is justifiably touted in the multigrid community, the FAS algorithm represents a low storage multigrid algorithm. No global matrices need be formed for simple point smoothers. For block Jacobi smoothers, storage is only required for a local block diagonal. Thus, FAS can be viewed as a matrix-free preconditioner, as in [118].

Finally, we mention an idea that is often exploited in the context of problems that are dominantly elliptic. There exists a technology of "fast Poisson solvers" [76] based on Fast Fourier Transforms (FFTs) or other fast transformations (including fast biharmonic solvers [18]). The FFT and the multidimensional fast Poisson solvers (accommodating some spatially uniform, but directionally varying diffusion coefficients) that can be assembled from it require workspace equal to just a few gridfunction vectors and operation count only a log-factor higher than linear in the size of the gridfunction vector. Such FFT-based preconditioners, defined simply by subroutine calls on vectors, with no explicit matrix formation or storage whatsoever, may be highly effective in linear problems, or in nonlinear problems solved by JFNK in which the nonlinearity is a relatively controllable perturbation of one of the elliptic operators for which a fast inversion is known.

3.6. *Nonlinear preconditioning*

We briefly mention here two views of nonlinear preconditioning. Then we provide a more detailed discription of the second, more recent view. As discussed by Chan and Jackson [47], their original motivation for "nonlinear preconditioning" was storage. Since the outer JFNK method did not form or store the Jacobian they desired a preconditioning process which had the same properties. They referred to this as nonlinear preconditioning, and we have also described it as matrix-free preconditioning in the previous subsection. Within the definition of [47], the use of FAS multigrid would also be called a nonlinear preconditioner [118].

A second view of nonlinear preconditioning has evolved recently [35]. Here the goal of nonlinear preconditioning is to attack the "nonlinear stiffness" of a problem and thus improve the global convergence properties of Newton's method. In [35] the phrase "unbalanced nonlinearities" was used in place of "nonlinear stiffness". We give a brief description of "nonlinear stiffness" and then describe the nonlinear preconditioner put forth in [35].

A system of ordinary differential equations is described as "stiff" if it encompasses a wide range of time scales. The discretization of an elliptic partial differential equation on a fine grid can also be referred to as “stiff”, because the Fourier representation of the error contains a wide range of wavenumbers. We use the phrase “nonlinear stiffness” to describe a problem in which the solution structure has disparate spatial scales as a result the nonlinearity of the problem. Classic examples are shock waves and reaction fronts. The location of a front or layer may be determined by a delicate balance involving nonlinear terms. Until the location is correct, Newton’s method tends to take very small steps. The advance of the front or layer typically occurs one grid point per step, while the solution away from the controlling feature barely changes.

As discussed in Section 2.4, the lack of a global convergence theory for Newton’s method is a severe drawback that has been met in practice with a variety of inventions. Some, generally those rooted in the physics known to lie behind particular discrete nonlinear systems, are applied outside of Newton’s method and exercise their beneficial effect by changing the system or the initial iterate fed to Newton’s method. Others, generally those rooted in mathematical assumptions about the behavior of $\mathbf{F}(\mathbf{u})$ near a root, are applied inside, and have their effect by modifying the strict Newton correction before it is accepted. A new technique, the additive Schwarz preconditioned inexact Newton (or “ASPIN”), nests multiple applications of Newton’s method. ASPIN involves a (generally nonlinear) transformation of the original rootfinding problem for $\mathbf{F}(\mathbf{u})$ to a new rootfinding problem, $\mathscr{F}(\mathbf{u}) = 0$, to which an outer Jacobian-free Newton method is applied. The formation of $\mathscr{F}(\mathbf{u})$ at a given point $\mathbf{u}$, which is required many times in the course of performing the outer Jacobian-free Newton–Krylov iteration, in turn involves the solution of possibly many smaller nonlinear systems by Newton’s method.

Without such a transformation, Newton’s method may stagnate for many iterations in problems that are “nonlinearly stiff”. A classical example is transonic compressible flow with a shock. The size of the global Newton step may be limited in such a problem by high curvature in the neglected terms of the multivariate expansion of $\mathbf{F}(\mathbf{u})$ coming from just a few degrees of freedom defined near the shock. Cai and collaborators [35,36,38] devised ASPIN to concentrate nonlinear work at such strong nonlinearities, and produce a more balanced global nonlinear problem, on which Newton behaves better, with less damping.

From an algebraic viewpoint, ASPIN is a generic transformation that requires only the unique solvability of subsystems of the original $\mathbf{F}(\mathbf{u})$ in the neighborhood of the root $\mathbf{u}_*$. From a physical viewpoint, ASPIN is a family of methods in which the subsystems may be chosen by domain decomposition, segregation of equations arising from different physical phenomena, identification of nonlinear stiffness, or still other criteria. As with all Schwarz methods, many flavors of nonlinear Schwarz preconditioning are possible – additive, multiplicative, or general polynomial combination of sub-operators; single-level or multi-level; overlapping or non-overlapping.

It is shown in [35] that Newton’s method applied to a nonlinearly preconditioned version of the velocity-vorticity driven cavity problem (without standard globalization), based on domain decomposition converges rapidly (e.g., in 5–10 Newton iterations) at Reynolds numbers far beyond those at which Newton’s method applied to the original discretization of the problem hopelessly stagnates. Globalization has also been used to find solutions at high Reynolds number, but more Newton iterations are required.

It is shown in [38] that Newton convergence of the nonlinearly transformed version of the problem of shocked flow in a variable cross-section duct is much less sensitive to mesh refinement than the original discretization.

As stated, the difficulties of Newton on the driven cavity problem can be ameliorated by standard globalization methods or by continuation in Reynolds number and the shocked flow problem through mesh sequencing in other contexts. Nevertheless, it is interesting to see that a purely algebraic method, ASPIN, is effective at rebalancing nonlinearities so that Newton converges easily. We expect that it will have wide applicability in problems with complex nonlinearities as a means of increasing nonlinear robustness.

Unfortunately, it is difficult to obtain direct approximations to the dense Jacobian of the transformed system, $\mathscr{J}$, so as to improve the *linear conditioning* of the resulting Newton correction problems. Therefore, these problems are subject to linear ill-conditioning as mesh resolution increases. To conquer this linear ill-conditioning, multi-level methods of ASPIN need to be devised. The straightforward FAS multigrid approach on $\mathscr{F}(\mathbf{u})$ may not be practical since the nonlinear correction to be computed at each coarse level requires an evaluation of the fine-grid residual, which is subsequently restricted to compute the coarse-grid defect that drives the correction. Since each fine-grid residual involves a host of fine-grid nonlinear subproblems, this is expensive. An alternative multi-level method is investigated, with promising results, in [36].

4. Applications

The focus of this section is to survey uses of JFNK in various fields of computational physics. We provide some references to the use of more standard Newton-based methods, as well. The subsequent section is an enumeration of "tricks" and techniques that span individual applications, illustrated in some of the work grouped by application domain here. In these sections (and previous ones as well) we have adopted a liberal referencing strategy. All related work that we are aware of is referenced. However, in many cases, we do not discuss the specific contributions found in each publication as this would add signifcant length to the current paper.

4.1. Fluid dynamics/aerodynamics

Computational fluid dynamics has been a rich area for algorithmic development, testing, and application. In this section we can only sample the diverse literature to JFNK and computational fluid dynamics. The majority of this work has been on steady-state BVPs.

Vanka and Leaf [181] were an early advocates of Newton's method for incompressible fluid flow, as were MacArthur and Patankar [116]. Early explorations of Newton's method in compressible flow can be traced back to Venkatakrishnan [182,183]. This work is representative of the finite volume/finite difference CFD community. There has also been extensive development of nonlinearly implicit algorithms within the finite element community [65,164].

The incompressible Navier–Stokes equations have been used as a testbed for much JFNK algorithm development and testing, with emphasis on standard test problems, such as the driven cavity [64] and the natural convection cavity [51]. In fact, the driven cavity problem was considered in one of the original JFNK papers [27]. McHugh and co-workers [81,121,122] studied inexact Newton methods and mesh sequencing, the performance of various Krylov methods within JFNK, the use of low-order spatial differencing within the preconditioner, as well as block ILU compared to point ILU preconditioning. The work of Shadid and co-workers [164,176] is not JFNK, but NK. It has elucidated important issues related to inexact Newton methods, oversolving, and backtracking in a CFD context. Knoll and co-workers [108,110,113,129] studied the ideas of multigrid preconditioning in JFNK in the context of the incompressible Navier–Stokes equations. They are also among the first to consider operator-split based preconditioning. Pernice and co-workers [139,141] studied hybrid combinations of nonlinear multigrid, the SIMPLE algorithm, and JFNK. Most notably, the work in [141] applied JFNK, with a SIMPLE/multigrid preconditioner, to a 3D 512 cubed grid thermally driven flow problem on up to 512 processors.

JFNK methods have been developed and applied to the compressible Euler and Navier–Stokes equations primarily by the aerodynamics community. Newton–Krylov–Schwarz (NKS) development occupied ICASE and Boeing in the mid-1990s [33,37,88,93,170,172]. The combined impact of parallelization, pseudo-transient continuation, and NKS is documented in [68]. Nearly matrix-free preconditioning techniques have been developed for the compressible Navier–Stokes equations [145]. Mavriplis studied the use of agglomeration multigrid as a preconditioner to JFNK on unstructured grids [118]. Other JFNK applications in compressible flow include [9,63,80,134,147].

4.2. Plasma physics

Problems in plasma physics provide a rich variety of time scales and nonlinearities. These result from the free electrons in an ionized gas (plasma) and the ability of the plasma to support and propagate electrostatic and electromagnetic waves. JFNK methods have made a significant impact in computational plasma physics within the past decade, and a number of plasma physics studies have been enabled by JFNK [21,45,46,96,97,106,114,144,190,192]. We briefly discuss the work in three separate areas of plasma physics.

4.2.1. Edge plasma modeling

The edge plasma (scrape-off, boundary layer) of a tokamak fusion experiment is that region of plasma which lies between the last closed flux surface and the vessel wall. This set of equations describes partially ionized flow with nonlinear and anisotropic heat conduction, thermal non-equilibrium (seperate ion and electron temperatures), and finite-rate ionization and recombination (i.e., chemistry).

The earliest use of JFNK in computational plasma physics was on the tokamak edge plasma fluid equations [154]. For the original use of Newton's method on the edge plasma equations see [94,101,102,109,186], with other Newton method applications following soon after [185,193]. The JFNK method has become the mainstay of edge plasma simulation within the US fusion community, as embodied in the UEDGE code [154].

JFNK applications in this area have utilized the numerical formation of a Jacobian and standard ILU factorization to perform the preconditioning process. In a set of papers [103,104] on two different edge plasma physics models it is demonstrated that the use of a stale Jacobian in the preconditioning process provides significant CPU savings. It is also demonstrated the a pseudo-transient approach (Section 2.4.1) provides a good globalization approach to this challenging boundary value problem.

