Copyright © 2011 Tech Science Press CMES, vol.71, no.3, pp.279-304, 2011

Simple "Residual-Norm" Based Algorithms, for the Solution of a Large System of Non-Linear Algebraic Equations, which Converge Faster than the Newton's Method

Chein-Shan Liu[1] **and Satya N. Atluri**[2]

Abstract: For solving a system of nonlinear algebraic equations (NAEs) of the type: $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, or $F_i(x_j) = 0,\ i,j = 1,\ldots,n$, a Newton-like algorithm has several drawbacks such as local convergence, being sensitive to the initial guess of solution, and the time-penalty involved in finding the inversion of the Jacobian matrix $\partial F_i/\partial x_j$. Based-on an invariant manifold defined in the space of $(\mathbf{x},t)$ in terms of the residual-norm of the vector $\mathbf{F}(\mathbf{x})$, we can derive a gradient-flow system of nonlinear ordinary differential equations (ODEs) governing the evolution of $\mathbf{x}$ with a fictitious time-like variable t as an independent variable. We can prove that in the present novel Residual-Norm Based Algorithms (RNBAs), the residual-error is automatically decreased to zero along the path of $\mathbf{x}(t)$. More importantly, we have derived three *iterative algoritms* which do not involve the fictitious time and its stepsize Δt. We apply the three RNBAs to several numerical examples, revealing exponential convergences with different slopes and displaying the high efficiencies and accuracies of the present iterative algorithms. All the three presently proposed RNBAs: (i) are easy to implement numerically, (ii) converge much faster than the Newton's method, (iii) do not involve the inversion of the Jacobian $\partial F_i/\partial x_j$, (iv) are suitable for solving a large system of NAEs, and (v) are purely iterative in nature.

Keywords: Nonlinear algebraic equations, Non-Linear Ordinary Differential Equations, Non-Linear Partial Differential Equations, Residual-Norm Based Algorithms (RNBAs), Fictitious time integration method (FTIM), Iterative algorithm

[1] Department of Civil Engineering, National Taiwan University, Taipei, Taiwan. E-mail: liucs@ntu.edu.tw

[2] Center for Aerospace Research & Education, University of California, Irvine

1 Introduction

In many practical nonlinear engineering problems, governed by ordinary or partial differential equations, the methods such as the finite element, boundary element, finite volume discretization, and the meshless method, etc., eventually lead to a system of nonlinear algebraic equations (NAEs). Many numerical methods used in computational mechanics, as demonstrated by Zhu, Zhang and Atluri (1998), Atluri and Zhu (1998a), Atluri (2002), Atluri and Shen (2002), and Atluri, Liu and Han (2006) lead to the solution of a system of linear algebraic equations for a linear problem, and of a NAEs system for a nonlinear problem.

Over the past forty years, two important contributions have been made towards the numerical solutions of NAEs. One of these methods has been called the “predictor-corrector" or “pseudo-arclength continuation" method. This method has its historical roots in the embedding and incremental loading methods which have been successfully used for several decades by engineers to improve the convergence properties when an adequate starting value for an iterative method is not available. Another is the so-called simplical or piecewise linear method. The monographs by Allgower and Georg (1990) and Deuflhard (2004) are devoted to the continuation methods for solving NAEs.

Liu and Atluri (2008) have pioneered a Fictitious Time Integration Method (FTIM) to solve a large system of NAEs, and Liu and his coworkers showed that high performance can be achieved by using the FTIM [Liu (2008, 2009a, 2009b, 2009c); Liu and Chang (2009)]. The FTIM whilst robust, still suffers from the drawbacks: (i) even though it does not involve the computations of $\partial F_i(x_j)/\partial x_j$ and the inversion of the Jacobian matrix $\partial F_i(x_j)/\partial x_j$ of $F_i(x_j) = 0,\ i, j = 1,\dots,n$, FTIM is still slow to converge; (ii) the convergence is only local; (iii) it still involves a time-step Δt in integrating the ODEs which are used as surrogates in solving the NAEs; (iv) it is not simply iterative in nature.

The aims of the present paper are to develop: (1) methods which converge faster than the Newton's method for solving a system of NAEs, simply based on the scalar norm of the residual error in solving $\mathbf{F}(\mathbf{x}) = \mathbf{0}$; (2) a method which does not involve the inversion of $\partial F_i(x_j)/\partial x_j$; (3) a method which is globally convergent, and (4) a method which is purely iterative in nature and does not involve actually integrating a system of ODEs, with a time step Δt in numerical schemes, such as GPS (group preserving scheme) developed by Liu (2001).

The remainder of this paper is arranged as follows. Some evolution methods for solving NAEs are briefly sketched in Section 2. In Section 3 we give a theoretical basis of the RNBA. We start from a continuous manifold defined in terms of residual-norm, and arrive at a system of ODEs using a "normality condition". Section 4 is devoted to deriving a scalar equation to keep $\mathbf{x}$ on the manifold, and then we propose *three strategies* to select the weighting factors for the regularized "gradient vector", which automatically have a convergent behavior of the residual-error curve. In Section 5 we give four numerical examples to test the RNBAs with different weighting factors. Finally, the iterative algorithms are summarized in Section 6, and the advantages of the newly developed RNBAs are emphasized.

2 Different evolution methods

We consider a system of nonlinear algebraic equations (NAEs) in their vector form:

$$\mathbf{F}(\mathbf{x}) = \mathbf{0}. \tag{1}$$

In order to eliminate the need for inverting a matrix in the iteration procedure, Ramm (2007) has proposed a lazy-bone method based on the following evolution equation:

$$\dot{\mathbf{x}} = -\mathbf{F}(\mathbf{x}), \tag{2}$$

which in general leads to a divergence of the solution.

Liu and Atluri (2008) have proposed another first-order system of nonlinear ODEs:

$$\dot{\mathbf{x}} = -\frac{\nu}{q(t)}\mathbf{F}(\mathbf{x}), \tag{3}$$

where ν is a nonzero constant and $q(t)$ may in general be a monotonically increasing function of t. In their approach, the term $\nu/q(t)$ plays a major role of a stabilized controller to help one obtain a solution even for a bad initial guess of solution, and speed up the convergence. Liu and Chang (2009) combined it with a nonstandard group preserving scheme for solving a system of ill-posed linear equations. Ku, Yeih, Liu and Chi (2009) employed a time-like function of $q(t) = (1+t)^m$, $0 < m \le 1$ in Eq. (3), and better performance was observed. In spite of its success, the FTIM has only a local convergence and needs to determine the viscous damping coefficients for different equations in the same problem.

To remedy the shortcoming of the vector homotopy method as initiated by Davidenko (1953), Liu, Yeih, Kuo and Atluri (2009) have proposed a scalar homotopy method with the following evolution equation:

$$\dot{\mathbf{x}} = -\frac{\frac{\partial h}{\partial t}}{\|\frac{\partial h}{\partial \mathbf{x}}\|^2}\frac{\partial h}{\partial \mathbf{x}}, \tag{4}$$

where

$$h(\mathbf{x},t) = \frac{1}{2}[t\|\mathbf{F}(\mathbf{x})\|^2 - (1-t)\|\mathbf{x}\|^2], \tag{5}$$

$$\frac{\partial h}{\partial t} = \frac{1}{2}[\|\mathbf{F}(\mathbf{x})\|^2 + \|\mathbf{x}\|^2], \tag{6}$$

$$\frac{\partial h}{\partial \mathbf{x}} = t\mathbf{B}^{\mathrm{T}}\mathbf{F} - (1-t)\mathbf{x}, \tag{7}$$

in which $\mathbf{B}$ is the Jacobian matrix with its ij-component being given by $B_{ij} = \partial F_i/\partial x_j$. This method has global convergence; however, the convergence speed is quite slow. Ku, Yeih and Liu (2010) combined this idea with an exponentially decaying scalar homotopy function, and developed a manifold-based exponentially convergent algorithm (MBECA), based on integrating a system of nonlinear ODEs:

$$\dot{\mathbf{x}} = -\frac{\nu}{(1+t)^m}\frac{\|\mathbf{F}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}\mathbf{B}^{\mathrm{T}}\mathbf{F}. \tag{8}$$

Two major drawbacks appear in the MBECA: *irregular bursts and flattened behavior appear in the trajectory of the residual-error*.

Before the derivation of our new algorithms, we also mention that Hirsch and Smale (1979) have derived a continuous Newton method governed by the following differential equation:

$$\dot{\mathbf{x}}(t) = -\mathbf{B}^{-1}(\mathbf{x})\mathbf{F}(\mathbf{x}). \tag{9}$$

It can be seen that the ODEs in Eq. (9) are difficult to calculate, because they involve an inverse matrix $\mathbf{B}^{-1}$. Usually it is difficult to derive a closed-form solution of Eq. (9); hence a numerical scheme, such as the Euler method, can be employed to integrate Eq. (9). For the Newton algorithm we can derive

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) - \Delta t\mathbf{B}^{-1}\mathbf{F}, \tag{10}$$

$$\dot{\mathbf{F}} = \mathbf{B}\dot{\mathbf{x}} = -\mathbf{F}, \tag{11}$$

$$\mathbf{F}(t+\Delta t) = \mathbf{F}(t) - \Delta t\mathbf{F}(t), \tag{12}$$

$$\frac{\|\mathbf{F}(t+\Delta t)\|}{\|\mathbf{F}(t)\|} = 1 - \Delta t, \tag{13}$$

where Δt is a time stepsize used in the Euler scheme. The last equation means that the ratio of two consecutive residual errors as given in Eq. (13) is $1-\Delta t$ for the Newton algorithm. All the above methods require to specify some parameters, such as ν, m and the time stepsize Δt used in the numerical integrations.

