 CMES, vol.90, no.4, pp.255-282, 2013

A Scalar Homotopy Method with Optimal Hybrid Search Directions for Solving Nonlinear Algebraic Equations

Weichung Yeih[1,2], Cheng-Yu Ku[1,2,3], Chein-Shan Liu[4], I-Yao Chan[1,2]

Abstract: In this paper, a scalar homotopy method with optimal hybrid search directions for solving nonlinear algebraic equations is proposed. To conduct the proposed method, we first convert the vector residual function to a scalar function by taking the square norm of the vector function and then, introduce a fictitious time variable to form a scalar homotopy function. To improve the convergence and the accuracy of the proposed method, a vector with multiple search directions and an iterative algorithm are introduced into the evolution dynamics of the solutions. Further, for obtaining the optimal search direction, linear and nonlinear optimization algorithms are develploped. Taking the advantages of finding the optimal search direction, the proposed novel method is able to consider hybrid search directions for solving the nonlinear algebraic equations. The formulation presented in this paper demonstrates a variety of flexibility with the use of the algorithm for finding optimal hybrid search directions. In addition, our proposed method does not necessarily need to calculate the inverse of the Jacobian matrix and has great numerical stability for solving nonlinear well-posed algebraic equations as well as the ill-posed problems which may have an ill-conditioned or singular Jacobian matrix. Results reveal that the proposed method can improve the convergence and increase the numerical stability for solving nonlinear algebraic equations

Keywords: the scalar homotopy method, scalar function, optimization, Jacobian, ill-posedness, Newton's method.

1 Introduction

For solving engineering problems, numerical methods used in computational mechanics lead to seeking the solution of a system of linear algebraic equations for a

[1] Department of Harbor and River Engineering, National Taiwan Ocean University, Keelung, Taiwan.
[2] Computation and Simulation Center, National Taiwan Ocean University, Keelung, Taiwan.
[3] Corresponding Author, E-mail: chkst26@mail.ntou.edu.tw
[4] Department of Civil Engineering, National Taiwan University, Taipei, Taiwan

linear problem, or that of a Non-Linear Algebraic Equations (NAEs) system for a non-linear problem. Over the past years, many contributions have been made towards the numerical solutions of linear or non-linear problems [Atluri (2002)] The iterative-based method, such as Newton's method, the method of steepest descent are most common methods. Newton's method converges quadratically; however, it is sensitive to the initial guess of solution and is very expensive in the computations of the inverse of the Jacobian matrix. Therefore, modifications of Newton's method, such as the continuous Newton method [Hirsch and Smale (1979)], the Jacobian-Free NewtonKrylov method [Knoll and Keyes (2004); Lemieux et al. (2010)], the fictitious time integration method [Liu and Atluri (2008); Ku, Yeih, Liu, and Chi (2009)], the scalar Newton-homotopy continuation method [Ku, Yeih, and Liu (2010)], and the dynamical Newton-like method [Ku, Yeih, and Liu (2011)], have been extensively developed for this purpose. All of these methods are based on a fixed search direction for solving problems. Accordingly, limitations such as slow convergence, numerical instability, or low accuracy may arise while encountering nonlinear algebraic equations especially for the ill-posed problems which may have an ill-conditioned or singular Jacobian matrices.

The nonlinear algebraic equations with ill-posedness such as ill-conditioned system or singular Jacobian matrix are frequently encountered in science and engineering and many problems in engineering and science require the solution of ill-posed problems and a good and stable numerical algorithm for solving the ill-posed nonlinear problems is very important. Over the past years, many contributions have been made towards the numerical solutions of ill-posed problems. Most of these methods are based on the so-called regularization methods. The main objective of regularization is to incorporate more information about the desired solution in order to stabilize the problem and find a useful and stable solution. Among these methods, the most common and well-known form of regularization is that of Tikhonov [Groetsch, (1984)] Tikhonov regularization is a method in which the regularized solution adopts two directions using the combination of the residual norm and a size constraint of the regularized solution From the regularization method, we can find that it is useful to incorporate more search directions for finding the solution of ill-posed problems. However, for conventional Tikhonov's regularization method to determine the regularization method requires a lot of computation efforts such as the L-curve method [Hanson, 1992] or the discrepancy principles [Morozov (1984, 1966)]. Liu and Kuo (2011) proposed a dynamic Tikhonov regularization method to solve the nonlinear ill-posed problems in which the searching direction is assumed to combine the direction of steepest descent vector and an unknown coefficient multiplying the direction of unknown vector. Liu and Atluri (2011a) adopted an iterative method using an optimal descent vector, for solving an ill-conditioned

system. In their study, two search directions including the combination of the steepest descent direction and the residual direction are used. From these two pioneer works, they show that it is possible to construct an optimal direction by determining the weighting factor between two directions. It is then interesting to know that how many directions are allowed when ill-posed problems are encountered as well as how to determine the unknown weights for each direction

In this study we propose a novel method based on a scalar homotopy function with optimal hybrid search directions for solving nonlinear algebraic equations. To improve the convergence and the accuracy of the proposed method, a vector composed by multiple search directions with unknown weights is used and an iterative algorithm based on the evolution of the residual vector is introduced into the evolution dynamics of the solutions. Further, for obtaining the optimal search directions, linear and nonlinear optimization algorithms are developed. Taking the advantages of finding the optimal search direction, the proposed novel method is able to consider hybrid search directions for solving the nonlinear algebraic equations. The proposed method is then adopted for the solution of ill-posed problems. First of all, the formulation of the method is described as follows.

2 Mathematical backgrounds

2.1 The scalar homotopy method

The early practical application of homotopy-like methods to numerical solution of nonlinear equations is commonly attributed to Davidenko (1953). Recently, more references can be found in the application of homotopy methods. Liao (2004) employed the basic ideas of homotopy to propose a general method for nonlinear problems, namely the homotopy analysis method, and this method has been successfully applied to solve many types of nonlinear problems. He (2005) studied the homotopy method through a series of different non-linear ordinary differential equations. In this study, we consider the following nonlinear algebraic equations as:

$$F_i(x_1,\ldots,x_n) = 0, i = 1,\ldots,n \tag{1}$$

Using $\mathbf{x} := (x_1,\ldots,x_n)^T$ and $\mathbf{F} := (F_1,\ldots,F_n)^T$, Eq. (1) can be written as:

$$\mathbf{F}(\mathbf{x}) = \mathbf{0} \tag{2}$$

the homotopy method represents a way to enhance the convergence from a local convergence to a global convergence All the homotopy methods are based on the construction of a vector function, $\mathbf{H}(\mathbf{x},\tau)$ which is called the homotopy function. The homotopy function serves the objective of continuously transforming a function $\mathbf{G}(\mathbf{x})$ into $\mathbf{F}(\mathbf{x})$ by introducing a homotopy parameter τ. The homotopy parameter τ can be treated as a time-like fictitious variable, and the homotopy function

can be any continuous function such that: $\mathbf{H}(\mathbf{x},0)=\mathbf{G}(\mathbf{x})$ and $\mathbf{H}(\mathbf{x},1)=\mathbf{F}(\mathbf{x})$. Hence we construct $\mathbf{H}(\mathbf{x},0)$ in such a way that its zeros are easily found while we also require that, once the parameter τ is equal to 1, then $\mathbf{H}(\mathbf{x},\tau)$ coincides with the original function $\mathbf{F}(\mathbf{x})$

Among the various homotopy functions that are generally used, the fixed point homotopy function, i.e. $\mathbf{G}(\mathbf{x})=\mathbf{x}-\mathbf{x_0}$ and the Newton homotopy function, i.e. $\mathbf{G}(\mathbf{x})=\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x_0})$ are simple and powerful ones that can be successfully applied to several different problems. The Newton homotopy function is

$$\mathbf{H}(\mathbf{x},\tau)=\tau\mathbf{F}(\mathbf{x})+(1-\tau)[\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{x_0})]=\mathbf{0} \tag{3}$$

where $\mathbf{x_0}$ is the given initial values and $\tau\in[0,1]$. In many vector-based homotopy methods, each step involves computing the inverse of the Jacobian matrix which often raises the difficulty of divergence in certain circumstances; in such cases each step is as costly as a Newton step. Liu, Yeih, Kuo and Atluri (2009) and Ku, Yeih, and Liu (2010) used the fixed point homotopy function and the Newton homotopy function respectively to make an anology for the scalar homotopy method to the theory of plasticity. Using the same concept, we first convert the vector equation of $\mathbf{F}=\mathbf{0}$ into a scalar equation by noticing that

$$\mathbf{F}=\mathbf{0}\Leftrightarrow\|\mathbf{F}\|^2=0 \tag{4}$$

where $\|\mathbf{F}\|^2=F_1^2+F_2^2+\ldots+F_n^2$. Obviously, the left-hand side implies the right-hand side. Conversely, by $\|\mathbf{F}\|^2=F_1^2+F_2^2+\ldots+F_n^2=0$ we have $F_1=F_2=\ldots=F_n=0$, and thus $\mathbf{F}=\mathbf{0}$

