MATHEMATICS OF COMPUTATION
Volume 74, Number 250, Pages 805–818
S 0025-5718(04)01689-8
Article electronically published on September 16, 2004

CONJUGATE GRADIENT PREDICTOR CORRECTOR METHOD FOR SOLVING LARGE SCALE PROBLEMS

MUHAMMED I. SYAM

Abstract. In this paper, we give a new method for solving large scale problems. The basic idea of this method depends on implementing the conjugate gradient as a corrector into a continuation method. We use the Euler method as a predictor. Adaptive steplength control is used during the tracing of the solution curve. We present some of our experimental examples to demonstrate the efficiency of the method.

1. Introduction

Continuation methods have long served as useful theoretical tools in modern mathematics. Their use can be traced back at least to such works as those of [29], [21] and [4]. Leray and Schauder [24] refined the tool and presented it as a global result in topology. The use of deformations to solve nonlinear systems of equations may be traced back at least to [23]. The classical continuation methods were the first deformation methods to be numerically implemented and may be regarded as a forerunner of the predictor corrector methods for the numerical path following.

Because of their versatility and robustness, numerical continuation methods have now been finding ever wider use in scientific applications. Introductions into aspects of the subject of numerical continuation methods may be found in the books [10], [13], [18], [31], [32], [36], and [1].

One of the primary applications of continuation methods involves the numerical solution of nonlinear eigenvalue problems; see [6], [19], [20], [17], [32], [7] and [28]. Such problems are likely to have arisen from a discretization of an operator equation in a Banach space context, which also involves an additional bifurcation parameter. For some specific examples see [1], [14]. As a result of the discretization, the corresponding finite dimensional problem $H(u) = 0$ where $H : \Re^{N+1} \to \Re^N$ may require that N be quite large. This leads to the task of solving large scale continuation problems.

There are many areas where large scale continuation methods exist. An area in which a considerable amount of experience concerning large scale continuation methods exists is structural mechanics; see [31] and the references cited therein. Also, some work was done on combining continuation methods with multigrid methods for solving large scale continuation problems arising from discretization of elliptic problems via finite differences; see [5], [3], and [28]. Another area where large

Received by the editor April 15, 2002 and, in revised form November 23, 2003.
2000 *Mathematics Subject Classification.* Primary 65.
Key words and phrases. Continuation methods, large scale problems, conjugate gradient method.

scale continuation problems have been treated concerns finite element discretizations of elliptic problems, which are then combined with a conjugate solver in the continuation algorithm; see [11], [30], and [2]. From this discussion, we see that a variety of combinations can be made of continuation algorithms and sparse solvers; see [1] and [12]. In this paper, we indicate more specifically how to incorporate a conjugate gradient method or a Krylov subspace method. In Section 2, the idea of the predictor corrector continuation method in general will be presented. In addition, we recall the nonlinear conjugate gradient method in Section 3 while the conjugate gradient predictor corrector algorithms will be developed in Section 4. Finally, numerical examples will be presented in Section 5.

2. Predictor corrector continuation method

To describe the idea of the predictor corrector continuation method, let us consider curves which are implicitly defined by an underdetermined system of equations

$$H(u) = 0$$

where $H : \Re^{N+1} \to \Re^N$ is a smooth map. We shall mean that a map is *smooth* if it has as many continuous derivatives as the context of the subsequent discussion requires. We call u a *regular point* of H if the Jacobian $H'(u)$ has maximal rank while v is a *regular value* of H if u is a regular point of H whenever $H(u) = v$. If a point or a value is not regular, then it is called *singular*. If $u_0 \in \Re^{N+1}$ is a regular point of H such that $H(u_0) = 0$, it follows from the Implicit Function Theorem that the solution set $H^{-1}(0)$ can be locally parametrized about u_0 with respect to some parameter, say arclength s; see [26]. We thus obtain the solution curve $c(s)$ of the equation $H(u) = 0$.

For convenience of the theoretical and practical discussions, we usually use parametrization with respect to arclength. Hence, by re-parametrization (according to arclength), we obtain a smooth solution curve $c : I \to \Re^{N+1}$ for some open interval I containing zero such that for all $s \in I$:

$$\begin{aligned} c(0) &= u_0, \\ H'(c(s))\,\dot{c}\,(s) &= 0, \\ \|\dot{c}\,(s)\| &= 1, \\ \det \begin{pmatrix} H'(c(s)) \\ \dot{c}\,(s)^* \end{pmatrix} &> 0. \end{aligned}$$

The above conditions uniquely determine the tangent $\dot{c}\,(s)$ with a specific orientation. Here and in the following, A^* denotes the Hermitian transpose of A, $\|u\|$ the Euclidean norm of u, H' the total derivative of H, and $\dot{c}$ the derivative of c with respect to arclength. The above solution curve $c(s)$ is characterized as the solution of the initial value problem

$$\dot{u} = t(H'(u)),\ u(0) = u_0$$

where $t(H'(u))$ is the tangent vector induced by $H'(u)$.

Since the solution curve c is characterized by the above initial value problem, it is evident that the numerical methods for solving initial value problems can immediately be used to numerically trace c. In fact, a typical path following method consists of a succession of two different steps.

Predictor step: An approximation step along the curve, usually in the general direction of the tangent of the curve. It is called the *Euler-predictor* which is given by

$$v = u + ht(H'(u))$$

where u is a point lying along the solution curve c, and $h > 0$ represents a stepsize.