In this paper we introduce novel and very simple iterative "Residual-Norm Based Algorithms (RNBAs)", which can be easily implemented to solve NAEs, and thereby a nonlinear system of partial differential equations when suitably discretized. The present RNBA can overcome the two major drawbacks as observed in the MBECA: *no irregular bursts and no flattened behavior appear in the trajectory of the residual-error*.

3 Theoretical basis–invariant manifold

We define a scalar function h, depending on the "Residual-Norm" in the error of $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, and a monotonically increasing function $Q(t)$, where t is a fictitious time-like parameter:

$$h(\mathbf{x},t) := \frac{1}{2}Q(t)\|\mathbf{F}(\mathbf{x})\|^2, \tag{14}$$

and define a surface

$$h(\mathbf{x},t) - C = 0. \tag{15}$$

This equation prescribes an invariant manifold in the space of $(\mathbf{x},t)$. By the above implicit function we in fact have required $\mathbf{x}$ to be a function of a fictitious time variable t. We do not need to specify the function $Q(t)$ *a priori*, but $\sqrt{2C/Q(t)}$ merely acts as a measure of the residual error in Eq. (1) in time. Hence, we impose in our algorithm that $Q(t) > 0$ is a monotonically increasing function of t. We let $Q(0) = 1$, and C to be determined by the initial condition $\mathbf{x}(0) = \mathbf{x}_0$ with

$$C = \frac{1}{2}\|\mathbf{F}(\mathbf{x}_0)\|^2. \tag{16}$$

Usually, $C > 0$, and $C = 0$ when the initial value $\mathbf{x}_0$ is just exactly the solution of Eq. (1). However, it is rare if this lucky coincidence happens.

We expect $h(\mathbf{x},t) - C = 0$ to be an invariant manifold in the space of $(\mathbf{x},t)$ for a dynamical system $h(\mathbf{x}(t),t) - C = 0$ to be specified further. When $C > 0$ and $Q > 0$, the manifold defined by Eq. (15) is continuous, and thus the following differential operation carried out on the manifold makes sense. As a "consistency condition", by taking the time differential of Eq. (15) with respect to t and considering $\mathbf{x} = \mathbf{x}(t)$, we have

$$\dot{h} = \frac{1}{2}\dot{Q}(t)\|\mathbf{F}(\mathbf{x})\|^2 + Q(t)(\mathbf{B}^{\mathrm{T}}\mathbf{F}) \cdot \dot{\mathbf{x}} = 0. \tag{17}$$

Eq. (17) cannot uniquely determine the evolution equation for $\mathbf{x}$; however, we assume a "normality condition" that

$$\dot{\mathbf{x}} = -\lambda \frac{\partial h}{\partial \mathbf{x}} = -\lambda Q(t)\mathbf{B}^{\mathrm{T}}\mathbf{F}, \tag{18}$$

where λ is to be determined. Inserting Eq. (18) into Eq. (17), we can solve for λ:

$$\lambda = \frac{\dot{Q}(t)\|\mathbf{F}\|^2}{2Q^2(t)\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}. \tag{19}$$

Thus we obtain an evolution equation for $\mathbf{x}$ defined by a "gradient-flow" or "normality-rule":

$$\dot{\mathbf{x}} = -q(t)\frac{\|\mathbf{F}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}\mathbf{B}^{\mathrm{T}}\mathbf{F}, \tag{20}$$

where

$$q(t) := \frac{\dot{Q}(t)}{2Q(t)}. \tag{21}$$

Hence, in our algorithm if $Q(t)$ can be guaranteed to be an increasing function of t, we may have an absolutely convergent property in solving the NAEs in Eq. (1):

$$\|\mathbf{F}(\mathbf{x})\|^2 = \frac{2C}{Q(t)}. \tag{22}$$

When t is large, the above equation will force the residual error $\|\mathbf{F}(\mathbf{x})\|$ to tend to zero, and meanwhile the solution of Eq. (1) is obtained approximately. Later in this paper, we prove that the ratio of residual errors derived from Eq. (20) is much better than that of the Newton algorithm in Eq. (13).

4 Dynamics of the present iterative algorithms

4.1 Discretizing, yet keeping $\mathbf{x}$ *on the manifold* $[h(\mathbf{x},t) - C = 0]$

Now we discretize the foregoing continuous time dynamics into discrete time dynamics:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) - \Delta t q(t)\frac{\|\mathbf{F}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}\mathbf{B}^{\mathrm{T}}\mathbf{F}, \tag{23}$$

which is obtained from the ODEs in Eq. (20) by applying the Euler scheme.

In order to keep $\mathbf{x}$ on the manifold defined by Eq. (22), we can consider the evolution of $\mathbf{F}$ along the path $\mathbf{x}(t)$ by:

$$\dot{\mathbf{F}} = \mathbf{B}\dot{\mathbf{x}} = -q(t)\frac{\|\mathbf{F}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}\mathbf{AF}, \tag{24}$$

where

$$\mathbf{A} := \mathbf{BB}^{\mathrm{T}}. \tag{25}$$

Suppose that we simply use the Euler scheme to integrate Eq. (24):

$$\mathbf{F}(t+\Delta t) = \mathbf{F}(t) - \Delta t q(t)\frac{\|\mathbf{F}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}\mathbf{AF}. \tag{26}$$

Taking the square-norms of both the sides of Eq. (26) and using Eq. (22), we can obtain

$$\frac{2C}{Q(t+\Delta t)} = \frac{2C}{Q(t)} - 2\Delta t\frac{2Cq(t)}{Q(t)}\frac{\mathbf{F}\cdot(\mathbf{AF})}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2} + (\Delta t)^2\frac{2Cq^2(t)}{Q(t)}\frac{\|\mathbf{F}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^4}\|\mathbf{AF}\|^2. \tag{27}$$

Thus we have the following scalar equation:

$$a(\Delta t)^2 - b\Delta t + 1 - \frac{Q(t)}{Q(t+\Delta t)} = 0, \tag{28}$$

where

$$a := q^2(t)\frac{\|\mathbf{F}\|^2\|\mathbf{AF}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^4}, \tag{29}$$

$$b := 2q(t). \tag{30}$$

As a result $h(\mathbf{x},t) - C = 0$, $t \in \{0,1,2,\ldots\}$ remains to be an invariant manifold in the space of $(\mathbf{x},t)$ for discrete time dynamical systems $h(\mathbf{x}(t),t) - C = 0$, which will be explored further in the next section.

4.2 Three simple and novel algorithms

Now we specify the discrete time dynamics $h(\mathbf{x}(t),t) = C$, $t \in \{0,1,2,\ldots\}$, through specifying the discrete time dynamics of $Q(t)$, $t \in \{0,1,2,\ldots\}$. Note that discrete time dynamics is an iterative dynamics, which in turn amounts to an iterative algorithm.

Inserting Eqs. (29) and (30) into Eq. (28) we can derive

$$a_0(q\Delta t)^2 - 2(q\Delta t) + 1 - \frac{Q(t)}{Q(t+\Delta t)} = 0, \tag{31}$$

where

$$a_0 := \frac{\|\mathbf{F}\|^2\|\mathbf{AF}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^4} \geq 1, \tag{32}$$

because of the condition

$$\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2 = \mathbf{F}\cdot(\mathbf{AF}) \leq \|\mathbf{F}\|\|\mathbf{AF}\|$$

by using the Cauchy-Schwarz inequality.

In our previous experience when $Q(t)$ is fixed to be a given function, such as an exponential function of t, the resultant algorithm has some drawbacks as observed by Ku, Yeih and Liu (2010). Thus, we let $Q(t)$ to be unspecified here. Instead, we let $Q(t)$ to be a quantity automatically derived from the new algorithms.

From Eq. (31) we let

$$s = a_0(q\Delta t)^2 - 2(q\Delta t) + 1 = \frac{Q(t)}{Q(t+\Delta t)}; \tag{33}$$

thus s signifies the ratio of $Q(t)/Q(t+\Delta t)$.