Using Eq. (4), we can transform the vector equation into a fictitious time dependent scalar function $h(\mathbf{x},\tau)$ as follows:

$$h(\mathbf{x},\tau)=\frac{1}{2}\|\mathbf{F}(\mathbf{x})\|^2+\frac{1}{2}(\tau-1)\|\mathbf{F}(\mathbf{x_0})\|^2=0 \tag{5}$$

Equation (5) holds for all $\tau\in[0,1]$. To motivate this study, we first consider a fictitious time function $Q(t)$ where t is the fictitious time and $Q(t)$ has to satisfy that $Q(t)>0$, $Q(0)=1$, and $Q(t)$ is a monotonically increasing function of t, and $Q(\infty)=\infty$. Then we introduce the proposed fictitious time function $Q(t)$ into Eq. (5) and have

$$h(\mathbf{x},t)=\frac{1}{2}\|\mathbf{F}(\mathbf{x})\|^2-\frac{1}{2}\frac{1}{Q(t)}\|\mathbf{F}(\mathbf{x_0})\|^2=0 \tag{6}$$

Using the fictitious time function, $Q(t)$, when the fictitious time $t=0$ and $t=\infty$, we can obtain

$$h(\mathbf{x},t=0)=\frac{1}{2}\|\mathbf{F}(\mathbf{x})\|^2-\frac{1}{2}\|\mathbf{F}(\mathbf{x_0})\|^2=0\Leftrightarrow\mathbf{F}(\mathbf{x})=\mathbf{F}(\mathbf{x_0}) \tag{7}$$

$$h(\mathbf{x},t=\infty)=\frac{1}{2}\left\|\mathbf{F}(\mathbf{x})\right\|^2=0\Leftrightarrow\mathbf{F}(\mathbf{x})=\mathbf{0}. \tag{8}$$

It is clear that the tracking of a solution path for the proposed scalar Newton homotopy function, as the homotopy parameter τ is gradually varied from 0 to 1, is equivalent to the fictitious time varying from $t=0$ to $t=\infty$.

If we assume that $h(\mathbf{x},t)=0$ is satisfied for any time greater than zero, multiplying $Q(t)$ at both sides of Eq. (6) one can construct a space-time manifold [Ku, Yeih and Liu (2010)] written as

$$h(\mathbf{x},t)=\frac{1}{2}Q(t)\left\|\mathbf{F}(\mathbf{x})\right\|^2-\frac{1}{2}\left\|\mathbf{F}(\mathbf{x_0})\right\|^2=0. \tag{9}$$

Then, we can use the 'consistency equation' to force the trajectory of the solution $\mathbf{x}$ always remain on this manifold which means the total derivate of $h(\mathbf{x},t)$ with respect to the time t should be zero as:

$$\frac{Dh}{Dt}=\frac{\partial h}{\partial t}+\nabla h\cdot\frac{d\mathbf{x}}{dt}=0 \tag{10}$$

where ∇ denotes the gradient operator. The derivatives of the scalar function, $h(\mathbf{x},t)$, with respect to $\mathbf{x}$ and t can be written as

$$\frac{\partial h}{\partial t}=\frac{1}{2}\dot{Q}(t)\left\|\mathbf{F}(\mathbf{x})\right\|^2\textbf{where}\dot{Q}(t)=\frac{dQ(t)}{dt}. \tag{11}$$

$$\nabla h=\frac{\partial h}{\partial \mathbf{x}}=Q(t)\mathbf{B^T}\mathbf{F}(\mathbf{x}) \tag{12}$$

where the superscript 'T' denotes the transpose of a matrix and $\mathbf{B}$ is the Jacobian matrix. Since Eq. (10) is a scalar equation, it is obviously impossible to determine the evolution dynamics of $\mathbf{x}$ (i.e., $\frac{d\mathbf{x}}{dt}$) from Eq. (10) uniquely. Now let us assume that the evolution dynamics of x is in the direction of a vector, say u. It means that we have

$$\dot{\mathbf{x}}=\frac{d\mathbf{x}}{dt}=\lambda\mathbf{u}. \tag{13}$$

where λ is a proportional constant.

Inserting Eqs. (11), (12) and (13) into Eq. (10), we then can derive

$$\lambda=-\frac{\dot{Q}(t)}{2Q(t)}\frac{\left\|\mathbf{F}(\mathbf{x})\right\|^2}{\mathbf{F^T}(\mathbf{x})\mathbf{Bu}}. \tag{14}$$

Inserting Eq. (14) into Eq. (13), we can obtain the evolution dynamics of $\mathbf{x}$ as:

$$\dot{\mathbf{x}}=-\frac{\dot{Q}(t)}{2Q(t)}\frac{\left\|\mathbf{F}(\mathbf{x})\right\|^2}{\mathbf{F^T}(\mathbf{x})\mathbf{Bu}}\mathbf{u}. \tag{15}$$

By defining $\mathbf{v} = \mathbf{Bu}$, one can rewrite Eq. (15) as:

$$\dot{\mathbf{x}} = -\frac{\dot{Q}(t)}{2Q(t)} \frac{\|\mathbf{F}(\mathbf{x})\|^2}{\mathbf{F}^{\mathbf{T}}(\mathbf{x})\mathbf{v}}\mathbf{u}. \tag{16}$$

2.2 Iteration algorithm based on the evolution of the residual vector

Let us take a look of the evolution of the residual vector $\mathbf{F}$ which can be written as:

$$\dot{\mathbf{F}}(\mathbf{x}(t)) = \mathbf{B}\dot{\mathbf{x}}. \tag{17}$$

Substituting Eq. (16) into Eq. (17), we then have:

$$\dot{\mathbf{F}}(\mathbf{x}(t)) = \frac{-\dot{Q(t)}}{2Q(t)} \frac{\|\mathbf{F}(\mathbf{x})\|^2}{\mathbf{F}^{\mathbf{T}}(\mathbf{x})\mathbf{v}}\mathbf{v}. \tag{18}$$

Using the forward Euler scheme, we can approximately express Eq. (18) as

$$\mathbf{F}(\mathbf{x}(t+\Delta t)) = \mathbf{F}(\mathbf{x}(t)) - \Delta t \frac{\dot{Q}(t)}{2Q(t)} \frac{\|\mathbf{F}(\mathbf{x})\|^2}{\mathbf{F}^T(\mathbf{x})\mathbf{v}}\mathbf{v} \tag{19}$$

where Δt is the time increment. For simplicity, we let

$$\beta := \Delta t \frac{\dot{Q}(t)}{2Q(t)}. \tag{20}$$

Since we require that the evolution path of $\mathbf{x}$ should always remain on the space-time manifold, we then can obtain the following expressions from Eq. (9):

$$\|\mathbf{F}(\mathbf{x}(t))\|^2 = \frac{\|\mathbf{F}(\mathbf{x}_0)\|^2}{Q(t)} \text{ and } \|\mathbf{F}(\mathbf{x}(t+\Delta t))\|^2 = \frac{\|\mathbf{F}(\mathbf{x}_0)\|^2}{Q(t+\Delta t)}. \tag{21}$$

Taking the square norm of Eq. (19) for both sides and using Eqs. (20) and (21), we can obtain

$$\frac{\|F(x_0)\|^2}{Q(t+\Delta t)} = \frac{\|F(x_0)\|^2}{Q(t)} - 2\beta\frac{\|F(x_0)\|^2}{Q(t)} + \beta^2\frac{\|F(x_0)\|^2}{Q(t)}\frac{\|F(x)\|^2}{(F^T(x)v)^2}\|v\|^2. \tag{22}$$

Rearranging the above equation, we can obtain an algebraic equation for β as

$$a_0\beta^2 - 2\beta + 1 - \frac{Q(t)}{Q(t+\Delta t)} = 0, \tag{23}$$

where

$$a_0 = \frac{\|\mathbf{F}(\mathbf{x})\|^2\|\mathbf{v}\|^2}{(\mathbf{F}^T(\mathbf{x})\mathbf{v})^2} = \left\{\frac{\|\mathbf{F}(\mathbf{x})\|\|\mathbf{v}\|}{(\mathbf{F}^T(\mathbf{x})\mathbf{v})}\right\}^2 = \left(\frac{1}{\cos\theta}\right)^2 \tag{24}$$

in which θ denotes the angle between the residual vector $\mathbf{F}$ and the vector $\mathbf{v}$. From the Cauchy-Schwarz inequality, it can be easily verified that $a_0 \geq 1$ Now let us define:

$$s := \frac{Q(t)}{Q(t+\Delta t)} = \frac{\|\mathbf{F}(\mathbf{x}(t+\Delta t))\|^2}{\|\mathbf{F}(\mathbf{x}(t))\|^2} \tag{25}$$

It can be found that this ratio s is the ratio between the square norm of the residual vector in the next state and the square norm of the residual vector in the current state. It is for sure that we hope $s \leq 1$, such that for each state the norm of the residual vector decreases. From Eq. (25), Eq. (23) now can be written as

$$a_0\beta^2 - 2\beta + 1 - s = 0. \tag{26}$$

We can obtain the solution for β as:

$$\beta = \frac{1-\sqrt{1-(1-s)\,a_0}}{a_0}.\mathbf{if}, 1-(1-s)a_0 \geq 0 \tag{27}$$

For simplicity, we let

$$1-(1-s)a_0 = r^2. \tag{28}$$

Eq. (28) can be rewritten as

$$s = 1 - \frac{1-r^2}{a_0}. \tag{29}$$

Substituting Eq. (24) into Eq. (29), one can obtain

$$s = 1 - \frac{\left(1-r^2\right)\left(\mathbf{F}^T(\mathbf{x})\,\mathbf{v}\right)^2}{\|\mathbf{F}(\mathbf{x})\|^2\,\|\mathbf{v}\|^2}. \tag{30}$$

Remember that s means the ratio between the square norm of the residual vector in the next state and the square norm of the residual vector in the current state, and it is better to minimize this parameter in our numerical scheme.

