Journal of Universal Computer Science, vol. 16, no. 6 (2010), 891-902
submitted: 6/10/09, accepted: 6/3/10, appeared: 28/3/10 © *J.UCS*

Newton Method for Nonlinear Dynamic Systems with Adaptive Time Stepping

Wensheng Shen
(State University of New York - Brockport, USA
wshen@brockport.edu)

Changjiang Zhang
(University of Kentucky, Lexington, USA
czhanb@csr.uky.edu)

Jun Zhang
(University of Kentucky, Lexington, USA
jzhang@cs.uky.edu)

Xiaoqian Ma
(South China University of Technology, Guangzhou, China
epxqma@scut.edu.cn)

Abstract: This paper presents a nonlinear solver based on the Newton-Krylov methods, where the Newton equations are solved by Krylov-subspace type approaches. We focus on the solution of unsteady systems, in which the temporal terms are discretized by the backward Euler method using finite difference. To save computational cost, an adaptive time stepping is used to minimize the number of time steps. The developed program can be applied to solve any nonlinear equations, provided the users could supply the discrete form of the equations. In particular, the nonlinear solver is implemented to solve unsteady reacting flows.
Key Words: Newton-Krylov method, nonlinear dynamics, diffusion flame, iterative solver
Category: J.2, J.6, G.1.8

1 Introduction

Developing efficient solvers for nonlinear system of equations arising from simulations in physical and chemical processes is still of big interest [Ern et al. 1995, Karaa et al. 2003, Noskov 2005, Shen et al. 2008, Tackenberg et al. 2009]. The difficulties associated with solving such problems stem from the large number of dependent unknowns, the nonlinear and coupling characteristics of the governing partial differential equations (PDEs), and the different length scales present in the problem [Nakamura et al. 2006]. Advanced computational techniques are in demand to solve these nonlinear systems accurately, efficiently, and robustly.

Newton's method is particularly robust in solving coupled nonlinear complex dynamic equations involving physics and chemistry, such as those describing reacting flows [Bennett and Smooke 1998, Smooke et al. 1989, Xu et al. 1993]. Theoretically, the Newton iteration converges quadratically under certain sufficient conditions, which means that the magnitude of the residual is squared with each Newton step so that it tends to zero very quickly. The disadvantage of Newton's method is that the size of the Jacobian matrix formed is extremely large and it is difficult to find a good initial guess. In order to stabilize the convergence at an early stage of the iteration and to save computing time, the damped Newton's method is used.

At each Newton step, a system of linear equations has to be solved, and the selection of linear system solver is not trivial. The splitting techniques of ADI and fractional-step are often used to solve multi-dimensional linear equations, however, they do not work well in situations that are highly nonlinear, such as the strongly coupled chemical reacting flow. As reported by Xu [Xu and Smooke 1993], difficulties had been encountered in obtaining a converged solution of the pressure and pressure correction equations when using the splitting method for simulating two-dimensional laminar diffusion flame. Current work focuses on the application of newly developed Krylov subspace iterative method to the solution of nonlinear problems that are modeled by PDEs.

In this work we are interested in developing a general purpose two-dimensional solver for nonlinear equations. It can be used to solve a variety of problems, as long as the discrete form of the equations are provided by the users. As an application, the program is applied to obtain the solution of laminar diffusion flames with one-step chemical reaction. The governing equations of this problem are highly nonlinear and the dependent variables are strongly coupled so that their solution constitutes a challenging test for nonlinear elliptic solvers.

2 Damped Newton Method

The Newton's method for a system of equations can be derived from multivariable Taylor expansion. Let $F(\mathbf{u}) = 0$ be the system of equations in residual form, obtained by discretizing the governing differential equations, the Taylor expansion of the left-hand side, $F(\mathbf{u})$, about the current location $\mathbf{u}^k$ is [Knoll and Keyes 2004]

$$\mathbf{F}(\mathbf{u}^{k+1}) = \mathbf{F}(\mathbf{u}^k) + \mathbf{F}'(\mathbf{u}^k)(\mathbf{u}^{k+1} - \mathbf{u}^k) + O\left((\mathbf{u}^{k+1} - \mathbf{u}^k)^2\right), \tag{1}$$

where, $\mathbf{F}$ is the nonlinear vector residual function and $\mathbf{u}$ is the vector of unknown dependent variables. By using the condition of $F(\mathbf{u}) = 0$, i.e., Eq. (1), and neglecting the second-order terms, the standard Newton iteration can be constructed as

