Mathematical Sciences
a SpringerOpen Journal

ORIGINAL RESEARCH Open Access

# Compact alternating group explicit method for the cubic spline solution of two point boundary value problems with significant nonlinear first derivative terms

Ranjan K Mohanty* and Jyoti Talwar

**Abstract**

In this paper, we report the application of two parameter coupled alternating group explicit (CAGE) iteration and Newton-CAGE iteration methods for the cubic spline solution of non-linear differential equation $u'' = f(r,u,u')$ subject to given natural boundary conditions. The error analysis for CAGE iteration method is discussed in details. We compared the results of proposed CAGE iteration method with the results of corresponding two parameter alternating group explicit (TAGE) iteration method to demonstrate computationally the efficiency of the proposed method.

**Keywords:** Two point boundary value problems, Nonlinear first derivative terms, Cubic spline approximations, CAGE method, Newton-CAGE method, Burgers' equation

**AMS(2010) Classification, 65 L10, 65 L12**

## Introduction

Consider the two point boundary value problem

$$L[u(r)] \equiv -u''(r) + f(r,u,u') = 0, 0 < r < 1 \tag{1}$$

with natural boundary conditions

$$u(0) = A, u(1) = B \tag{2}$$

where $A$ and $B$ are constants. We assume that for $0 < r < 1$ and $-\infty < u,v < \infty$

(i) $f(r,u,v)$ is continuous,
(ii) $\frac{\partial f}{\partial u}$ and $\frac{\partial f}{\partial v}$ exist and are continuous, and
(iii) $\frac{\partial f}{\partial u} > 0$ and $\left|\frac{\partial f}{\partial v}\right| \leq W$ for some positive constant $W$.

These conditions assure that the boundary value problem (1)-(2) has a unique solution (see Keller [1]).

During last four decades, there has been a growing interest in the theory of splines and their applications (see [2-6]). Bickley [7], Albasiny and Hoskins [8,9], and Fyfe [10] have demonstrated the use of cubic spline function for obtaining the second order approximation solution for two point boundary value problems. Later, Chawla and Subramanian [11] have constructed a fourth order cubic spline method for second-order mildly nonlinear two point boundary value problems. In 1983, Jain and Aziz [12] have first developed fourth order accurate numerical method based on cubic spline approximation for the solution of more general non-linear two point boundary value problems. In the recent past, many authors (see [13-19]) have suggested various numerical methods based on cubic spline approximations for the solution of linear singular two point boundary value problems.

In 1985 Evans [20] developed group explicit method for solving large linear systems arising due to the discretization of differential equations. Later, Sukon and Evans [21] have introduced two parameter alternating group explicit (TAGE) iterative methods for the solution of tri-diagonal linear system of equations. Using the technique given in [20,21], Mohanty *et al.*

* Correspondence: rmohanty@maths.du.ac.in
Department of Mathematics, Faculty of Mathematical Sciences, University of Delhi, Delhi 110 007, India

[22-24] have discussed the application of TAGE iterative method to fourth order accurate cubic spline approximation for the solution of non-linear singular two point boundary problems. In this paper, we discuss two parameter coupled alternating group explicit (CAGE) and Newton-CAGE iteration methods, and fourth order cubic spline finite difference approximation and their application to linear and nonlinear differential equations with singular coefficients. In the next section, we discuss cubic spline approximation and its application to singular problems. In section 3, we discuss the CAGE and Newton-CAGE iteration methods and convergence analysis. In section 4, we compare the computational results obtained by using the proposed CAGE iterative method with the corresponding TAGE iterative method. Concluding remarks are given in section 5.

## Cubic spline approximation and application

To obtain a cubic spline solution of the boundary value problem (1) and (2), we choose uniform mesh spacing $h > 0$ along the $r$-direction. The interval [0, 1] is divided into a set of points with interval of $h = 1/(N + 1)$, $N$ being a positive integer. The cubic spline approximation to equation (1) is obtained on [0,1] which consists of the central point $r_k = kh$ and the two neighboring points $r_{k+1} = r_k + h$ and $r_{k-1} = r_k - h, k = 1(1)N$, where $r_0 = 0$ and $r_{N+1} = 1$. Let $U_k = u(r_k)$ be the exact solution of $u$ at the grid point $r_k$ and is approximated by $u_k$.

At each internal mesh point $r_k$, we denote:

$$M_k = u''(r_k) = f(r_k, u(r_k), u'(r_k)), k = 0(1)N + 1.$$

Given the values $u_0, u_1, \ldots, u_{N+1}$ of the function $u(r)$ at the mesh points $r_0, r_1, \ldots, r_{N+1}$ and the values of the second derivatives of $u$ at the end points $u''_0$ and $u''_{N+1}$, there exists a unique interpolating cubic spline function $S(r)$ with the following properties:

(i) S(r) coincides with a polynomial of degree three on each $[r_{k-1}, r_k], k = 1(1)N + 1$
(ii) $S(r) \in C^2[0, 1]$ and
(iii) $S(r_k) = u_k, k = 0(1)N + 1$

The interpolating cubic spline polynomial may be written as:

$$S(r) = \frac{(r_k - r)^3}{6h} M_{k-1} + \frac{(r - r_{k-1})^3}{6h} M_k + \left(u_{k-1} - \frac{h^2}{6} M_{k-1}\right) \frac{(r_k - r)}{h} + \left(u_k - \frac{h^2}{6} M_k\right) \frac{(r - r_{k-1})}{h}, r_{k-1} \le r \le r_k, k = 1(1)N + 1 \quad (3)$$

We consider the following approximations:

$$r_{k\pm\eta} = r_k \pm \eta h, 0 < \eta \le 1, \quad (4.1)$$

$$\bar{m}_k = \bar{u}'_k = \frac{(u_{k+1} - u_{k-1})}{2h}, \quad (4.2)$$

$$\bar{m}_{k\pm1} = \frac{(\pm 3u_{k\pm1} \mp 4u_k \pm u_{k\mp1})}{2h}, \quad (4.3)$$

$$\bar{f}_k = f(r_k, u_k, \bar{m}_k), \quad (4.4)$$

$$\bar{f}_{k\pm1} = f(r_{k\pm1}, u_{k\pm1}, \bar{m}_{k\pm1}), \quad (4.5)$$

$$\bar{\bar{u}}_{k\pm\eta} = \eta u_{k\pm1} + (1 - \eta)u_k + h^2\left(p\bar{f}_{k\pm1} + q\bar{f}_k\right), \quad (4.6)$$

$$\bar{\bar{m}}_{k\pm\eta} = \pm\frac{1}{h}(u_{k\pm1} - u_k) \pm h\left(p^*\bar{f}_{k\pm1} + q^*\bar{f}_k\right), \quad (4.7)$$

$$\hat{m}_k = \bar{m}_k - \frac{h}{12}\left(\bar{f}_{k+1} - \bar{f}_{k-1}\right), \quad (4.8)$$

$$\bar{\bar{f}}_{k\pm\eta} = f\left(r_{k\pm\eta}, \bar{\bar{u}}_{k\pm\eta}, \bar{m}_{k\pm\eta}\right), \quad (4.9)$$

$$\hat{f}_k = f(r_k, u_k, \hat{m}_k), \quad (4.10)$$

where $p = \frac{\eta(\eta^2-1)}{6}, p^* = \frac{dp}{d\eta} = \frac{1}{2}\left(\eta^2 - \frac{1}{3}\right)$,

$$q = \frac{(1-\eta)\left[(1-\eta)^2 - 1\right]}{6}, \; q^* = \frac{dq}{d\eta} = \frac{1}{2}\left[\frac{1}{3} - (1-\eta)^2\right]$$

Then the cubic spline method with order of accuracy four for the differential equation (1) may be written as:

$$U_{k+1} - 2U_k + U_{k-1} = \frac{h^2}{12\eta^2}\left[\bar{\bar{f}}_{k+\eta} + \bar{\bar{f}}_{k-\eta} + \left(12\eta^2 - 2\right)\hat{f}_k\right] + T_k, 0 < \eta \le 1, k = 1(1)N \quad (5)$$

where $T_k = O(h^6)$ (See Jain and Aziz [12]) with $u_0 = A$ and $u_{N+1} = B$.