Now by using the forward Euler scheme on Eq. (16), one can rewrite the evolution of solution as:

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) - (1-r)\frac{(\mathbf{F}^T(\mathbf{x})\mathbf{v})}{\|\mathbf{v}\|^2}\mathbf{u}. \tag{31}$$

Eq. (31) may be viewed as an iterative formula, rewritten as:[Liu and Atluri, 2011b]

$$\mathbf{x}_{k+1} = \mathbf{x}_k - (1-r)\frac{\mathbf{F}^T(\mathbf{x}_k)\,\mathbf{v}}{\|\mathbf{v}\|^2}\mathbf{u}. \tag{32}$$

2.3 Finding search directions using nonlinear algorithm

A previous study [Liu and Atluri, 2011c] developed an algorithm to find the optimal vector for $\mathbf{v} = \mathbf{v}_1 + \alpha \mathbf{v}_2$. The optimal value of α can be obtained by substituting the above expression into Eq. (30), then take the derivative of s (or equivalently the derivative of a_0 with respect to α) to be zero and the following expression can be obtained: [Liu and Atluri, 2011c]

$$\alpha = \frac{(\mathbf{v}_1 \cdot \mathbf{F})(\mathbf{v}_1 \cdot \mathbf{v}_2) - (\mathbf{v}_2 \cdot \mathbf{F}) \|\mathbf{v}_1\|^2}{(\mathbf{v}_2 \cdot \mathbf{F})(\mathbf{v}_1 \cdot \mathbf{v}_2) - (\mathbf{v}_1 \cdot \mathbf{F}) \|\mathbf{v}_2\|^2}. \tag{33}$$

Based on a similar aspect, we proposed a new optimization method which can find the optimal vector for the solution using optimal hybrid search directions. First, we assume that

$$\mathbf{u} = \sum_{k=1}^{m} \alpha_k \mathbf{u}_k \tag{34}$$

which means that the searching direction is a linear combination of m known vectors with undetermined coefficients. It follows directly that

$$\mathbf{v} = \mathbf{B}\mathbf{u} = \sum_{k=1}^{m} \alpha_k (\mathbf{B}\mathbf{u}_k) = \sum_{k=1}^{m} \alpha_k \mathbf{v}_k. \tag{35}$$

By a similar procedure, we can obtain a system of m nonlinear algebraic equations to determine the unknown coefficients:

$$(\mathbf{F} \cdot \mathbf{v})(\mathbf{v} \cdot \mathbf{v}_k) - (\mathbf{v}, \mathbf{v})(\mathbf{F}, \mathbf{v}_k) = 0, for \mathbf{k} = 1, 2, 3 \ldots, m. \tag{36}$$

Eq. (36) is a necessary condition for finding the extreme value of s or equivalentlya_0. To seek for the minimum value of s, one can expect that the inner product between the vectors $\mathbf{v}$ and $\mathbf{F}$ should not be zero, i.e. the angle between vectors $\mathbf{v}$ and $\mathbf{F}$ should not be πvyg We now prove that an additional constraint for the undetermined coefficients is required to ensure the solution of Eq. (36) to be unique. Now assume another vector as

$$\mathbf{u}* = p\mathbf{u} \tag{37}$$

where p is a proportional constant and u can be written as Eq. (34) and satisfy Eq. (36). Now we substitute u^* into Eq. (37) to replace u in Eq. (35) then use the expression of $\mathbf{v}$ and substitute it into Eq. (36), it can be easily found that Eq. (36) can be satisfied automatically. It then means that when a direction is the optimal one, any vector lies in this direction can be the optimal one. This means that infinitely

many solutions exist for the set of equations written in Eq. (36). In addition, we can check the evolution of x by using $p\mathbf{u}$ to replace u in Eq. (31) and using pv to replace $\mathbf{v}$ in Eq. (31), it will be found that p can be cancelled and make no difference in the evolution of x at all. From the above argument, we can conclude that some constraint is required to make the possible solutions be finite. For example, one can use the following equation as additional constraint:

$$\sum_{k=1}^{m} \alpha_k^2 = 1. \tag{38}$$

Summing up the previous arguments, we say the optimal direction for many vectors can be determined by solving Eqs.(36) and (38) together. This method we named it as the nonlinear algorithm and summed up as follows:

Nonlinear algorithm:

Step 1: Solving $(\mathbf{F}\cdot\mathbf{v})(\mathbf{v}\cdot\mathbf{v}_k)-(\mathbf{v},\mathbf{v})(\mathbf{F},\mathbf{v}_k)=0$ *for k=1,... m and* $\sum_{k=1}^{m}\alpha_k^2=1$ *together. (Inner iteration)*

Step 2: After obtaining α*, Eq.(32) is used to solve the solution x (Outer iteration)*

In order to determine the optimal direction, a set of nonlinear algebraic equations should be solved first which for sure makes the numerical scheme more complicated. In reality, only for the first several steps we may pay a lot of effort on seeking the solution of the unknown coefficient vector α. If we select the 'solution' of α in the previous step as the initial guess of the current step, soon the solution of α can be found for each step quickly enough.

2.4 Finding search directions using linear algorithm

It is well known that for the n-dimensional problem, the dimension of solution vector x is n. That means at most $k=n$, that is at most we can provide n-linearly independent vectors such that they can represent any vector in the space. For simplicity, if one make

$$\mathbf{u}_k := \begin{bmatrix} \delta_{1k} \\ \delta_{2k} \\ \vdots \\ \delta_{nk} \end{bmatrix} for k = 1,\ldots,n \tag{39}$$

with $\delta_{pq} = \begin{cases} 1, p=q \\ 0, p\neq q \end{cases}$ the Kronecker delta symbol.

It then can be seen that in order to minimize a_0, it can be achieved if $\mathbf{v} = \mathbf{F}$. It then follows

$$\begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_n \end{bmatrix}. \tag{40}$$

It is known that the solution of the unknown coefficient vector α can be uniquely determined if $\det[\ \mathbf{v}_1 \quad \mathbf{v}_2 \quad \ldots \quad \mathbf{v}_n\ ] \neq 0$. Accordingly, Eq. (40) can be further reduced to

$$B\alpha = F. \tag{41}$$

Solving Eq. (41) is much simpler than solving Eqs. (36) and (38) together using nonlinear algorithm. But notice that it can be only achieved while n independent directions of u_k are given. We will provide numerical examples to support our viewpoints later.

When we only have m directions where $m < n$, we still can have the following equations which is similar to Eq. (40):

$$\begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_n \end{bmatrix}. \tag{42}$$

To solve Eq. (42), one can solve it in the sense of least square or adopt the so-called pseudo-inverse concept. Actually solving Eq. (42) by using the least square method is equivalent to the following minimization problem:

$$\min \left(\begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix} - \mathbf{F} \right)^2. \tag{43}$$

It then can be seen that in Eq. (43), we try to select the unknown coefficients such that the vector $\mathbf{v}$ is mostly close to the vector $\mathbf{F}$ and this also implies that the angle between $\mathbf{v}$ and $\mathbf{F}$ is then mostly close to zero. Solving the problem in Eq. (43) is much simpler than solving Eqs. (36) and (38) using nonlinear algorithm together. Furthermore, solving Eq. (43) only the minimum value of s is obtained

and solving Eqs.(36) and (38) may result in obtaining the maximum value of s but not the minimum value of s

To solve Eq. (43), it is equivalent to solve the following linear algebraic equation as

$$\mathbf{V}^T\mathbf{V}\begin{bmatrix}\alpha_1\\ \alpha_2\\ \vdots\\ \alpha_m\end{bmatrix}=\mathbf{V}^T\begin{bmatrix}F_1\\ F_2\\ \vdots\\ F_n\end{bmatrix} \tag{44}$$

in which $\mathbf{V}\equiv\begin{bmatrix}\mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m\end{bmatrix}$.