We search for a minimum of s with respect to Δt by setting to zero of the differential of Eq. (33) with respect to Δt:

$$\Delta t = \frac{1}{qa_0}. \tag{34}$$

Inserting it into Eq. (33) we can derive the minimum of s:

$$s = 1 - \frac{1}{a_0} < 1 \tag{35}$$

due to the fact that $a_0 \geq 1$ as shown in Eq. (32). The above property is very important. From Eqs. (22) and (33) it follows that

$$\frac{\|\mathbf{F}(t+\Delta t)\|}{\|\mathbf{F}(t)\|} = \sqrt{s}. \tag{36}$$

Thus, Eq. (35) means that the ratio of two consecutive residual errors is smaller than one:

$$\frac{\|\mathbf{F}(t+\Delta t)\|}{\|\mathbf{F}(t)\|} = \sqrt{1-a_0^{-1}} < 1. \tag{37}$$

Inserting the value of Δt from Eq. (34) into Eq. (23) we obtain the *first algorithm*:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) - \frac{1}{a_0}\frac{\|\mathbf{F}\|^2}{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}\mathbf{B}^{\mathrm{T}}\mathbf{F} = \mathbf{x}(t) - \frac{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}{\|\mathbf{A}\mathbf{F}\|^2}\mathbf{B}^{\mathrm{T}}\mathbf{F}. \tag{38}$$

It is interesting that in the above algorithm no parameters and no Δt are required. Furthermore, the property in Eq. (37) is very important, since it guarantees the new algorithm to be absolutely convergent to the true solution. Corresponding to the gradient vector $\mathbf{B}^{\mathrm{T}}\mathbf{F}$, we can understand that $\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2\mathbf{B}^{\mathrm{T}}\mathbf{F}/\|\mathbf{A}\mathbf{F}\|^2$ is a regularized gradient vector. In front of it, some weighting factor η may be placed to speed-up the convergence speed.

The above Δt in Eq. (34) may be too conservative. Thus we specify a certain constant $s = s_0 < 1$, and from Eq. (33) we have

$$a_0(q\Delta t)^2 - 2(q\Delta t) + 1 - s_0 = 0. \tag{39}$$

We can take the solution of Δt to be

$$\Delta t = \frac{1+\sqrt{1-(1-s_0)a_0}}{qa_0}, \text{ if } 1-(1-s_0)a_0 \geq 0, \tag{40}$$

$$\Delta t = \frac{1}{qa_0}, \text{ if } 1-(1-s_0)a_0 < 0. \tag{41}$$

Inserting the above two Δt's into Eq. (23) we can derive the *second algorithm*:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) - \eta\frac{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}{\|\mathbf{A}\mathbf{F}\|^2}\mathbf{B}^{\mathrm{T}}\mathbf{F}, \tag{42}$$

where

$$\eta = 1+\sqrt{1-(1-s_0)a_0}, \text{ if } 1-(1-s_0)a_0 \geq 0, \tag{43}$$

$$\eta = 1, \text{ if } 1-(1-s_0)a_0 < 0. \tag{44}$$

This algorithm includes a given parameter $s_0 < 1$, but does not need Δt, whose ratio of residual errors is given by

$$\frac{\|\mathbf{F}(t+\Delta t)\|}{\|\mathbf{F}(t)\|} = \sqrt{s_0}, \text{ if } 1-(1-s_0)a_0 \geq 0, \tag{45}$$

$$\frac{\|\mathbf{F}(t+\Delta t)\|}{\|\mathbf{F}(t)\|} = \sqrt{1-a_0^{-1}}, \text{ if } 1-(1-s_0)a_0 < 0. \tag{46}$$

In Eq. (38) the weighting factor η is $\eta = 1$. In contrast, the weighting factor η in Eq. (42) is larger or equal to 1.

Instead of the constant s_0, we may allow s to be a function of a_0. We observe that the following

$$s = 1 - \frac{1}{a_0^2} \tag{47}$$

automatically satisfies $1-(1-s)a_0 \geq 0$. Hence by solving Eq. (33) for Δt with the above s we can derive

$$\Delta t = \frac{1+\sqrt{1-a_0^{-1}}}{qa_0}, \tag{48}$$

and inserting it into Eq. (23) we can obtain the *third algorithm*:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) - \eta \frac{\|\mathbf{B}^{\mathrm{T}}\mathbf{F}\|^2}{\|\mathbf{AF}\|^2}\mathbf{B}^{\mathrm{T}}\mathbf{F}, \tag{49}$$

where the weighting factor η is given by

$$\eta = 1+\sqrt{1-a_0^{-1}} > 1. \tag{50}$$

This algorithm also does not involve specifying any parameter and time stepsize. The ratio of residual errors of this algorithm is

$$\frac{\|\mathbf{F}(t+\Delta t)\|}{\|\mathbf{F}(t)\|} = \sqrt{1-a_0^{-2}} < 1. \tag{51}$$

Below we give some numerical tests of the newly proposed Residual-Norm Based Algorithms (RNBAs), which are respectively labelled in this paper as **Algorithm 1**, **Algorithm 2** and **Algorithm 3**.

5 Numerical comparisons of three RNBAs

Before the comparisons of presently developed three algorithms, we must stress that these algorithms do not need the stepsize. However, in order to compare them with the Newton method we require Δt to be inserted into Eq. (13) to obtain the ratio of residual errors belong to the Newton scheme. Thus we use Eq. (34) to calculate Δt for **Algorithm 1**, Eqs. (40) and (41) to calculate Δt for **Algorithm 2**, and Eq. (48) to calculate Δt for **Algorithm 3**. Here we fix $q(t) = 100/(1+t)$ for a reasonable stepsize of Δt. Now we apply the new methods of RNBAs to some nonlinear algebraic equations derived from PDE, ODE, Brown's problem, and a nonlinear problem with $\mathbf{B}$ singular.

5.1 Example 1

We consider a nonlinear heat conduction equation:

$$u_t = \alpha(x)u_{xx} + \alpha'(x)u_x + u^2 + h(x,t), \tag{52}$$
$$\alpha(x) = (x-3)^2, \;\; h(x,t) = -7(x-3)^2 e^{-t} - (x-3)^4 e^{-2t}, \tag{53}$$

with a closed-form solution being $u(x,t) = (x-3)^2 e^{-t}$.

By applying the new algorithms to solve the above equation in the domain of $0 \le x \le 1$ and $0 \le t \le 1$ we fix $n_1 = n_2 = 15$, which are numbers of nodal points in a standard finite difference approximation of Eq. (52). Because a_0 defined in Eq. (32) is a very important factor of our new algorithms we show it in Fig. 1(a) for **Algorithm 1**, while the ratio of residual errors is shown in Fig. 1(b), the stepsize is shown in Fig. 1(c), and the residual errors with respect to the number of steps up to 200 are shown in Fig. 1(d). From Fig. 1(b) we can see that the numerical performance of **Algorithm 1** is better than the Newton method, because we have a much smaller ratio of residual errors than that of the Newton method, which is calculated from Eq. (13) by inserting the stepsize as shown in Fig. 1(c). The results obtained from **Algorithms 2 and 3** are, respectively, shown in Figs. 2 and 3. In **Algorithm 2** we set $s_0 = 0.9$. No matter which algorithm is used the performances are better than the Newton method as shown in Figs. 1(b), 2(b) and 3(b). It is interesting to note that the three algorithms lead to three quite different a_0 as shown in Figs. 1(a), 2(a) and 3(a). The residual errors for the three new algorithms were compared in Fig. 3(d). Up to 200 steps they give almost the same residual error; however, their convergence behaviors are slightly different.

5.2 Example 2

In this example we apply the new algorithms to solve the following boundary value problem:

$$u'' = \frac{3}{2}u^2, \tag{54}$$
$$u(0) = 4, \;\; u(1) = 1. \tag{55}$$

The exact solution is

$$u(x) = \frac{4}{(1+x)^2}. \tag{56}$$

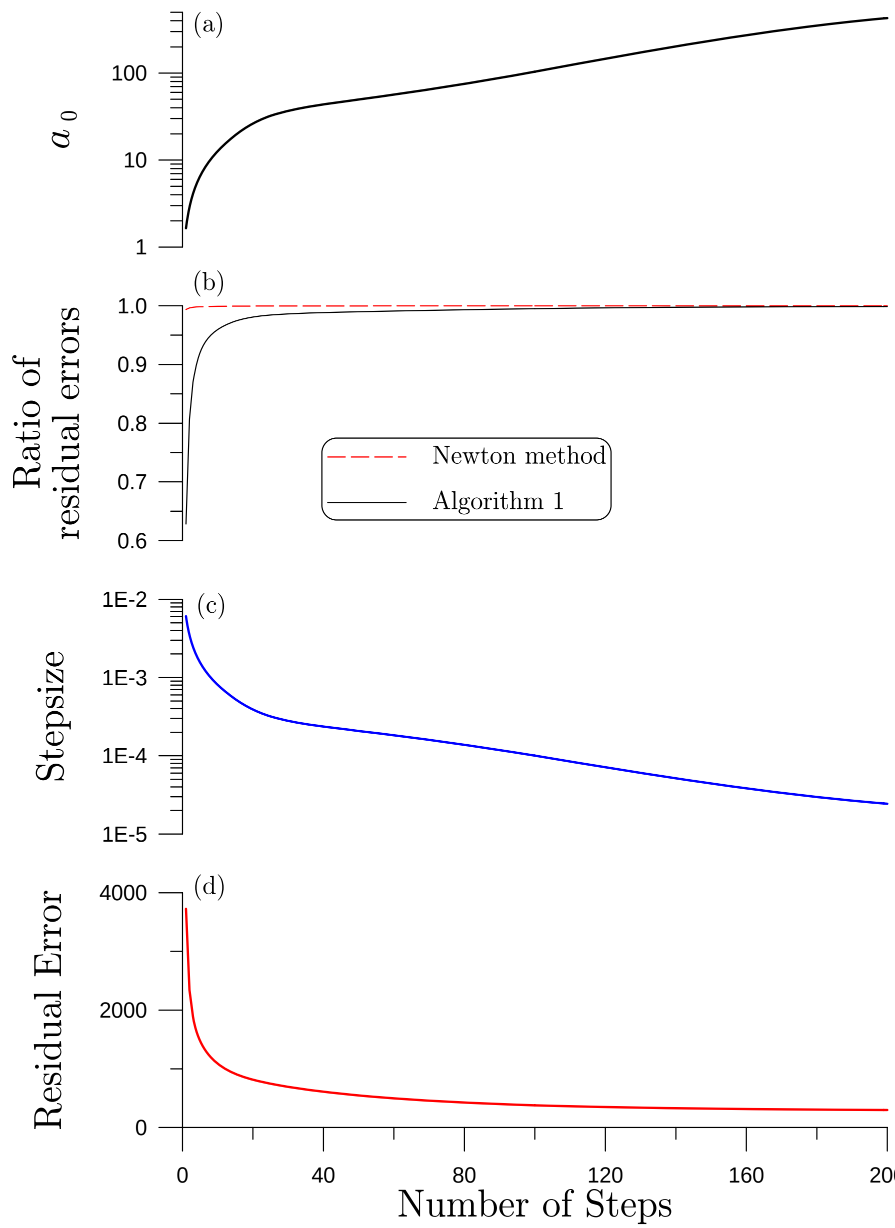

Figure 1: For example 1 by the first algorithm showing (a) a_0, (b) the comparison of the ratios of residual errors of **Algorithm 1**, and of the Newton method, (c) stepsize, and (d) residual error.

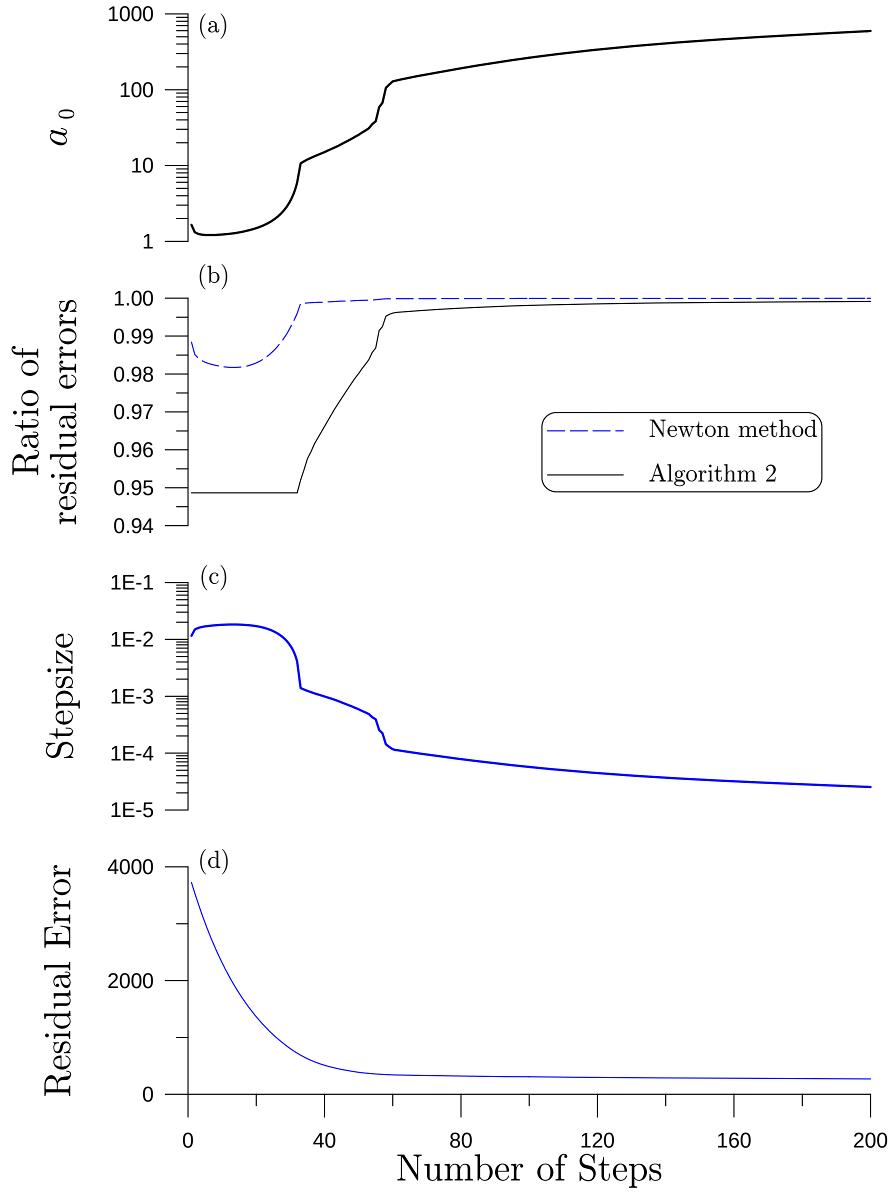

Figure 2: For example 1 by the second algorithm showing (a) a_0, (b) the comparison of the ratios of residual errors of **Algorithm 2**, and of the Newton method, (c) stepsize, and (d) residual error.

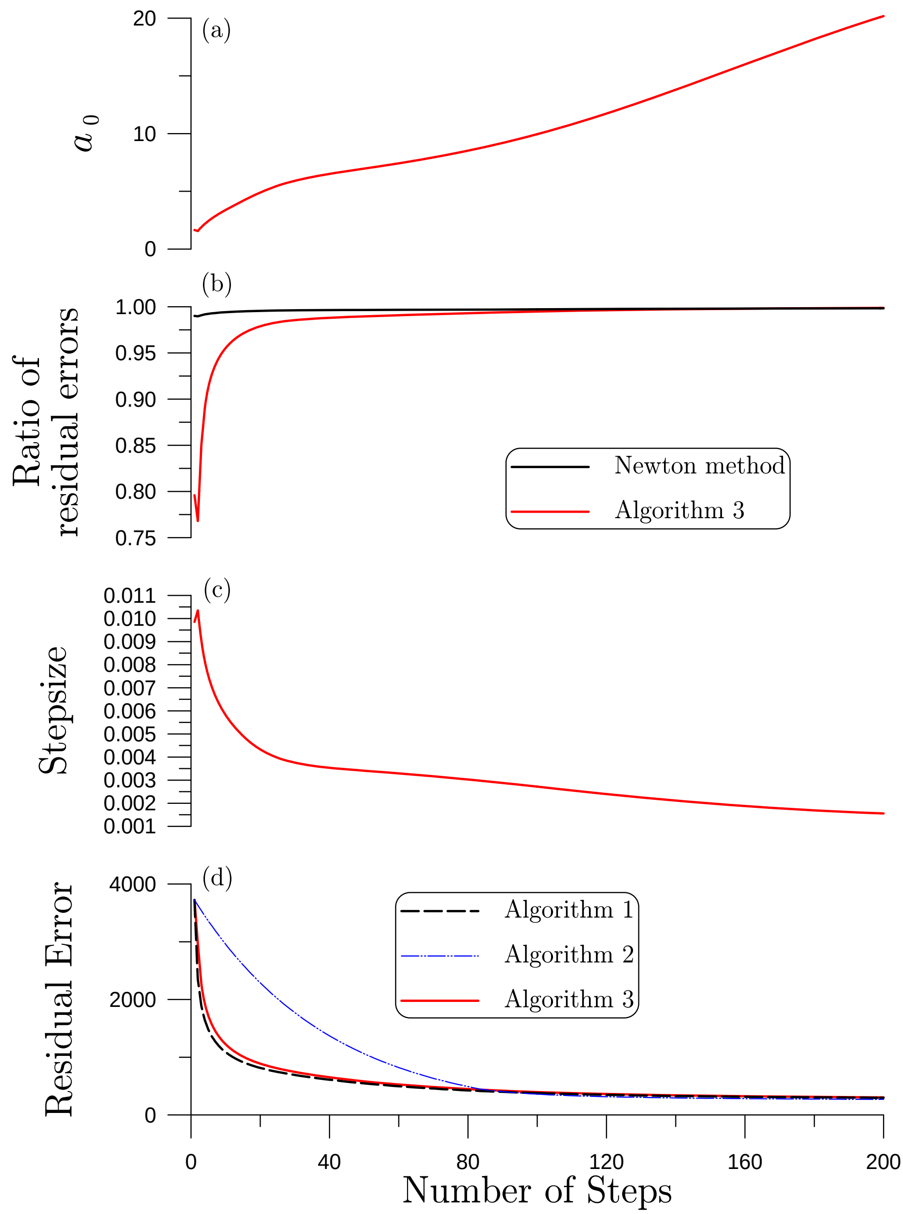

Figure 3: For example 1 by the third algorithm showing (a) a_0, (b) the comparison of the ratios of residual errors of **Algorithm 3**, and of the Newton method, (c) stepsize, and (d) residual error.