Let us discuss the application of the difference formula (5) to the following singular problems

$$u'' = D(r)u' + E(r)u + f(r), 0 < r < 1 \quad (6)$$

and

$$\nu u'' = B(r)u' + uu' + C(r)u + g(r), 0 < r < 1 \quad (7)$$

where $\nu = R^{-1} > 0$ is a constant and $D(r) = -\alpha/r$ and $E(r) = \alpha/r^2, B(r) = -\alpha\nu/r$ and $E(r) = \alpha\nu/r^2$.

For $\alpha = 1$ and 2, the linear singular equation (6) becomes cylindrical and spherical problems, respectively, and for $\alpha = 0$, 1 and 2, the non-linear singular problem (7) represents steady-state Burger's equation in Cartesian, cylindrical and spherical coordinates respectively.

Now applying the difference formula (5) to the singular equations (6) and (7) and using the technique discussed by Mohanty *et al.* [22], we may obtain the following fourth order difference scheme

$$a_k u_{k-1} + 2b_k u_k + c_k u_{k+1} = d_k, 0 < \eta \le 1, k = 1(1)N, \tag{8}$$

for the numerical solution of the differential equation (6), where

$$a_k = -1 + \frac{\alpha}{12\eta}\left(\frac{6\eta}{k} + \frac{1}{2k^3}(6\eta + \alpha\eta - 2\alpha) + \frac{(2-\alpha\eta)}{k^2}\right),$$

$$b_k = 1 + \frac{\alpha}{12\eta}\left(\frac{1}{k^2}(6\eta + \alpha\eta - 2) + \frac{1}{2k^4}(6\eta + \alpha\eta - 2\alpha)\right),$$

$$c_k = -1 + \frac{\alpha}{12\eta}\left(\frac{-6\eta}{k} - \frac{1}{2k^3}(6\eta + \alpha\eta - 2\alpha) + \frac{(2-\alpha\eta)}{k^2}\right),$$

$$d_k = \frac{-h^2}{12\eta}\left[\eta\left(12f_k + \frac{h\alpha}{k}f'_k + h^2 f''_k\right) + \frac{\alpha}{k^2}f_k(3\eta^2 - 2\eta)\right].$$

and the following fourth order difference scheme

$$\begin{aligned}
\phi(u_{k-1}, u_k, u_{k+1}) \equiv & -\nu[u_{k+1} - 2u_k + u_{k-1}] \\
& + \frac{h^2}{12}[I_1 u_k + I_2(u_{k+1} - u_{k-1}) + I_3(u_{k+1} - 2u_k + u_{k-1}) \\
& + I_4 u_k^2 + I_5 u_k(u_{k+1} - u_{k-1}) + I_6 u_k(u_{k+1} - 2u_k + u_{k-1}) \\
& + I_7(u_{k+1}^2 - u_{k-1}^2) + I_8(u_{k+1}^2 - u_{k-1}^2)(u_{k+1} - u_{k-1}) \\
& + I_9 u_k^2(u_{k+1} - 2u_k + u_{k-1}) + I_{10} u_k (u_{k+1} - u_{k-1})^2 \\
& + \Sigma f] = 0, k = 1(1)N,
\end{aligned} \tag{9}$$

for the numerical solution of the differential equation (7), where

$$I_1 = \frac{12\alpha\nu}{(r_k)^2} + \frac{\alpha\nu h^2(6-\alpha)}{(r_k)^4} - \nu^{-1}h^2 f'_k,$$

$$I_2 = \frac{-6\alpha\nu}{hr_k} + \frac{\alpha\nu h(\alpha - 6)}{2(r_k)^3} - \frac{\nu^{-1}h}{2}f_k,$$

$$I_3 = \frac{\alpha\nu(2-\alpha)}{(r_k)^2}, I_4 = \frac{2\alpha h^2}{(r_k)^3}, I_5 = \frac{4}{h} - \frac{2\alpha h}{3(r_k)^2}, I_6 = \frac{2\alpha}{(r_k)},$$

$$I_7 = \frac{1}{h} + \frac{\alpha h}{3(r_k)^2}, I_8 = \frac{\nu^{-1}}{6}, I_9 = -\nu^{-1}, I_{10} = -\frac{\nu^{-1}}{3},$$

and

$$\Sigma f = 12f_k + h^2 f''_k + \frac{\alpha h^2}{(x_k)^2}(f_k + x_k f'_k).$$

In order to avoid the numerical complexity, we consider $\eta = 1$.

If the differential equation is linear, we can apply the two parameter CAGE iterative method and in the non-linear case, we can use the Newton-CAGE iterative method to obtain the solution.

## CAGE Algorithm and convergence analysis

The linear system (8) in matrix form may be written as:

$$\boldsymbol{Au} = \mathbf{RH} \tag{10}$$

where

$$A = \begin{bmatrix} 2b_1 & c_1 & & & 0 \\ a_2 & 2b_2 & c_2 & & \\ & & \ddots & & \\ & & a_{N-1} & 2b_{N-1} & c_{N-1} \\ 0 & & & a_N & 2b_N \end{bmatrix}_{N\times N}, \quad u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ \vdots \\ u_N \end{bmatrix}_{N\times 1}$$

$$\text{and } RH = \begin{bmatrix} \sum f_1 - a_1 u_0 \\ \sum f_2 \\ \vdots \\ \vdots \\ \sum f_N - c_N u_{N+1} \end{bmatrix}_{N\times 1} = \begin{bmatrix} RH_1 \\ RH_2 \\ \vdots \\ \vdots \\ RH_N \end{bmatrix}_{N\times 1} \quad \text{(say)}.$$

To implement the CAGE iterative method, we split the coefficient matrix $\boldsymbol{A}$ into two sub-matrices $A = G_1 + G_2$, where $G_1$ and $G_2$ satisfy the conditions:

(i) $G_1+\omega_1 I$ and $G_2+\omega_2 I$ are non-singular for any $\omega_1 > 0$ and $\omega_2 > 0$.
(ii) For any vectors $\nu_1$ and $\nu_2$ and $\omega_1 > 0$ , $\omega_2 > 0$, it is 'convenient' to solve the system explicitly, *i.e.* $z_1 = (G_1 + \omega_1 I)^{-1}\nu_1$ and $z_2 = (G_2 + \omega_2 I)^{-1}\nu_2$ for vectors $z_1$ and $z_2$, respectively.

We shall be concerned here with the situation where $G_1$ and $G_2$ are small $(2 \times 2)$ block systems.

Now we discuss the case when N is odd (with $x_0 = 0$, $x_{N+1} = 1$).

Let

$$G_1 = \begin{bmatrix} b_1 & & & & & 0 \\ & b_2 & c_2 & & & \\ & a_3 & b_3 & & & \\ & & & \ddots & & \\ 0 & & & & b_{N-1} & c_{N-1} \\ & & & & a_N & b_N \end{bmatrix}_{N\times N},$$

and

$$G_2 = \begin{bmatrix} b_1 & c_1 & & & & 0 \\ a_2 & b_2 & & & & \\ & & \ddots & & & \\ & & & b_{N-1} & c_{N-1} & \\ & & & a_N & b_N & \\ 0 & & & & & b_N \end{bmatrix}_{N\times N}$$

So that the system (10) can be re-written as

$$(G_1 + G_2)u = RH \tag{11}$$

Then a two parameter AGE method for solving the above system may be written as

$$(G_1 + \omega_1 I)z^{(s)} = RH - (G_2 - \omega_1 I)u^{(s)}, s = 0, 1, 2, \ldots \tag{12}$$

$$(G_2 + \omega_2 I)u^{(s+1)} = RH - (G_1 - \omega_2 I)z^{(s)}, s = 0, 1, 2, \ldots \tag{13}$$

where $z^{(s)}$ is an intermediate vector.