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

or

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \delta\mathbf{u}^k = \mathbf{u}^k - \mathbf{J}(\mathbf{u}^k)^{-1}\mathbf{F}(\mathbf{u}^k), k = 0, 1, ..., \tag{3}$$

where $\mathbf{J} \equiv \mathbf{F}'$ is the Jacobian matrix, and k indicates the k-th Newton iteration. The damped Newton's method is consequently written in the following form:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \lambda^k\delta\mathbf{u}^k = \mathbf{u}^k - \lambda^k\mathbf{J}(\mathbf{u}^k)^{-1}\mathbf{F}(\mathbf{u}^k), k = 0, 1, ..., \tag{4}$$

where λ^k is the k-th damping parameter, taking as $0 < \lambda^k \leq 1$. Since there are no practical means to suggest an initial guess which is within the domain of convergence, damping is observed to be necessary for convergence [Xu et al. 1993]. The damping parameter is taking as $\lambda = 1$ at the first iteration and a minimum function defined by Eq. (5) at other iterations [Xu and Smooke 1993].

$$\lambda^k = \begin{cases} 1, & k = 0 \\ min(\lambda_p^k, 1), & k > 0 \end{cases} \tag{5}$$

where λ_p^k is calculated as

$$\lambda_p^k = \begin{cases} max\left|(\mathbf{u}_m^{max} - \mathbf{u}_{i,j,m}^k)/\delta\mathbf{u}_{i,j,m}^k\right|, & \delta\mathbf{u}_{i,j,m}^k > 0 \\ max\left|(\mathbf{u}_{i,j,m}^k - \mathbf{u}_m^{min})/\delta\mathbf{u}_{i,j,m}^k\right|, & \delta\mathbf{u}_{i,j,m}^k < 0 \end{cases} \tag{6}$$

If the projected update $\delta\mathbf{u}^{k+1}$ using Eq. (4) corresponding to λ^k is larger than the update at the kth iteration $\delta\mathbf{u}^k$, the damping parameter will be recursively taken a half until $\delta\mathbf{u}^{k+1} \leq \delta\mathbf{u}^k$, or until λ^k is smaller than predefined tolerance.

The Newton iteration is terminated when the 2-norm of the update $\delta\mathbf{u}$ satisfies the pre-determined convergence tolerance. To be specific, following the procedure described in [Ern et al. 1995], we scale each of the dependent variables in the nonlinear system such that each of them is of a size similar to the others of equal importance, and the 2-norm of the discrete vector $\delta\mathbf{u}^n$ can be written as [Ern et al. 1995]

$$\|\delta\mathbf{u}^n\| = \sqrt{\frac{1}{N}\sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3}\left(\frac{1}{\alpha_k}\delta\mathbf{u}^n\right)^2}, \tag{7}$$

where $N = n_1 n_2 n_3$, α_k is the scaling factor of the k-th component of the unknown vector, and n_1, n_2, and n_3 are the number of points in the first space direction, the number of points in the second space direction, and the number of elements of the unknown vector $\mathbf{u}$ respectively. In each of the Newton iteration, the inner linear system is solved by a Krylov type linear solver with preconditioners.

3 Implementation

3.1 Construction of Numerical Jacobian

The governing equations and boundary conditions are discretized by using an implicit finite-diffference technique on a nine-point stencil. The implicit formulation allows for 3 orders of magnitude fewer time steps for describing the complete transient, as compared to explicit formulations. Diffusion terms are approximated by centered differences and convective terms by a monotonicity-preserving upwind scheme. Thus, the partial differential equations are transformed into N_{eq} coupled nonlinear algebraic equations, where N_{eq} equals to the number of unknowns multiplied by the number of mesh points in the computational domain. The resulting system of equations, written in residual form, is solved by a damped Newton's method.