In [95] a higher-order, nonlinear, convection scheme is applied to the edge plasma fluid equations. It is demonstrated that forming the preconditioner from a low-order spatial discretization provided a simultaneous savings in memory requirements and CPU time, as compared to using the higher-order method in both the preconditioner and the residual evaluation. In [107] some initial investigation is done on the application of one-level Schwarz preconditioning to the edge plasma equations. In [155] Newton–Krylov–Schwarz methods are applied to the edge plasma equation for parallelism. Also in [155], the superiority of JFNK over an operator split approach is demonstrated.

4.2.2. Fokker–Planck

The Fokker–Planck equation is used to model semi-collisional transport of charged particles in phase space. There are a wide variety of applications of the Fokker–Planck approach in plasma physics. JFNK has been applied to Fokker–Planck-based models of the tokamak edge plasma [127] and of inertial electrostatic confinement (IEC) devices [40–42,46]. The major challenge of the Fokker–Planck model is that it is nonlinear integro-differential. Thus a standard implementation of Newton's method results in a dense Jacobian matrix [58]. The storage requirements of such a problem limit its use, although it is well understood that such an implicit implementation has other significant advantages [58].

In [127] the Landau form of the Fokker–Planck equation is solved where the integral effect arises through coefficients that are integral functions of the distribution function (1D in configuration space and 1D in phase space). The preconditioner is formed by lagging the integral coupling while maintaining this coupling in the residual evaluation. A standard ILU method is applied to the resulting sparse preconditioning matrix. This preconditioner is frequently used by others as a solver. However, as the collision frequency increases, the fastest time scale in the problem is being evaluated at the previous iteration. It is clearly demonstrated in [127] that as collision frequency increased, the JFNK method significantly outperformed the method which lagged the integral coupling.

In [40,41] the Fokker–Planck equation is solved in Rosenbluth form in 2D phase space. Here, two subsidiary potential (elliptic) equations must be solved. This is done inside the residual evaluations using a multigrid-GMRES method. The preconditioning matrix for JFNK (2D, 9-point) is evaluated with the coefficients lagged and the preconditioning matrix is approximately inverted using simple multigrid ideas [100]. This approach is shown to have superior accuracy and scale efficiently to fine grids.

4.2.3. MHD

The equations of magnetohydrodynamics (MHD) represent a combination of the Navier–Stokes equations and Maxwell's equations without displacement current (the speed of light time scale has been removed). The MHD equations are typically used to simulate plasma phenomena on the ion time scale, with the fast electron time scales removed. By retaining the Hall term in the generalized Ohm's law (so-called Hall MHD) some electron physics is retained. This adds an additional fast wave to the system, the Whistler wave. As compared to the Navier–Stokes equations, the MHD equations are more complicated since they support a family of waves that propagate at different speeds in an anisotropic manner. In [44] JFNK is applied to a 2D incompressible MHD equation system (a three-equation system). A semi-implicit, physics-based, preconditioner is developed and the resulting scalar elliptic problems are approximately solved with simple multigrid methods (Section 3.3). The resulting implicitly balanced [98] algorithm is demonstrated to have a second-order accurate time step. It is shown that this algorithm can efficiently step over stiff wave time step constraints while maintaining a second-order accurate time step. Excellent time step and grid refinement scaling is demonstrated, as well as significant CPU gains as compared to an explicit simulation.

In [43] JFNK is applied to a 2.5D incompressible Hall MHD equation system (a five-equation system). The parabolic operator that arises from the semi-implicit treatment of the Whistler wave is fourth order in space. In one approach, the conjugate gradient method is used to approximately invert this system in the preconditioner. In another approach the fourth-order equation is re-cast as two second-order equations and this system is approximately inverted with multigrid. In both cases the JFNK method with physics-based preconditioning is shown to significantly outperform an explicit approach.

4.3. Reactive flows and flows with phase change

Reactive flows and flows with phase change are examples of nonlinear multiple time scale problems. There are many engineering and industrial applications that are simulated by such systems. Operator splitting has been used extensively to simulate reactive flows [136]. Early Newton method research as applied to reactive flows may be found in [87,165,167].

In [105,123] the performance of the JFNK method is studied in low Mach number compressible combustion. The base model is a BVP and represents a laminar diffusion flame. In [105] the SER method is employed for pseudo-transient continuation with standard ILU type preconditioners (similar to [104]). In [123] (within a Schwarz context) standard ILU preconditioners are compared to block ILU preconditioners, where the blocksize follows the number of conservation equations within a control volume. Block ILU preconditioning is shown to be superior in memory and in terms of preconditioner performance. The edge plasma Navier–Stokes neutrals model discussed in Section 4.2.1 and in [95,104] also contains finite-rate "chemistry" in table look-up form. This model has mass, momentum, and energy exchange between "phases" as a result of the finite-rate "chemistry". A particularly challenging version of this problem results from including both molecules and atoms in the neutral fluid [114]. The addition of molecules brings in fast (stiff) reactions. Here, block ILU is essential to effective preconditioning. The standard ILU machinery can be thought of as attacking the intra-equation coupling, while the blocking attacks the inter-equation coupling.

Another recent example of the application of JFNK methods to reactive flows is the work of Mukadi and Hayes [131]. Their application is the transient simulation of an automotive catalytic convertor. In this study the effects of spatial discretization on preconditioner performance are considered.

Shadid et al. [164] use standard inexact Newton–Krylov methodology (not JFNK) to simulate reactive flows in 3D, on unstructured grids, and on massively parallel architectures. Simulations with several chemical species and reactions have been performed [162].

JFNK methods are applied to phase change heat conduction problems in [99,113]. This is done for pure material (isothermal phase change) alloys and pure materials with multiple phase transitions. The key to implementing the JFNK methodology on this class of problems is the use of the enthalpy formulation [99] (enthalpy as the dependent energy variable). In the initial study of this problem, the phase-change physics is ignored in the preconditioner and the preconditioning operator is formed only from heat conduction. In a subsequent study, the "effective heat capacity" method [113] (a linearized solver for this problem) is used as a preconditioner. This is an approximate route to bringing the phase change physics into the preconditioner. This is shown to provide a factor of five reduction in GMRES iterations and a factor of four improvement in execution time.

4.4. Radiation diffusion and radiation hydrodynamics

Radiation diffusion and radiation hydrodynamics are further examples of nonlinear multiple time-scale systems. The equations of radiation hydrodynamics are used to simulate astrophysical phenomena and problems in inertial confinement fusion. These equations are formed by combining the compressible Navier–Stokes equations with one of many models for radiation transport. In many regimes of interest there is strong nonlinear coupling between the flow field (often called the "material") and the radiation (photon) field. Operator splitting has been used extensively to simulate radiation hydrodynamics [20]. Early Newton method research as applied to radiation hydrodynamics is found in [189]. The simplest of radiation transport models is the diffusion model, and this is where one finds most of the initial JFNK effort. As with many other multiple time-scale systems, the non-equilibrium radiation diffusion problem has both a dynamical time-scale and normal modes. Reaction and diffusion time scales can be very fast compared to the thermal front time scale. One wants to be able to follow the dynamical time scale [152].

Early 1D work [111,112] focused on presenting the ideas behind JFNK to the radiation transport community, as well as elucidating the ability of JFNK to provide increases in both accuracy and efficiency as compared to more standard linearized and operator split methods. The work in [111] demonstrates that the JFNK approach has superior nonlinear convergence rates compared to a Picard iteration.

Two-dimensional examples have focused on the development of multigrid-based preconditioning strategies [153] and on the use of operator-splitting as a preconditioner [130]. Analysis and results of operator-split based preconditioning on such problems indicate that the approach can break down [29]. Thus, there can exist a window where JFNK can integrate a system accurately but traditional operator splitting does not provide adequate preconditioning. This happens when there is an extremely large spread between the dynamical time scale (on which JFNK can integrate a system accurately) and the normal modes of the system. Options to overcome this hurdle include Schur complement approaches considered in [29], coupled multigrid preconditioning, or augmenting the operator splitting with a two-stage approach that includes a defect correction and a fine grid block Jacobi smoother (blocked at the component level) on the Jacobian [128].

Two recent efforts in radiation transport couple JFNK with nonlinear multigrid. In [8] JFNK is used as a smoother in an FAS scheme for a radiation transport problem. The problem considered has integral coupling, which is an opportune setting for JFNK. In [118] FAS is used as a preconditioner to JFNK for a non-equilibrium radiation diffusion problem similar to that in [130]. This system is solved on an unstructured grid using agglomeration multigrid in the preconditioner.

Coupled radiation hydrodynamics problems are beginning to come under investigation using JFNK. Typically, in radiation hydrodynamics the radiation transport is done implicitly and coupled via operator splitting to an explicit method for the hydrodynamics. The research in [189] considers the implicit Newton solution to the coupled problem. The work in [12] compares a nonlinearly consistent (JFNK) method applied to the radiation model and coupled to the hydrodynamics model in a predictor–corrector fashion. While this new approach is more accurate than the standard linearize-and-split approach, it is still able to achieve only first-order accuracy in time.

4.5. Geophysical flows

Problems in porous media flow and atmospheric flows can possess widespread time scales and/or strong nonlinearities. Both of these problems motivate the consideration of JFNK methods. We briefly mention results coming from two different applications of JFNK methods to subsurface flow and one application of JFNK to atmospheric flow.

In [173] JFNK is applied to Richard's equation, a nonlinear parabolic model for variably saturated flow, and the perfomance of various Krylov methods is considered. In [82] JFNK is applied to Richards' equation. Semicoarsening multigrid [5,160] and simpler multigrid [5] are applied to approximately invert the preconditioning matrix. The preconditioner is the the symmetric part of the complete Jacobian. The work in [79], while not Jacobian-free, develops effective two-level preconditioners for the Newton–Krylov solution of Richard's equation.

In [52] JFNK methods are applied to multiphase flow in a permeable media. A two-stage preconditioner is developed. The first stage is a decoupling stage (similar to ABF [10]) while the second stage solves separate (scalar) elliptic systems, as promoted in [113,130]. The two-stage preconditioner is shown to outperform an additive split-based preconditioner. In the recent work of Hammond and co-workers [73] JFNK and an operator-split preconditioner are applied to a multicomponent reactive subsurface transport model. The JFNK method is shown to provide advantages relative to conventional methods in this field.

There is a growing interest in increasing the predictive nature of atmospheric flow simulations such as those involved in wildfire modeling [150] and hurricane modeling. Both of these problems are by nature highly nonlinear and contain multiple time scales. Currently, simulation efforts in this community are dominated by operator splitting approaches, and thus they may contain unmonitored numerical error. The work in [126,149,151] represents an initial effort to bring JFNK into this community. In [126] JFNK is applied to the shallow water wave equations in 2D, including the Coriolis effect. Physics-based preconditioning with simple multigrid is shown to be very effective on this classic stiff wave problem. Here, the outer JFNK method is integrating a three-component hyperbolic system, while the preconditioner requires only the approximate implicit solution of a scalar parabolic equation. It is clearly shown that a second-order in time discretization, solved with a JFNK method, can integrate this system at the dynamical time scale (stepping over the stiff wave time scale) to obtain excellent time accuracy. The work in [151] extends the work in [126] by considering the compressible Euler equations, and thus a more complicated equation of state. This requires a more sophisticated physics-based preconditioner. Here, the outer JFNK method is integrating a four-component hyperbolic system while the preconditioner only requires the approximate implicit solution of a scalar parabolic equation. The JFNK method developed in [151] is being used for simulating wildfires and hurricanes.