Corrector steps: One or more iterative steps for solving $H(u) = 0$ which bring the predicted point back to the curve. It is called the *Gauss-Newton-Corrector* which is given by

$$W = v - H'(v)^{+}H(v).$$

We should note that if A is an $N \times (N+1)$ matrix with maximal rank, then $A^{+} = A^{*}(AA^{*})^{-1}$ and it is called the *Moore-Penrose inverse of A.* We can use a fixed stepsize h or adaptive steplength control. We now describe the Newton adaptive steplength control. Let $f : Range(c) \to \Re$ be a smooth functional. One choice of f is

$$f(x) = (x - u_0)^{*}t_0$$

where $t_0 = t(H'(u_0))$.

Suppose that the predictor-corrector method produced two successive points $c(s_{k+1})$ and $c(s_k)$ such that $f(c(s_{k+1}))\times\ f(c(s_k)) < 0$. Then, it is reasonable to replace the usual steplength adaptation used to traverse the curve c by a Newton-steplength adaptation which is motivated by the following one-dimensional Newton method for solving the equation $f(c(s)) = 0$:

$$s_{n+1} = s_n - \frac{f(c(s_n))}{f'(c(s_n))C'(s_n)}.$$

The last equation suggests that we can take the new steplength

$$h = -\frac{f(c(s_n))}{f'(c(s_n))C'(s_n)}$$

at $u = c(s_n)$ in order to obtain a predictor point $v = u + ht(H'(u))$, which should lead to a better approximation of a zero point of f on c. In addition, under the normal hypotheses of smoothness, if the initial value of $c(s_n)$ is sufficiently near a zero point of f, then the usual quadratic convergence of the iterative process can be expected. For more detail, see [1], [34], [35], and [33].

3. Nonlinear conjugate gradient method

Let us assume that the problem to be solved is

$$\min_{u}\{f(u) : u \in \Re^{N+1}\} \tag{1}$$

where $f : \Re^{N+1} \to \Re$ is a smooth nonlinear functional, usually having an isolated local minimal point $\widetilde{u}$ which we desire to approximate. We recall the nonlinear conjugate gradient method of Polak and Ribiere [30]. The following is an outline of the conjugate gradient method due to them.

Algorithm 3.1 (Nonlinear conjugate gradient).

```
Input : $u_0 \in \Re^{N+1}$          % initial point
Output: $u_0, u_1, \ldots$           % converging to $\widetilde{u}$
Step 1: $g_0 = \bigtriangledown f(u_0)$ and $d_0 = g_0$;        % initial gradients
Step 2: For $j = 0, 1, \ldots$ do steps 3-7
Step 3: Set $\rho_j = \arg\ \min_{\rho>0} f(u_j - \rho d_j)$;      % line search
Step 4: Set $u_{j+1} = u_j - \rho_j d_j$;
Step 5: Set $g_{j+1} = \bigtriangledown f(u_{j+1})$;
Step 6: Set $\gamma_j = \frac{(g_{j+1}-g_j)^* g_{j+1}}{\|g_j\|^2}$;
Step 7: Set $d_{j+1} = g_j + \gamma_j d_j$;        % new conjugate gradient
Step 8: Stop.
```

For more detail about the nonlinear conjugate gradient method and its properties, see [9], [12], and [16]. This method has some difficulties and disadvantages. We mention some of them.

1) The convergence of the conjugate gradient method is sometimes slow. Its superlinear convergence is somewhat unsatisfactory; see [8] and [27].

2) There are various possibilities for obtaining the factors γ_j in Algorithm 3.1. The choice of γ_j will affect the convergence of the conjugate gradient method. Hence, a wrong choice will make the algorithm diverge.

3) In practice and in order to obtain an acceptable ρ_j, one need not do a very precise one-dimensional minimization in the line search step since it is costly. Most of the convergence rate proofs require cyclic reloading, i.e., setting $\gamma_j = 0$ after $N+1$ steps. The general idea of such proofs involves the approximation of $f(u)$ via the Taylor formula by

$$f(u) \simeq f(\widetilde{u}) + f'(\widetilde{u})(u-\widetilde{u}) + (u-\widetilde{u})^* \bigtriangledown f'(\widetilde{u})(u-\widetilde{u})$$

and then using the convergence results for the quadratic case. Actually, even in the simple cases, because of the presence of rounding errors, we cannot expect that stopping will occur after $N+1$ steps. Instead, we should regard the conjugate gradient method, even in this case, as an iterative method which makes a substantial improvement after k steps. For more detail, see [12]. For these reasons, authors combine the conjugate gradient with a preconditioning to make this method more suitable.

The ideal convergence (one step!) would occur when the condition number is one, i.e., when all of the eigenvalues of $\bigtriangledown f'$ are equal. Intuitively, the next best situation would occur when the eigenvalues have as few cluster points as possible. This observation is used to motivate the idea of preconditioning for the conjugate gradient method. For more detail, see [12].

In the preconditioned conjugate gradient (PCG) method, we will use the incomplete Cholesky factorization as a preconditioner. The idea behind the PCG method is to compute the incomplete Cholesky factorization for the Hessian matrix $\bigtriangledown f(u_j)$. Then, we use this factorization in Steps 1 and 5 in Algorithm 3.1 instead of $\bigtriangledown f(u_j)$. In this technique, the preconditioning matrix will vary from iteration to iteration. This will affect the cost of this method. We will describe this preconditioner in Section 4. In Section 5, we will compare the PCG method and our contribution in Algorithm 4.2. In the next section, we present our new method, which is the conjugate gradient predictor corrector method.