As discussed in Section 2, to solve the nonlinear system of equations with Newton's method, we have to perform function evaluation and compute the derivative function. For many nonlinear problems, the analytical calculation of Jacobian matrix may be very tedious in practice. As an alternative, we approximate the Jacobian matrix by forward difference. The Jacobian matrix in Eqs. (2) $\sim$ (4) is computed numerically through the following procedure. Assuming that there are k components in the unknown vector, i.e., $\mathbf{u} = [u_1\ u_2\ \dots\ u_k]^T$, we can denote the corresponding residual vector as $\mathbf{F} = [F_1\ F_2\ \dots\ F_k]^T$. At each mesh point (i,j), there is a corresponding dense square block with dimensions of $k \times k$,

$$\frac{\partial \mathbf{F}}{\partial \mathbf{u}} = \begin{bmatrix} \frac{\partial F_1}{\partial u_1} & \frac{\partial F_1}{\partial u_2} & \cdots & \frac{\partial F_1}{\partial u_k} \\ \frac{\partial F_2}{\partial u_1} & \frac{\partial F_2}{\partial u_2} & \cdots & \frac{\partial F_2}{\partial u_k} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_k}{\partial u_1} & \frac{\partial F_k}{\partial u_2} & \cdots & \frac{\partial F_k}{\partial u_k} \end{bmatrix}. \tag{8}$$

Each element in the above $k \times k$ block is evaluated by finite difference approximation, which can avoid the computation of the derivative and provide savings in linear algebra work, but pays the price of slower rate of convergence [Kelley 2003]. The finite difference approximation to the jth column of the Jacobian matrix can be written as [Xu et al. 1993]

$$\frac{\partial F_i}{\partial u_j} = \frac{F_i(u_j + \delta u_j) - F_i(u_j)}{\delta u_j}, \tag{9}$$

where δu_j is a small perturbation of the jth element of vector u, and it is implemented as $\delta u_j = \xi u_j + \eta$, where ξ and η are taking as the square root of the machine precision. One new function evaluation is required for each column of Jacobian J_j, so a total of N function evaluations are resulted for an unknown vector of size N.

The cost of constructing the Jacobian is substantial. Due to the sparse structure of Jacobian, the total number of function evaluations can be reduced by applying a vector function evaluation technique, where several columns of the Jacobian are evaluated simultaneously. After the Jacobian is obtained, the Newton equation may be solved by any linear algebraic solvers, which will be discussed later.

3.2 Adaptive Time Stepping

The unsteady nonlinear system of equations in discrete form can be written as

$$C\frac{\partial u}{\partial t} + F(u) = 0, \tag{10}$$

where C is a scaling matrix. Using backward Euler discretization of the temporal term, we obtain the following equation

$$C\frac{u^n - u^{n-1}}{t^n - t^{n-1}} + F(u^n) = 0, \tag{11}$$

where u^n and u^{n-1} represents the unknown vectors at time step n and $n-1$ respectively. The nonlinear system of equations, Eq (11), has to be solve by Newton's method at each time step. The new Jacobian matrix including unsteady terms is simply

$$J'(u^n) = \frac{C}{t^n - t^{n-1}} + J(u^n). \tag{12}$$

The unsteady term in Eq. (12) contributes to the diagonal terms in the Jacobian matrix, which makes the Jacobian more diagonally dominant than that of the steady-state equations. The resulting linear system is actually better conditioned and it can be solved more easily by an iterative method [Kelley 2003].

It is clear from Eq. (12) that a small time step will produce a better conditioned matrix. However, the selection of a small time step will substantially increase the computation time for a steady-state solution. Therefore, the length of time steps are adaptively chosen to minimize the number of time steps and save computation cost. The length of time steps is determined by monitoring the local truncation error of the time discretization. Following procedures in [Xu and Smooke 1993] by applying Taylor expansion, the local truncation error at the n^{th} time step due to backward Euler discretization can be approximated as

$$T_e \approx \frac{\gamma(\Delta t^{n+1})^2}{2} = \varepsilon, \tag{13}$$

where ε is a specified tolerance, and γ is calculated as

$$\gamma = max\left|\frac{\partial^2 u}{\partial t^2}\right|_{t^n \leq t \leq t^{n+1}}. \tag{14}$$

The second order partial derivative in Eq. (14) is approximated in the calculation by taking central difference at the time step t^{n-1}

$$\gamma = max \left| \frac{2}{t^n - t^{n-2}} \left(\frac{u_{ij}^n - u_{ij}^{n-1}}{t^n - t^{n-1}} - \frac{u_{ij}^{n-1} - u_{ij}^{n-2}}{t^{n-1} - t^{n-2}} \right) \right| \tag{15}$$

where i and j indicate the node position in the x and y directions respectively. The new time step at time level $n+1$ can then be predicted as

$$\Delta t^{t+1} = \sqrt{\frac{2\varepsilon}{\gamma}}. \tag{16}$$

3.3 Numerical Algorithm

Numerical methods described in the paper for solving multi-dimensional multi-variable unsteady nonlinear equations are summarized in **Algorithms** 1 and 2, where **Algorithm** 1 expresses the procedure of Newton iteration with damping parameters, while **Algorithm** 2 presents the steps of time iteration, in which the length of time steps is adaptively calculated based on the local truncation error.