4.6. Other applications

In addition to the well-established applications mentioned above, one can find recent application of JFNK methods to granular flow [50], plasma reactor simulation [168], and semiconductor simulation [19].

5. Illustrations

This section provides computational illustrations of some of the techniques and "tricks" of JFNK methods – making them work, and making them work effectively on real problems. As the heading indicates, this section is illustrative, not exhaustive. We make reference back to the appropriate areas in Sections 2–4.

5.1. Jacobian lagging

Here, we present results from [104], which considers an eight-equation system for the coupled edge plasma/Navier–Stokes neutral model. We consider the effect of Jacobian lagging only in the preconditioner (as discussed in Section 3.1), versus Jacobian lagging in the outer Newton iteration. Here ILU is used to approximately invert the preconditioning matrix. In Table 2 (Table 1 in [104]) a single grid simulation is considered using pseudo-transient continuation. SNK is a standard Newton–Krylov method (not Jacobian-free) and forms the Jacobian every Newton iteration. JFNK performs the Newton–Krylov iteration matrix-free, while the Jacobian used in the preconditioner is formed with a frequency of p Newton iterations. MNK (modified Newton–Krylov) is a standard Newton–Krylov method but the Jacobian is formed with a frequency of p Newton iterations for use in both the matrix vector multiply and the preconditioner. Within the pseudo-transient continuation method, to be consistent, the time step is advanced only every p Newton iterations. This frequency is somewhat arbitrary and intended to demonstrate possible savings. Recall that it is not our goal to be time accurate, but to under-relax the Newton iteration. Table 2 clearly shows an execution time advantage for the JFNK approach. The potential for CPU savings will be sensitive to p, the continuation method being used, and the application.

5.2. Mesh sequencing

The nonlinear convergence rate enhancement resulting from mesh sequencing has been investigated in several studies for BVPs and is discussed in Section 2.4.2. Table 3 (which is Table 4 in [104]) presents single grid and mesh sequencing results from an eight-equation system for the coupled edge plasma/Navier–Stokes neutral model in 2D. The impact of mesh sequencing for this BVP is clear, and the impact increases with grid refinement.

5.3. Multigrid preconditioning

We provide some results on the impact of multigrid as a preconditioner to JFNK (discussed in Section 3.3 as NKMG). We consider the steady-state solution of the incompressible Navier–Stokes BVP in the stream function–vorticity formulation. Fig. 1 is from [110] and plots the average number of GMRES iterations per Newton iteration as a function of grid dimension for five different solution methods. The solution methods vary in the preconditioner MG (multigrid), ILU(0), or BSGS (block symmetric Gausss-Sidel) and the number of GMRES vectors stored before restart. Advection is ignored in all preconditioners. It can be seen that the simple multigrid-based preconditioner significantly outperforms ILU(0) as the grid is refined. Furthermore, this allows storage of fewer GMRES vectors. While restart is employed on this

Table 2
Iteration and execution time performance for an 8-equation 2D BVP (64×32 grid) for a coupled edge plasma/Navier–Stokes neutral model (from [104])

Solution method	Newton iterations	GMRES iterations	Avg. GMRES per Newton	Rel. CPU
SNK, $p = 1$	158	485	3.1	3.56
MNK, $p = 5$	681	318	0.5	3.26
JFNK, $p = 5$	234	1570	6.7	1.46
JFNK, $p = 10$	182	2624	14.4	1.0

Table 3
Effect of mesh squencing on total execution time for an 8-equation 2D BVP coupled edge plasma/Navier–Stokes neutral model (from [104])

Problem size	Without mesh sequencing (h)	With mesh sequencing (h)	Rel. speedup
32×16	0.4	0.4	1.0
64×32	4.3	0.84	5.1
128×64	14.7	1.5	9.8

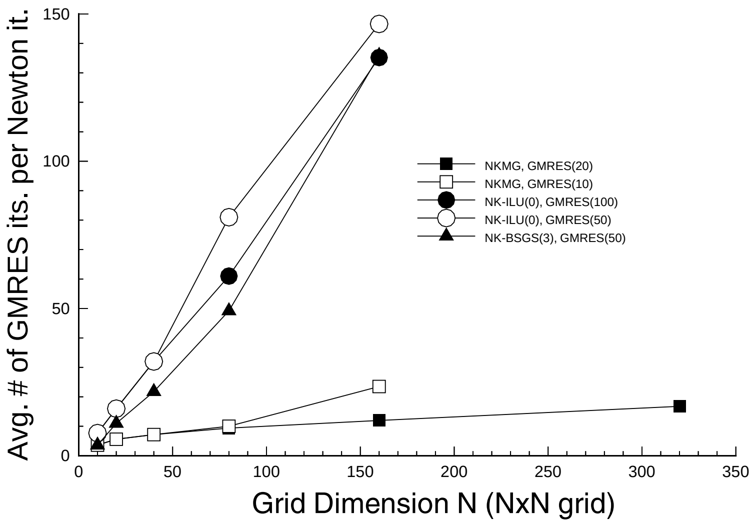

Fig. 1. Comparison of NKMG and NK-ILU(0) on 2D Navier–Stokes, varying number of GMRES vectors stored (from [110]).

problem, allowing 200 total GMRES iterations, its success is limited. We employ the standard restarting algorithm referred to as "Algorithm 6.11" in [157]. In terms of normalized execution time for a converged solution on the 160×160 grid, we have: NKMG/GMRES(20) = 1.0, NKMG/GMRES(10) = 2.6, NK-ILU(0)/GMRES(100) = 3.1, NK-ILU(0)/GMRES(50) = 2.7, and NK-BSGS(3)/GMRES(50) = 5.3. Only NKMG/GMRES(20) converges in a reasonable time on the 320×320 grid, with an execution time of 5.6, relative to the four times smaller 160×160 case. On the 160×160 grid, NKMG would not converge with less than 10 GMRES vectors stored, while NK-ILU(0) would not converge with less than 50 GMRES vectors stored.

Fig. 2 is from [108], and is a plot of CPU time scaling, as a function of grid dimension, using both the distributed and the coupled multigrid approaches in the preconditioner, piecewise constant restriction and prolongation, and a Galerkin coarse grid operator. The problem being solved is the natural convection problem simulated using a stream-function, vorticity, energy equation system. The data are for 80×80, 160×160, and 320×320 problems. We include a reference line for linear scaling, and we see that both approaches scale better than linearly.

5.4. Physics-based preconditioning

Here, we present some results from physics-based preconditioning, concentrating on the stiff-wave problem discussed in Section 3.4.1. First, we present a result from [126] where the JFNK method is applied

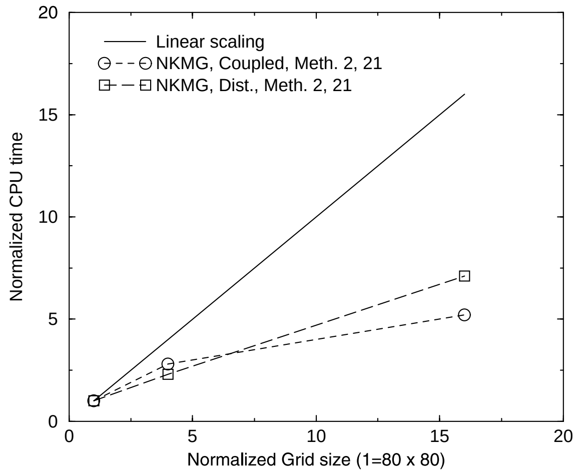

Fig. 2. Scaling (CPU time vs problem size) of Newton–Krylov-Multigrid (NKMG) for coupled and distributed preconditioners, for $Gr = 1.0 \times 10^5$ (from [108]).

to the 2D shallow water wave equations with the Coriolis force. This is a three-component hyperbolic system with a stiff gravity wave. As discussed in Section 3.4.1, a semi-implicit method is used to construct the preconditioner. Thus, the preconditioner action only requires an approximate inversion of a scalar parabolic equation, and this approximate inversion is accomplished with low complexity multigrid. In [126] a stiff-wave model problem is used to demonstrate that a nonlinearly consistent method (JFNK) can use time steps on the order of the dynamical time scale while maintaining comparable accuracy to a semi-implicit method run at the stiff-wave explicit CFL. An example of the algorithmic scaling of the method is given in Table 4, which is from [126]. As a result of spatial discretization mismatch between the semi-implicit method and the true nonlinear function, the preconditioner actually improves under grid refinement. The number of Newton iterations per time step is fairly constant, while the average number of GMRES iterations per time step decreases. As a result, the actual execution time beats the linear execution time scaling, which would be a factor of eight for each refinement (a factor of two in each space dimension and two in time).

Table 5 is from [44] and demonstrates the algorithmic scaling of physics-based preconditioning (with MG) on a three-equation MHD problem. This MHD problem contains a stiff Alfvén wave whose time scale is typically well separated from the dynamical time scale of interest. The data in Table 5 are from three different stiff wave CFL time step sizes over a range of grids. The scaling in terms of nonlinear iterations per time step and linear iterations per nonlinear iteration is good over a range of grids. As in the shallow water wave problem, the parabolic problem in each preconditioner application is approximately inverted using simple multigrid methods.

Table 4
Algorithm performance study (from [126])

$NX \times NY$	Newton/Timestep	GMRES/Newton	Advection CFL	Normalized CPU scaling	Linear CPU scaling
32×32	4.12	26.26	0.1394	1.00	1.0
64×64	4.01	15.68	0.2322	5.00	8.0
128×128	4.00	8.45	0.2421	24.19	64.0
256×256	4.00	5.22	0.2435	397.57	512.0

Table 5
Algorithm performance study, on various grids, with $\Delta t = 20, 40, 160 \times \Delta t_{\text{CFL}}$ for the tearing instability in 2D MHD

Grid	Newton/timestep	GMRES/Newton	GMRES/timestep	CPU (s)
$\Delta t = 20\Delta t_{\text{CFL}}$				
32×32	3.0	2.6	7.8	12.8
64×64	3.0	2.0	5.9	102
128×128	2.8	1.4	3.8	793
256×256	3.0	1.0	3.0	6537
$\Delta t = 40\Delta t_{\text{CFL}}$				
32×32	3.0	3.8	11.5	8.2
64×64	3.0	3.3	10.0	73.6
128×128	3.0	2.0	6	517
256×256	3.0	1.6	5.0	4248
$\Delta t = 160\Delta t_{\text{CFL}}$				
32×32	3.0	9.3	28.0	4.2
64×64	3.0	6.3	19.0	29
128×128	3.1	4.6	14.2	234
256×256	3.6	5.9	21.5	3220

Δt_{CFL} is stiff wave CFL (from [44]).