Eliminating $z^{(s)}$ and combining equations (12) and (13), we obtain the iterative method

$$\begin{aligned}(G_2 + \omega_2 I)u^{(s+1)} = &\left[I - (\omega_1 + \omega_2)(G_1 + \omega_1 I)^{-1}\right] \\ &\times (G_2 - \omega_1 I)u^{(s)} + (\omega_1 + \omega_2) \\ &\times (G_1 + \omega_1 I)^{-1} RH, s = 0, 1, 2, \ldots\end{aligned} \tag{14}$$

or

$$u^{(s+1)} = T_w u^{(s)} + RH_w, s = 0, 1, 2, \ldots \tag{15}$$

where

$$\begin{aligned}T_w = &(G_2 + \omega_2 I)^{-1}[(G_2 - \omega_1 I) - (\omega_1 + \omega_2) \\ &\times (G_1 + \omega_1 I)^{-1}(G_2 - \omega_1 I)]\end{aligned}$$

and

$$RH_w = (\omega_1 + \omega_2)(G_2 + \omega_2 I)^{-1}(G_1 + \omega_1 I)^{-1} RH$$

The new iterative method (14) or (15) is called the two parameter CAGE iterative method and the matrix $\boldsymbol{T_w}$ is called the CAGE iteration matrix.

To prove the convergence of the method, we need to prove that $S(\boldsymbol{T}_w) \le 1$, where $S(\boldsymbol{T}_w)$ denotes the spectral radius of $\boldsymbol{T_w}$.

Let $\lambda_i$ and $\mu_i$, $i = 1(1)N$, be the eigen values of $G_1$ and $G_2$, respectively.

Since $(G_2 + \omega_2 I)^{-1}$, $(G_2 - \omega_1 I)$, and $(G_1 + \omega_1 I)^{-1}$, $(G_2 - \omega_1 I)$, commute with each other

$$\begin{aligned}\|T_w\|_2 &= \max_{\lambda_i, \mu_i}\left|\frac{\mu_i - \omega_1}{\mu_i + \omega_2} - \frac{(\omega_1 + \omega_2)(\mu_i - \omega_1)}{(\mu_i + \omega_2)(\lambda_i + \omega_1)}\right| \\ &= \max_{\lambda_i}\left|\frac{(\lambda_i - \omega_2)}{(\lambda_i + \omega_1)}\right| \max_{\mu_i}\left|\frac{\mu_i - \omega_1}{\mu_i + \omega_2}\right|\end{aligned} \tag{16}$$

The eigen values $\lambda_i$ of sub-matrices $\boldsymbol{G}_1$, satisfy the equation

$$\begin{vmatrix} b_i - \lambda_i & c_i \\ a_{i+1} & b_{i+1} - \lambda_i \end{vmatrix} = 0 \tag{17}$$

or

$$\lambda_i^2 - (b_i + b_{i+1})\lambda_i + (b_i b_{i+1} - a_{i+1}c_i) = 0 \tag{18}$$

Simplifying, we get

$$\lambda_i = \frac{1}{2}\left[(b_i + b_{i+1}) \pm \sqrt{(b_i - b_{i+1})^2 + 4a_{i+1}c_i}\right] \tag{19}$$

It is easy to verify that $\mathrm{Re}(\lambda_i) > 0$, hence

$$\left|\frac{\lambda_i - \omega_2}{\lambda_i + \omega_1}\right| < 1, i = (1)N \text{ for all } \omega_1 > 0, \omega_2 > 0 \tag{20}$$

Similarly, the eigenvalues $\mu_i$ of the sub-matrices of $G_2$ satisfy $\mathrm{Re}(\mu_i) > 0$, hence

$$\left|\frac{\mu_i - \omega_1}{\mu_i + \omega_2}\right| < 1, i = 1(1)N \text{ for all } \omega_1 > 0, \omega_2 > 0 \tag{21}$$

Therefore from the equation (16), we conclude that

$$S(T_w) = \|T_w\|_2 < 1 \tag{22}$$

Hence, the convergence of the CAGE method (15) follows.

Now we discuss the CAGE algorithm, when $N$ is odd.

For simplicity we denote:

$$p_k = b_k + \omega_1, q_k = b_k - \omega_1, r_k = b_k + \omega_2 \text{ for } k = 1(1)N$$

and for $(p_k p_{k+1} - c_k a_{k+1}) \neq 0$, we define $d_k = 1/(p_k p_{k+1} - c_k a_{k+1})$ for $k = 1(1)N - 1$.

By carrying out the necessary algebra in equation (14), we obtain the following CAGE parallel algorithm:

Let

$$\Delta_1 = r_1 r_2 - c_1 a_2 \neq 0,$$

$$S_1 = q_1 u_1^{(s)} + c_1 u_2^{(s)} - \frac{2\omega}{p_1}\left[q_1 u_1^{(s)} + c_1 u_2^{(s)} - RH_1\right],$$

$$\begin{aligned}S_2 = a_2 u_1^{(s)} + q_2 u_2^{(s)} - 2\omega d_2 &\left[a_2 p_3 u_1^{(s)} + p_3 q_2 u_2^{(s)}\right. \\ &\left.- c_2 q_3 u_3^{(s)} - c_2 c_3 u_4^{(s)} - p_3 RH_2 + c_2 RH_3\right]\end{aligned}$$

then

$$u_1^{(s+1)} = \frac{(S_1 r_2 - S_2 c_1)}{\Delta_1}, s = 0, 1, 2, \ldots \tag{23}$$

$$u_2^{(s+1)} = \frac{(S_2 r_1 - S_1 a_2)}{\Delta_1}, s = 0, 1, 2, \ldots \tag{24}$$

For $k = 3(2)\ N\text{-}2$

$$\Delta = r_k r_{k+1} - c_k a_{k+1} \neq 0,$$

**Table 1 Problem 1: the RMS errors**

| N | TAGE method |  |  | CAGE method |  |  | RMS errors (for both TAGE and CAGE method) |
|---|---|---|---|---|---|---|---|
|  | $\omega_{1opt} = \omega_{2opt} = \omega_{opt}$ | iter | CPU time (in sec) | $\omega_{1opt} = \omega_{2opt} = \omega_{opt}$ | iter | CPU time (in sec) |  |
|  | $\beta=10$ |  |  |  |  |  |  |
| 10 | 0.725 | 24 | 0.0016 | 0.55 | 12 | 0.0003 | 0.1619(−03) |
| 20 | 0.41 | 48 | 0.0034 | 0.35 | 21 | 0.0008 | 0.1169(−04) |
| 30 | 0.28 | 70 | 0.0062 | 0.25 | 31 | 0.0015 | 0.2428(−05) |
| 40 | 0.21 | 100 | 0.0108 | 0.19 | 41 | 0.0025 | 0.7884(−06) |
| 60 | 0.15 | 150 | 0.0228 | 0.13 | 61 | 0.0054 | 0.1599(−06) |
| 80 | 0.11 | 200 | 0.0398 | 0.106 | 79 | 0.0092 | 0.5131(−07) |
|  | $\beta=100$ |  |  |  |  |  |  |
| 10 | 6.0 | 18 | 0.0014 | 4.72 | 09 | 0.00024 | 0.8820(−01) |
| 20 | 2.4 | 17 | 0.0019 | 2.37 | 06 | 0.00027 | 0.1977(−01) |
| 30 | 1.61 | 20 | 0.0024 | 1.60 | 06 | 0.00036 | 0.6125(−02) |
| 40 | 1.22 | 26 | 0.0035 | 1.19 | 07 | 0.00049 | 0.2331(−02) |
| 60 | 0.82 | 38 | 0.0065 | 0.83 | 08 | 0.00077 | 0.5187(−03) |
| 80 | 0.62 | 50 | 0.0105 | 0.59 | 11 | 0.00133 | 0.1684(−03) |