4 Krylov-subspace based Method

Krylov-subspace based iterative methods build the iteration by evaluating matrix-vector product in the appropriate Krylov subspace. In Krylov method, the solution to a linear system $Ax = b$ is expressed as a summation of the following form [Kelley 2003]

$$x_k = x_0 + \sum_{i=0}^{k-1} \gamma_i A^i r_0, \tag{17}$$

where $r_0 = b - Ax_0$ and x_0 is the initial iterate, $x_k \in \mathcal{K}_k$, and the kth Krylov subspace is defined as

$$\mathcal{K}_k = span(r_0, Ar_0, \ldots, A^{k-1} r_0). \tag{18}$$

Among them, the most successful general purpose Krylov iterative methods for solving non-symmetrical matrices include GMRES (generalized minimal residual method) [Saad 1996] and BiCGSTAB (bi-conjugate gradient stabilized method) [Vorst 1992]. GMRES method has robust theoretical properties, but its major drawback is that the required storage at each iteration increases linearly with the iteration count. On the contrary, BiCGSTAB has the advantage of fixed storage throughout the linear iteration, but may break down due to a division by zero [Kelley 2003]. Since there might be a significant amount of unknown variables in a nonlinear system, storage could be a big concern. As a result, both GMRES and BiCGSTAB are implemented as the linear solvers, preconditioned with block Gauss-Seidel [Ern et al. 1995].

Algorithm 1 Newton iteration

provide a guessed solution, set $convergence = false$ and $k = 0$
while $k < maxNumberIter$ or $convergence == false$ **do**
  $k++$
  construct Jacobian using Eq. (8)
  Solve Eq. (2) for δu^k using preconditioned Krylov methods
  calculate damping parameters using Eq. (5)
  compute new solution by Eq. (4)
  project δu^{k+1} using existing Jacobian and Eq. (2)
  if $\delta u^{k+1} \leq \varepsilon$ **then**
    set $convergence = true$
    break
  end if
  while $\delta u^{k+1} \geq \delta u^k$ **do**
    set $\lambda = \frac{1}{2}\lambda$
    if $\lambda < \lambda_{min}$ **then**
      set $convergence = false$
      set $stop = true$
      break
    end if
    project δu^{k+1} using existing Jacobian and Eq. (2)
  end while
  if $stop == true$ **then**
    break
  end if
end while

Algorithm 2 Unsteady Newton

specify Δt and t_{max}
let $t = 0$, $t = t + \Delta t$
while $t < t_{max}$ **do**
  call Newton iteration for the unsteady equations (Eq. (12))
  while $convergence == false$ **do**
    let $\Delta t = \frac{1}{2}\Delta t$
    call Newton iteration for the unsteady equations (Eq. (12))
  end while
  set $t = t + \Delta t$
  calculate the next adaptive time step using Eq. (16)
end while

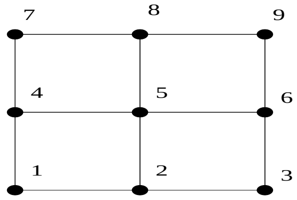


Figure 1: A sample 9-point grid.

5 Numerical Example

The Newton solver is applied to solve high-nonlinear diffusion flame, which is modeled by the Navier-Stokes equations and the transport equation of mixture fraction [Ern et al. 1995, Xu et al. 1993].