5.5. Newton–Krylov–Schwarz: parallelism and scaling

We conclude this section with an discussion of the use of JFNK to parallelize a legacy application code. We consider an aerodynamics application based on the code FUN3D, a tetrahedral, vertex-centered unstructured mesh code originally developed by W.K. Anderson of the NASA Langley Research Center for compressible and incompressible Euler and Navier–Stokes equations [3,4]. FUN3D employs a control volume discretization with a variable-order Roe scheme for approximating the convective fluxes and a Galerkin discretization for the viscous terms. FUN3D has been used for design optimization of airplanes, automobiles, and submarines, with irregular meshes comprising several million mesh points. The optimization involves many analyses, typically sequential. Thus, reaching the steady-state solution in each analysis cycle in a reasonable amount of time is crucial to conducting the design optimization. A representative achievement to date for million meshpoint simulations on thousands of processors is about 10 μs per degree of freedom for convergence of the steady-state residuals below the square-root of machine precision.

In work that was recognized with a 1999 Gordon Bell Prize in the "special" category, and subsequently published in [69], FUN3D was ported to the PETSc [7] JFNK framework, using the single program multiple data (SPMD) message-passing programming model, supplemented by multithreading at the physically shared memory level. For a fixed-size problem with 11 million degrees of freedom, the time decreased from 2589 to 154 s for solving the BVP on 128–3072 processors, which amounts to an efficiency of 70%.

Achieving high sustained performance with an emphasis on total solution time requires attention to three factors. The first is a scalable implementation, in the sense that time per iteration is reduced in inverse proportion to the number of processors (strong scaling), or that time per iteration is constant as problem size and processor number are scaled proportionally (weak scaling). The second is good per processor performance on contemporary cache-based microprocessors. The third is algorithmic scalability, in the sense that the number of iterations to convergence does not grow with increased numbers of processors (or problem size). The third factor arises because the requirement of a scalable implementation generally forces parameterized changes in the algorithm as the number of processors grows. If the convergence is allowed to degrade, however, the overall execution is not scalable, and this must be countered algorithmically.

The following is an incomplete list of parameters that need to be tuned in various phases of a pseudo-transient Newton–Krylov–Schwarz algorithm:

- *Nonlinear robustness continuation parameters*: discretization order, initial time step, pseudo-time step evolution law.
- *Newton parameters*: convergence tolerance on each time step, globalization strategy (line search or trust region parameters), refresh frequency for Jacobian preconditioner.
- *Krylov parameters*: convergence tolerance for each Newton correction, restart dimension of Krylov subspace, overall Krylov iteration limit, orthogonalization mechanism.
- *Schwarz parameters*: subdomain number, amount of subdomain overlap, coarse grid usage.
- *Subdomain parameters*: incomplete factorization fill level, number of sweeps.

Many of these parameters received comment in the presentation of the JFNK and ΨNKS algorithms in Sections 2 and 3. In relation to the FUN3D example, we point out that although convergence is to a second-order convection scheme discretization, much of the early nonlinear iteration uses a first-order convection scheme, until the location of the shock has stabilized. Only after this is the discretization sharpened up in the right-hand side nonlinear residuals (and consequently in the vector finite difference approximation of the Jacobian-vector products). Otherwise, Newton is difficult to use on this problem, even with pseudo-timestepping. First-order discretization for the convective terms is used for the Jacobian preconditioner throughout.

Choices for these parameters are extensively studied in [69], where there is no claim that the optimal combination has been found. JFNK algorithms are evidently rich in options. This can be overwhelming to a new user, but it provides a great deal of architectural and application adaptivity to the experienced user.

6. PDE-constrained optimization

It is increasingly recognized that PDE-based analyses are rarely ends in themselves, but more properly part of a scientific process that includes some type of sensitivity analysis or optimization, in which the system of PDEs serves as a constraint. The optimization process may arise, for instance, in design, in control, in parameter identification (inverse problems), or in data assimilation. Some property, such as the integrated dissipation rate or the norm of some discrepancy between measured and modeled outcomes, is to be minimized, subject to the constraint that a governing system of PDEs is satisfied. Without the PDE constraint, the optimization algorithm may find more optimal values of the objective that are physically infeasible, and therefore uninteresting.

Jacobian-free Newton–Krylov methods have important roles to play in PDE-constrained optimization. At the very least, the fact that PDEs need to be solved in the inner loop of a conventional constrained optimization algorithm requires time- and memory-efficient PDE solvers. A Newton method makes good use of a "warm" initial guess, and therefore performs well in projecting the result of an optimization step onto the constraint manifold inside an iterative optimization method such as Reduced Sequential Quadratic Programming (RSQP) [191]. However, it has become apparent in recent years that there are potentially much more efficient classes of optimization algorithms that employ a Jacobian-free Newton–Krylov method as the *outer* optimization loop, not as the *inner* projection step. These methods search for saddle points of the Lagrangian formulation of the constrained optimization by looking for the roots of the gradient of the Lagrangian with respect to all of its parameters – the design parameters, the state variables of the PDE, and the Lagrange multipliers. In these contemporary optimization methods, called Lagrange-Newton–Krylov (LNK) methods [14,15,92], the PDE constraints are not necessarily satisfied accurately at every step, but are only guaranteed to be satisfied asymptotically, as a design parameter optimum is approached.

Our discussion of the LNK family of methods in this survey is introductory only, since a systematic treatment demands first a survey of constrained optimization methodology, and also since this application of JFNK is relatively young. However, a presentation of the prospects for JFNK would be incomplete without mentioning its use in LNK, and such a presentation can be self-contained if one accepts treating equality-constrained optimization as a nonlinear rootfinding problem.

One of the chief practical differences between JFNK applied to PDEs and JFNK applied to Lagrangian constrained optimization is that the Jacobian of the direct analysis involves only first derivatives of the conservation laws with respect to the state variables, whereas the Jacobian of the Lagrangian problem (the so-called Karush–Kuhn–Tucker matrix) involves second derivatives of the objective function and the PDE conservation laws. It is relatively routine to obtain approximations to first derivatives of well-scaled objects in standard floating point contexts with finite differences. This is not true for second derivatives since a second difference amplifies the roundoff to levels generally unacceptable for standard double-precision floating point. Therefore, the subject of automatic differentiation makes an important appearance in optimization. Moreover, the Jacobian of the Lagrangian problem involves in a fundamental way the transpose of the Jacobian of the PDE constraints with respect to the state variables. In the JFNK context, it is not known how to form Jacobian-transpose-vector products with finite Fréchet derivatives. Automatic differentiation, whose so-called "reverse mode" permits efficient Jacobian-transpose applications, is therefore important for this reason, as well. Indeed, it is no coincidence that LNK is only now becoming a practically important method, with the advent of quality automatic differentiation software [16,17].

6.1. Newton's method in constrained optimization

Equality constrained optimization leads, as mentioned, through the Lagrangian formulation, to a multivariate nonlinear rootfinding problem for the gradient (the first-order necessary conditions), which is amenable to treatment by Newton's method. To establish notation, consider the following canonical framework, in which we enforce equality constraints on the state variables only. (Design variable constraints require additional notation, and inequality constraints require additional algorithmics, so we leave their generalization to the literature [191].) Choose design variables $u \in R^m$ to minimize the scalar objective function, $\phi(u,x)$, subject to state constraints, $h(u,x) = 0$, where $x \in R^n$ is the vector of state variables. In the Lagrange framework, a stationary point of the scalar Lagrangian function

$$\mathscr{L}(x,u,\lambda) \equiv \phi(x,u) + \lambda^{\mathrm{T}} h(x,u)$$

is sought, where $\lambda \in R^n$. When Newton's method is applied to the first-order optimality conditions, a linear system known as the Karush–Kuhn–Tucker (KKT) system arises at each step. There is a natural "outer" partitioning: the vector of parameters is often of lower dimension than the vectors of states and multipliers. This suggests an approximate Schur complement-like block preconditioning process at the outer level. Within the state-variable subproblem, in turn, Schwarz provides a natural "inner" partitioning for concurrency.

To emphasize differences of computational scale relevant to the algorithmics, we mention three classes of PDE-constrained optimization:

- *Design optimization*: (especially shape optimization): u parametrizes the domain geometry of the PDE (e.g., a lifting surface) and ϕ is a cost-to-benefit ratio of forces, energy expenditures, etc. Typically, m is small compared with n and does not scale directly with it as the mesh is refined. However, m may still be hundreds or thousands in industrial applications.
- *Optimal control*: u parametrizes a continuous control function acting in part of the domain or on part of the boundary of the domain and ϕ is the norm of the difference between desired and actual responses of the system. For boundary control, $m \propto n^{2/3}$.

- *Parameter identification/data assimilation*: u parametrizes an unknown continuous constitutive or forcing function defined throughout the domain and ϕ is the norm of the difference between measurements and simulation results. Typically, $m \propto n$.

  Written out in partial detail, the first-order optimality conditions are:

$$\frac{\partial \mathscr{L}}{\partial x} \equiv \frac{\partial \phi}{\partial x} + \lambda^{\mathrm{T}} \frac{\partial h}{\partial x} = 0, \quad \frac{\partial \mathscr{L}}{\partial u} \equiv \frac{\partial \phi}{\partial u} + \lambda^{\mathrm{T}} \frac{\partial h}{\partial u} = 0, \quad \frac{\partial \mathscr{L}}{\partial \lambda} \equiv h = 0.$$

Newton's method iteratively seeks a correction,

$$\begin{pmatrix} \delta x \\ \delta u \\ \delta \lambda \end{pmatrix} \text{ to the iterate } \begin{pmatrix} x \\ u \\ \lambda \end{pmatrix}$$

to reduce the gradient of the Lagrangian to zero. With subscript notation for the partial derivatives, the Newton correction (KKT) equations are:

$$\begin{bmatrix} (\phi_{,xx} + \lambda^{\mathrm{T}} h_{,xx}) & (\phi_{,xu} + \lambda^{\mathrm{T}} h_{,xu}) & h_{,x}^{\mathrm{T}} \\ (\phi_{,ux} + \lambda^{\mathrm{T}} h_{,ux}) & (\phi_{,uu} + \lambda^{\mathrm{T}} h_{,uu}) & h_{,u}^{\mathrm{T}} \\ h_{,x} & h_{,u} & 0 \end{bmatrix} \begin{pmatrix} \delta x \\ \delta u \\ \delta \lambda \end{pmatrix} = - \begin{pmatrix} \phi_{,x} + \lambda^{\mathrm{T}} h_{,x} \\ \phi_{,u} + \lambda^{\mathrm{T}} h_{,u} \\ h \end{pmatrix}$$

or

$$\begin{bmatrix} W_{xx} & W_{ux}^{\mathrm{T}} & J_x^{\mathrm{T}} \\ W_{ux} & W_{uu} & J_u^{\mathrm{T}} \\ J_x & J_u & 0 \end{bmatrix} \begin{pmatrix} \delta x \\ \delta u \\ \lambda_+ \end{pmatrix} = - \begin{pmatrix} g_x \\ g_u \\ h \end{pmatrix}, \tag{43}$$

where

$$W_{ab} \equiv \frac{\partial^2 \phi}{\partial a \, \partial b} + \lambda^{\mathrm{T}} \frac{\partial^2 h}{\partial a \, \partial b}, \quad J_a \equiv \frac{\partial h}{\partial a} \quad \text{and} \quad g_a = \frac{\partial \phi}{\partial a},$$

for $a, b \in \{x, u\}$, and where $\lambda_+ = \lambda + \delta\lambda$.