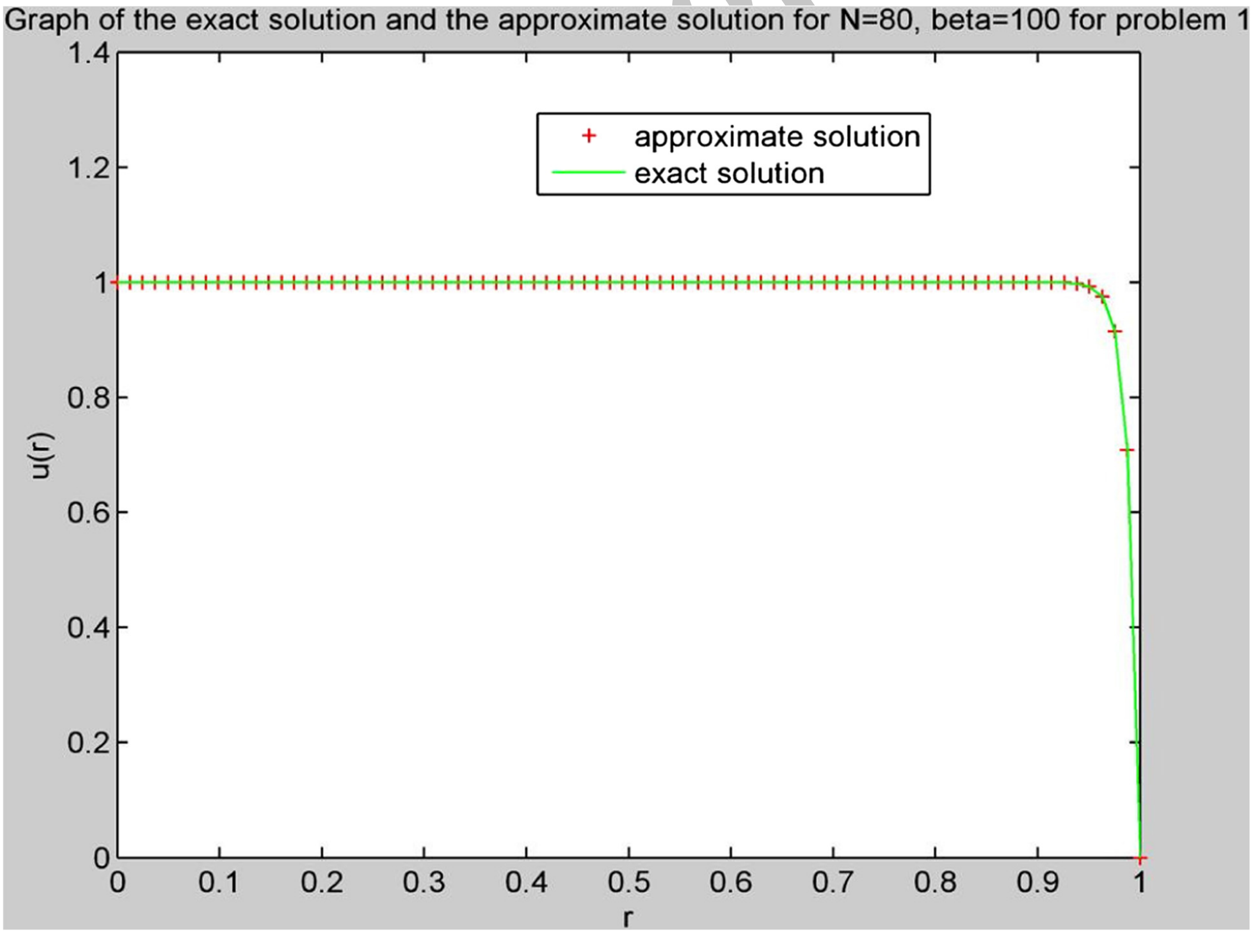

**Figure 1** Graph of the exact solution and the approximate solution for N = 80, beta = 100 for problem 1.

**Table 2 Problem 2: the RMS errors**

| N | TAGE method | | | | CAGE method | | | | RMS errors (for both TAGE and CAGE method) |
|---|---|---|---|---|---|---|---|---|---|
| | $\omega_{1opt}$ | $\omega_{2opt}$ | iter | CPU time (in sec) | $\omega_{1opt}$ | $\omega_{2opt}$ | iter | CPU time ( in sec) | |
| | $\alpha$=1 | | | | | | | | |
| 10 | 0.573 | 0.574 | 32 | 0.0017 | 0.510 | 0.530 | 19 | 0.0004 | 0.6666(−03) |
| 20 | 0.305 | 0.319 | 60 | 0.0038 | 0.390 | 0.290 | 37 | 0.0012 | 0.5173(−04) |
| 30 | 0.218 | 0.224 | 86 | 0.0070 | 0.185 | 0.225 | 53 | 0.0025 | 0.1093(−04) |
| 40 | 0.166 | 0.158 | 114 | 0.0128 | 0.167 | 0.166 | 67 | 0.0040 | 0.3575(−05) |
| 60 | 0.109 | 0.109 | 163 | 0.0234 | 0.120 | 0.113 | 95 | 0.0082 | 0.7282(−06) |
| 80 | 0.0478 | 0.0475 | 319 | 0.0600 | 0.091 | 0.086 | 124 | 0.0139 | 0.2331(−07) |
| | $\alpha$=2 | | | | | | | | |
| 10 | 0.660 | 0.678 | 29 | 0.0019 | 0.451 | 0.455 | 19 | 0.0004 | 0.7861(−03) |
| 20 | 0.350 | 0.369 | 53 | 0.0035 | 0.240 | 0.241 | 38 | 0.0013 | 0.6101(−04) |
| 30 | 0.242 | 0.250 | 77 | 0.0085 | 0.244 | 0.240 | 53 | 0.0025 | 0.1289(−04) |
| 40 | 0.186 | 0.187 | 100 | 0.0105 | 0.179 | 0.163 | 71 | 0.0042 | 0.4213(−05) |
| 60 | 0.126 | 0.126 | 144 | 0.0200 | 0.083 | 0.100 | 101 | 0.0086 | 0.8590(−06) |
| 80 | 0.087 | 0.087 | 200 | 0.0370 | 0.090 | 0.083 | 131 | 0.0146 | 0.2760(−06) |