Continuity equation[Tannehill et al. 1997]

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0. \tag{19}$$

Momentum equation[Tannehill et al. 1997]

$$\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v}) = \rho \mathbf{g} - \nabla p + \nabla \cdot \left(\mu \left(\nabla \mathbf{v} + (\nabla \mathbf{v})^T - \frac{2}{3} \mathbf{I} (\nabla \cdot \mathbf{v}) \right) \right). \tag{20}$$

Mixture fraction equation[Xu et al. 1993]

$$\frac{\partial (\rho \phi)}{\partial t} + \nabla \cdot (\rho \mathbf{v} \phi) = \nabla \cdot (\rho D \nabla \phi). \tag{21}$$

The configuration of the jet flow in this paper is similar to that reported in [Mohammed et al. 1998]. The fuel jet and the coflowing oxidizer jet are co-centered, where the inner fuel jet has a radius of $R_I = 0.2$ cm, and the outer oxidizer jet has a radius of $R_O = 2.5$ cm. The fuel is nitrogen-diluted consisting of CH_4 and N_2, where the mass fraction of CH_4 is about 0.52. The oxidizer is the air, where the mass fraction of O_2 is 0.232. Both the fuel and the oxidizer are set with a velocity of $v_I = v_O = 35$ cm/s. The computational domain expands from $r = 0$ to $r = 7.5$ cm in the radial direction and from $z = 0$ to $z = 30$ cm in the axial direction, such that the radial dimension of the domain is much larger than the radius of the coflowing oxidizer jet, R_O, and the axial dimension of the domain is much larger than the flame length [Ern et al. 1995], L_f. The computational domain is covered by a non-uniformly distributed mesh of size 129×161. Using the simplest 3×3 mesh as an example (shown in Fig. 1), for the 9 point stencil, the structure of the Jacobian matrix is presented in Eq. (22).

$$\mathbf{J} = \begin{bmatrix} \square & \square & & & \square & \square & & & \\ \square & \square & \square & \square & \square & \square & & & \\ & \square & \square & & \square & \square & & & \\ \square & \square & & \square & \square & & \square & \square & \\ \square & \square & \square & \square & \square & \square & \square & \square & \square \\ & \square & \square & & \square & \square & & \square & \square \\ & & & \square & \square & & \square & \square & \\ & & & \square & \square & \square & \square & \square & \square \\ & & & & \square & \square & & \square & \square \end{bmatrix} \tag{22}$$

As discussed in Section 3.1, if there are k dependent variables, each element in the above matrix would contain a $k \times k$ dense block, as shown in Eq. (8). Let n_c, n_r, and n_z denote the number of components, number of points in the radial direction, number of points in the axial direction respectively, then the number of rows or the number of unknowns is $n_c \times n_r \times n_z$, the maximum number of nonzero elements in each row is $9 \times n_c$, and the maximum number of nonzeros in the Jacobian matrix is $9 \times n_c^2 \times n_r \times n_z$.

Following the flame sheet model, the one-step irreversible reaction for the methane-air diffusion flame can be written in the following form, as shown in [Bennett and Smooke 1998, Xu et al. 1993],

$$\nu_{CH_4} CH_4 + \nu_{O_2} O_2 + N_2 \rightarrow \nu_{H_2O} H_2O + \nu_{CO_2} CO_2 + N_2, \tag{23}$$

where the stoichiometric coefficients ν_{CH_4}, ν_{O_2}, ν_{H_2O}, and ν_{CO_2}, are taken as 1, 2, 2, and 1 respectively. We use this equation to recover the temperature and the mass fraction of major species. Under the assumption of infinite rate chemistry, the reaction zone becomes infinitely thin under the stoichiometric condition [Ern et al. 1995]. In three dimensions the infinitely thin reaction zone forms a surface called stoichiometric surface, while in two dimensions, the reaction zone forms a curve. The combustion domain has been separated into two parts: the fuel side where oxidizer is absent and the oxidizer side where fuel is absent. At the stoichiometric surface, the mass fractions of both fuel and oxidizer are zero, i.e., $Y_{CH_4} = Y_{O_2} = 0$, so the stoichiometric mixture fraction is constant there, and determined as [Xu et al. 1993]

$$\phi_s = \frac{1}{1 + \frac{W_{CO_2} \nu_{CO_2}}{W_{CH_4} \nu_{CH_4}} \frac{(Y_{CH_4})_f}{(Y_{CO_2})_o}}, \tag{24}$$

where the subscript s, o, and f indicates the stoichiometric surface, oxidizer side, and fuel side respectively. We have to keep track of the location of the stoichiometric surface to determine the scopes of fuel side region and oxidizer side region, since different mechanisms are used in computing the temperature and the mass fraction of major species in different regions. For example, on the fuel side, we have