6.2. Newton-RSQP and LNK

The RSQP method [191] consists of a three-stage iteration. We follow the language and practice of [14,15].

- *Design step* (Schur complement for middle blockrow):

$$H \delta u = f,$$

  where H and f are the reduced Hessian and gradient, respectively:

$$H \equiv W_{uu} - J_u^{\mathrm{T}} J_x^{-\mathrm{T}} W_{ux}^{\mathrm{T}} + \left(J_u^{\mathrm{T}} J_x^{-\mathrm{T}} W_{xx} - W_{ux}\right) J_x^{-1} J_u,$$
$$f \equiv -g_u + J_u^{\mathrm{T}} J_x^{-\mathrm{T}} g_x - \left(J_u^{\mathrm{T}} J_x^{-\mathrm{T}} W_{xx} - W_{ux}\right) J_x^{-1} h.$$

- *State step* (last blockrow):

$$J_x \delta x = -h - J_u \delta u.$$

- *Adjoint step* (first blockrow):

$$J_x^{\mathrm{T}} \lambda_+ = -g_x - W_{xx} \delta x - W_{ux}^{\mathrm{T}} \delta u.$$

In each overall iteration, we must form and solve with the reduced Hessian matrix H, and we must solve separately with J_x and J_x^{T}. The latter two solves are almost negligible compared with the cost of forming H, which is dominated by the cost of forming the sensitivity matrix $J_x^{-1}J_u$. Because of the quadratic convergence of Newton, the number of overall iterations is few (asymptotically independent of m). However, the cost of forming H at each design iteration is m solutions with J_x. These are potentially concurrent over the m independent columns of J_u, but prohibitive.

In order to avoid computing any Hessian blocks, the design step may be approached in a quasi-Newton (e.g., BFGS) manner [191]. Hessian terms are dropped from the adjoint step right-hand side.

- *Design step* (severe approximation to middle blockrow):

  $$Q\delta u = -g_u + J_u^{\mathrm{T}}J_x^{-\mathrm{T}}g_x,$$

  where Q is a quasi-Newton approximation to the reduced Hessian, H.
- *State step* (last blockrow):

  $$J_x\delta x = -h - J_u\delta u.$$

- *Adjoint step* (approximate first blockrow):

  $$J_x^{\mathrm{T}}\lambda_+ = -g_x.$$

In each overall iteration of this quasi-Newton RSQP, we must perform a low-rank update on Q or its inverse and we must solve with J_x and J_x^{T}. This strategy vastly reduces the cost of an iteration; however, it is no longer a Newton method. The number of overall iterations is many. Since BFGS is equivalent to unpreconditioned CG for quadratic objective functions, $\mathcal{O}(m^p)$ sequential cycles ($p > 0$, $p \approx 1/2$) may be anticipated. Hence, quasi-Newton RSQP is not scalable in the number of design variables, and no ready form of parallelism can address this convergence-related defect.

To summarize, conventional RSQP methods apply a (quasi-)Newton method to the optimality conditions: solving an approximate $m \times m$ system to update u, solving an $n \times n$ system to update x and λ consistently, and iterating. The unpalatable expense arises from the exact linearized analyses for updates to x and λ that appear in the inner loop. We therefore consider replacing the exact elimination steps of RSQP with preconditioning steps in an outer loop, arriving at LNK.

Consider applying a Krylov–Schwarz method directly to the $(2n + m) \times (2n + m)$ KKT system, Eq. (43). For this purpose, we require the action of the full matrix on the full-space vector and a good full-system preconditioner, for algorithmic scalability. One Newton SQP iteration is a perfect preconditioner – a block factored solver, based on forming the reduced Hessian of the Lagrangian H – but, of course, far too expensive. Backing off wherever storage or computational expense becomes impractical for large-scale PDEs generates a family of attractive methods.

To precondition the full system, we need approximate inverses to the three left-hand side matrices in the first algorithm of Section 6.2, namely, H, J, and J^{T}. If a preconditioner is available for H, and exact solves are available for J and J^{T}, then it may be shown [90] that conjugate gradient Krylov iteration on the (assumed symmetrizable) reduced system and conjugate gradient iteration on the full system yield the same sequence of iterates. The iterates are identical in the sense that if one were to use the values of u arising from the iteration on the reduced system in the right-hand side of the block rows for x and λ, one would reconstruct the iterates of the full system, when the same preconditioner used for H in the reduced system is used for the W_{uu} block in the full system. Moreover, the spectrum of the full system is simply the spectrum of the reduced system supplemented with a large multiplicity of unit eigenvalues. If one retreats from exact solves with J and J^{T}, this equivalence no longer holds; however, if good preconditioners are used for these Jacobian blocks, then the cloud of eigenvalues around unity is still readily shepherded by a Krylov method, and convergence should be nearly as rapid as in the case of exact solves.

This Schur-complement-based preconditioning of the full system was proposed in this equality-constrained optimization context by Biros and Ghattas in 1998 [14]. From a purely algebraic point of view (divorced from optimization), the same Schur-complement-based preconditioning was advocated by Keyes and Gropp in 1987 [90] in the context of domain decomposition. There, the reduced system was a set of unknowns on the interface between subdomains, and the savings from the approximate solves on the subdomain interiors more than paid for the modest degradation in convergence rate relative to interface iteration on the Schur complement. The main advantage of the full system problem is that the Schur complement never needs to be formed. Its exact action is felt on the design variable block through the operations carried out on the full system.

Biros and Ghattas have demonstrated the large-scale parallel effectiveness of the full system algorithm on a 3D Navier–Stokes flow boundary control problem, where the objective is dissipation minimization of flow over a cylinder using suction and blowing over the back portion of the cylinder as the control variables [15]. They performed this optimization with domain-decomposed parallelism on 128 processors of a T3E, using an original optimization toolkit add-on to the PETSc [7] toolkit. To quote one result from [15], for 6×10^5 state constraints and 9×10^3 controls, full-space LNK with approximate subdomain solves beat quasi-Newton RSQP by an order of magnitude (4.1 versus 53.1 h).

Automatic differentiation has two roles in the new JFNK optimization algorithm: formation of the action on a Krylov vector of the full KKT matrix, including the full second-order Hessian blocks, and supply of approximations to the elements of J (and J^T) for the preconditioner. LNK will generally be applied to large problems of n state variables and m parameters. Upon surveying existing AD tools, it is concluded in [92] that the preconditioned matrix-vector product can be formed in time linear in these two parameters.

7. Conclusions and prospects

We conclude with brief remarks on future directions for JFNK methodology, as influenced by directions for scientific and engineering applications, computer architecture, mathematical software, and the on-going development of other numerical techniques.

Computational simulation of systems governed by PDEs is being relied upon as never before for accurate results on which to base massive economic investments, public and corporate, as well as critical governmental policies. For instance, the ASCI program is intended to provide a computational alternative to nuclear weapons testing and the SciDAC program to help target investments in fusion energy devices and next-generation accelerators. The 40 Teraflop/s, 5120-processor Japanese Earth Simulator is intended to allow unprecedented resolution and forward time integration horizons for climate prediction.

The typical governing equation system confronted in these Grand Challenge problems is nonlinear, coupled, multiscale, and multirate. The typical computational environment is distributed shared memory. To address this combination requires scalable nonlinear implicit solvers, for which we propose preconditioned Jacobian-free Newton–Krylov methods. As shown here, JFNK methods offer asymptotically rapid nonlinear convergence, and with proper preconditioning can also be both linearly scalable and efficiently parallelizable.

Preconditioning is where the battle for scalability is won or lost, both algorithmic scalability and parallel scalability. Therefore, in this paper we have reviewed a set of preconditioning techniques so varied that all that some of them have in common is that their action does not directly rely on the matrix elements of the true Jacobian. The distinction between the implicit forward action of the true Jacobian and the inverse action of an approximate Jacobian, which may be defined only by a subroutine call that maps a residual into an approximate delta correction, is fundamental to the culture of JFNK. As new ideas and implementations for preconditioners evolve, the JFNK method readily absorbs them.

The generality of preconditioning and multiplicity of Jacobian representations exploited in JFNK dictates an open software infrastructure, such as, e.g., the PETSc [7] or Aztec [175] solver frameworks, and invites the reuse of valuable existing user application solver code, which is reinterpreted as a component of the preconditioner.

Future work planned by the authors include the release, in one or more of these JFNK software frameworks, of tutorial examples of the advanced use of JFNK in a variety of fields. We also invite existing and new users of JFNK to post us with their own successes and challenges and to join in expanding the algorithm and application scope of this compelling methodology.

Acknowledgements

The authors thank all of their co-authors and co-explorers in Newton–Krylov methodology over the past dozen years, especially S. Balay, J.U. Brackbill, X.-C. Cai, L. Chacon, O. Ghattas, W.D. Gropp, P. Hovland, D. Kaushik, P.R. McHugh, L. McInnes, V.A. Mousseau, M. Pernice, W.J. Rider, J.N. Shadid, B. F. Smith, and D.P. Young, for contributing results from onging research as well as for helpful suggestions in the presentation of this material. This work was supported under the auspices of the US Department of Energy under DOE contract W-7405-ENG-36 at Los Alamos National Laboratory under DOE subcontract B347882 from Lawrence Livermore National Laboratory to Old Dominion University, and DOE cooperative agreement DE-FC02-01ER25476 with Old Dominion University.

References

[1] E. Allgower, K. Georg, Continuation and path following, in: Acta Numerica 1993, Cambridge University Press, Cambridge, 1993, pp. 1–65.

[2] D.A. Anderson, J.C. Tannehill, R.H. Pletcher, Computational Fluid Dynamics and Heat Transfer, Hemisphere Publishing, New York, 1984.

[3] W.K. Anderson, D.L. Bonhaus, An implicit upwind algorithm for computing turbulent flows on unstructured grids, Comp. Fluids 23 (1994) 1–21.

[4] W.K. Anderson, R.D. Rausch, D.L. Bonhaus, Implicit/multigrid algorithms for incompressible turbulent flows on unstructured grids, J. Comput. Phys. 128 (1996) 391–408.

[5] S.F. Ashby, R.D. Falgout, A parallel multigrid preconditioned conjugate gradient algorithm for groundwater flow simulations, Nucl. Sci. Engrg. 124 (1996) 145–159.

[6] O. Axelsson, Iterative Solution Methods, Cambridge University Press, Cambridge, 1994.

[7] S. Balay, W.D. Gropp, L.C. McInnes, B.F. Smith, PETSc users manual, Technical Report, ANL-95/11 – Revision 2.1.0, Argonne National Laboratory, April 2001.

[8] D. Balsara, Fast and accurate discrete ordinates methods for multidimensional radiative transfer. part I, basic methods, J. Quant. Spectr. Radiat. Transfer 69 (2001) 671–707.

[9] K. Banas, A Newton–Krylov solver with multiplicative Schwarz preconditioning for finite element compressible flow simulations, Comm. Numer. Methods Engrg. 18 (2002) 275–296.

[10] R. Bank, T. Chan, W. Coughran, R. Smith, The alternate block factorization procedure for systems of partial differential equations, BIT 29 (1989) 938–954.

[11] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. van der Vorst, Templates for the solution of linear systems: building blocks for iterative methods, SIAM (1994).