4. Conjugate gradient predictor corrector method

Let us now return back to our major problem, namely minimizing the functional

$$f(u) = \frac{1}{2}\|H(u)\|^2 \tag{2}$$

where $H : \Re^{N+1} \to \Re^N$. The minimal points of (2) obviously form the solution curve which can be traced by the continuation method. From (2), we have

$$\begin{aligned} \nabla f(u) &= H'(u)^* H(u); \\ \nabla f'(u) &= H'(u)^* H'(u) + \bigcirc(\|H(u)\|). \end{aligned}$$

The gradient $\nabla f(u) = H'(u)^* H(u)$ is orthogonal to the tangent vector $t(H'(u))$. This motivates the idea for implementing the conjugate method (Algorithm 3.1) as a corrector into a continuation method. Analogous to the case when the minimization problem has an isolated solution at which the Hessian is positive definite, we may expect local superlinear convergence of the conjugate gradient method for the functional (2).

The solution will be a point $\widetilde{u} \in H^{-1}(0)$ which is essentially nearest to a predictor point v which is taken as the starting point for the (nonlinear) conjugate gradient method. Numerical experience suggests that local superlinear convergence holds, as in the case of the functional f with an isolated minimal point. We propose the above conjugate gradient method as a reasonable corrector procedure nevertheless, provided once again that an effective preconditioning is incorporated. We will propose a preconditioning of the dependent variables, which we now describe in more detail.

Let us consider the following transformation of f:

$$\varphi(u) = \frac{1}{2} \left\|L^{-1} H(u)\right\|^2 \tag{3}$$

where L is an as yet unspecified nonsingular $N \times N$ matrix. By computing the first and the second derivatives of $\varphi(u)$, we have

$$\begin{aligned} \nabla\varphi(u) &= H'(u)^*(LL^*)^{-1}\, H(u); \\ \nabla\varphi'(u) &= H'(u)^*(LL^*)^{-1} H'(u) + \bigcirc(\|H(u)\|). \end{aligned} \tag{4}$$

If we assume that our continuation method furnishes predictor points which are already near $H^{-1}(0)$, we may neglect the $\bigcirc(\|H(u)\|)$ term in (4). Furthermore, if $H(u) \approx 0$, then an ideal choice would be to take L such that

$$LL^* = H'(u)H'(u)^*$$

is the Cholesky decomposition. Thus, we have

$$\begin{aligned} \nabla\varphi(u) &= H'(u)^*(LL^*)^{-1} H(u) \\ &= H'(u)^*(H'(u)H'(u)^*)^{-1} H(u) \\ &= H'(u)^+ H(u), \end{aligned}$$

where $H'(u)^+ = (H'(u))^*(H'(u)H'(u)^*)^{-1}$. Hence in this case, the gradient $\nabla\varphi(u) = H'(u)^+ H(u)$ coincides with the usual Newton direction which we used as the standard corrector in Section 2.

If we want to use the Cholesky decomposition, then we would relinquish whatever advantage sparseness may have offered. Also, we would prefer to determine L with a small computational expense and in such a way that the linear equations $LL^*x = y$ are cheaply solved for x. For these reasons, our preconditioning involves computing an incomplete Cholesky factorization. To describe this factorization, let $A = H'(u)H'(u)^*$. The main idea behind this factorization is to calculate a lower triangular matrix L with the property that L has some tractable sparsity structure and is somehow "close" to A's exact Cholesky factor. The preconditioning is then taken to be LL^*. We should note that $L_{ij} = 0$ if the corresponding $a_{ij} = 0$. This leads to the following algorithm.

Algorithm 4.1 (Incomplete Cholesky Preconditioner)**.**

```
INPUT: $A$
        $n$        % the size of the matrix $A$,
OUTPUT: L   % lower triangular matrix so that $LL^* \approx A$.
STEP 1: For $k = 1 : n$ do steps 2-7.
STEP 2: Set $a_{kk} = \sqrt{a_{kk}}$.
STEP 3: For $i = k + 1 : n$ do step 4
STEP 4: If $a_{ik} \neq 0,\ a_{ik} = \frac{a_{ik}}{a_{kk}}$, end
STEP 5: For $j = k + 1 : n$, do steps 6-7
STEP 6: For $i = j : n$, do step 7.
STEP 7: If $a_{ij} \neq 0,\ a_{ij} = a_{ij} - a_{ik}a_{jk}$, end.
STEP 8: Stop.
```

Note that the matrix A and its incomplete Cholesky factor L would be stored in an appropriate data structure and the looping in Algorithm 4.1 would take on a very special appearance. This preconditioner would relinquish whatever advantage sparseness may have offered. For more detail, see [22].

We are ready to present our new technique in Algorithm 4.2.