By introducing a finite difference discretization of u at the grid points we can obtain

$$F_i = \frac{1}{(\Delta x)^2}(u_{i+1} - 2u_i + u_{i-1}) - \frac{3}{2}u_i^2, \tag{57}$$

$$u_0 = 4, \;\; u_{n+1} = 1, \tag{58}$$

where $\Delta x = 1/(n+1)$ is the grid length.

We fix $n = 9$. In Fig. 4 we compare a_0, ratios of residual errors, and residual errors up to 2000 steps. From Fig. 4(c) the three new algorithms were convergent very fast with exponential decaying by different slopes. **Algorithm 1** and **Algorithm 2** with $s_0 = 0.9$ almost have the same convergence speed, and are better than **Algorithm 3**. As shown in Fig. 5 the three new algorithms can give accurate numerical solutions with maximum error smaller than 0.005. It is interesting that a_0 defined in Eq. (32) are all tending to some constants as shown in Fig. 4(a), which indicates that there exist "attracting sets" in the state space $\mathbf{x}$ for the above three algorithms. A further study will be the behavior of these "attracting sets".

Under the above same conditions we also apply the FTIM and scalar homotopy method to this problem, where ν and time stepsize used for FTIM are respectively 0.2 and 0.01, and the time stepsize used for scalar homotopy method is 0.0001. From Fig. 6 we can observe that **Algorithm 1** is faster than the FTIM, and much more faster than the scalar homotopy method.

5.3 Example 3

We consider an almost linear Brown's problem [Brown (1973)]:

$$F_i = x_i + \sum_{j=1}^{j=n} x_j - (n+1), \;\; i = 1, \ldots, n-1, \tag{59}$$

$$F_n = \prod_{j=1}^{j=n} x_j - 1, \tag{60}$$

with a closed-form solution $x_i = 1, \; i = 1, \ldots, n$.

For $n = 5$, in Fig. 7 we show a_0, the ratios of residual errors, and the residual errors up to 308 steps, which with an initial guess $x_i = 0.5, \; i = 1, \ldots, 5$ is convergent under the convergence criterion $\varepsilon = 10^{-5}$ by applying **Algorithm 1** to solve the above nonlinear algebraic equations. The accuracy is very good with a maximum error of $x_i, \; i = 1, \ldots, 5$ with 5.38×10^{-5}. From Fig. 7(c) it can be seen that **Algorithm 1** is exponentially convergent, with three different slopes.

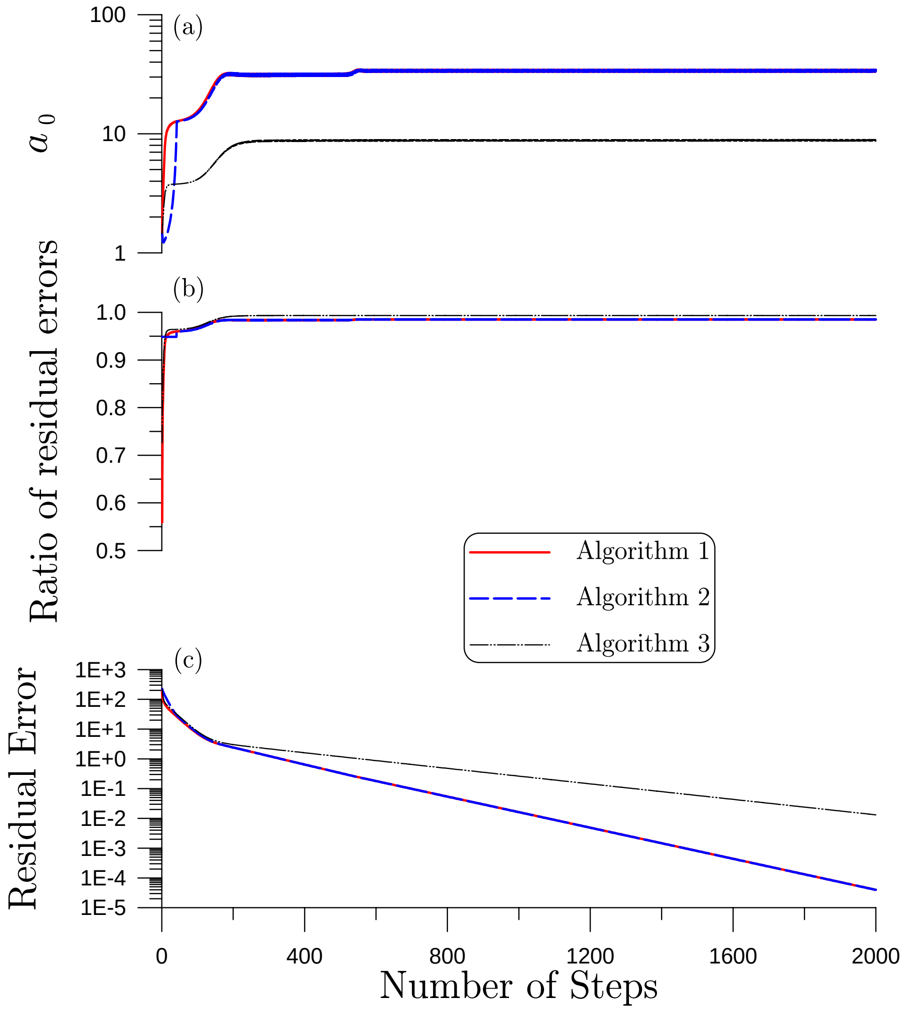

Figure 4: For example 2 solved by new algorithms showing (a) a_0, (b) the ratio of residual errors, and (c) the residual errors of three algorithms.

As demonstarted by Han and Han (2010), Brown (1973) solved this problem by the Newton method, and gave an incorrectly converged solution

$$(-0.579, -0.579, -0.579, -0.579, 8.90).$$

For $n = 10, 30$, Brown (1973) found that the Newton method diverged quite rapidly. Now, we apply our algorithms to this tough problem with $n = 30$. Under the convergence criterion $\varepsilon = 10^{-5}$ by applying **Algorithm 1** to solve the above nonlinear

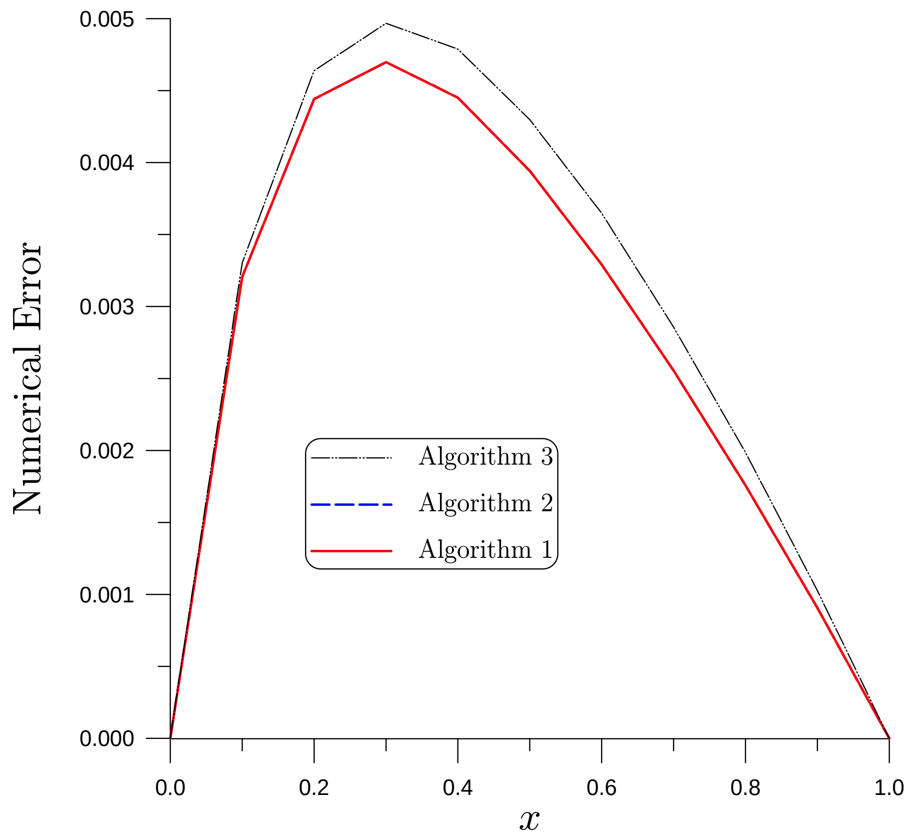

Figure 5: For example 2 solved by three new algorithms: a comparison of the numerical errors.

algebraic equations, the accuracy is very good with a maximum error of x_{30} with 2.09×10^{-4}, and other errors of $x_i,\ i = 1,\dots,29$ are the same 6.987×10^{-6}. From Fig. 8(a) it can be seen that **Algorithm 1** is exponential convergent with several different slopes.