[12] J.W. Bates, D.A. Knoll, W.J. Rider, R.B. Lowrie, V.A. Mousseau, On consistent time-integration methods for radiation hydrodynamics in the equilibrium diffusion limit: low energy density regime, J. Comput. Phys. 167 (2001) 99–130.

[13] M. Benzi, Preconditioning techniques for large linear systems: a survey, J. Comput. Phys. 182 (2002) 418–477.

[14] G. Biros, O. Ghattas, Parallel Newton–Krylov methods for PDE-constrained optimization, in: Proceedings of SC99, IEEE Computer Society, Silver Spring, MD, 1999.

[15] G. Biros, O. Ghattas, A Lagrange-Newton–Krylov–Schur method for PDE-constrained optimization, SIAG/OPT Views-and-News 11 (2000) 1–6.

[16] C. Bischof, A. Carle, G. Corliss, A. Griewank, P. Hovland, ADIFOR – generating derivative codes from Fortran programs, Sci. Program. 1 (1992) 1–29.
[17] C. Bischof, L. Roh, A. Mauer, ADIC – an extensible automatic differentiation tool for ANSI-C, Software – Practice Exp. 27 (1997) 1427–1456.
[18] P. Bjørstad, Fast numerical solution of the biharmonic Dirichlet problem on rectangles, SIAM J. Numer. Anal. 20 (1983) 59–71.
[19] F. Bosisio, S. Micheletti, R. Sacco, A discretization scheme for an extended drift-diffusion model including trap-assisted phenomena, J. Comput. Phys. 159 (2000) 197–212.
[20] R.L. Bowers, J.R. Wilson, Numerical Modeling in Applied Physics and Astrophysics, Jones and Bartlett, Boston, 1991.
[21] J.U. Brackbill, D.A. Knoll, Transient magnetic reconnection and unstable shear layers, Phys. Rev. Lett. 86 (2001) 2329–2332.
[22] A. Brandt, Multi-level adaptive solutions to boundary value problems, Math. Comp. 31 (1977) 333.
[23] A. Brandt, Multigrid techniques: 1984 guide with applications to fluid dynamics, Technical Report, von Karman Institute, 1984.
[24] P.N. Brown, A local convergence theory for combined inexact-Newton/finite-difference projection methods, SIAM J. Numer. Anal. 24 (1987) 407–434.
[25] P.N. Brown, A.C. Hindmarsh, Matrix-free methods for stiff systems of ode's, SIAM J. Numer. Anal. 23 (1986) 610–638.
[26] P.N. Brown, A.C. Hindmarsh, Reduced storage matrix methods in stiff ODE systems, J. Appl. Math. Comput. 31 (1989) 40–91.
[27] P.N. Brown, Y. Saad, Hybrid Krylov methods for nonlinear systems of equations, SIAM J. Sci. Stat. Comput. 11 (1990) 450–481.
[28] P.N. Brown, Y. Saad, Convergence theory for nonlinear Newton–Krylov algorithms, SIAM J. Opt. 4 (1994) 297–330.
[29] P.N. Brown, C. Woodward, Preconditioning strategies for fully implicit radiation diffusion with material-energy coupling, SIAM J. Sci. Comput. 23 (2001) 499–516.
[30] X.-C. Cai, M. Dryja, M. Sarkis, RASHO: a restricted additive Schwarz preconditioner with harmonic overlap, in: Proceedings of the 13th International Conference on Domain Decomposition Methods, CIMNE, 2002, pp. 337–344.
[31] X.-C. Cai, C. Farhat, M. Sarkis, Schwarz methods for the unsteady compressible Navier–Stokes equations on unstructured meshes, in: Proceedings of the Eighth International Conference on Domain Decomposition Methods, Wiley, New York, 1997, pp. 453–460.
[32] X.-C. Cai, C. Farhat, M. Sarkis, A minimum overlap restricted additive Schwarz preconditioner and applications in 3d flow simulations, in: Proceedings of the Tenth International Conference on Domain Decomposition Methods, AMS, 1998, pp. 238–244.
[33] X.-C. Cai, W.D. Gropp, D.E. Keyes, R.G. Melvin, D.P. Young, Parallel Newton–Krylov–Schwarz algorithms for the transonic full potential equation, SIAM J. Sci. Comput. 19 (1998) 246–265.
[34] X.-C. Cai, W.D. Gropp, D.E. Keyes, M.D. Tidriri, Newton–Krylov–Schwarz methods in CFD, in: Proceedings of the International Workshop on Numerical Methods for the Navier–Stokes Equations, Vieweg, Braunschweig, 1995, pp. 17–30.
[35] X.-C. Cai, D.E. Keyes, Nonlinearly preconditioned inexact Newton algorithms, SIAM J. Sci. Comput. 24 (2002) 183–200.
[36] X.-C. Cai, D.E. Keyes, L. Marcinkowski, Nonlinear additive Schwarz preconditioners and applications in computational fluid dynamics, Int. J. Numer. Methods Fluids 40 (2002) 1463–1470.
[37] X.-C. Cai, D.E. Keyes, V. Venkatakrishnan, Newton–Krylov–Schwarz: an implicit solver for CFD, in: Proceedings of the Eighth International Conference on Domain Decomposition Methods, Wiley, New York, 1997, pp. 387–400.
[38] X.-C. Cai, D.E. Keyes, D.P. Young, A nonlinearly additive Schwarz preconditioned inexact Newton method for shocked duct flow, in: Proceedings of the 13th International Conference on Domain Decomposition Methods, Domain Decomposition Press, 2002, pp. 345–352.
[39] X.-C. Cai, M. Sarkis, A restricted additive Schwarz preconditioner for general sparse linear systems, SIAM J. Sci. Stat. Comput. 21 (1999) 792–797.
[40] L. Chacon, Fokker–Planck modeling of spherical inertial electrostatic, virtual-cathode fusion systems, PhD thesis, University of Illinois, 2000.
[41] L. Chacon, D.C. Barnes, D.A. Knoll, G.H. Miley, An implicit energy-conservative 2D Fokker–Planck algorithm: II Jacobian-free Newton–Krylov solver, J. Comput. Phys. 157 (2000) 654–682.
[42] L. Chacon, D.C. Barnes, D.A. Knoll, G.H. Miley, A bounce-averaged Fokker–Planck code for Penning fusion devices, Comput. Phys. Comm. 134 (2001) 182–208.
[43] L. Chacon, D.A. Knoll, A 2D high-β Hall MHD implicit nonlinear solver, J. Comput. Phys. 188 (2003) 573–592.
[44] L. Chacon, D.A. Knoll, J.M. Finn, An implicit nonlinear reduced resistive MHD solver, J. Comput. Phys. 178 (2002) 15–36.
[45] L. Chacon, D.A. Knoll, J.M. Finn, Hall MHD effects in the 2-d Kelvin–Helmholtz/tearing instability, Phys. Lett. A 308 (2003) 187–197.
[46] L. Chacon, G.H. Miley, D.C. Barnes, D.A. Knoll, Energy gain calculations in Penning fusion systems using a bounce-averaged Fokker–Planck model, Phys. Plasmas 7 (2000) 4547–4560.
[47] T.F. Chan, K.R. Jackson, Nonlinearly preconditioned Krylov subspace methods for discrete Newton algorithms, SIAM J. Sci. Stat. Comput. 5 (1984) 533–542.

[48] T.S. Coffey, C.T. Kelley, D.E. Keyes, Pseudo-transient continuation and differential-algebraic equations. SIAM J. Sci. Comput., 2003 (to appear).
[49] D.E. Culler, J.P. Singh, A. Gupta, Parallel Computer Architecture, Morgan-Kaufmann, Los Altos, CA, 1998.
[50] S.J. Cummins, J.U. Brackbill, An implicit particle-in-cell method for granular materials, J. Comput. Phys. 180 (2002) 506–548.
[51] G.D.V. Davis, Natural convection of air in a square cavity: a benchmark numerical solution, Int. J. Numer. Methods Fluids 3 (1983) 249.
[52] C. Dawson, H. Klie, M. Wheeler, C. Woodward, A parallel, implicit, cell centered method for two-phase flow with a preconditioned Newton–Krylov solver, Comp. Geosci. 1 (1997) 215–249.
[53] R. Dembo et al., Inexact Newton methods, SIAM J. Numer. Anal. 19 (1982) 400–408.
[54] J.E. Dennis, R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equa tions, Prentice-Hall, Englewood Cliffs, NJ, 1983.
[55] M. Dryja, O.B. Widlund, An additive variant of the Schwarz alternating method for the case of many subregions, Technical Report 339, Courant Institute, NYU, 1987.
[56] S.C. Eisenstat, H.F. Walker, Globally convergent inexact Newton methods, SIAM J. Optim. 4 (1994) 393–422.
[57] S.C. Eisenstat, H.F. Walker, Choosing the forcing terms in a inexact Newton method, SIAM J. Sci. Comput. 17 (1996) 16–32.
[58] E.M. Epperlein, Implicit and conservative difference scheme for the Fokker–Planck equation, J. Comput. Phys. 112 (1994) 291.
[59] A. Ern, V. Giovangigli, D.E. Keyes, M.D. Smooke, Towards polyalgorithmic linear system solvers for nonlinear elliptic systems, SIAM J. Sci. Comput. 15 (1994) 681–703.
[60] R. Freund, A transpose-free quasi-minimal residual algorithm for non-Hermitian linear systems, SIAM J. Sci. Comput. 14 (1993) 470–482.
[61] M. Garbey, R. Hoppe, D.E. Keyes, Y. Kuznetsov, J. Périaux (Eds.), Proceedings of the 13th International Conference on Domain Decomposition Methods, CIMNE, 2002. Available from www.ddm.org.
[62] C.W. Gear, Y. Saad, Iterative solution of linear equations in ode codes, SIAM J. Sci. Stat. Comput. 4 (1983) 583–601.
[63] P. Geuzaine, Newton–Krylov strategy for compressible turbulent flows on unstructured mesh, AIAA J. 39 (2001) 528–531.
[64] U. Ghia, K. Ghia, C. Shin, High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method, J. Comput. Phys. 48 (1982) 387.
[65] R. Glowinski, Numerical Methods for Nonlinear Variational Problems, Springer Verlag, Berlin, 1984.
[66] G.H. Golub, D. O'Leary, Some history of the conjugate gradient and Lanczos algorithms: 1948–1976, SIAM Rev. (1989) 50–102.
[67] A. Greenbaum, Iterative Methods for Solving Linear Systems, SIAM, Philadelphia, PA, 1997.
[68] W. Gropp, D.E. Keyes, L. McInnes, M. Tidriri, Globalized Newton–Krylov–Schwarz algorithms and software for parallel implicit CFD, Int. J. High Performance Comput. Appl. 14 (2000) 102–136.
[69] W.D. Gropp, D.K. Kaushik, D.E. Keyes, B.F. Smith, High performance parallel implicit CFD, Parallel Comput. 27 (2001) 337–362.
[70] H. Gummel, A self-consistent iterative scheme for one-dimensional steady state transistor calculations, IEEE Trans. Electron. Dev. ED-11 (1964) 455–465.
[71] M.H. Gutknecht, Lanczos-type solvers for nonsymmetric linear systems of equations, in: Acta Numerica 1997, Cambridge University Press, Cambridge, 1997, pp. 271–398.
[72] W. Hackbusch, Iterative Methods for Large Sparse Linear Systems, Springer, Berlin, 1993.
[73] G.E. Hammond, A.J. Valocchi, P.C. Lichtner, Modeling multicomponent reactive transport on parallel computers using Jacobian-Free Newton–Krylov with operator-splt preconditioning, Devel. Water Resour. 47 (2002) 727–734.
[74] F. Harlow, A. Amsden, A numerical fluid dynamical calculation method for all flow speeds, J. Comput. Phys. 8 (1971) 197–214.
[75] M.R. Hestenes, E.L. Stiefel, Methods of conjugate gradients for solving linear systems, J. Res. Natl. Bureau of Standards, Section B 49 (1952) 409–436.
[76] R. Hockney, Meth. Comp. Phys. 9 (1970) 135–211.
[77] P. Hovland, L.C. McInnes, Parallel simulation of compressible flow using automatic differentiation and PETSc, Parallel Comput. 27 (2001) 503–519.
[78] B.R. Hutchinson, G.D. Raithby, A multigrid method based on the additive correction strategy, Numer. Heat Transfer 9 (1986) 511–537.
[79] E.W. Jenkins, C.E. Kees, C.T. Kelley, C.T. Miller, An aggregation-based domain decomposition preconditioner for groundwater flow, SIAM J. Sci. Stat. Comput. 23 (2001) 430–441.
[80] H. Jiang, P.A. Forsyth, Robust linear and nonlinear strategies for solution of the transonic Euler equations, Comp. Fluids 24 (1995) 753–770.
[81] R.W. Johnson, P.R. McHugh, D.A. Knoll, High-order scheme implementation using Newton–Krylov solution methods, Numer. Heat Trans., Part B 31 (1997) 295–312.
[82] J.E. Jones, C.S. Woodward, Newton–Krylov-multigrid solvers for large scale, highly heterogeneous, variably saturated flow problems, Adv. Water Resour. 24 (2001) 763–774.