Algorithm 4.2 (CG Predictor Corrector Method)**.**

```
INPUT: $u_0 \in \Re^{N+1}$    % such that $H(u_0) \approx 0$; initial point,
        $t \in \Re^{N+1}$       % initial approximation to $t(H'(u_0)) \approx 0$,
        $h > 0$          % initial steplength.
OUTPUT: $u_i,\ i = 0, 1, 2, \ldots$   % approximating the solution curve.
STEP 1: For $i = 1, 2, \ldots$ do steps 2-14.
STEP 2: Set $v = u_{i-1} + ht$      % predictor step.
STEP 3: Calculate $LL^* \approx H'(v)H'(v)^*$    % preconditioner.
STEP 4: Set $g_v = H'(v)^*(LL^*)^{-1}H(v),\ d = g_v$,  % gradients.
STEP 5: Do steps 6-11 until convergence       % corrector loop.
STEP 6: Set $\overline{\rho} = \arg\ \min_{\rho \geq 0} \left\| L^{-1}H(v - \rho d) \right\|^2$,
STEP 7: Set $w = v - \overline{\rho}d$      % corrector step, nonlinear CG.
STEP 8: Set $g_w = H'(w)^*(LL^*)^{-1}H(w)$;    % new gradient.
STEP 9: Set $\gamma = \frac{(g_w - g_v)^* g_w}{\|g_v\|^2}$;
STEP 10: Set $d = g_w + \gamma d$;   % new conjugate gradient,
STEP 11: Set $v = w$ and $g_v = g_w$.
STEP 12: adapt steplength $h > 0$.
STEP 13: Set $t = \frac{u_{i-1} - w}{\|u_{i-1} - w\|}$;    % approximation to $t(H'(w))$.
STEP 14: Set $u_i = w$;      % new point approximately on $H^{-1}(0)$.
STEP 15: Stop.
```

Now, we discuss the details of the method for solving the line search problem

$$\overline{\rho} = \min_{\rho \geq 0} \left\| L^{-1}H(v - \rho d) \right\|^2. \tag{5}$$

Let us approximate $\varphi(v - \rho d)$ via Taylor's formula of order two by

$$\varphi(v - \rho d) \approx \varphi(v) - \rho\varphi'(v)d + \frac{1}{2}\rho^2 d^* \nabla\varphi'(v)d. \tag{6}$$

We approximate $\overline{\rho}$ by the minimum of the right-hand side of equation (6). This is minimized exactly when

$$\overline{\rho} = \frac{\varphi'(v)d}{d^*\nabla\varphi'(v)d} \tag{7}$$

provided $d^*\nabla\varphi'(v)d > 0$. Furthermore, for $\varphi(v - \rho d) = \frac{1}{2}\left\|L^{-1}H(v-\rho d)\right\|^2$, we have

$$\varphi'(v)d = H(v)^*(LL^*)^{-1}\, H'(v)d = g_v^* d \tag{8}$$

and

$$\begin{aligned} d^*\nabla\varphi'(v)d &= d^*H'(v)^*(LL^*)^{-1}\, H'(v)d + \bigcirc(\|H(v)\|\,\|d\|) \\ &\approx \left\|L^{-1}\, H'(v)d\right\|^2. \end{aligned} \tag{9}$$

Since the evaluation of $H'(v)d$ is costly for large scale problems, we will use an inexpensive approximation of it using the central difference formula

$$H'(v)d = \frac{H(v+\epsilon\frac{d}{\|d\|}) - H(v-\epsilon\frac{d}{\|d\|})}{2\epsilon}\,\|d\| + \bigcirc(\epsilon^2) \tag{10}$$

for an appropriate $\epsilon > 0$. If we choose $d = e_i = (\delta_{i1}, \delta_{i2}, \ldots, \delta_{i(N+1)})$, where

$$\delta_{ij} = \left\{\begin{array}{ll} 1, & j = i \\ 0, & j \neq i \end{array}\right\},$$

then

$$H'(v)e_i = H_{x_i}(v) = \frac{H(v+\epsilon e_i) - H(v-\epsilon e_i)}{2\epsilon} + \bigcirc(\epsilon^2).$$

Therefore,

$$H'(v) = \begin{bmatrix} H_{x_1}(v) \\ H_{x_2}(v) \\ \vdots \\ H_{x_{N+1}}(v) \end{bmatrix} \simeq \frac{1}{2\epsilon}\begin{bmatrix} H(v+\epsilon e_1) - H(v-\epsilon e_1) \\ H(v+\epsilon e_2) - H(v-\epsilon e_2) \\ \vdots \\ H(v+\epsilon e_{N+1}) - H(v-\epsilon e_{N+1}) \end{bmatrix}. \tag{11}$$

We will use (10) and (11) for computing $H'(v)d$ and $H'(w)$. For more detail on how to choose ϵ, see [1] and [25]. Since the evaluation of $H'(w)$ is very costly ($2N+2$ evaluations of H for large N), we hold it fixed in the corrector loop (i.e., in Steps 5-11 in Algorithm 4.2). Since the predictor corrector steps of a continuation method are performed in such a way that all generated points are close to the solution curve in $H^{-1}(0)$, the quadratic approximation considered in (7) will give good results in our situation. Thus, we substitute (9)-(11) in (7). For the same reason, the approximation of $H'(v)$ in (11) will also give good results and it will not have any effect on the results. In Step 12 of Algorithm 4.2, we use the adaptive steplength h which is described in Section 2.

As we saw before, $\nabla\varphi(u) = H'(u)^+H(u)$ coincides with the usual Newton direction which we used as the standard corrector in Section 2. This means that the corrector in Algorithm 4.2 will be a rapidly converging corrector such as the standard corrector in Section 2. Moreover, Algorithm 4.2 has been based upon the Euler predictor, which is of local order two. This leads to the fact that the sequence, which is generated by Algorithm 4.2, is quadratically convergent. Also, it has the upper bound

$$\|u_{i+1} - u_i\| \leq O(h^2).$$

For more detail about the last inequality, see [1].