$$T = T_f\phi + \left(T_o + (Y_{CO_2})_o \frac{Q}{c_p}\frac{W_{CH_4}\nu_{CH_4}}{W_{O_2}\nu_{O_2}}\right)(1-\phi), \tag{25}$$

$$Y_{CH_4} = (Y_{CH_4})_f\phi + (Y_{O_2})_o\frac{W_{CH_4}\nu_{CH_4}}{W_{O_2}\nu_{O_2}}(\phi - 1), \tag{26}$$

and on the oxidizer side, we have

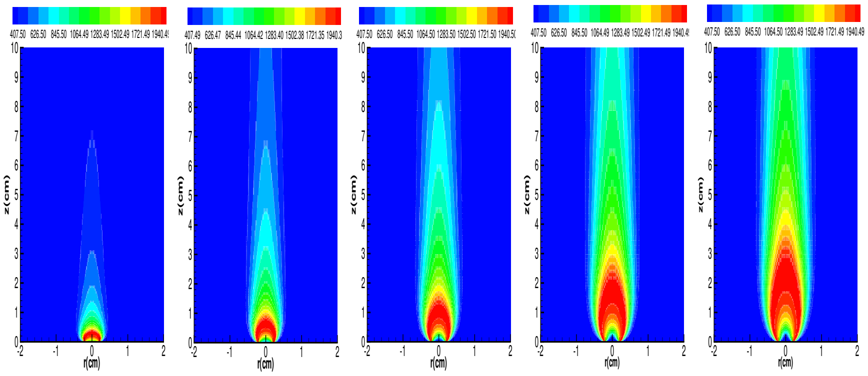
$$T = T_o(1-\phi) + \left(\frac{Q}{c_p}(Y_{CH_4})_f + T_f\right)\phi, \tag{27}$$

$$Y_{O_2} = (Y_{O_2})_o(1-\phi) - (Y_{CH_4})_f\frac{W_{O_2}\nu_{O_2}}{W_{CH_4}\nu_{CH_4}}\phi. \tag{28}$$

The mass fraction of other species can be obtained in a similar way, as presented in [Ern et al. 1995, Xu et al. 1993]. In Eqs. (25) $\sim$ (28), Q is the heat release per unit mass of the fuel, c_p the specific heat of the mixture, W the molecular weight of each species. In the mixture fraction equation, fluid properties, such as the viscosity, the mixture density, the mixture diffusivity, are temperature dependent. After the temperature is obtained, the related properties can be found as follows. The temperature dependence of viscosity is approximated by the power law [Xu et al. 1993] $\mu = \mu_0\left(\frac{T}{T_0}\right)^r$, where $T_0 = 298$ K, $r = 0.7$, and $\mu_0 = 1.857\times10^{-4}$ gm/cm. From the definition of the Prandtl number $Pr = \frac{\mu c_p}{k}$, we obtain the relation of $\frac{c_p}{k} = \frac{Pr}{\mu}$. The Lewis number is assumed to be one $Le = \frac{k}{\rho D c_p} = 1$, and we further obtain $\rho D = \frac{k}{c_p} = \frac{\mu}{Pr}$, where k is the thermal conductivity of the mixture, and the Prandtl number is taken as $Pr = 0.75$.

The nonlinear coupled equations of the combustion problems are very difficult to solve. A good initial solution guess is absolutely necessary for the convergence of the Newton's method and very important for fast solution of the iterative process. In the current investigation, the initial guess is set to zero for u, v, and ϕ, and 298 K for T. The Newton iteration is considered to have converged if the 2-norm of the update of the scaled dependent variable, defined in Eq. (7), is less than 1.0×10^{-4}, and GMRES or BiCGSTAB is considered to have converged when the 2-norm of the scaled residual vector is less than or equal to $\frac{1}{10}$ of the Newton tolerance [Ern et al. 1995].

The computation is performed on a Sun-Blade-100 machine with a single 500 MHz SPARC processor and 2 GB memory. The evolution of the laminar diffusion flame with time is presented in Fig. 2, which shows the development of flame temperature profile in the early stage of a flame. At $t = 0.125$ s, the obtained temperature profile of the flame is close to the steady-state solution, with highest temperature of 2050 K and highest axial velocity of 2.61 m/s. The unsteady diffusion flame simulated in the paper agrees qualitatively well with that in [Mohammed et al. 1998]. As expected, it has been observed in several



(a) (b) (c) (d) (e)

Figure 2: The dependency of flame temperature on time: (a) $t = 0.025$ s, (b) $t = 0.05$ s, (c) $t = 0.075$ s, (d) $t = 0.1$ s, and (e) $t = 0.125$ s.

running cases that about 50% $\sim$ 80% computational time has been saved due to the introduction of adaptive time stepping.