Notice that if $m=n$ Eq. (44) is the same as Eq. (41) which will result in that the best choice for vector $\mathbf{v}$ is in the direction of $\mathbf{B}^{-1}\mathbf{F}$ if the inverse of Jacobian matrix exists. And the iteration formula becomes the well-known Newton method if $r=0$ in Eq.(32) is selected. It is more interesting when the Jacobian matrix becomes singular, then the inverse of Jacobian matrix does not exist at all. For most inverse problems, we all know that the numerical algorithm becomes unstable due to the inherent ill-posedness. It then can be found that it is impossible to numerically calculate $\mathbf{B}^{-1}$ at all. For solving n unknowns by n equations this becomes difficult because some of these equations are nearly linearly dependent to others. That means the rank of the Jacobian matrix is not equal to n such that one cannot construct n linearly independent vectors (directions, which are $\mathbf{u_i}$). To use less directions is then commonly seen in the techniques for solving the inverse problems. For example, the well-known Tikhonov's method adopts two directions; one is in the direction of $\mathbf{B^TF}$ with known coefficient $\alpha_1=1$ and the other direction is in the direction of $\mathbf{x}$ (the unknown vector) with an unknown coefficient to be determined. The Landweber iteration method adopts the direction of $\mathbf{B^TF}$ only, but it does not obey the property of manifold-based algorithm as mentioned in this paper. The scalar homotopy method [Ku, Yeih and Liu (2010)] selects the direction of $\mathbf{B^TF}$ and it follows the property of manifold-based algorithm. The fictitious time integration method [Liu and Atluri (2008)] selects the direction of $\mathbf{F}$ but it does not obey the property of manifold-based algorithm. On the other hand, the dynamic Jacobian inverse free method [Ku, Yeih and Liu (2011)] selects the direction of $\mathbf{F}$ and it follows the property of manifold-based algorithm. All these methods are proved to have the ability to overcome the numerical instability but they all suffer from slow convergence. To our best knowledge it is not known up to date that how many directions are allowed when ill-posed problems are encountered. In addition, how to determine the unknown weights of these selected directions for number of directions is bigger than two is still an open question.

In the following, we proposed a modified algorithm to select as many directions as

possible such that the convergence can be fastened.

Modified algorithm (for singular Jacobian matrix)

Step 1: Constructing the equations (n equations) for unknowns (m unknowns)

Step 2: Constructing the Jacobian matrix $\mathbf{B}$ *(which is n by m)*

Step 3: Checking the rank of $S=\mathbf{V^T V}$*(m by m matrix), one can adopt the following definition:*

$$tol(S) = m * norm(S) * \varepsilon \tag{45}$$

where the function $tol(S)$ *gives the tolerance value, the function norm returns the norm value of matrix S by using the largest singular value of S and* ε *is the priori-selected precision which depends on the problem itself and the accuracy of the computer. The rank of matrix S is now defined as the number of singular values which are bigger than* $tol(S)$

Step 4: Building m directions (u_i*) and using the following algorithm to pick only k directions where k is equal to the rank of matrix* $\mathbf{S}$:

Step 4.1 Building vectors $v_i = Bu_i$

Step 4.2 Calculating the norm of mismatch vectors, $h_i \equiv \left\| \frac{\mathbf{F^T F}}{\mathbf{v_i^T F}} \mathbf{v_i} - \mathbf{F} \right\|$.

Step 4.3 Pick the first k directions whose mismatch h_i *is smaller.*

The following examples demonstrate our proposed methods for solving NAES.

3 Numerical illustrations

3.1 Example 1

Let us consider a system of nonlinear algebraic equations as:

$$\begin{cases} x^2 - 2y - 1 = 0 \\ x - e^y = 0 \end{cases} \tag{46}$$

It is obvious that one of the solutions is (1,0). We now first test the case where $\mathbf{u_1}$ and $\mathbf{u_2}$ are known vectors. We assign $\mathbf{u}_1 = [\ 1 \quad 0\ ]^T$ and $\mathbf{u}_2 = [\ 0 \quad 1\ ]^T$. It can be seen that these two vectors are linearly independent vectors and therefore according to our direction the optimal direction should be selected such that $\mathbf{v}$ is parallel to $\mathbf{F}$. We first use the nonlinear algorithm to determine the optimal direction. For solving this nonlinear system with the use of the inner iteration on the nonlinear algorithm, we use the manifold-based algorithm developed by Liu and Atluri [Liu and Atluri, (2011b)] which is equivalent to let the searching direction be $\mathbf{u} = \mathbf{BF}^T$ in Eq. (32) with assigning $r = 0.01$. The convergent criterion for the inner iteration is set as $\|\mathbf{F}_\alpha\| \leq 10^{-20}$ where $\mathbf{F}_\alpha$ denotes the residual vector for solving Eqs.(36) and (38)

together. We records the number of inner iteration steps for each outer iteration in the nonlinear algorithm for solving unknown vector **x**. For the convergence criterion of Eq. (46) in the outer iteration, we set as the root mean square error is less than 10^{-5}. For the first outer iteration step, we give the initial guess for α is $[\ 1 \quad 0\ ]^T$. For the following outer iteration steps, we adopt the final value of α in the previous step as the initial guess. For the outer iteration, we give the initial guess of $(x,y) = (1,1)$

It is reported that after 8 outer iteration steps, the approximate solution is obtained as $(1.0023, 0.0023)$ which is very close to the solution (1,0). The root mean square error versus the number of outer iteration step is shown in Figure 1. The cosine function values of the angles between two vectors **v** and **F** for outer iteration steps are illustrated in Figure 2. It can be seen that these cosine function values are all close to 1 which means that **v** is approximately parallel to **F**. Remember that all numerical algorithm have errors, it is then not possible to exactly find the vector **v** which is parallel to **F** in numerical sense. In addition, we also record the number of inner iteration steps for each outer iteration step as the black dash line shown in Figure 3. Now we change the requirement to $\|\mathbf{F}_\alpha\| \leq 10^{-10}$, the result is shown as the red line in Figure 3. It can be seen that for the beginning, the convergence criterion can be reached very quickly. However, while the numerical solution is almost close to the real solution the required inner iteration increases which means that the Jacobian matrix for the system involving Eqs. (36) and (38) using the inner iteration for nonlinear algorithm varies too dramatically such that to use the value of α for the previous step as the initial guess for the current step is not suitable now.

Next, we assign $\mathbf{u_1} = \mathbf{B^T F}$ and $\mathbf{u}_2 = [\ x \quad y\ ]^T$. This means now the searching direction is no longer the fixed direction but relates to the residual vector and unknown vector. We set the convergence criterion for the inner iteration in the nonlinear algorithm as $\|\mathbf{F}_\alpha\| \leq 10^{-10}$ and all other conditions are the same as stated above. We first use the nonlinear algorithm to determine the value of α. In Figure 4, the cosine function values for the angles between the vectors **v** and **F** are illustrated. We can find out that for some iteration steps, the values are zero which means **v** is perpendicular to **F** and that means maximum value of s but not the minimum value is found. It then can be said that to use Eqs. (36) and (38) together using the nonlinear algorithm is not a robust method. We then change the algorithm for finding the value of α by using Eq. (44). The cosine function values for the angles between the vectors **v** and **F** are illustrated also in Figure 4 by the solid red line. It can be found that now we can see the angle between **v** and **F** is zero.

In the final stage for this example, we now use $\mathbf{u}_1 = [\ 1 \quad 0\ ]^T$ and $\mathbf{u}_2 = [\ 0 \quad 1\ ]^T$. For solving the optimal value of α, we use Eq. (44) to determine it. The conditions are all the same as the previous stage. The root mean square error versus the itera-

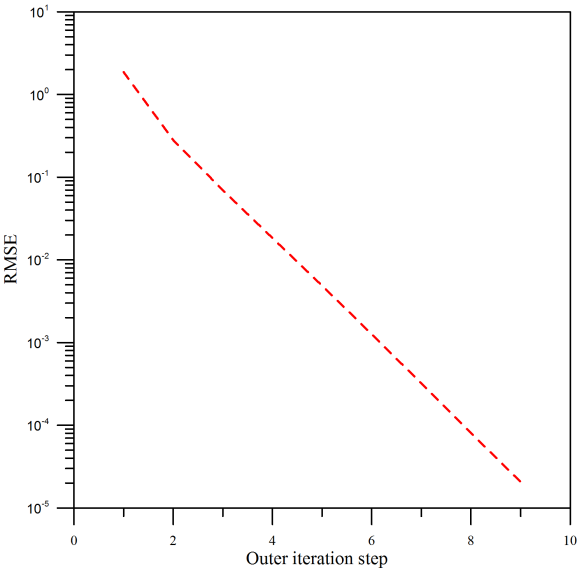

Figure 1: RMSE versus outer iteration step.