The above discussion ensures that Algorithm 4.2 safely follows the solution curve faster than the traditional CG which has superlinear convergence. Also, the PCG method will be faster than the traditional CG method but it will not be quadratically convergent. For more detail, see [15]. We can see the difference between these three methods in the next section.

5. Numerical results

In this section, we present three numerical experiments. In each example, we make a comparison between the traditional conjugate gradient, the preconditioning conjugate gradient, and the conjugate gradient predictor corrector method (Algorithm 4.2). We compare them according to the computational time in seconds, absolute error, and the number of iterations. We did all of these calculations on a 2.00Ghz Pinetum 4 computer. The first example is an application in fluid dynamics. We will start with it to demonstrate the practicality of our technique in real life problems.

Example 5.1 (The problem of thermal ignition in a finite cylinder)**.** In this example we study the equations modelling thermal ignition in a finite cylinder with aspect ratio $\gamma = \frac{R}{L}$ where R and L denote the radius and length of the cylinder, respectively. Assume the cylinder has the origin as center and the z-axis as its rotational axis. Thus, the partial differential equation that represents this application takes the form

$$\begin{aligned} -\Delta u &= \delta\, e^{\frac{u}{1+\epsilon u}}, \\ u &= 0 \text{ on the boundary of the cylinder}, \\ \frac{\partial u}{\partial z} &= 0 \text{ on the plane } z = 0, \\ \frac{\partial u}{\partial r} &= 0 \text{ on the axis } r = 0, \end{aligned} \tag{12}$$

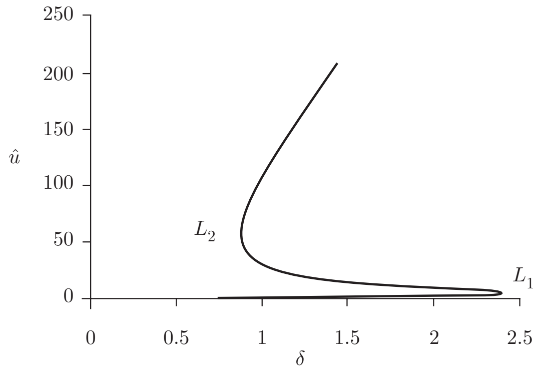

FIGURE 1. The variation of the peak temperature $\hat{u}$ with δ

where Δ is the Laplace operator in cylindrical coordinates, u is the nondimensional temperature, δ is the exothermicity, and ϵ is the inverse activation energy. Replacing the axial coordinate z by γz, we obtain

$$-\frac{1}{r}\frac{\partial}{\partial r}r\frac{\partial u}{\partial r} - \frac{1}{\gamma^2}\frac{\partial^2 u}{\partial z^2} = \delta\, e^{\frac{u}{1+\epsilon u}}.$$

The nonlinear thermal equation was discretized in a standard Galerkin finite-element method, using nine-noded quadrilateral elements with biquadratic interpolation. The details of the finite-element formulation are similar to those in the bifurcation studies of the Taylor and Benard problems (for more detail, see [2]) and will not be repeated here. As a result of the discretization, we get the large scale problem of the form $H(u, \delta) = 0$, where $H : \Re^{521} \to \Re^{520}$. Then we apply our technique which is described in Algorithm 4.2 on the discretization form. To show how our technique works efficiently, we compare it with the traditional conjugate gradient method and the preconditioning conjugate gradient method. As was discussed in Section 4, we use the incomplete Cholesky factorization as preconditioning for the nonlinear conjugate gradient.

For $\gamma = 4$, $\epsilon = 0.15$ and $R = 1$, a starting point $\delta = 0$ and $\overline{u} = 0$ is available on the solution curve for the discretized problem. First, we study the behavior of the solution, as represented by the peak temperature $\hat{u}$, as δ varies. The points of ignition and extinction are marked by L_1 and L_2, respectively, on the solution curve. From Figure 1, we note that the peak temperature $\hat{u}$ of the discretized

Table 1. The absolute errors in the temperature

(r, z)	$Error1$	$Error2$	$Error3$
$(0.2, 0.2)$	$1.1 * 10^{-4}$	$3.2 * 10^{-8}$	$2.0 * 10^{-13}$
$(0.4, 0.4)$	$9.3 * 10^{-4}$	$2.1 * 10^{-7}$	$2.0 * 10^{-13}$
$(0.8, 0.5)$	$6.5 * 10^{-4}$	$5.9 * 10^{-8}$	$2.9 * 10^{-13}$
$(1, 0.7)$	$9.9 * 10^{-4}$	$2.8 * 10^{-7}$	$8.4 * 10^{-12}$
$(1.4, 1)$	$8.9 * 10^{-4}$	$9.8 * 10^{-8}$	$6.5 * 10^{-12}$
$(1.9, 1.2)$	$4.7 * 10^{-4}$	$1.0 * 10^{-7}$	$7.2 * 10^{-13}$

Table 2. The total computational time in seconds

γ	N	$Time1$	$Time2$	$Time3$
1	200	163	80	43
2	308	201	97	52
4	520	277	123	70
6	642	355	146	80

Table 3. The number of iterations

γ	N	$Iter1$	$Iter2$	$Iter3$
1	200	128	60	32
2	308	164	82	42
4	520	230	110	61
6	642	301	142	77

solution when plotted against δ has a characteristic S-shaped curve whose two-fold points correspond to L_1 and L_2.