6 Conclusions and Future Work

A two-dimensional nonlinear solver has been developed to solve coupled differential equations using damped Newton's method. The program has been applied to find the solution of a diffusion flame, where the air flow is coupled with chemical reactions, and reasonable results are obtained. This indicates that the Newton's method is very efficient for the simulation of coupled high nonlinear physical processes such as the complicated combustion problems. To make the Newton iterations more robust, Krylov-subspace based iterative methods have been used to solve the linear system of equations at each Newton step. The backward Euler method has been used to discretize the unsteady terms, and adaptive time stepping has been applied to save computation time. In future study, we would like to further speed up the steady-state solution of diffusion flame by using multiscale multigrid high performance computing techniques.

References

[Bennett and Smooke 1998] B. Bennett and M. Smooke, "Local rectangular refinement with application to axisymmetric laminar flames", *Combust. Theory Modelling*, **2**, 221-258 (1998)

[Ern et al. 1995] A. Ern, C.C. Douglas, M.D. Smooke, "Detailed chemistry modeling of laminar diffusion flames on parallel computers" *Int J Supercomput Appl High Perf Comput*, **9**, 167-186 (1995)

[Karaa et al. 2003] S. Karaa, J. Zhang, and C.C. Douglas, "Preconditioned multigrid simulation of an axisymmetric laminar diffusion flame" *Math. Comput. Modeling*, **38**, 269-279 (2003)

[Kelley 2003] C.T. Kelley *Solving Nonlinear Equations with Newton's Method*, SIAM, 2003.

[Knoll and Keyes 2004] D.A. Knoll and D.E. Keyes Jacobian-free Newton-Krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, **193**, 357-397 (2004)

[Mohammed et al. 1998] R. Mohammed, M. Tanoff, M.D. Smooke, and A. Schaffer, "Computational and experimental study of a forced, time-varying, axisymmetric, laminar diffusion flame", In *Twenty-Seventh Symposium (International) on Combustion*, The Combustion Institute, 603-702 (1998)

[Nakamura et al. 2006] K. Nakamura, T. Higuchi, and N. Hirose, "Sequential data assimilation: information fusion of numerical simulation and large scale observation data", *Journal of Universal Computer Science*, **12**, 608-626, 2006.

[Noskov 2005] M. Noskov "An implicit compact scheme solver with application to chemically reacting flows", *J. of Comput. Phys.*, **203**, 700-730 (2005)

[Saad 1996] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Pub., New York (1996)

[Shen et al. 2008] W. Shen, C. Zhang, M. Fannon, K. Forsten-Williams, and J. Zhang, "A computational model of FGF-2 binding and HSPG regulation under flow", *IEEE Trans. on Biomed. Eng.*, **56**, 2147-2155 (2009)

[Smooke et al. 1989] M.D. Smooke, R.E. Mitchell, and D.E. Keyes, "Numerical solution of two-dimensional axisymmetric laminar diffusion flames", *Combustion Science and Technology*, **67**, 85-122 (1989)

[Tackenberg et al. 2009] S. Tackenberg, B. Kausch, S. Duckwitz, C.M. Schlick, and S. Karahancer, "Organizational simulation of complex process engineering projects in the chemical industry", *Journal of Universal Computer Science*, **15**, 1746-1765 (2009).

[Tannehill et al. 1997] J.C. Tannehill, R.H. Pletcher, D.A. Anderson, R.H. Pletcher, *Computational Fluid Mechanics and Heat Transfer*, 2nd edition, Hemisphere Pub., 1997.

[Vorst 1992] 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**, 631-644 (1992)

[Xu and Smooke 1993] Y. Xu and M.D. Smooke, "Application of a primitive variable Newton's method for the calculation of an axisymmetric laminar diffusion flame", *J. of Comput. Phys.*, **104**, 99-109 (1993)

[Xu et al. 1993] Y. Xu, M.D. Smooke, P. Lin, M.B. Long, "Primitive variable modeling of multidimensional laminar flames", *Combust. Sci. and Tech.*, **90**, 289-313 (1993)