By applying **Algorithm 2** with a given $s_0 = 0.5$, the accuracy is very good with a maximum error of x_{30} with 9.79×10^{-5}, and other errors of $x_i,\ i = 1,\dots,29$ are the same 3.21×10^{-6}. **Algorithm 2** is accurate than **Algorithm 1**, even from Fig. 8(b) it can be seen that **Algorithm 2** is exponentially convergent up to 50 steps. We also applied **Algorithm 3** to this case with an initial guess $x_i = 2,\ i = 1,\dots,30$. This algorithm converges much slower than **Algorithms 1 and 2** as shown in Fig. 8(c), and as shown in Fig. 9 the accuracy is also much worse than in **Algorithms 1 and 2**.

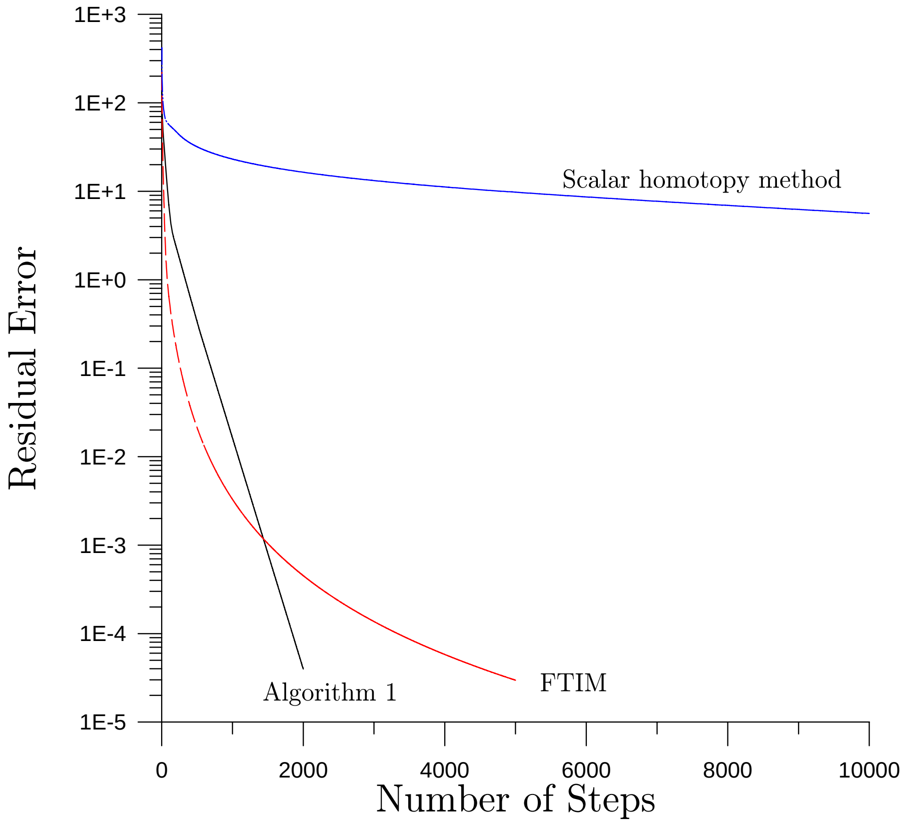

Figure 6: For example 2 solved by three different algorithms: a comparison of the residual errors.

When n is quite large, the last row of the matrix $\mathbf{B}$ at the initial point is almost zero. So the resulting nonlinear equations are very stiff and ill-conditioned. In Fig. 10 we show the residual error and numerical error for an extremely ill-posed case of the Brown's problem with $n = 100$. By applying **Algorithm 2** with a given $s_0 = 0.5$, the accuracy is very good with a maximum error of x_{100} with 3.02×10^{-4}, and other errors of x_i, $i = 1, \ldots, 99$ are the same as 3×10^{-6}. Under a convergence criterion $\varepsilon = 10^{-5}$, **Algorithm 2** converges within 223 iterations.

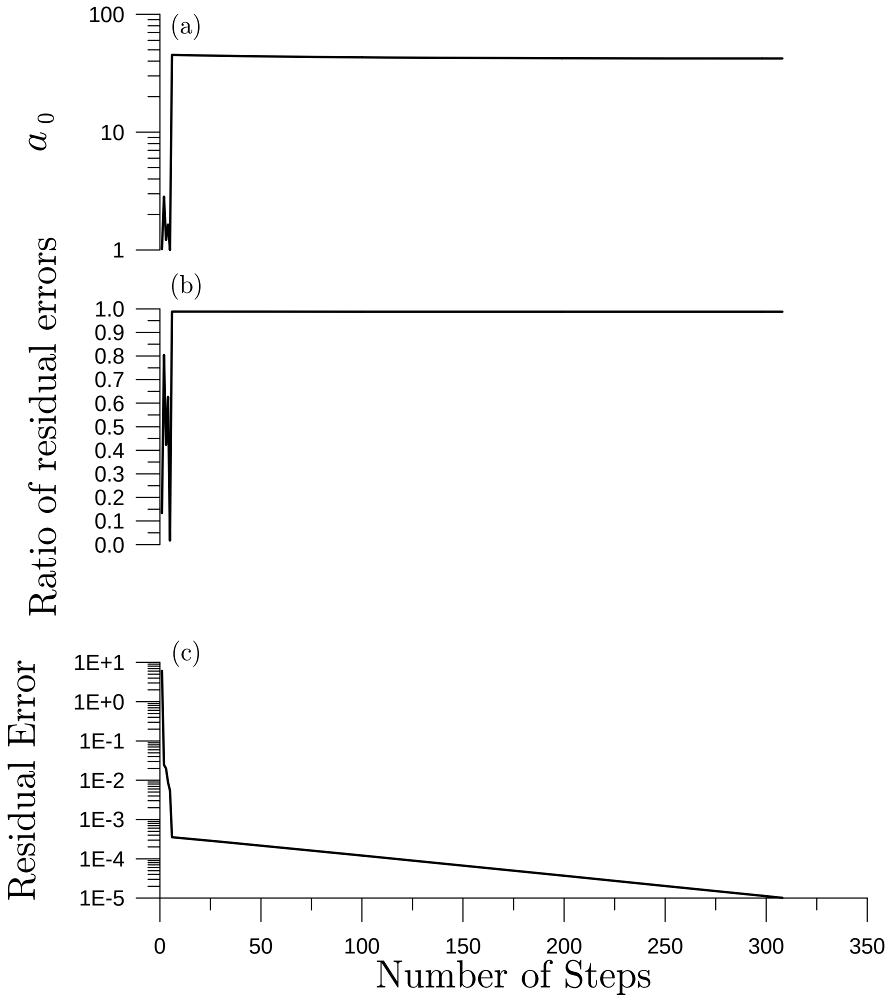

Figure 7: For example 3 with $n = 5$ solved by **Algorithm 1** showing (a) a_0, (b) the ratio of residual errors, and (c) residual error.

5.4 Example 4

We consider a singular case of **B** obtained from the following two nonlinear algebraic equations [Boggs (1971)]:

$$F_1 = x_1^2 - x_2 + 1, \tag{61}$$

$$F_2 = x_1 - \cos\left(\frac{\pi}{2}x_2\right), \tag{62}$$

$$\mathbf{B} = \begin{bmatrix} 2x_1 & -1 \\ 1 & \frac{\pi}{2}\sin\left(\frac{\pi}{2}x_2\right) \end{bmatrix}. \tag{63}$$