Next, we compare the traditional CG method, the PCG method and the CG predictor corrector method (Algorithm 4.2) when $\delta = 0.5$. Assume that $Error1$, $Error2$, and $Error3$ mean the absolute error at the given point using the traditional CG method, the PCG method and Algorithm 4.2, respectively. The results obtained are given in Table 1. It is clear that the error produced using Algorithm 4.2 is much smaller than that of the PCG method. Also, the error produced using the PCG method is smaller than that of the traditional CG method. In terms of computational time where $Time1$, $Time2$, and $Time3$ refer to the total computational time in seconds using the traditional CG method, the PCG method and Algorithm 4.2, respectively, the results are given in Table 2. Again it is clear that Algorithm 4.2 is superior to the traditional CG and PCG methods. Moreover, in terms of the number of iterations where $Iter1$, $Iter2$, and $Iter3$ refer to the number of iterations using the traditional CG method, the PCG method and Algorithm 4.2, respectively, the results are given in Table 3. Again it is clear that Algorithm 4.2 is superior to the traditional CG and PCG methods.

Example 5.2. Consider the two point boundary value problem

$$\begin{aligned} y'' + xe^{-x}yy' + (1+x^2)y &= (2+x+x^3)e^x,\ -1 < x < 1, \\ y(-1) &= e^{-1} \text{ and } y(1) = e. \end{aligned} \tag{13}$$

Using finite difference method to approximate the derivatives

$$f'(x) \simeq \frac{f(x+\epsilon) - f(x-\epsilon)}{2\epsilon} \quad \text{and} \quad f''(x) \simeq \frac{f(x+\epsilon) - 2f(x) + f(x-\epsilon)}{\epsilon^2},$$

we obtain a nonlinear system of the form $F(Y) = 0$ where $F : \Re^N \to \Re^N$. Simple calculations give us

$$F_i(Y) = y_{i+1} + \alpha_i y_{i+1} y_i + \beta_i y_i + \gamma_i y_i y_{i-1} + y_{i-1} - \delta_i,$$

where

$$\begin{aligned} \alpha_i &= \frac{\epsilon x_i e^{-x_i}}{2}, \qquad \beta_i = -2 + \epsilon^2(1 + x_i^2), \\ \gamma_i &= -\frac{\epsilon x_i e^{-x_i}}{2}, \qquad \delta_i = -\epsilon^2(2 + x_i + x_i^3)e^{x_i}, \\ Y &= (y_1, y_2, \dots, y_N), \quad \text{and} \quad x_i = -1 + \frac{2i}{N+1} \text{ for } i = 0 : N. \end{aligned}$$

Define the homotopy $H : \Re^N \times [0,1] \to \Re^N$ by $H(Y,\lambda) = \lambda F(Y) + (1-\lambda)G(Y)$, where G is the finite difference discertization of the linear boundary value problem

$$\begin{aligned} y'' &= (2+x+x^3)e^x,\ -1 < x < 1, \\ y(-1) &= e^{-1} \text{ and } y(1) = e. \end{aligned} \tag{14}$$

Assume that Y_0 is the solution of the linear system $G(Y) = 0$. Then a starting point $(Y_0, 0)$ is available on the solution curve for the large scale problem $H(Y,\lambda) = 0$.

Table 4. The maximum of the absolute errors

N	$Error1$	$Error2$	$Error3$
100	$3.3 * 10^{-5}$	$1.2 * 10^{-8}$	$2.1 * 10^{-14}$
150	$4.1 * 10^{-5}$	$4.4 * 10^{-8}$	$2.4 * 10^{-14}$
200	$6.4 * 10^{-5}$	$8.3 * 10^{-8}$	$2.7 * 10^{-14}$
250	$7.3 * 10^{-5}$	$8.8 * 10^{-8}$	$3.4 * 10^{-14}$
300	$8.5 * 10^{-5}$	$9.1 * 10^{-8}$	$3.8 * 10^{-14}$
350	$9.7 * 10^{-5}$	$9.4 * 10^{-8}$	$4.4 * 10^{-14}$

Table 5. The total computational time in seconds

N	$Time1$	$Time2$	$Time3$
100	155	77	29
150	191	89	36
200	226	111	63
250	290	139	77
300	333	162	84
350	378	180	91

Table 6. The number of iterations

N	$Iter1$	$Iter2$	$Iter3$
100	96	68	37
150	142	80	44
200	194	106	58
250	241	139	66
300	292	148	76
350	344	160	81

Then, we apply our technique described in Algorithm 4.2, the traditional CG, and the preconditioning conjugate gradient (PCG) method. The unique solution of problem (14) is $y(x) = (-24+19x-6x^2+x^3)e^x+(\frac{25}{e}-5e)x-(\frac{25}{e}+5e)$. Throughout this example we use the following notation.

$Error1 = Max\{|y(x_i) - y_i| : i = 0, N\}$ using the traditional nonlinear conjugate gradient.

$Error2 = Max\{|y(x_i) - y_i| : i = 0, N\}$ using the preconditioned nonlinear conjugate gradient.

$Error3 = Max\{|y(x_i) - y_i| : i = 0, N\}$ using Algorithm 4.2.