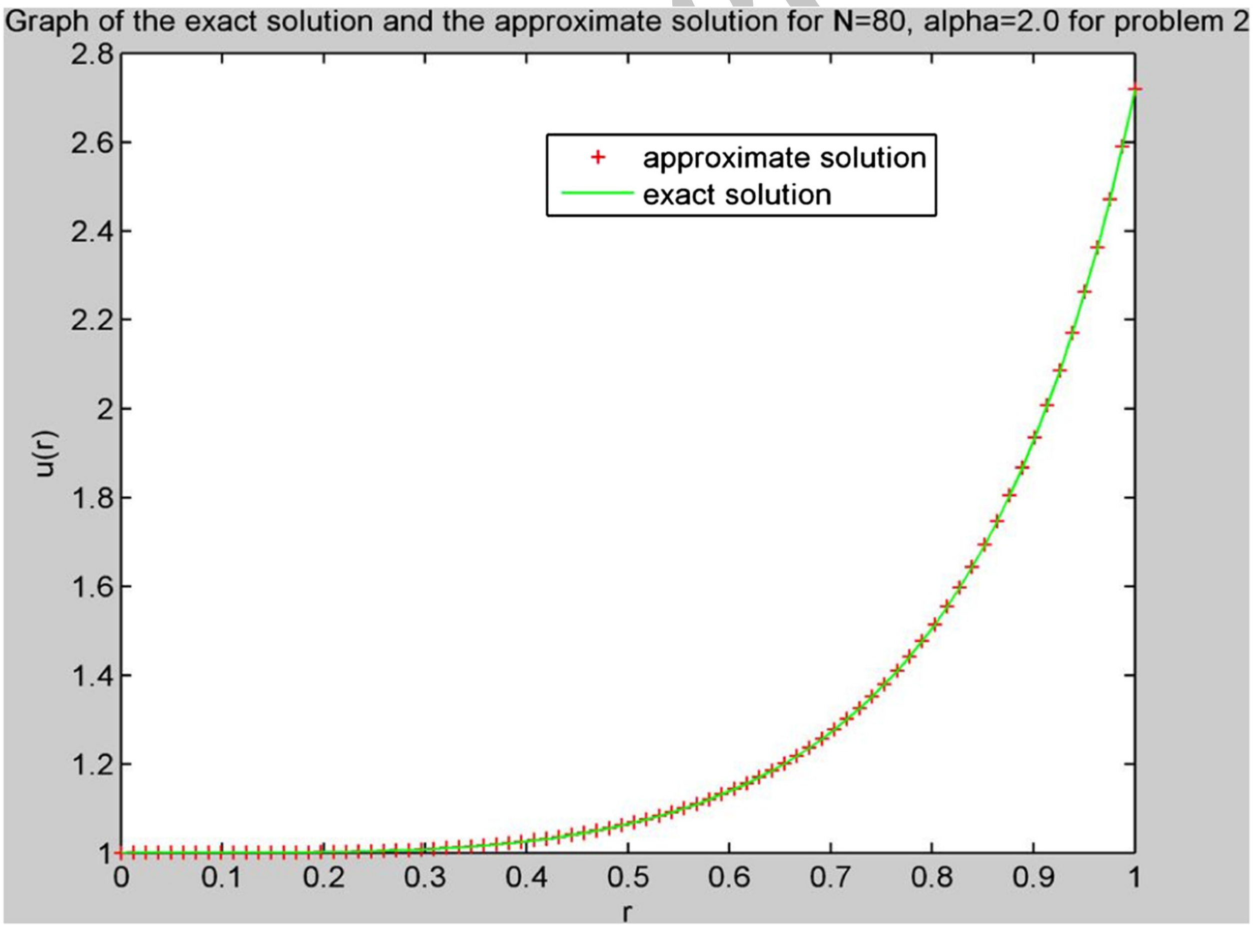

**Figure 2** Graph of the exact solution and the approximate solution for N = 80, alpha = 2.0 for problem 2.

$$S_3 = q_k u_k^{(s)} + c_k u_{k+1}^{(s)} - 2\omega d_{k-1}\Big[-a_{k-1}a_k u_{k-2}^{(s)} - a_k q_{k-1} u_{k-1}^{(s)} + p_{k-1} q_k u_k^{(s)} + p_{k-1} c_k u_{k+1}^{(s)} + a_k RH_{k-1} - p_{k-1} RH_k\Big],$$

$$S_4 = a_{k+1} u_k^{(s)} + q_{k+1} u_{k+1}^{(s)} - 2\omega d_{k+1}\Big[p_{k+2} a_{k+1} u_k^{(s)} + p_{k+2} q_{k+1} u_{k+1}^{(s)} - c_{k+1} q_{k+2} u_{k+2}^{(s)} + c_{k+1} c_{k+2} u_{k+3}^{(s)} - p_{k+2} RH_{k+1} + c_{k+1} RH_{k+2}\Big]$$

then

$$u_k^{(s+1)} = \frac{(S_3 r_{k+1} - S_4 c_k)}{\Delta}, s = 0, 1, 2, \ldots \tag{25}$$

$$u_{k+1}^{(s+1)} = \frac{(S_4 r_k - S_3 a_{k+1})}{\Delta}, s = 0, 1, 2, \ldots \tag{26}$$

Finally, for $k = N$:

$$u_N^{(s+1)} = \frac{q_N u_N^{(s)} - 2\omega d_{N-1}[-a_{N-1} a_N u_{N-2}^{(s)} - a_N q_{N-1} u_{N-1}^{(s)} + p_{N-1} q_N u_N^{(s)} + a_N RH_{N-1} - p_{N-1} RH_N]}{r_N}, \quad s = 0, 1, 2, \ldots \tag{27}$$

Similarly, we can write the CAGE algorithm when $N$ is even.

Now we discuss the two parameter Newton-CAGE iterative method for the non-linear difference equation (9). We follow the approaches given by Evans [25].

Let us define

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{bmatrix}_{N\times 1}, \phi(u) = \begin{bmatrix} \phi_1(u) \\ \phi_2(u) \\ \vdots \\ \phi_N(u) \end{bmatrix}_{N\times 1}$$

and

$$a_k(u) = \frac{\partial \phi_k}{\partial u_{k-1}}, k = 2(1)N$$

$$2b_k(u) = \frac{\partial \phi_k}{\partial u_k}, k = 1(1)N \tag{28}$$

$$c_k(u) = \frac{\partial \phi_k}{\partial u_{k+1}}, k = 1(1)N-1$$

**Table 3 Problem 3: the RMS errors**

| N | Newton-TAGE method | | | Newton-CAGE method | | | RMS errors (for both Newton-TAGE and Newton-CAGE method) |
|---|---|---|---|---|---|---|---|
| | $\omega_{1opt} = \omega_{2opt} = \omega_{opt}$ | iter | CPU time (in sec) | $\omega_{1opt} = \omega_{2opt} = \omega_{opt}$ | iter | CPU time (in sec) | |
| | $R = 10, \beta = 1/2$ | | | | | | |
| 20 | 0.0270 | 15 | 0.0190 | 0.0245 | 09 | 0.0019 | 0.6970(−06) |
| 30 | 0.0202 | 21 | 0.0212 | 0.0180 | 12 | 0.0034 | 0.1452(−06) |
| 40 | 0.0154 | 28 | 0.0248 | 0.0142 | 15 | 0.0044 | 0.4719(−07) |
| 60 | 0.0105 | 43 | 0.0353 | 0.0100 | 21 | 0.0082 | 0.9581(−08) |
| 80 | 0.0093 | 57 | 0.0505 | 0.0076 | 27 | 0.0128 | 0.3081(−08) |
| | $R = 50, \beta = 1/2$ | | | | | | |
| 20 | 0.0100 | 06 | 0.0176 | 0.0110 | 05 | 0.0014 | 0.1992(−03) |
| 30 | 0.0080 | 06 | 0.0180 | 0.0106 | 06 | 0.0023 | 0.4113(−04) |
| 40 | 0.0054 | 08 | 0.0191 | 0.0060 | 06 | 0.0024 | 0.1295(−04) |
| 60 | 0.0041 | 09 | 0.0208 | 0.0041 | 06 | 0.0034 | 0.2571(−05) |
| 80 | 0.0029 | 12 | 0.0246 | 0.0031 | 07 | 0.0047 | 0.8188(−06) |
| | $R = 100, \beta = 1/2$ | | | | | | |
| 20 | 0.0070 | 05 | 0.0175 | 0.0210 | 05 | 0.0016 | 0.1016(−02) |
| 30 | 0.0078 | 05 | 0.0182 | 0.0102 | 05 | 0.0021 | 0.3038(−03) |
| 40 | 0.0044 | 06 | 0.0183 | 0.0068 | 05 | 0.0024 | 0.1518(−03) |
| 60 | 0.0041 | 06 | 0.0195 | 0.0044 | 06 | 0.0037 | 0.3085(−04) |
| 80 | 0.00303 | 07 | 0.0212 | 0.0031 | 06 | 0.0047 | 0.9571(−05) |

Then the Jacobian of $\boldsymbol{\varphi}(\boldsymbol{u})$ can be written as the $N$th-order tri-diagonal matrix

$$T = \frac{\partial \boldsymbol{\varphi}(\boldsymbol{u})}{\partial \boldsymbol{u}} = \begin{bmatrix} 2b_1(\boldsymbol{u}) & c_1(\boldsymbol{u}) & & & \boldsymbol{0} \\ a_2(\boldsymbol{u}) & 2b_2(\boldsymbol{u}) & c_2(\boldsymbol{u}) & & \\ & & \ddots & & \\ & & & \ddots & \\ \boldsymbol{0} & & & a_N(\boldsymbol{u}) & 2b_N(\boldsymbol{u}) \end{bmatrix}_{N\times N} \tag{29}$$