[83] D.E. Kaushik, D.E. Keyes, B.F. Smith, On the interaction of architecture and algorithm in the domain-based parallelization of an unstructured grid incompressible flow code, in: Proceedings of the Tenth International Conference on Domain Decomposition Methods, AMS, 1998, pp. 311–319.
[84] C.T. Kelley, Iterative Methods for Linear and Nonlinear Equations, SIAM, Philadelphia, 1995.
[85] C.T. Kelley, D.E. Keyes, Convergence analysis of pseudo-transient continuation, SIAM J. Numer. Anal. 35 (1998) 508–523.
[86] T. Kerkhoven, Y. Saad, On acceleration methods for coupled nonlinear elliptic systems, Numer. Math. 60 (1992) 525–548.
[87] D.E. Keyes, Domain decomposition methods for the parallel computation of reacting flows, Comp. Phys. Commun. 53 (1989) 181–200.
[88] D.E. Keyes, Aerodynamic applications of Newton–Krylov–Schwarz solvers, in: Proceedings of the 14th Conference on Numerical Methods in Fluid Dynamics, Springer, Berlin, 1995, pp. 1–20.
[89] D.E. Keyes, How scalable is domain decomposition in practice? in: Proceedings of the 11th International Conference on Domain Decomposition Methods, Domain Decomposition Press, 1999, pp. 286–297.
[90] D.E. Keyes, W.D. Gropp, A comparison of domain decomposition techniques for elliptic partial differential equations and their parallel implementation, SIAM J. Sci. Stat. Comput. 8 (1987) s166–s202.
[91] D.E. Keyes, D.K. Kaushik, B.F. Smith, Prospects for CFD on petaflops systems, in: CFD Review 1998, World Scientific, Singapore, 1998, pp. 1079–1096.
[92] D.E. Keyes, L. McInnes, W. Samyono, Using automatic differentiation for second-order matrix-free methods in PDE-constrained optimization, in: G. Corliss et al. (Eds.), Automatic Differentiation Algorithms: From Simulation to Optimization, Springer, Berlin, 2002, pp. 35–50.
[93] D.E. Keyes, V. Venkatakrishnan, Newton–Krylov–Schwarz methods: interfacing sparse linear solvers with nonlinear applications, Z. Angew. Math. Mech. 76 (Suppl. 1) (1996) 147–150.
[94] D.A. Knoll, Development and application of a direct Newton solver for the two-dimensional tokamak edge plasma fluid equations, PhD thesis, University of New Mexico, 1991.
[95] D.A. Knoll, An improved convection scheme applied to recombining divertor plasma flows, J. Comput. Phys. 142 (1998) 473–488.
[96] D.A. Knoll, J.U. Brackbill, The Kelvin–Helmholtz instability, differential rotation, and 3-D, localized, magnetic reconnection, Phys. Plasmas 9 (2002) 3775–3782.
[97] D.A. Knoll, L. Chacon, Magnetic reconnection in the two-dimensional Kelvin–Helmholtz instability, Phys. Rev. Lett. 88 (2002), p. art. no. 215003.
[98] D.A. Knoll, L. Chacon, L.G. Margolin, V.A. Mousseau, On balanced approximations for the time integration of multiple time scale systems, J. Comput. Phys. 185 (2003) 583–611.
[99] D.A. Knoll, D.B. Kothe, B. Lally, A new nonlinear solution method for phase-change problems, Numer. Heat Trans., Part B 35 (1999) 439–459.
[100] D.A. Knoll, G. Lapenta, J. Brackbill, A multilevel iterative field solver for implict, kinetic plasma simulation, J. Comput. Phys. 149 (1999) 377–388.
[101] D.A. Knoll, P.R. McHugh, NEWEDGE: a 2-D fully implicit edge plasma fluid code for advanced physics and complex geometries, J. Nucl. Mater. 196-198 (1992) 352–356.
[102] D.A. Knoll, P.R. McHugh, An inexact Newton algorithm for solving the Tokamak edge plasma fluid equations on a multiply connected domain, J. Comput. Phys. 116 (1995) 281–291.
[103] D.A. Knoll, P.R. McHugh, Newton–Krylov methods applied to a system of convection–diffusion–reaction equations, Comput. Phys. Commun. 88 (1995) 141–160.
[104] D.A. Knoll, P.R. McHugh, Enhanced nonlinear iterative techniques applied to a nonequilibrium plasma flow, SIAM J. Sci. Comput. 19 (1998) 291–301.
[105] D.A. Knoll, P.R. McHugh, D.E. Keyes, Newton–Krylov methods for low Mach number compressible combustion, AIAA J. 34 (1996) 961–967.
[106] D.A. Knoll, P.R. McHugh, S.I. Krasheninnikov, D.J. Sigmar, Simulation of dense recombining divertor plasmas with a Navier–Stokes neutral transport model, Phys. Plasmas 3 (1996) 293–303.
[107] D.A. Knoll, P.R. McHugh, V.A. Mousseau, Newton–Krylov–Schwarz methods applied to the tokamak edge plasma fluid equations, in: Domain-Based Parallelism and Problem Decomposition in Computational Science and Engineering, SIAM, Philadelphia, 1995, pp. 75–96.
[108] D.A. Knoll, V.A. Mousseau, On Newton–Krylov multigrid methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 163 (2000) 262–267.
[109] D.A. Knoll, A.K. Prinja, R.B. Campbell, A direct Newton solver for the two-dimensional tokamak edge plasma fluid equations, J. Comput. Phys. 104 (1993) 418–426.
[110] D.A. Knoll, W.J. Rider, A multigrid preconditioned Newton–Krylov method, SIAM J. Sci. Comput. 21 (2000) 691–710.
[111] D.A. Knoll, W.J. Rider, G.L. Olson, An efficient nonlinear solution method for non-equilibrium radiation diffusion, J. Quant. Spectrosc. Radiat. Transfer 63 (1999) 15–29.

[112] D.A. Knoll, W.J. Rider, G.L. Olson, Nonlinear convergence, accuracy, and time step control in non-equilibrium radiation diffusion, J. Quant. Spectrosc. Radiat. Transfer 70 (2001) 25–36.
[113] D.A. Knoll, W.B. VanderHeyden, V.A. Mousseau, D.B. Kothe, On preconditioning Newton–Krylov methods in solidifying flow applications, SIAM J. Sci. Comput. 23 (2002) 381–397.
[114] S.I. Krasheninnikov et al., Plasma recombination in tokamak divertors and divertor simulators, Phys. Plasmas 4 (1997) 1638.
[115] A.R. Larzelere, Creating simulation capabilities, IEEE Comput. Sci. Engrg. 5 (1988) 27–35.
[116] J.W. MacArthur, S.V. Patankar, Robust semidirect finite difference methods for solving the Navier–Stokes and energy equations, Int. J. Numer. Methods Fluids 9 (1989) 325–340.
[117] D.J. Mavriplis, Multigrid strategies for viscous flow solvers on anisotropic unstructured meshes, J. Comput. Phys. 145 (1998) 141.
[118] D.J. Mavriplis, An assessment of linear versus non-linear multigrid methods for unstructured mesh solvers, J. Comput. Phys. 175 (2002) 302–325.
[119] D.J. Mavriplis, V. Venkatakrishnan, Agglomeration multigrid for the two-dimensional viscous flows, Comp. Fluids 24 (1995) 533.
[120] S.F. McCormick, Multilevel Adaptive Methods for Partial Differential Equations, SIAM, Philadelphia, 1989.
[121] P.R. McHugh, D.A. Knoll, Fully implicit finite volume solutions of the incompressible Navier–Stokes and energy equations using inexact Newton's method, Int. J. Numer. Methods Fluids 18 (1994) 439–455.
[122] P.R. McHugh, D.A. Knoll, Inexact Newton's method solution to the incompressible Navier–Stokes and energy equations using standard and matrix-free implementations, AIAA J. 32 (1994).
[123] P.R. McHugh, D.A. Knoll, D.E. Keyes, Application of a Newton–Krylov–Schwarz algorithm to low Mach number combustion, AIAA J. 36 (1998) 290–292.
[124] P.R. McHugh, D.A. Knoll, V.A. Mousseau, G.A. Hansen, An investigation of Newton–Krylov solution techniques for low mach number compressible flow, in: Proceedings of the ASME Fluids Engineering Division Summer Meeting, 1995.
[125] J.C. Meza, R.S. Tuminaro, A multigrid preconditioner for the semiconductor equations, SIAM J. Sci. Comput. 17 (1996) 118–132.
[126] V. Mousseau, D.A. Knoll, J. Reisner, An implicit nonlinearly consistent method for the two-dimensional shallow-water equations with Coriolis force, Mon. Weather Rev. 130 (2002) 2611–2625.
[127] V.A. Mousseau, D.A. Knoll, Fully implicit kinetic solution of collisional plasmas, J. Comput. Phys. 136 (1997) 308–323.
[128] V.A. Mousseau, D.A. Knoll, New physics-based preconditioning of implicit methods for non-equilibrium radiation diffusion, J. Comput. Phys. (2003).
[129] V.A. Mousseau, D.A. Knoll, W.J. Rider, A multigrid Newton–Krylov solver for nonlinear systems, in: Multigrid Methods VI: Lecture Notes in Computational Science and Engineering, Springer, Berlin, 2000, pp. 200–206.
[130] V.A. Mousseau, D.A. Knoll, W.J. Rider, Physics-based preconditioning and the Newton–Krylov method for non-equilibrium radiation diffusion, J. Comput. Phys. 160 (2000) 743–765.
[131] L.S. Mukadi, R.E. Hayes, Modelling the three-way catalytic converter with mechanistic kinetics using the Newton–Krylov method on a parallel computer, Comput. Chem. Engrg. 26 (2002) 439–455.
[132] W. Mulder, B.V. Leer, Experiments with implicit upwind methods for the Euler equations, J. Comput. Phys. 59 (1985) 232–246.
[133] N.M. Nachtigal, S.C. Reddy, L.N. Trefethen, How fast are nonsymmetric matrix iterations?, SIAM J. Matrix Anal. Appl. 13 (1992) 778–795.
[134] M. Nemec, D.W. Zingg, Newton–Krylov algorithm for aerodynamic design using the Navier–Stokes equations, AIAA J. 40 (2002) 1146–1154.
[135] C.W. Oosterlee, T. Washio, An evaluation of parallel multigrid as a solver and a preconditioner for singularly perturbed problems, SIAM J. Sci. Stat. Comput. 19 (1998) 87–110.
[136] E.S. Oran, J.P. Boris, Numerical Simulation of Reactive Flow, Elsevier, New York, 1987.
[137] J. Ortega, W. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, Boston, 1970.
[138] S.V. Patankar, Numerical Heat Transfer and Fluid Flow, Hemisphere Publishing, New York, 1980.
[139] M. Pernice, A hybrid multigrid method for the steady-state incompressible Navier–Stokes equations, Elec. Trans. Numer. Anal. 10 (2000) 74–91.
[140] M. Pernice, Private communication, 2003.
[141] M. Pernice, M. Tocci, A multigrid-preconditioned Newton–Krylov method for the incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 23 (2001) 398–418.
[142] M. Pernice, H.F. Walker, NITSOL: a Newton iterative solver for nonlinear systems, SIAM J. Sci. Comput. 19 (1998) 302–318.
[143] M. Pernice, L. Zhou, H.F. Walker, Parallel solution of nonlinear partial differential equations using a globalized inexact Newton–Krylov–Schwarz method, Technical Report 48, University of Utah Center for High Performance Computing, 1997.
[144] G.D. Porter et al., Detailed comparison of simulated and measured plasma profiles in the scrape-off layer and edge plasma of DIII-D, Phys. Plasmas 7 (2000) 3663–3680.
[145] N. Qin, D.K. Ludlow, S.T. Shaw, A matrix-free preconditioned Newton/GMRES method for unsteady Navier–Stokes solutions, Int. J. Numer. Methods Fluids 33 (2000) 223–248.