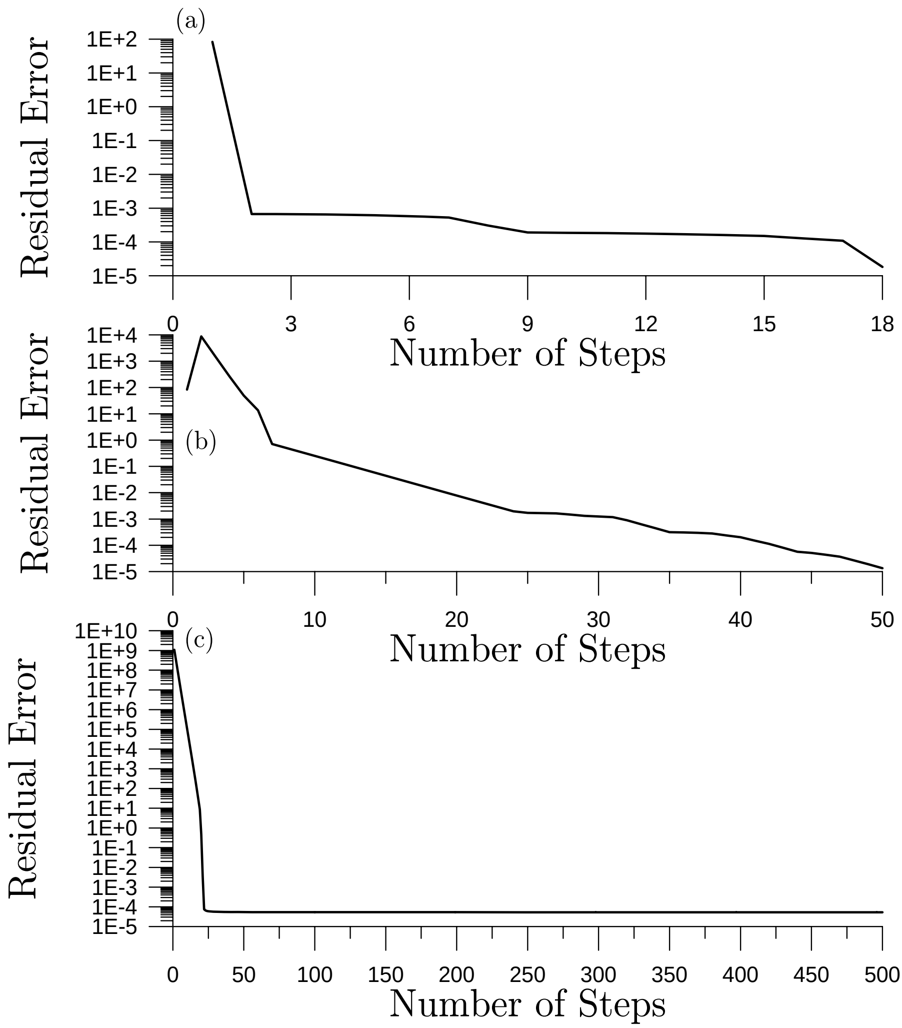

Figure 8: For example 3 with $n = 30$ showing the residual errors for (a) **Algorithm 1**, (b) **Algorithm 2**, and (3) **Algorithm 3**.

Obviously, on the curve of $\pi x_1 \sin(\pi x_2/2) + 1 = 0$, $\mathbf{B}$ is singular, i.e., $\det(\mathbf{B}) = 0$. They have a closed-form solution $(0, 1)$.

As demonstrated by Boggs (1971), the Newton method does not converge to $(0, 1)$, but rather it crosses the singular curve and converges to $(-\sqrt{2}/2, 3/2)$. We apply **Algorithm 1** to solve this problem within 126 iterations, and the results are shown in Fig. 11 for a_0, the ratio of residual errors, and the residual error by the solid

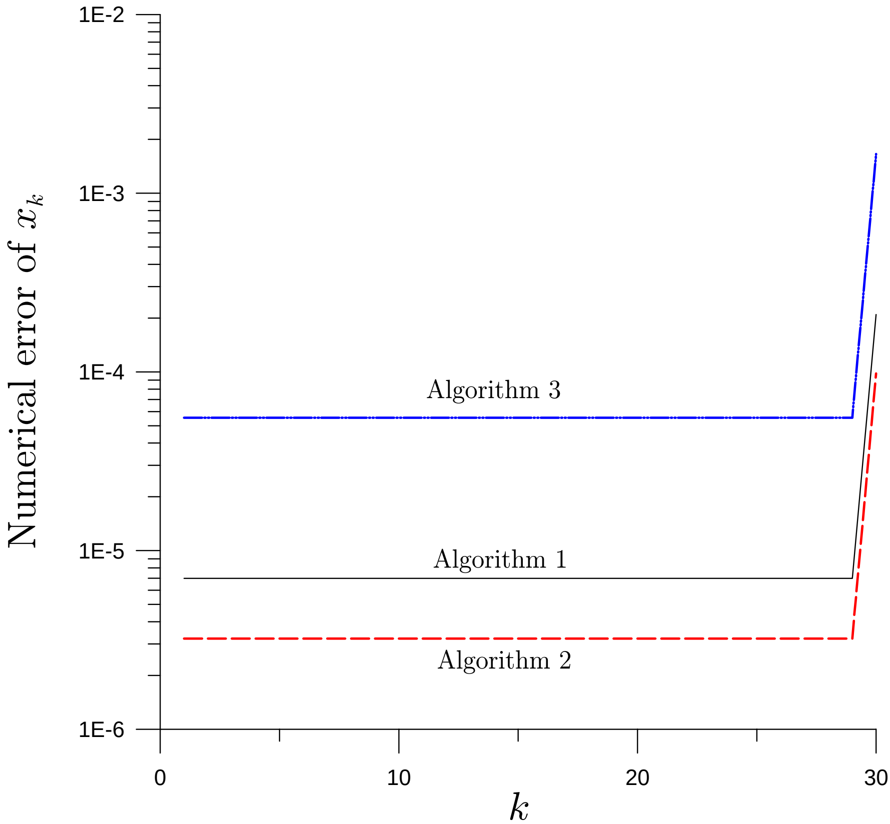

Figure 9: For example 3 with $n = 30$ solved by three new algorithms: a comparison of the numerical errors.

lines. In the termination of iterative process we found that the accuracy of x_1 is 1.77×10^{-8}, and of x_2 is 9.50×10^{-9}. When we apply **Algorithm 3** to solve this problem with 144 iterations, satisfying the convergence criterion $\varepsilon = 10^{-8}$, the results are shown in Fig. 11 for a_0, the ratio of residual errors, and the residual error by the dashed lines. The accuracy of x_1 is 1.3×10^{-8}, and of x_2 is 9.54×10^{-9}. **Algorithm 3** is slightly more accurate than **Algorithm 1**. From Fig. 11(b) it can be seen that even in the terminated step the ratios are still within 0.9, which show that the two **Algorithms 1 and 3** can further get even more accurate solutions, if we let them run more steps. The accuracy and efficiency obtained in the present algorithms are much better than those obtained by Boggs (1971), and Han and Han (2010).

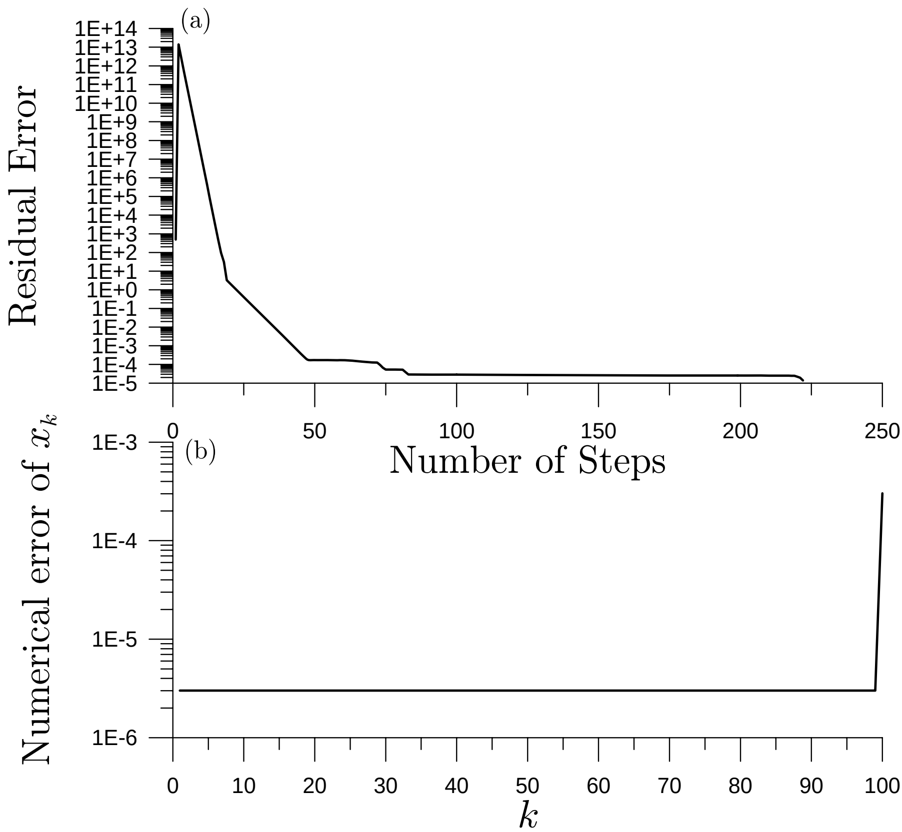

Figure 10: For example 3 with $n = 100$ solved by **Algorithm 2** showing (a) residual error, and (b) numerical error.