Now with any initial vector $\boldsymbol{u}^{(0)}$, we define

$$\boldsymbol{u}^{(s+1)} = \boldsymbol{u}^{(s)} + \Delta\boldsymbol{u}^{(s)}, \quad s = 0, 1, 2, \ldots \tag{30}$$

where $\Delta\boldsymbol{u}^{(s)}$ is the solution of the nonlinear system

$$T\Delta\boldsymbol{u}^{(s)} = -\boldsymbol{\varphi}\left(\boldsymbol{u}^{(s)}\right), \quad s = 0, 1, 2, \ldots \tag{31}$$

For the Newton-CAGE method, we consider the case when $N$ is odd. We split the matrix $\boldsymbol{T}$ as $\boldsymbol{T} = \boldsymbol{T}_1 + \boldsymbol{T}_2$, where

$$T_1 = \begin{bmatrix} b_1 & & & & & 0 \\ & b_2 & c_2 & & & \\ & a_3 & b_3 & & & \\ & & & \ddots & & \\ 0 & & & & b_{N-1} & c_{N-1} \\ & & & & a_N & b_N \end{bmatrix}_{N\times N}, \tag{32}$$

and

$$T_2 = \begin{bmatrix} b_1 & c_1 & & & & 0 \\ a_2 & b_2 & & & & \\ & & \ddots & & & \\ & & & b_{N-1} & c_{N-1} & \\ & & & a_N & b_N & \\ 0 & & & & & b_N \end{bmatrix}_{N\times N} \tag{33}$$

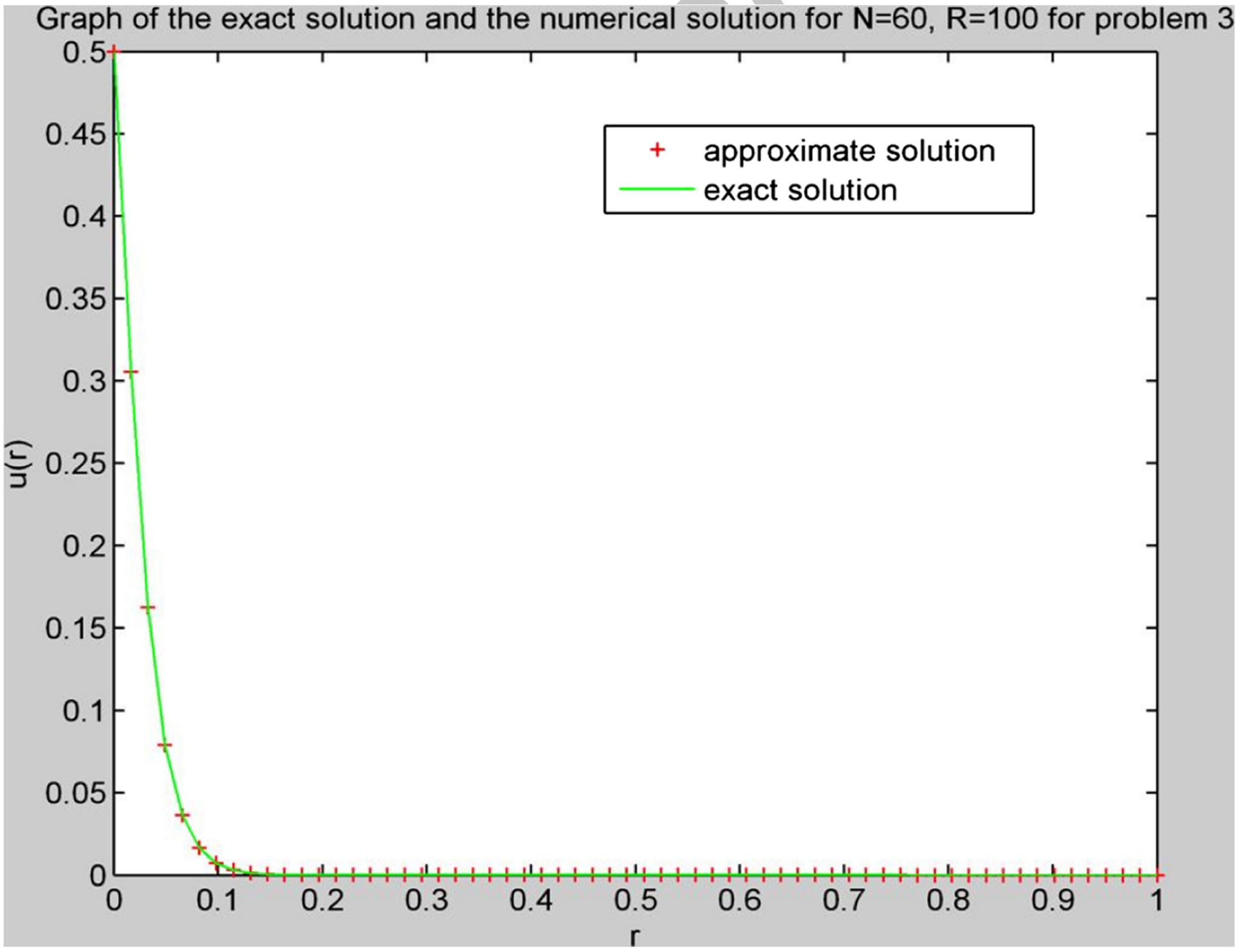

**Figure 3** Graph of the exact solution and the approximate solution for N = 60, R = 100 for problem 3.

**Table 4 Problem 4: the RMS errors**

| N | Newton-TAGE method | | | | Newton-CAGE method | | | | RMS errors (for both Newton-TAGE and Newton-CAGE method) |
|---|---|---|---|---|---|---|---|---|---|
| | $\omega_{1opt}$ | $\omega_{2opt}$ | iter | CPU time (in sec) | $\omega_{1opt}$ | $\omega_{2opt}$ | iter | CPU time (in sec) | |
| | $R = 10,\ \alpha = 1$ | | | | | | | | |
| 40 | 0.0174 | 0.0182 | 21 | 0.1853 | 0.0173 | 0.0188 | 13 | 0.1626 | 0.1238(−05) |
| 50 | 0.0157 | 0.0153 | 26 | 0.2798 | 0.0150 | 0.0153 | 15 | 0.1934 | 0.5160(−06) |
| 60 | 0.0128 | 0.0131 | 30 | 0.3790 | 0.0108 | 0.0126 | 18 | 0.2440 | 0.2516(−06) |
| 70 | 0.0110 | 0.0113 | 37 | 0.5395 | 0.0108 | 0.0100 | 22 | 0.3070 | 0.1369(−06) |
| 80 | 0.0092 | 0.00935 | 41 | 0.6778 | 0.0086 | 0.0096 | 25 | 0.3796 | 0.8075(−07) |
| | $R = 50,\ \alpha = 1$ | | | | | | | | |
| 40 | 0.0056 | 0.0057 | 16 | 0.1445 | 0.0054 | 0.006 | 12 | 0.1437 | 0.5441(−05) |
| 50 | 0.0050 | 0.0044 | 18 | 0.1988 | 0.0043 | 0.0048 | 13 | 0.1797 | 0.2268(−05) |
| 60 | 0.0035 | 0.0039 | 21 | 0.2697 | 0.0052 | 0.0041 | 13 | 0.2000 | 0.1106(−05) |
| 70 | 0.0035 | 0.0031 | 25 | 0.3691 | 0.0040 | 0.0033 | 14 | 0.2265 | 0.6022(−06) |
| 80 | 0.0028 | 0.0031 | 27 | 0.4513 | 0.0029 | 0.00294 | 16 | 0.2700 | 0.3551(−06) |
| | $R = 10,\ \alpha = 2$ | | | | | | | | |
| 40 | 0.0200 | 0.0210 | 19 | 0.1696 | 0.0180 | 0.0188 | 15 | 0.1612 | 0.1374(−05) |
| 50 | 0.0161 | 0.0151 | 24 | 0.2580 | 0.0169 | 0.0151 | 18 | 0.2147 | 0.5727(−06) |
| 60 | 0.0129 | 0.0132 | 29 | 0.3660 | 0.0117 | 0.0134 | 22 | 0.2760 | 0.2794(−06) |
| 70 | 0.0106 | 0.0101 | 36 | 0.5241 | 0.0118 | 0.0112 | 25 | 0.3380 | 0.1521(−06) |
| 80 | 0.0106 | 0.0101 | 38 | 0.6287 | 0.0099 | 0.0100 | 29 | 0.4199 | 0.8970(−07) |
| | $R = 50,\ \alpha = 2$ | | | | | | | | |
| 40 | 0.0055 | 0.0064 | 15 | 0.1371 | 0.0045 | 0.0060 | 13 | 0.1370 | 0.5713(−05) |
| 50 | 0.0053 | 0.0046 | 16 | 0.1785 | 0.0054 | 0.0038 | 14 | 0.1764 | 0.2382(−05) |
| 60 | 0.0037 | 0.00385 | 20 | 0.2585 | 0.0034 | 0.0047 | 14 | 0.2063 | 0.1162(−05) |
| 70 | 0.0027 | 0.003 | 24 | 0.3557 | 0.0042 | 0.0032 | 15 | 0.2378 | 0.6325(−06) |
| 80 | 0.0029 | 0.0032 | 26 | 0.4345 | 0.0039 | 0.0026 | 17 | 0.2811 | 0.3730(−06) |