$Time1 =$ The total computational time in seconds using the traditional nonlinear conjugate gradient.

$Time2 =$ The total computational time in seconds using the preconditioned nonlinear conjugate gradient.

$Time3 =$ The total computational time in seconds using Algorithm 4.2.

$Iter1 =$ The number of iterations using the traditional nonlinear conjugate gradient.

$Iter2 =$ The number of iterations using the preconditioned nonlinear conjugate gradient.

$Iter3$ = The number of iterations using Algorithm 4.2
N = the size of the large scale problem.

The numerical results for this example are given in Tables 4, 5 and 6. Again it is clear that Algorithm 4.2 proves to be superior to the traditional CG and PCG methods.

Example 5.3. Consider the two point boundary value problem

$$\begin{aligned} y'' + (\sin x)(y')^2 + y &= 2e^x + e^{2x}\sin x,\ 0 < x < 2, \\ y(-0) &= 1 \text{ and } y(2) = e^2. \end{aligned} \tag{15}$$

Using the same procedure and notation as in Example 5.2, the numerical results are given in Tables 7, 8 and 9.

We want to end this section with the following remarks.

Remark 1. As a result of the discretization of the problems in Examples 5.1–5.3, the corresponding finite dimensional problem is $H : \Re^{n+1} \to \Re^n$ where n is large. This leads to the task of solving large scale continuation problems.

Remark 2. The matrices in Algorithm 4.2 are sparse matrices. If we rewrite $H(w) = BU$, then B is a band matrix with bandwidth 2 in Examples 5.2 and 5.3 and with

Table 7. The maximum of the absolute errors

N	$Error1$	$Error2$	$Error3$
100	$3.3 * 10^{-5}$	$5.1 * 10^{-9}$	$2.5 * 10^{-15}$
150	$4.9 * 10^{-5}$	$5.9 * 10^{-9}$	$2.9 * 10^{-15}$
200	$6.2 * 10^{-5}$	$6.5 * 10^{-9}$	$3.8 * 10^{-15}$
250	$7.1 * 10^{-5}$	$7.9 * 10^{-9}$	$4.2 * 10^{-15}$
300	$8.9 * 10^{-5}$	$9.1 * 10^{-9}$	$4.8 * 10^{-15}$
350	$9.9 * 10^{-5}$	$1.1 * 10^{-8}$	$5.3 * 10^{-15}$

Table 8. The total computational time in seconds

N	$Time1$	$Time2$	$Time3$
100	154	75	35
150	162	84	41
200	201	109	60
250	255	133	71
300	298	155	80
350	320	170	88

Table 9. The number of iterations

N	$Iter1$	$Iter2$	$Iter3$
100	77	41	20
150	105	55	28
200	133	72	35
250	184	91	45
300	220	119	59
350	297	148	71

bandwidth 4 in Example 5.1. We use the benefit of the bandness when we store these matrices and when we multiply them by each other or by another vector.

Remark 3. The computational time in seconds for Algorithm 4.2 is less than that of the preconditioned conjugate gradient method (PCG method). For example, in Example 5.3, when $n = 300$, the computational time for Algorithm 4.2 is 80 seconds while for the PCG methods it is 155 seconds.

Remark 4. The absolute error, as defined in Examples 5.1–5.3, in Algorithm 4.2 is less than for the PCG method. For example, in Example 5.3, when $n = 300$, the absolute error for Algorithm 4.2 is $4.8 * 10^{-15}$ while for PCG methods it is $9.1 * 10^{-9}$.

Remark 5. The number of iterations, as defined in Examples 5.1–5.3, in Algorithm 4.2 is less than for the PCG method. For example, in Example 5.2, when $n = 300$, the number of iterations for Algorithm 4.2 is 76 while for the PCG methods it is 148.

Remark 6. From Tables 1–9, we see that the traditional CG method gives the worst results while Algorithm 4.2 gives the best.

From the above discussion, we see that Algorithm 4.2 works efficiently and it is better than the traditional CG method and the preconditioned conjugate gradient method. Finally, we give Figure 1 to let the reader see how Algorithm 4.2 takes care for the fold points, which is one of the serious problems in the continuation methods. Moreover, this figure coincides with the real life practicality of the problem. For more detail, see [37].

Acknowledgments

We want to express our thanks to the referees for their valuable comments.

References

[1] E. L. Allgower and K. Georg, Numerical path following. In P. G. Ciarlet and J. L. Lions, editors, Handbook of Numerical Analysis, volume 5, North-Holland (1997). MR98i:65001

[2] E. L. Allgower, C.-S. Chien, K. Georg, C.-F. Wang, Conjugate gradient methods for continuation problems. J. Comput. Appl. Math. 38, (1991), 1-16. MR92j:65161

[3] R. E. Bank and T. F. Chan, PLTMGC: A multi-grid continuation program for parameterized nonlinear elliptic systems, SIAM J. Sci. Statis. Comput., 7 (1986), 540-559. MR87d:65125

[4] S. Bernstein, Sur la generalisation du probleme de Dirichlet., Math. Ann., 69 (1910), 82-136.