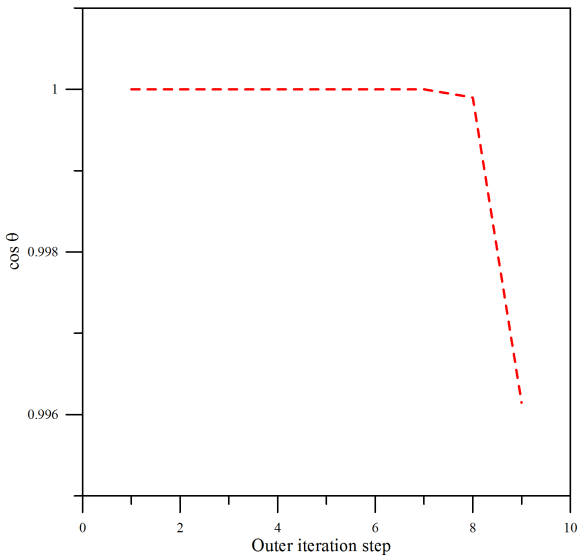

Figure 2: Cosine function value of angles between v and F versus outer iteration step.

tion step is illustrated in Figure 5, it can be seen that totally 9 steps are required to reach convergence. In addition, the cosine values of the angles between **v** and **F** for

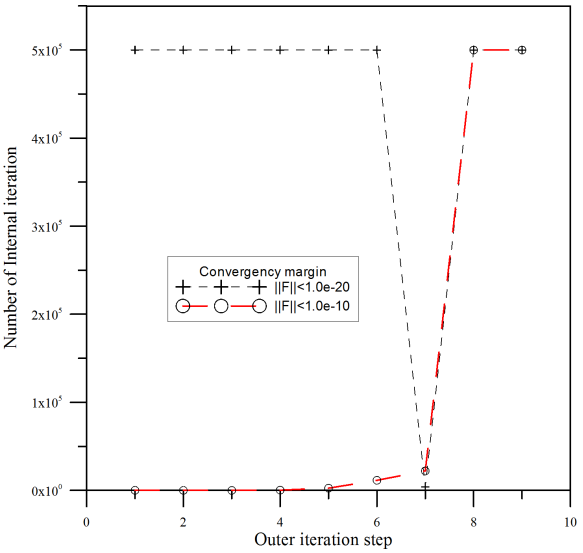

Figure 3: The influence of convergence criterion for the internal iteration on the required internal steps.

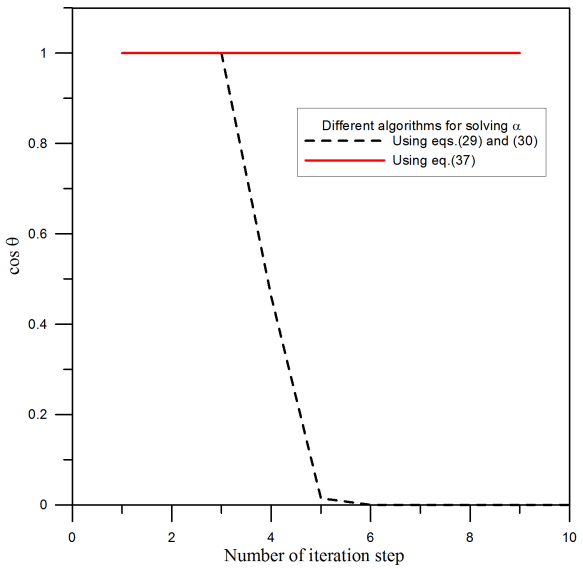

Figure 4: The influence of different algorithms for solving α

all iteration steps are illustrated in Figure 6. It can be seen that exactly the angle between **v** and **F** keeps zero during the iteration process. The approximate solution

obtained is $(1.0025, 0.0025)$.

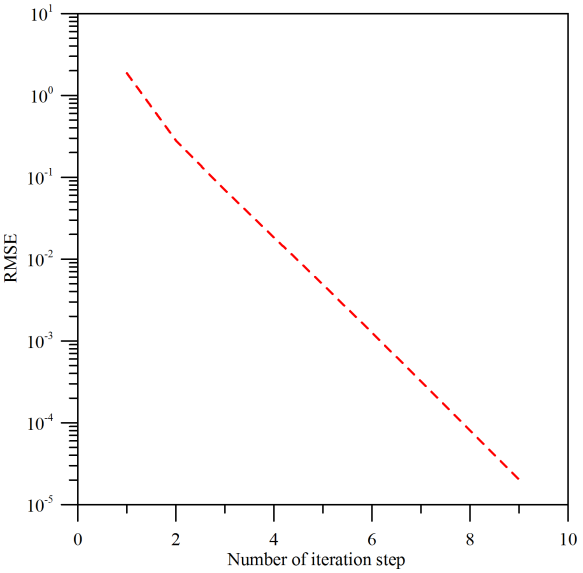

Figure 5: RMSE versus iteration step using Eq. (41) to determine αu

3.2 Example 2

In the second example, we solve for the following nonlinear equations:

$$\begin{cases} xy + y^2 z - 2 = 0 \\ x + 2y - 3z = 0 \\ xyz - e^{z-1} = 0 \end{cases} \tag{47}$$

There exists one obvious solution $(1,1,1)$. First, we assign $\mathbf{u}_1 = [\ 1 \quad 0 \quad 0\ ]^T, \mathbf{u}_2 = [\ 0 \quad 1 \quad 0\ ]^T$ and. $\mathbf{u}_3 = [\ 0 \quad 0 \quad 1\ ]^T$ We first use the nonlinear algorithm to solve this problem. The initial guess for the solution $(x_0, y_0, z_0) = (4,3,2)$ and the initial guess for $(\alpha_1, \alpha_2, \alpha_3) = (1/\sqrt{3}, 1/\sqrt{3}, 1/\sqrt{3})$. The convergence criterion for solving the undetermined coefficients for αs is set to be 10^{-4} and the convergence criterion for finding the solution (x, y, z) is set as the root mean square error is less than 10^{-10}. The results of the cosine values of the angles between $\mathbf{v}$ and $\mathbf{F}$ for all outer iteration steps are illustrated in Figure 7. It can be seen that the searching direction follows the direction of $\mathbf{B^{-1}F}$. One can also use Eq. (44) to determine the undetermined coefficients α and it is easy to verify that the searching direction really follows the direction of $\mathbf{B^{-1}F}$. Now let us assume we only select two directions, say $\mathbf{u}_1 = \mathbf{B}^{-1}\left[\ 1 \quad 0 \quad 0\ \right]^T$ and $\mathbf{u}_1 = \mathbf{B}^{-1}\left[\ 0 \quad 1 \quad 0\ \right]^T$. We then can say

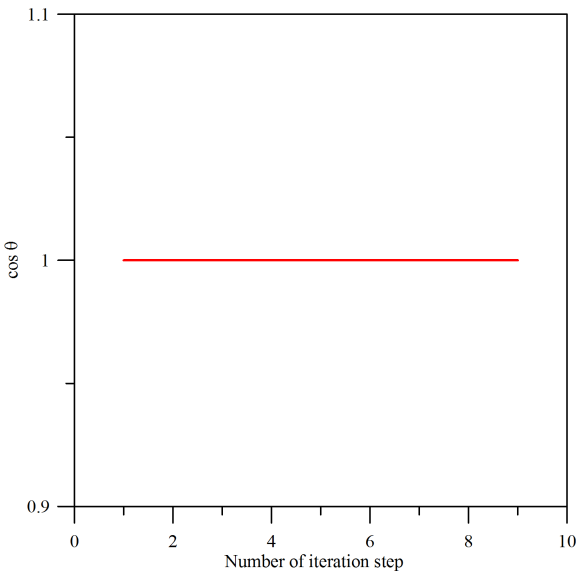

Figure 6: Cosine function value of angles between **v** and **F** versus outer iteration step using Eq. (41) to determine αu

the optimal selection of the undetermined coefficients (α_1, α_2) must make the vector $\mathbf{v} = \begin{bmatrix} F_1 & F_2 & 0 \end{bmatrix}^T$. We define the mismatch as $(v_1 - F_1)^2 + (v_2 - F_2)^2$which theoretically should be zero. Eq. (44) is used to solve the unknown undetermined coefficients (α_1, α_2) and mismatches for the first ten steps are illustrated in Figure 8 and it can be said that the current proposed algorithm really can make **v** and **F** as closely as possible.