then we write Newton-CAGE method as:

$$\begin{aligned}(\boldsymbol{T}_2 + \omega_2\boldsymbol{I})\Delta\boldsymbol{u}^{(s+1)} &= \left[\boldsymbol{I} - (\omega_1+\omega_2)(\boldsymbol{T}_1+\omega_1\boldsymbol{I})^{-1}\right]\\ &\quad\times(\boldsymbol{T}_2-\omega_1\boldsymbol{I})\Delta\boldsymbol{u}^{(s)} - (\omega_1+\omega_2)\\ &\quad\times(\boldsymbol{T}_1+\omega_1\boldsymbol{I})^{-1}\boldsymbol{\varphi}\left(\boldsymbol{u}^{(s)}\right),\\ &\quad s = 0,1,2,\ldots \end{aligned} \tag{34}$$

where $\omega_1 > 0, \omega_2 > 0$ are relaxation parameters and $(\boldsymbol{T}_1 + \omega_1\boldsymbol{I})$ and $(\boldsymbol{T}_2 + \omega_2\boldsymbol{I})$ are non-singular.

Since $(\boldsymbol{T}_1 + \omega_1\boldsymbol{I})$ and $(\boldsymbol{T}_2 + \omega_2\boldsymbol{I})$ consists of $(2 \times 2)$ sub-matrices, they can be easily inverted.

$$(\boldsymbol{T}_1+\omega_1\boldsymbol{I})^{-1} = \begin{bmatrix} 1/p_1 & & & 0 \\ & \begin{pmatrix} p_3 & -c_2 \\ -a_3 & p_2 \end{pmatrix}/\Delta_2 & & \\ & 0 & \ddots & \\ & & & \begin{pmatrix} p_N & -c_{N-1} \\ -a_N & p_{N-1} \end{pmatrix}/\Delta_{N-1} \end{bmatrix}_{N\times N} \tag{35}$$

and

$$(\boldsymbol{T}_2+\omega_2\boldsymbol{I})^{-1} = \begin{bmatrix} \begin{pmatrix} r_2 & -c_1 \\ -a_2 & r_1 \end{pmatrix}/\Delta_1 & & 0 \\ & \ddots & \\ 0 & \begin{pmatrix} r_{N-1} & -c_{N-2} \\ -a_{N-1} & r_{N-2} \end{pmatrix}/\Delta_{N-2} & \\ & & 1/r_N \end{bmatrix}_{N\times N} \tag{36}$$

with $p_k = b_k + \omega_1, k = 1(1)N$ and $\Delta_k = p_k p_{k+1} - c_k a_{k+1}, k = 2(2)N-1$ and $r_k = b_k + \omega_2, k = 1(1)N$ and $\Delta_k = r_k r_{k+1} - c_k a_{k+1}, k = 1(2)N-2$.

Further the matrices $(T_2 + \omega_2 I)^{-1}(T_1 + \omega_1 I)^{-1}(T_2 - \omega_1 I)$ and $(T_2 + \omega_2 I)^{-1}(T_1 + \omega_1 I)^{-1}$ can be evaluated in a manner suitable for parallel computing. In order for this Newton-CAGE method to converge, it is sufficient that the initial vector $\boldsymbol{u}^{(0)}$ be close to the solution.

In a similar manner, we can write the Newton-CAGE algorithm when $N$ is even.

## Numerical illustrations

To illustrate the proposed CAGE iterative methods, we have solved the following four problems whose exact solutions are known. We have also compared the proposed CAGE iterative methods with the corresponding TAGE iterative methods. The right-hand side functions and boundary conditions can be obtained by using the exact solutions. The initial vector ***0*** is used in all iterative methods, and iterations were stopped when $|u^{(s+1)} - u^{(s)}| \leq 10^{-10}$ was achieved. While solving non-linear difference equations, we have considered five inner iterations only.

### *Problem 1*

$$u'' = \beta u', 0 < r < 1 \quad (\text{Convection} - \text{Diffusion equation}) \tag{37}$$

The exact solution is $u(x) = (1 - e^{-\beta(1-r)})/(1 - e^{-\beta})$.The root mean square (RMS) errors and the number of iterations both for CAGE and TAGE methods are presented in Table 1 for various values of β. The graph of the exact solution and the approximate solution for $N$ = 80, β = 100 is give in the Figure 1.

### *Problem 2*

$$u'' + \frac{\alpha}{r}u' - \frac{\alpha}{r^2}u = f(r), 0 < r < 1, \qquad \alpha = 1, 2 \;\; (\text{Linear Singular Equation}) \tag{38}$$

The exact solution is $u(r) = e^{r^4}$. The RMS errors and the number of iterations for both CAGE and TAGE methods are presented in Table 2 for α = 1,2. The graph of the exact solution and the approximate solution for $N$ = 80, α = 2 is give in the Figure 2.