6 Conclusions

Three "Residual-Norm Based Algorithms" (RNBAs) were established in this paper. Although we were starting from a continuous invariant manifold based on the simple residual-norm and specifying a "gradient-flow" ODEs to govern the evolution of unknown variables, we were able to derive the final novel algorithms of iterative type without resorting on the fictitious time steps. In summary, the three novel algorithms could be written concisely as:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta \frac{\|\mathbf{B}_k^{\mathrm{T}}\mathbf{F}_k\|^2}{\|\mathbf{A}_k\mathbf{F}_k\|^2}\mathbf{B}_k^{\mathrm{T}}\mathbf{F}_k, \tag{64}$$

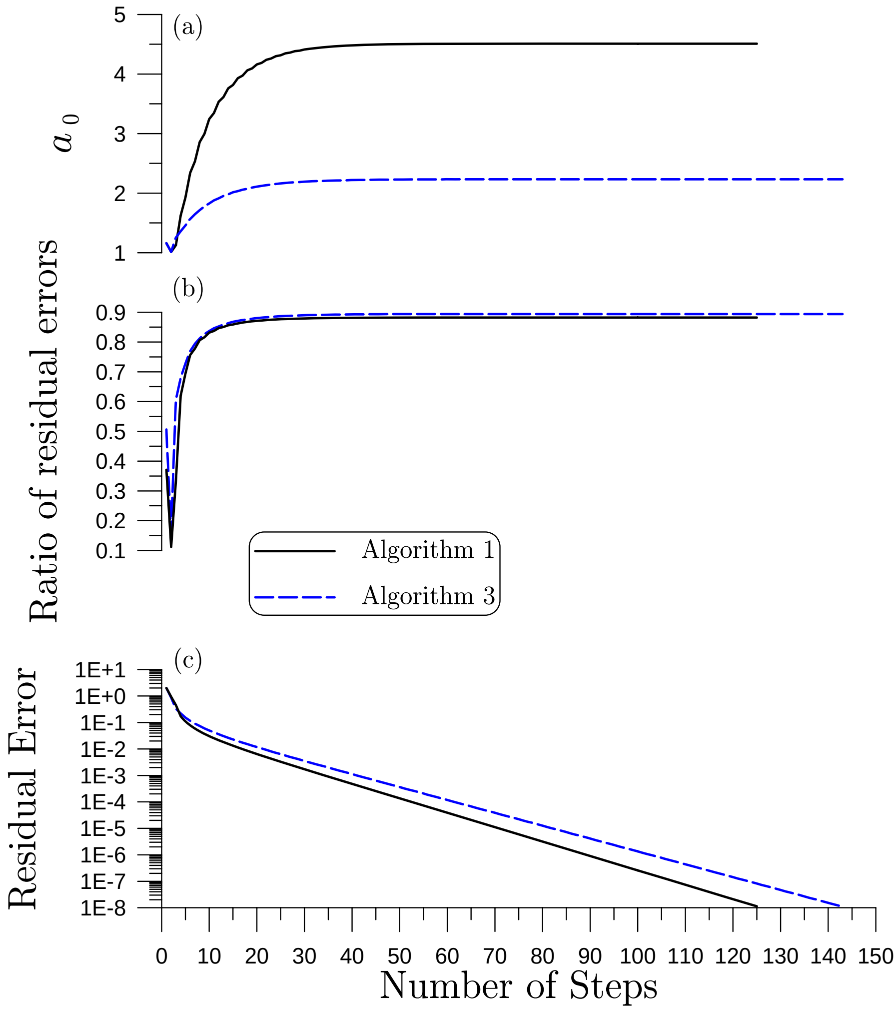

Figure 11: For example 4 solved by **Algorithms 1 and 3** showing (a) a_0, (b) ratios of residual errors, and (c) residual errors.

in which

$$\textbf{Algorithm 1:}\ \ \eta = 1, \tag{65}$$

$$\textbf{Algorithm 2:}\ \ \eta = \begin{cases} 1+\sqrt{1-(1-s_0)a_k} & \text{if } 1-(1-s_0)a_k \geq 0, \\ 1 & \text{if } 1-(1-s_0)a_k < 0, \end{cases} \tag{66}$$

$$\textbf{Algorithm 3:}\ \ \eta = 1+\sqrt{1-a_k^{-1}}, \tag{67}$$

where

$$a_k = \frac{\|\mathbf{F}_k\|^2 \|\mathbf{A}_k \mathbf{F}_k\|^2}{\|\mathbf{B}_k^{\mathrm{T}} \mathbf{F}_k\|^2}. \tag{68}$$

Algorithms 1 and 3 possess a major advantage that they do not need any parameter in the formulations; however, a suitable choice of $s_0 < 1$ in **Algorithm 2** can sometimes speed-up the convergence of iterations. We have proved that all the three novel algorithms are convergent automatically, and all are much better than that of the Newton method. Several numerical examples of nonlinear PDE, nonlinear ODE, nonlinear Brown problem with large dimension, and a singular nonlinear equations system, were tested to show the efficiency and accuracy of RNBAs. Indeed, in most situations we observed exponential convergences with different slopes in the iteration process. The RNBAs are easy to implement numerically, do not involve the inversions of the Jacobian matrices, and they can solve a large system of nonlinear algebraic equations very rapidly.

Acknowledgement: Taiwan's National Science Council project NSC-99-2221-E-002-074-MY3 granted to the first author is highly appreciated. The second author's research was supported by the World Class University (WCU) program through the National Research Foundation of Korea, funded by the Ministry of Education, Science & Technology (Grant no: R33-10049), during his visit to Pusan National University.

References

Allgower, E. L.; Georg, K. (1990): Numerical Continuation Methods: an Introduction. Springer, New York.

Atluri, S. N. (2002): Methods of Computer Modeling in Engineering and Sciences. Tech. Science Press, 1400 pages.

Atluri, S. N.; Liu, H. T.; Han, Z. D. (2006): Meshless local Petrov-Galerkin (MLPG) mixed collocation method for elasticity problems. *CMES: Computer Modeling in Engineering & Sciences*, vol. 14, pp. 141-152.

Atluri, S. N.; Shen, S. (2002): The meshless local Petrov-Galerkin (MLPG) method: a simple & less-costly alternative to the finite and boundary element methods. *CMES: Computer Modeling in Engineering & Sciences*, vol. 3, pp. 11-51.

Atluri, S. N.; Zhu, T. L. (1998a): A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics. *Comp. Mech.*, vol. 22, pp. 117-127.

Atluri, S. N.; Zhu, T. L. (1998b): A new meshless local Petrov-Galerkin (MLPG) approach to nonlinear problems in computer modeling and simulation. *Comp. Model. Simul. Eng.*, vol. 3, pp. 187-196.

Boggs, P. T. (1971): The solution of nonlinear systems of equations by A-stable integration technique. *SIAM J. Numer. Anal.*, vol. 8, pp. 767-785.

Brown, K. M. (1973): Computer oriented algorithms for solving systems of simultaneous nonlinear algebraic equations. In Numerical Solution of Systems of Nonlinear Algebraic Equations, Byrne, G. D. and Hall C. A. Eds., pp. 281-348, Academic Press, New York.

Davidenko, D. (1953): On a new method of numerically integrating a system of nonlinear equations. *Doklady Akad. Nauk SSSR*, vol. 88, pp. 601-604.

Deuflhard, P. (2004): Newton Methods for Nonlinear Problems: Affine Invariance and Adaptive Algorithms. Springer, New York.

Han, T.; Han Y. (2010): Solving large scale nonlinear equations by a new ODE numerical integration method. *Appl. Math.*, vol. 1, pp. 222-229.

Hirsch, M.; Smale, S. (1979): On algorithms for solving $f(x) = 0$. *Commun. Pure Appl. Math.*, vol. 32, pp. 281-312.

Ku, C. Y.; Yeih, W.; Liu, C.-S. (2010): Solving non-linear algebraic equations by a scalar Newton-homotopy continuation method. *Int. J. Non-Linear Sci. Num. Simul.*, vol. 11, pp. 435-450.

Liu, C.-S. (2001): Cone of nonlinear dynamical system and group preserving schemes. *Int. J. Non-Linear Mech.*, vol. 36, pp. 1047-1068.

Liu, C.-S. (2008): A time-marching algorithm for solving non-linear obstacle problems with the aid of an NCP-function. *CMC: Computers, Materials & Continua*, vol. 8, pp. 53-65.

Liu, C.-S. (2009a): A fictitious time integration method for a quasilinear elliptic boundary value problem, defined in an arbitrary plane domain. *CMC: Computers, Materials & Continua*, vol. 11, pp. 15-32.

Liu, C.-S. (2009b): A fictitious time integration method for the Burgers equation. *CMC: Computers, Materials & Continua*, vol. 9, pp. 229-252.

Liu, C.-S. (2009c): A fictitious time integration method for solving delay ordinary differential equations. *CMC: Computers, Materials & Continua*, vol. 10, pp. 97-116.

Liu, C.-S.; Atluri, S. N. (2008): A novel time integration method for solving a large system of non-linear algebraic equations. *CMES: Computer Modeling in Engineering & Sciences*, vol. 31, pp. 71-83.

Liu, C.-S.; Chang, C. W. (2009): Novel methods for solving severely ill-posed linear equations system. *J. Marine Sciences & Tech.*, vol. 17, pp. 216-227.

Liu, C.-S.; Yeih, W.; Kuo, C. L.; Atluri, S. N. (2009): A scalar homotopy method for solving an over/under-determined system of non-linear algebraic equations. *CMES: Computer Modeling in Engineering & Sciences*, vol. 53, pp. 47-71.

Ramm, A. G. (2007): Dynamical System Methods for Solving Operator Equations. Mathematics in Science and Engineering; Vol. 208 (series editor: Chu, C.K.), Elsevier, Amsterdam, Netherlands.

Zhu, T.; Zhang, J.; Atluri, S. N. (1998): A meshless local boundary integral equation (LBIE) method for solving nonlinear problems. *Comp. Mech.*, vol. 22, pp. 174-186.

Zhu, T.; Zhang, J.; Atluri, S. N. (1999): A meshless numerical method based on the local boundary integral equation (LBIE) to solve linear and non-linear boundary value problems. *Eng. Anal. Bound. Elem.*, vol. 23, pp. 375-389.