[5] T. F. Chan and H. B. Keller, Arc-length continuation and multi-grid techniques for nonlinear eigenvalue problems, SIAM J. Sci. Statis. Comput., 3 (1982), 173-194. MR83d:65152

[6] S. N. Chow and J.K. Hale, Methods of bifurcation theory, Springer Verlag, New York, 1982. MR84e:58019

[7] K. A. Cliffe and A. Spence, in Numerical methods for bifurcation problems (T. Kupper, H. D. Mittleman, and H. Weber, Eds.), ISNM, Birkhauser, 1984. MR86j:65008

[8] A. I. Cohen, Rate of convergence of several congugate gradient algorithms, SIAM J. Numer. Anal., 9 (1972), 248-259. MR47:1284

[9] R. Fletcher, Practical methods of optimization, J. Wiley, New York, second ed., 1987. MR89j:65050

[10] C.B. Garci and W. I. Zangwill, Pathways to solutions, fixed points, and equilibria, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[11] R. Glowinski, H.B. Keller, and L. Reinhart, Continuation conjugate gradient methods for the least squares solution of nonlinear boundary value problems, SIAM J. Sci. Statist. Comput., 6 (1985), 793-832. MR86j:65156

[12] G. H. Golub and C. F. Van Loan, Matrix computations, J. Hopkins University Press, London, Second ed., 1989. MR90d:65055

[13] F.J. Gould and J.W. Tolle, Complementary pivoting on a pseudomanifold structure with applications on the decision sciences, Vol. 2 of Sigma Series in Applied Mathematics, Heldermann Verlag, Berlin, 1983. MR85h:90126

[14] K. Georg, Matrix-Free Numerical Continuation and Bifurcation, Numerical Functional Analysis and Optimization, vol. 22 (2001), 303-320. MR2002j:65127

[15] P. Gill, W. Murray and M. Wright, Practical optimization, Academic Press, London, 1981. MR83d:65195

[16] M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, J. Res. Nat. Bur. Standards, 49 (1952), 409-436. MR15:651a

[17] J. P. Keener and H. B. Keller, Perturbed bifurcation theory, Arch. Rational Mech. Anal., 50 (1974), 159-175. MR49:1253

[18] H. B. Keller, Lectures on numerical methods in bifurcation problems, Springer Verlag, Berlin, Heidelberg, New York, 1987. MR89f:58031

[19] H. B. Keller, Nonlinear bifurcation, J. Diff. Equ., 7 (1970), 417- 434. MR41:8851

[20] H. B. Keller, Numerical solution of bifurcation and nonlinear eigenvalue problems, Applications of Bifurcation Theory, P.H. Rabinowitz, ed., New York, London, 1977, Academic Press, 359-384. MR56:13592

[21] F. Klein, Neue beitrage zur Riemannschen Funktionentheorie, Math. Ann., 21(1882-1883).

[22] D. S. Kershaw, The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations, J. Comput. Phys. 26 I (Jan.) (1978), 43-65. MR58:8190

[23] E. Lahaye, Une methode de resolution d'une categorie d'equations transcendantes, C. R. Acad. Sci. Paris, 198 (1934), 1840-1842.

[24] J. Leray and J. Schauder, Topologie et equations fonctioneles, Ann. Sci. Ecole Norm. Sup., 51(1934), 45-78.

[25] W. Mackens, Numerical differentiation of implicitly defined space curves, Computing, 41 (1989), 237-260. MR90b:65030

[26] J. E. Marsden and A. J. Tromba, Vector calculus, Third edition, W. H. Freeman and Company, New Your, 1988, 75.

[27] G. P. McCormick and K. Ritter, Alternate proofs of the convergence properties of the conjugate-gradient method, J. Optim. Theory Appl., 13(1974), 497-518.

[28] H. D. Mittelmann, D. Roose, eds., Continuation Techniques and Bifurcation Problems, Vol. 92 of ISNM, Birkhauser, 1990. MR90m:65005

[29] H. Poincare, Sur les courbes define par une equation differentielle, I-IV, Oeuvres I. Gauthier-Villars, Paris, 1881-1886.

[30] E. Polak and G. Ribiere, Note sur la convergence de methodes de directions conjugees, Rev. Francaise Informate. Recherche Operatonelle, 3(1969), 35-43. MR40:8232

[31] W. C. Rheinboldt, Numerical analysis of continuation methods for nonlinear structural problems, Comp. and Structures, 13(1981), 103-113. MR82i:73043

[32] R. Seydel, From Equilibrium to Chaos, Practical Bifurcation and Stability Analysis, Elsevier, New York, 1988. MR89e:58084

[33] H.I. Siyyam and M.I. Syam, The modified trapezoidal rule for line integrals, J. Comput. Appl. Math. 84 (1997), 1-14.

[34] M. I. Syam and H. I. Siyyam, Numerical differentiation of implicitly defined curves, J. of Compu. Appl. Math., 108(1999), 131-144. MR2000d:65040

[35] M. I. Syam, Interpolation predictors over implicitly defined curves, Computers & Mathematics with Applications, 44(2002), 1067-1076. MR2003j:65053

[36] M. J. Todd, The computation of fixed points and applications, vol. 124 of Lecture Notes in Economics and Mathematical Systems, Springer Verlag, Berlin, Heidelberg, New York, 1976. MR53:14478

[37] K. H. Winters, K. A. Cliffe, The prediction of critical points for thermal explosions in a finite volume, Combustion and Flame, 62 (1985), 13-20.

Department of Mathematics and Computer Science, United Arab Emirates University, Al-Ain, United Arab Emirates

E-mail address: m.syam@uaeu.ac.ae