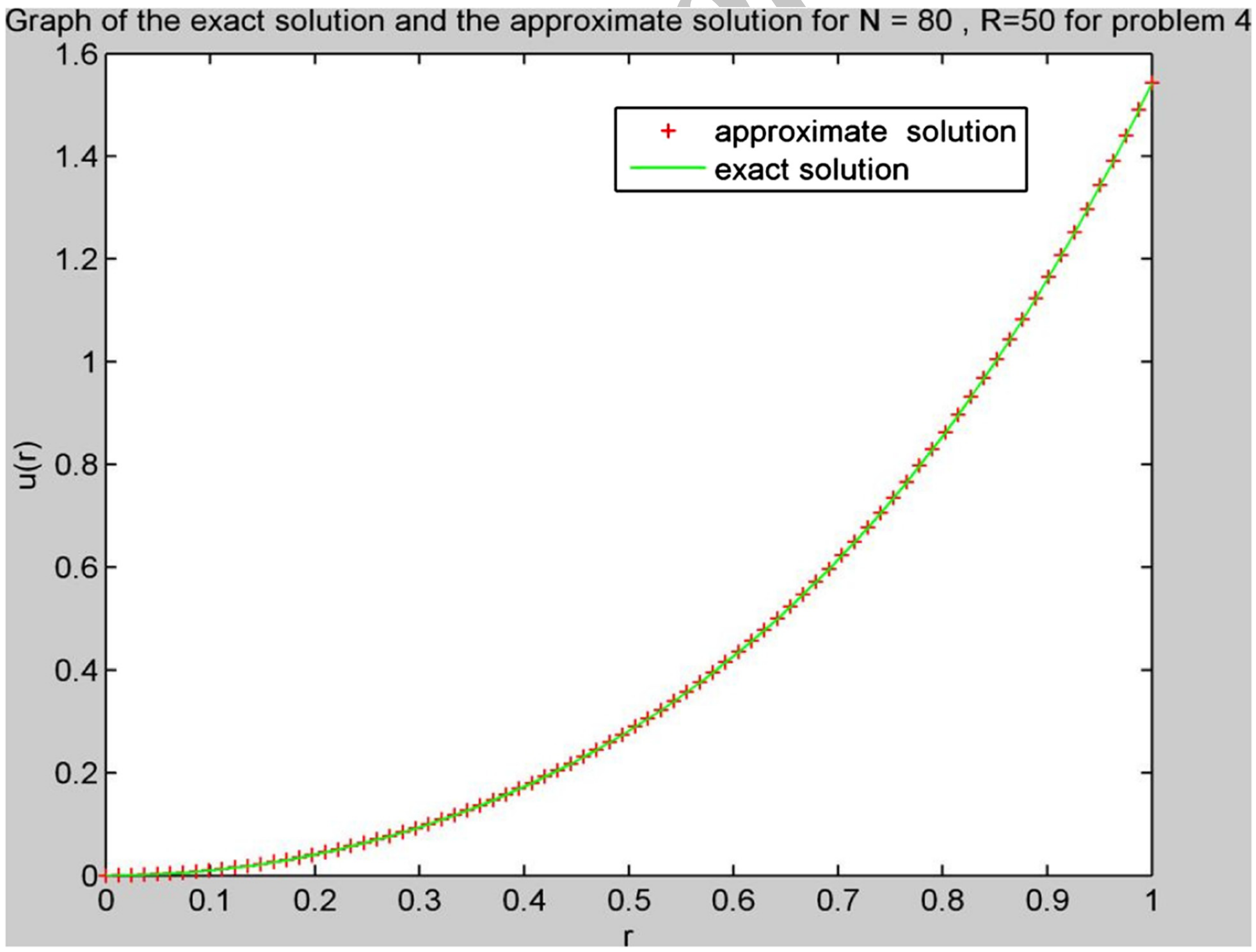

**Figure 4** Graph of the exact solution and the approximate solution for N = 80, R = 50 for problem 4.

***Problem 3***

$$\nu u'' = (u-\beta)u', 0 < r < 1 \quad \text{(Burgers'equation)} \tag{39}$$

The exact solution is $u(r) = \beta[1 - \tanh(\beta r/2\nu)]$. The root mean square (RMS) errors and the number of iterations both for both Newton-CAGE and Newton-TAGE methods are presented in Table 3 for β = 1/2 and various values of $R = \nu^{-1}$. The graph of the exact solution and the approximate solution for $N$ = 60, $R$ = 100 is given in the Figure 3.

***Problem 4***

$$\frac{1}{R}\left[u'' + \frac{\alpha}{r}u' - \frac{\alpha}{r^2}u\right] = uu' + g(r), 0 < r < 1, \quad \alpha = 1, 2 \quad \text{(Burgers'equation)} \tag{40}$$

The exact solution is $u(r) = r^2 \cosh(r)$. The RMS errors and the number of iterations for both Newton-CAGE and Newton-TAGE methods are presented in Table 4 for α = 1,2 and various values of Re.The graph of the exact solution and the approximate solution for $N$ = 80, R = 50 is given in the Figure 4.

## Final remarks

The TAGE method requires two sweeps to solve a problem and also, it requires a lot of algebra for computational work, whereas the CAGE method requires only one-sweep to solve the problem. Experimentally, as compared to the TAGE method the corresponding CAGE method is requires very less number of iterations and better time because it uses less intermediate variables. We have solved four benchmark problems and numerical results show the efficiency of the proposed CAGE method. The results conclude that the two parameter CAGE method is competitive to solve the one-dimensional problem and it can be extended to solve multi-dimensional problems.

**Competing interests**
The authors declare that they have no competing interests.

**Author's contribution**
RKM derived the cubic spline approximation and discussed its application to singular problems. JT carried out the CAGE algorithm, its convergence analysis and performed all computational work. All authors read and approved the final manuscript.

**Acknowledgements**
This research was supported by 'The Council of Scientific and Industrial Research' under research grant No. 09/045(0836)2009-EMR-I.

Received: 26 May 2012 Accepted: 21 July 2012
Published: 29 October 2012

**References**
1. Keller, H.B.: Numerical Methods for Two Point Boundary Value Problem. Blaisdell Pub. Co., Waltham, MA (1992)
2. Ahlberg, J.H., Nilson, J.H., Walsh, E.N.: The Theory of Splines and Their Applications. Academic Press, San Diego (1967)
3. De Boor, C.: Practical Guide to Splines. Springer, Berlin (1978)
4. Micula, G.: Handbook of Splines. Kluwer Academic, Dordercht (1999)
5. Prenter, P.M.: Splines and Variational Methods. Wiley, New York (1975)
6. Regan, D.O.: Theory of Singular Boundary Value Problems. World Scientific, Singapore (1994)
7. Bickley, W.G.: Piecewise cubic interpolation and two point boundary value problems. Comput. J. **11**, 206–212 (1968)
8. Albasiny, E.L., Hoskins, W.D.: Cubic splines solutions to two point boundary value problems. Comput. J. **12**, 151–153 (1969)
9. Albasiny, E.L., Hoskins, W.D.: Increased accuracy cubic spline solutions to two point boundary value problems. IMA J. Numer. Anal. **9**(1), 47–55 (1972)
10. Fyfe, D.J.: The use of cubic splines in the solution of two point boundary value problems. Comput. J. **12**, 188–192 (1969)
11. Chawla, M.M., Subramanian, R.: A new fourth order cubic spline method for second-order nonlinear two point boundary value problems. J of Comp and Appl Math **23**, 1–10 (1988)
12. Jain, M.K., Aziz, T.: Cubic spline solution of two point boundary value problems with significant first derivatives. Comput Meth Appl Mech Eng **39**, 83–91 (1983)
13. Al-Said, E.A., Noor, M.A., Al-Shejari, A.A.: Numerical solution for system of second order boundary value problems. J. Appl. Math. Comput. **5**, 669–680 (1998)
14. Al-Said, E.A.: Cubic spline method for solving two point boundary value problems. J. Appl. Math. Comput. **5**, 669–680 (1998)
15. Al-Said, E.A.: The Use of Cubic Splines in the Numerical Solution of Systems of Second-Order Boundary Value Problems. Comput Math Applics **42**, 861–869 (2001)
16. Khan, A.: Parametric cubic spline solution of two point boundary value problems. Appl. Math. Comput. **154**, 175–182 (2004)
17. Ravi Kanth, A.S.V., Reddy, Y.N.: Cubic spline for a class of singular two-point boundary value problems. Appl. Math. Comput. **170**, 733–740 (2005)
18. Al-Said, E.A., Noor, M.A.: Cubic spline method for solving fourth order obstacle problems. Appl. Math. Comput **174**, 180–187 (2006)
19. Kumar, M.: Higher order method for singular boundary problems by using spline function. Appl. Math. Comput. **192**, 175–179 (2007)
20. Evans, D.J.: Group explicit methods for solving large linear systems. Int. J. Comput. Math. **17**, 81–108 (1985)
21. Sukon, K.S., Evans, D.J.: Two Parameter AGE Method for the Solution of Tridiagonal Linear System of Equations. Int. J. Comput. Math. **60**, 265–278 (1996)
22. Mohanty, R.K., Evans, D.J.: A fourth order accurate cubic spline alternating group explicit method for non-linear singular two point boundary value problems. Int. J. Comput. Math. **80**, 479–492 (2003)
23. Mohanty, R.K., Sachdev, P.L., Jha, N.: An $O(h^4)$accurate cubic spline TAGE method for non-linear singular two point boundary value problems. Appl. Math. Comput. **158**, 853–868 (2004)
24. Mohanty, R.K., Evans, D.J.: Highly accurate two parameter CAGE parallel algorithms for non-linear singular two point boundary value problems. Int. J. Comput. Math. **82**, 433–444 (2005)
25. Evans, D.J.: Iterative methods for solving non-linear two point boundary value problems. Int. J. Comput. Math. **72**, 395–401 (1999)

doi:10.1186/2251-7456-6-58
**Cite this article as:** Mohanty and Talwar: **Compact alternating group explicit method for the cubic spline solution of two point boundary value problems with significant nonlinear first derivative terms.** *Mathematical Sciences* 2012 **6**:58.