3.3 Example 3

In this example, we study the following system of two NAEs:

$$\begin{aligned} F_1(x_1, x_2) &= x_1^2 + x_2^2 - 2 = 0, \\ F_2(x_1, x_2) &= e^{(x_1 - 1)} + x_2^2 - 2 = 0, \end{aligned} \tag{48}$$

where

$$\mathbf{B} = \begin{bmatrix} 2x_1 & 2x_2 \\ e^{(x_1-1)} & 2x_2 \end{bmatrix} \tag{49}$$

This is an interesting example because the iteration for Newton's method fails when the initial guess is selected as (3,5) as shown in Figure 9. As the trajectory approaches to $x_1 = 3.5192$ during the iteration, it happens $x_2 \approx 0.0$. It then is found

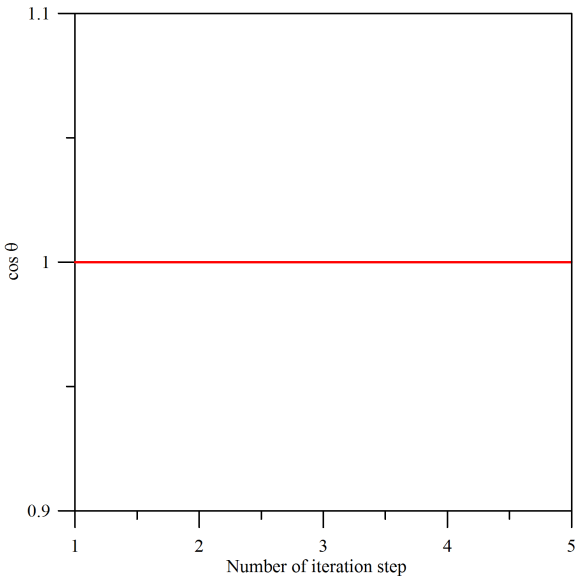

Figure 7: Cosine function value of angles between **v** and **F** versus outer iteration step using Eqs.(36) and Eq. (37) to determine α

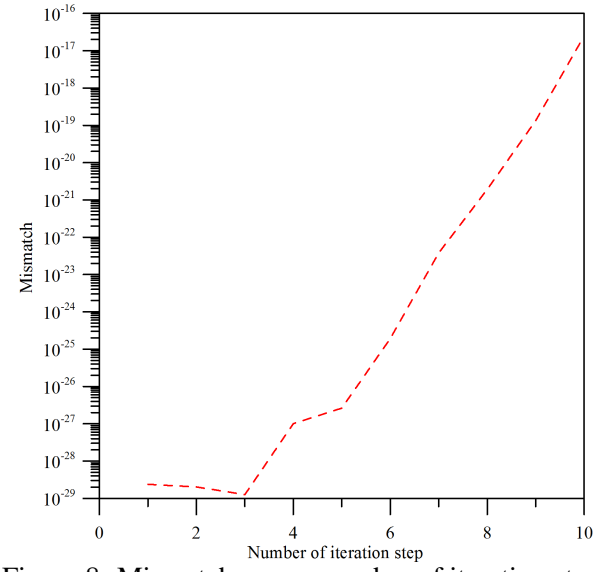

Figure 8: Mismatch versus number of iteration step.

from Eq. (49) that the Jacobian matrix now is nearly singular. This leads the trajectory of (x_1, x_2) oscillates at the axis for $x_1 = 3.5192$ as shown in Figure 9.

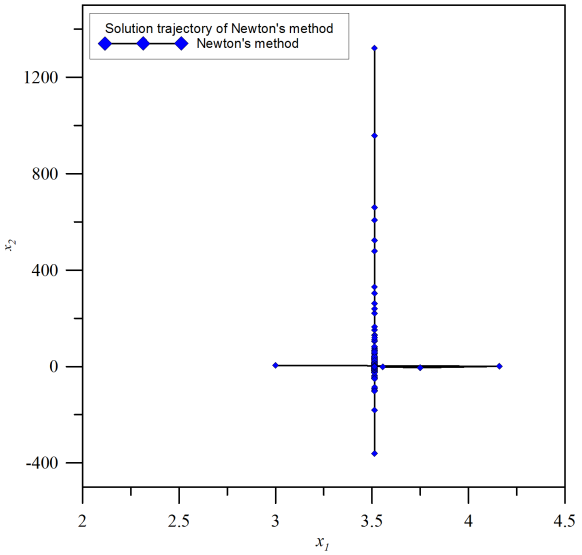

Figure 9: Trajectory for the solution using Newton's method.

The convergence criterion is selected as the root mean square error is less than 10^{-6}. We now use the modified algorithm mentioned in the previous section and make the precision in the modified algorithm: $\varepsilon = 0.01$. The initial guess is set as (x_1, x_2)=(3,5). While the rank of the Jacobian matrix is not equal to 2, according to our modified algorithm we can only choose one direction. We set our two directions as $\mathbf{u}_1 = \mathbf{F}$ and $\mathbf{u}_2 = \mathbf{B}^T\mathbf{F}$. The following three methods are tried as mentioned in the followings:

Method 1. When $\mathbf{rank}(\mathbf{B}) = 1$, set $\mathbf{u} = \mathbf{u}_1$ always.

Method 2. When $\mathbf{rank}(\mathbf{B}) = 1$, set $\mathbf{u} = \mathbf{u}_2$ always.

Method 3. When $\mathbf{rank}(\mathbf{B}) = 1$, set $\mathbf{u}$ by the modified algorithm.

We first check the convergence of root mean square error (RMSE) as shown in Figure 10. We can find that Newton's method fails to converge. Other methods such as method 1 to method 3 can converge. Actually when we check the trajectory of $(x_1, x_2)$for all methods, we can find that method 1 happens to be the same as method 3. These two methods all converge after 11 iterations. Method 2 converges after 35 iterations. It is worth mentioned that method 1 (method 3 also) converges to the solution (x_1, x_2)=(1,-1) and method 2 converges to other solution (x_1, x_2)=(1,1). The trajectories for all methods are shown in Figure 11.

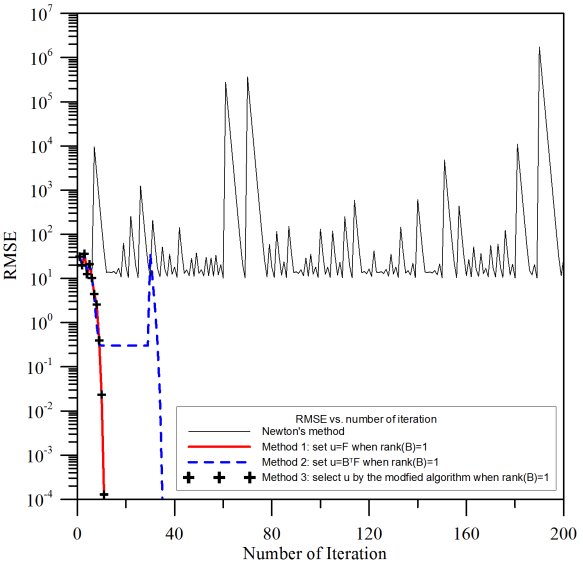

Figure 10: Evolutions of root mean square errors for example 3 using different methods.

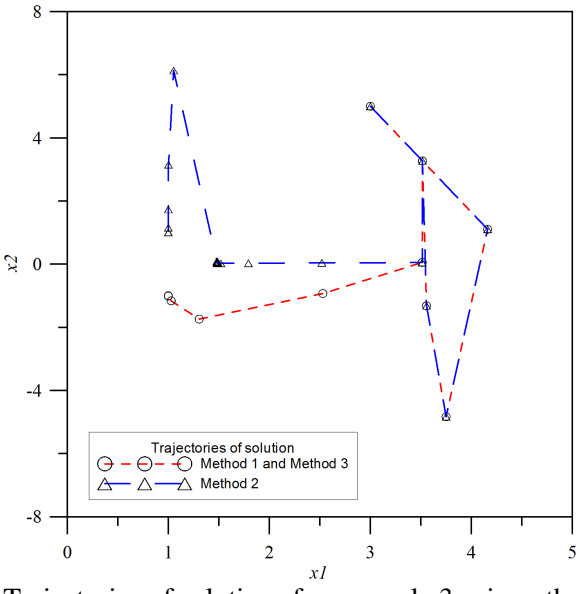

Figure 11: Trajectories of solutions for example 3 using other methods.

3.4 Example 4

In this example, we consider an almost linear problem [Brown (1973)]:

$$F_i = x_i + \sum_{j=1}^{j=n} x_j - (n+1), i = 1, \cdots, n-1, \tag{50}$$

$$F_n = \prod_{j=1}^{j=n} x_j - 1 \tag{51}$$

with a closed-form solution as $x_i = 1$, for $i = 1, \cdots, n$.

It is easy to find that if the path of solution approaches to the origin ($\mathbf{x} = \mathbf{0}$) the Jacobian matrix becomes singular and results in the failure of Newton's method. We now adopt the modified algorithm stated in the previous section to solve this problem. In our study, the number of equations is set as $n = 10$. The parameter ε is set to be 10^{-16}, the convergence criterion is RMSE$\leq 10^{-6}$ and the maximum number of iteration steps is 20,000 steps. The parameter r in Eq. (32) is set to be zero. The initial guess is given as $\mathbf{x}_0 = [0.1, 0.1, 0.1, 0.1, 0.3, 0.1, 0.1, 0.1, 0.1, 0.2]^T$ We study the following three methods. Method 1 is that we only use one direction, $u_1 = B^{\mathbf{T}} F$ Method 2 is that we construct ten directions by adopting $u_1 = \mathbf{B^T F}/norm(\mathbf{B^T})$ and $\mathbf{u}_k = \mathbf{B u}_{k-1}/norm(\mathbf{B}), k = 2, ..., 10$. The modified method stated in the previous section is then used to select appropriate directions. Method 3 is that we construct ten directions by using ten unit vectors. The modified method stated in the previous section is then used to select appropriate directions.