[146] G.H.G.R.W. Freund, N.M. Nachtigal, Iterative solution of linear systems, in: Acta Numerica 1992, Cambridge University Press, Cambridge, 1992, pp. 57–100.
[147] P. Rasetarinera, M. Hussaini, An efficient implicit discontinuous spectral Galerkin method, J. Comput. Phys. 172 (2001) 718–738.
[148] J.K. Reid, On the method of conjugate gradients for the solution of large sparse systems of linear equations, in: J.K. Reid (Ed.), Large Sparse Sets of Linear Equations, Academic Press, New York, 1971, pp. 231–254.
[149] J. Reisner, V. Mousseau, D.A. Knoll, Application of the Newton–Krylov method to geophysical flows, Mon. Weather Rev. 129 (2001) 2404–2415.
[150] J. Reisner, S. Wynne, L. Margolin, R. Linn, Coupled atmospheric-fire modeling employing the method of averages, Mon. Weather Rev. 128 (2000) 3683–3691.
[151] J. Reisner, A. Wyszogrodzki, V. Mousseau, D.A. Knoll, An efficient physics-based preconditioner for the fully implicit solution of small-scale thermally driven atmospheric flows, J. Comput. Phys. 189 (2003) 30–44.
[152] W.J. Rider, D.A. Knoll, Time step size selection for radiation diffusion calculations, J. Comput. Phys. 152 (1999) 790–795.
[153] W.J. Rider, D.A. Knoll, G.L. Olson, A multigrid preconditioned Newton–Krylov method for multimaterial equilibrium radiation diffusion, J. Comput. Phys. 152 (1999) 164–191.
[154] T.D. Rognlien, J.L. Milovich, M.E. Rensink, G.D. Porter, A fully implicit, time-dependent 2-d fluid code for modeling tokamak edge plasmas, J. Nucl. Mater. 196–198 (1992) 347–351.
[155] T.D. Rognlien, X.Q. Xu, A.C. Hindmarsh, Application of parallel implicit methods to edge-plasma numerical simulations, J. Comput. Phys 175 (2002) 249–268.
[156] Y. Saad, A flexible inner–outer preconditioned GMRES algorithm, SIAM J. Sci. Stat. Comput. 14 (1993) 461–469.
[157] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS Publishing Company, Boston, 1996.
[158] Y. Saad, M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving non-symetric linear systems, SIAM J. Sci. Stat. Comput. 7 (1986) 856.
[159] Y. Saad, H.A. van der Vorst, Iterative solution of linear systems in the 20th century, J. Comp. Appl. Math. 123 (2000) 1–33.
[160] S. Schaffer, A semi-coarsening multigrid method for elliptic partial differential equations with highly discontinuous and anisotropic coefficients, SIAM J. Sci. Stat. Comput. 20 (1999) 228–242.
[161] A. Settari, K. Aziz, A generalization of the additive correction methods for the iterative solution of matrix equations, SIAM J. Numer. Anal. 10 (1973) 506–521.
[162] J.N. Shadid et al., Efficient parallel computation of unstructured finite element reacting flow solutions, Parallel Comput. 23 (1997) 1307–1325.
[163] J.N. Shadid, R.S. Tuminaro, A comparison of preconditioned nonsymmetric krylov methods on a large-scale mimd machine, SIAM J. Sci. Comput. 15 (1994) 440–459.
[164] J.N. Shadid, R.S. Tuminaro, H.F. Walker, An inexact Newton method for fully coupled solution of the Navier–Stokes equations with heat and mass transport, J. Comput. Phys. 137 (1997) 155–185.
[165] M.D. Smooke, Solution of burner-stabilized premixed laminar flames by boundary value methods, J. Comput. Phys. 48 (1982) 72–105.
[166] M.D. Smooke, R.M. Mattheij, On the solution of nonlinear two-point boundary value problems on successively refined grids, Appl. Numer. Math. 1 (1985) 463–487.
[167] M.D. Smooke, R.E. Mitchell, D.E. Keyes, Numerical solution of two-dimensional axisymmetric laminar diffusion flames, Combust. Sci. Technol. 67 (1989) 85–122.
[168] E. Suetomi et al., Two-dimensional fluid simulation of plasma reactors for the immobilization of krypton, Comput. Phys. Commun. 125 (2000) 60–74.
[169] M.D. Tidriri, Schwarz-based algorithms for compressible flows, Technical Report 96–4, ICASE, January 1996.
[170] M.D. Tidriri, Preconditioning techniques for the Newton–Krylov solution of compressible flows, J. Comput. Phys. 132 (1997) 51–61.
[171] M.D. Tidriri, Hybrid Newton–Krylov/domain decomposition methods for compressible flows, in: Proceedings of the Ninth International Conference on Domain Decomposition Methods in Sciences and Engineering, 1998, pp. 532–539.
[172] M.D. Tidriri, Development and study of Newton–Krylov–Schwarz algorithms, Int. J. Comput. Fluid Dyn. 15 (2001) 115–126.
[173] M.D. Tocci, C.T. Kelley, C.T. Miller, C.E. Kees, Inexact Newton methods and the method of lines for solving Richards' equation in two space dimensions, Comp. Geosci. 2 (1998) 291–309.
[174] U. Trottenberg, A. Schuller, C. Oosterlee, Multigrid, Academic Press, New York, 2000.
[175] R.S. Tuminaro, M. Heroux, S.A. Hutchinson, J.N. Shadid, Official Aztec user's guide, Technical Report SAND99-8801J, Sandia National Laboratory, December 1999.
[176] R.S. Tuminaro, H.F. Walker, J.N. Shadid, On backtracking failure in Newton-GMRES methods with a demonstration for the Navier–Stokes equations, J. Comput. Phys. 180 (2002) 549–558.
[177] K. Turner, H.F. Walker, Efficient high-accuracy solutions with GMRES(*m*), SIAM J. Sci. Stat. Comput. 13 (1992) 815–825.

[178] US Department of Energy, Scientific discovery through advanced computing, Technical Report, US Department of Energy, March 2000.
[179] H.A. van der Vorst, Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 13 (1992) 631–644.
[180] H.A. van der Vorst, C. Vuik, A comparison of some GMRES-like methods, Linear Algebra Appl. 160 (1992) 131–162.
[181] S.P. Vanka, G.K. Leaf, Fully coupled solution of pressure-linked fluid-flow equations, Technical Report ANL-83-73, Argonne National Laboratory, 1983.
[182] V. Venkatakrishnan, Newton solution of invisid and viscous problems, AIAA J. 27 (1989).
[183] V. Venkatakrishnan, Viscous computations using a direct solver, Comp. Fluids 18 (1990).
[184] V. Venkatakrishnan, D.J. Mavriplis, Agglomeration multigrid for the three-dimensional Euler equations, AIAA J. 33 (1995) 633.
[185] R.A. Vesey, D. Steiner, A 2-dimensional finite-element model of the edge plasma, J. Comput. Phys. 116 (1995) 300–313.
[186] H.X. Vu, Plasma collection by an obstacle, PhD thesis, California Institute of Technology, 1990.
[187] G. Wang, D.K. Tafti, Performance enhancement on microprocessors with hierarchical memory systems for solving large sparse linear systems, Int. J. Supercomput. Appl. High Perform. Comput. 13 (1999) 63–79.
[188] P. Wesseling, An Introduction to Multigrid Methods, John Wiley & Sons, Chichester, 1992.
[189] K.A. Winkler, M.L. Norman, D. Mihalas, Implicit adaptive-grid radiation hydrodynamics, in: J. Brackbill, B. Cohen (Eds.), Multiple Time Scales, Academic Press, New York, 1985.
[190] F. Wising, D.A. Knoll, S.I. Krasheninnikov, T.D. Rognlien, D.J. Sigmar, Simulation of the Alcator C-Mod divertor with an improved neutral fluid model, Contrib. Plasma Phys. 36 (1996) 136.
[191] S.J. Wright, J. Nocedal, Numerical Optimization, Springer, Berlin, 1999.
[192] X.Q. Xu, R.H. Cohen, T.D. Rognlien, J. Myra, Low-to-high confinement transition simulations in divertor geometry, Phys. Plasmas 7 (2000) 1951.
[193] R. Zanino, Advanced finite element modeling of the tokamak plasma edge, J. Comput. Phys. 138 (1997) 881–906.