We first examine the RMSE versus the number of iteration steps for three methods as shown in Fig. 12. It can be found that all three methods can obtain the solution. It takes 2516 steps for method 1, 22 steps for method 2 and 8 steps for method 3. We further examine the rank(S) for each step using method 2 or method 3 as shown in Fig. 13. It can be seen for the first step, the rank of the S matrix is not equal to 10 but 9 for method 3 and the rank of the S matrix is 10 for method 1. From the second step to the final step, the rank of S matrix for both method 1 and method 3 is 10. It is then can be said that method 1 is totally equivalent to Newton's method. The reason why method 3 converges faster than method 1 is because that after the first step method 3 makes the location of $\mathbf{x}$ at a better point such that further using the Newton's method is more efficient.

3.5 Example 5

A classical example of an ill-posed problem is a nonlinear Fredholm integral equation of the first kind which is well known as a nonlinear ill-posed problem. In the

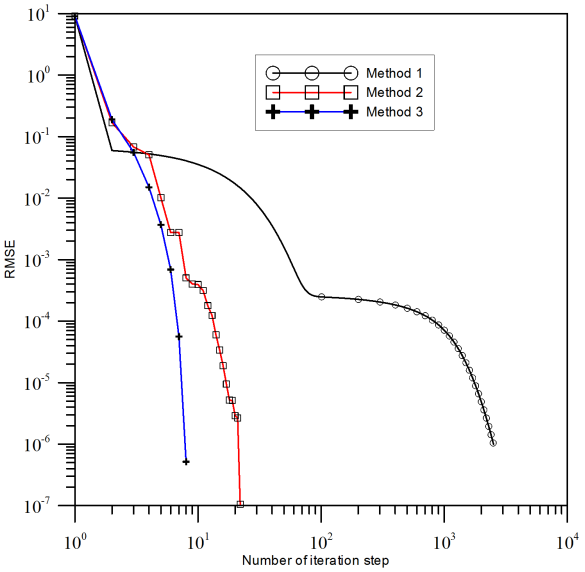

Figure 12: RMSE versus number of iteration for three different methods in example 4.

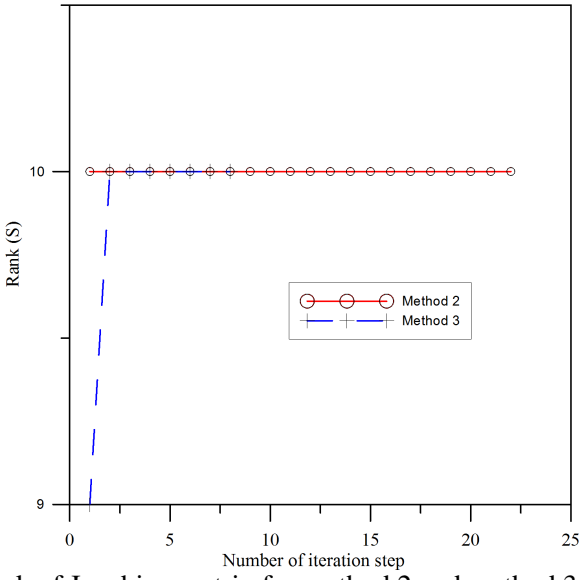

Figure 13: Rank of Jacobian matrix for method 2 and method 3 in example 4.

final example, we study the example as follows.

$$\int_0^1 x(s)\,x(t)\,dt = A\cos(\beta s), A > 0$$

where A and β are constants. We let $A = 1$ and $\beta = 3$ in the followings. We give data for $A\cos(\beta s)$ in the region $s \in [0,1]$ by equally dividing the region into 20 segments, that means totally 21 data points are used. Therefore, we also use these 21 points as the integration quadrature points and the trapezoidal rule is used for integration. Two exact solutions exist: $x(s) = \pm\sqrt{\frac{A\beta}{\sin\beta}}\cos(\beta s) = \pm\sqrt{\frac{3}{\sin 3}}\cos(3s)$. [Polyanin and Manzhirov, 2007]. The initial guess are given as $x(t) = 10.0$ for $t \in [0,1]$ and the following three methods are used to find the solutions:

Method 1: We only adopt one direction as $\mathbf{u}_1 = \frac{\mathbf{F}}{\|\mathbf{F}\|}$

Method 2: We only adopt one direction as $\mathbf{u}_1 = \frac{\mathbf{B}^T\mathbf{F}}{\|\mathbf{B}^T\mathbf{F}\|}$

Method 3: We only adopt two directions as $\mathbf{u}_1 = \frac{\mathbf{F}}{\|\mathbf{F}\|}, \mathbf{u}_2 = \frac{\mathbf{B}^T\mathbf{F}}{\|\mathbf{B}^T\mathbf{F}\|}$ and then determine the optimal coefficients using the linear algorithm.

Method 4: We first construct 21 directions as $\mathbf{u}_1 = \frac{\mathbf{F}}{\|\mathbf{F}\|}$ and $\mathbf{u}_k = \frac{\mathbf{B}^T\mathbf{u}_{k-1}}{\|\mathbf{B}^T\mathbf{u}_{k-1}\|}$ for $k = 2,...,21$ and then determine the optimal coefficients using the modified algorithm. The parameter $\varepsilon = 10^{-10}$ is used.

All these methods use the following termination criterions: (1) when the root mean square error is less than 10^{-3}, or (2) when the number of iteration steps exceeds 1,000 steps. In addition, the parameter r in Eq. (32) is set to be zero.

We will examine the influence of noise by examining results for random relative noises level $\sigma = 0\%$ and 1% in data. We first examine the root mean square error versus iteration steps as shown in Figure 14. In this figure, we only show results that obtained by using data without any noise for different methods. We can see that method 1 and method 2 which only adopt one searching direction cannot reach the convergence criterion within 1,000 steps. On the contrary, method 3 and method 4 can reach the convergence criterion in 9 steps. In method 3, we adopt two specific directions but in method 4 we use the modified algorithm which can automatically select as many searching directions as possible. We illustrate rank($\mathbf{S}$) for method 4 as shown in Figure 15. We can see that for method 4, rank is 2 for all steps that means method 4 selects two directions only. This is similar to the method 3. One may wonder if method 4 chooses the same directions as method 3 does. Remember that the first two directions $\mathbf{u}_1$ and $\mathbf{u}_2$ are exactly the specific directions used in method 3. We check this doubt by showing Figure 16. We can find out that actually method 4 adopts different directions in comparison with method 3.

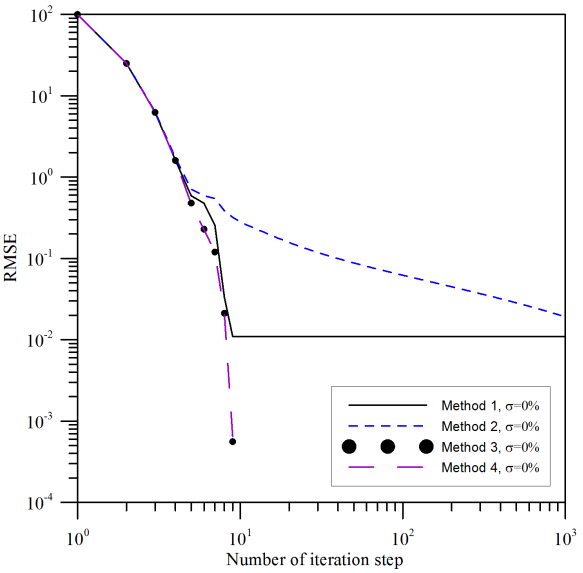

Figure 14: RMSE versus number of iteration for different methods in Example 5.

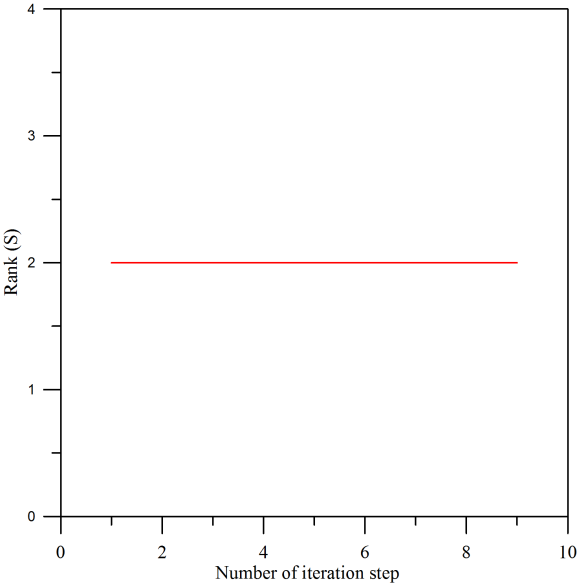

Figure 15: Rank for S matrix for each step versus number of iteration step in method 4.

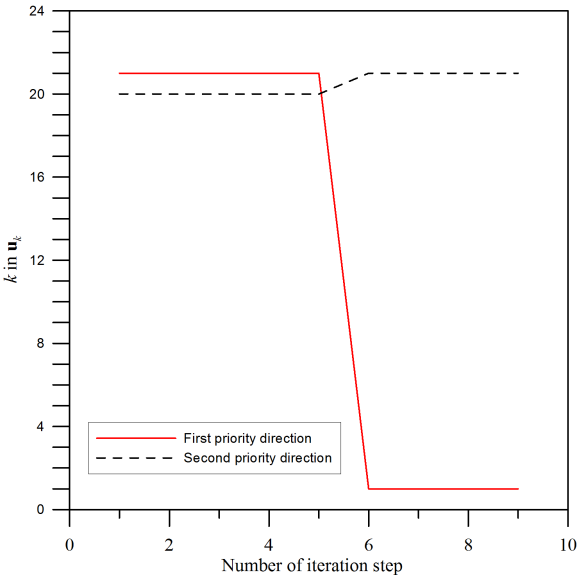

Figure 16: Selected directions for method 4.

In the final, we will examine the results by only show the results for those with noise level of 1% as shown in Figure 17. It can be seen that although method 1 and method 2 basically already catch the shape for the exact solution but deviation between numerical solution and exact solution can be easily found. Method 3 and method 4 obtain very close solution and deviation is negligible. Remember that in this figure, all data are added with random noise. It then can be said that our proposed method has a good noise resistance. One more thing worth mentioned here is that method 3 and method 4 can reach solution very quickly and this fast convergence property makes our method more interesting. We believe this proposed method has a very promising future in solving the ill-posed inverse problems but we have to admit more research should be carried out to fulfill this.

4 Conclusions

In this paper, a scalar homotopy method with optimal hybrid search directions for solving nonlinear algebraic equations is proposed. The important fundamental concepts and the construction of the scalar homotopy method, the iteration algorithm based on the evolution of the residual vector, and the optimization of multiple hybrid search directions are clearly addressed. Several numerical illustrations for solving the problems with ill-posedness which may have ill-conditioned or singular Jacobian matrix are conducted. Findings from this study are drawn as follows.

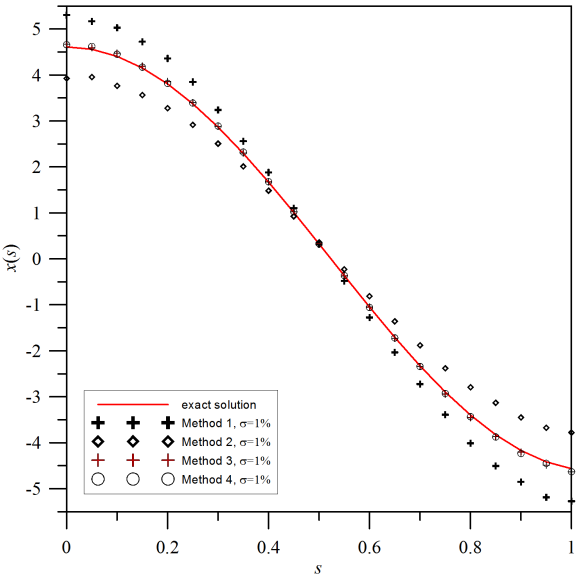

Figure 17: Solution for example 5 using different methods.Solution for example 5 using different methods.

1. The proposed method is based on the construction of a scalar homotopy function with the use of the optimization algorithm to find optimal hybrid search directions. With the novel formulation proposed in this study, the present method is applied to the solution of nonlinear algebraic equations. Results reveal that the proposed method can improve the convergence and increase the numerical stability for solving ill-posed problems.

2. Taking the advantages of finding the optimal search directions, the proposed novel method is able to consider hybrid search directions for solving nonlinear algebraic equations. The formulation presented in this paper demonstrates a variety of flexibility with the use of the algorithm for finding optimal hybrid search directions which can be used to improve the convergence and increase the numerical stability.

3. Numerical illustrations reveal that the selection of the optimal vector plays a crucial role for finding the stable solution. With the use of the proper optimal vector, the proposed method reaches the solution very quickly and this fast convergence property makes our method more interesting. In addition, difficulties such as slow convergence, numerical instability, or low accuracy for solving the ill-posed problems that arose previously in many numerical

methods may be overcome by means of our proposed method.

Acknowledgement: This study was supported by the National Science Council under project grant NSC100-2628-E-019-054-MY3. The authors would like to thank the National Science Council for the generous financial support.

References

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

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.

Groetsch N. (1984): The Theory of Tikhonov Regularization for Fredholm Integral Equations of the First Kind. London: Pitman.

Hansen, P. C. (1992): Analysis of discrete ill-posed problems by means of the L-curve. SIAM Rev., vol. 34, pp. 561-580.

He, J. H. (2005): Homotopy perturbation method for bifurcation of nonlinear problems, *International Journal of Nonlinear Science and Numerical Simulation*, v6. pp. 207-208.

Hirsch, M. W.; Smale, S. (1979): On Algorithms for Solving $f(x) = 0$, *Communications on Pure and Applied Mathematics*, Volume 32, Issue 3, pp. 281–312.

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

Ku, C.-Y.; Yeih, W.; Liu, C.-S.; Chi, C.-C. (2009): Applications of the fictitious time integration method using a new time-like function. *CMES: Computer Modeling in Engineering & Sciences*, Vol. 43, No. 2, pp. 173-190.

Ku, C.-Y.; Yeih, W.; Liu, C.-S. (2010): "Solving Non-Linear Algebraic Equations by a Scalar Newton-homotopy Continuation Method," International Journal of Nonlinear Sciences and Numerical Simulation, 11(6), pp. 435-450.

Ku, C.-Y.; Yeih, W.; Liu, C.-S. (2011): "Dynamical Newton-Like Methods for Solving Ill-Conditioned Systems of Nonlinear Equations with Applications to Boundary Value Problems," CMES: Computer Modeling in Engineering & Sciences, Vol.

76, No. 2, pp. 83-108, 2011.

Ku, C.-Y.; Yeih, W.; Liu, C.-S.; Chi, C.-C. (2009): "Applications of the Fictitious Time Integration Method Using a New Time-Like Function", CMES: Computer Modeling in Engineering & Sciences, Vol. 43, No. 2.

Liao, S. J. (2004): On the homotopy analysis method for nonlinear problems. *Appl. Math. Comput.*, 47, pp. 49-513.

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.; 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.

Liu, C.-S. ; Atluri, S. N. (2011a): An Iterative Method Using an Optimal Descent Vector, for Solving an Ill-Conditioned System $\mathbf{BX} = \mathbf{b}$, Better and Faster than the Conjugate Gradient Method. *CMES: Computer Modeling in Engineering & Sciences*, vol. 80, pp. 275-298.

Liu, C.-S.; Atluri, S. N. (2011b): 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. *CMES: Computer Modeling in Engineering & Sciences*, vol. 71, pp. 279-304.

Liu, C.-S. ; Atluri, S. N. (2011c): An Iterative Algorithm for Solving a System of Nonlinear Algebraic Equations, $\mathbf{F}(\mathbf{x}) = \mathbf{0}$, Using the System of ODEs with an Optimum α in $\dot{\mathbf{x}} = \lambda \left[\alpha \mathbf{F} + (1-\alpha)\mathbf{B}^{\mathbf{T}}\mathbf{F}\right]; B_{ij} = \partial F_i / \partial x_j$. *CMES: Computer Modeling in Engineering & Sciences*, vol. 73, pp. 395-431.

Liu, C.-S; Kuo, C.-L.(2011): A Dynamic Tikhonov Regularization Method for Solving Nonlinear Ill-posed Problems. *CMES: Computer Modeling in Engineering & Sciences*, vol. 76, No.2, pp.109-132.

Lemieux, J.F.; Tremblay, Bruno; Sedlacek, Jan; Tupper, Paul; Thomas, Stephen; Huard, David; Auclair, Jean-Pierre (2010): Improving the numerical convergence of viscous-plastic sea ice models with the Jacobian-free Newton-Krylov method. *Journal of Computational Physics*, 229, pp. 2840-2852.

Morozov, V. A. (1966): On regularization of ill-posed problems and selection of regularization parameter. J. Comp. Math. Phys., vol. 6, pp. 170-175.

Morozov, V. A. (1984): Methods for Solving Incorrectly Posed Problems. Springer, New York.

Polyanin, A. D., Manzhirov, A. V. (2007): Handbook of Integral Equations. Second Edition, Caoman & Hall/CRC, Taylor & Francis Group, New York.