Iranian Journal of Numerical Analysis and Optimization

Volume 5 , Number 2

Summer 2015

Ferdowsi University of Mashhad, Iran

In the Name of God

Iranian Journal of Numerical Analysis and Optimization (IJNAO)

This journal is authorized under the registration No. 174/853 dated 1386/2/26, by the Ministry of Culture and Islamic Guidance.

Volume 5, Number 2, Summer 2015

ISSN: 2423-6977

Publisher: Faculty of Mathematical Sciences, Ferdowsi University of Mashhad

Published by: Ferdowsi University of Mashhad Press

Circulation: 100

Address: Iranian Journal of Numerical Analysis and Optimization

Faculty of Mathematical Sciences, Ferdowsi University of Mashhad

P.O. Box 1159, Mashhad 91775, Iran.

Tel. : +98-51-38806222 , **Fax:** +98-51-38807358

E-mail: mjms@um.ac.ir

Website: http://ijnao.um.ac.ir

This journal is indexed by:

• Mathematical Review

• Zentralblatt

• ISC

• SID

به اطلاع کلیه محققان، پژوهشگران، اساتید ارجمند، دانشجویان تحصیلات تکمیلی و نویسندگان محترم می رساند که نشریه ایرانی آنالیز عددی و بهینه سازی - IJNAO - طبق مجوز شماره ۳/۱۸/۵۴۸۹۱۳/ مورخه ۱۳۹۲/۱۰/۲۵ مدیر کل محترم سیاست گذاری و برنامه ریزی امور پژوهشی وزارت علوم، تحقیقات و فناوری، علمی – پژوهشی میباشد.
بر اساس نامه شماره ۹۲/۱۵۳۶/پ مورخه ۱۳۹۲/۱۱/۲۱ سرپرست محترم معاونت پژوهشی وفناوری پایگاه استنادی علوم جهان اسلام، نشریه ایرانی آنالیز عددی و بهینه سازی در پایگاه ISC نیز نمایه می شود.

Iranian Journal of Numerical Analysis and Optimization

Volume 5, Number 2, Summer 2015

Ferdowsi University of Mashhad - Iran

Iranian Journal of Numerical Analysis and Optimization

This journal is published under the auspices of Ferdowsi University of Mashhad

---

* Full Professor

** Associate Professor

We would like to acknowledge the help of Narjes khatoon Zohorian in the preparation of this issue.

Letter from the Editor in Chief

I would like to welcome you to the Iranian Journal of Numerical Analysis and Optimization (IJNAO). This journal is published biannually and supported by the Faculty of Mathematical Sciences at the Ferdowsi University of Mashhad. Faculty of Mathematical Sciences with three centers of excellence and three research centers is well-known in mathematical communities in Iran.

The main aim of the journal is to facilitate discussions and collaborations between specialists in applied mathematics, especially in the fields of numerical analysis and optimization, in the region and worldwide.

Our vision is that scholars from different applied mathematical research disciplines, pool their insight, knowledge and efforts by communicating via this international journal.

In order to assure high quality of the journal, each article is reviewed by subject-qualified referees.

Our expectations for IJNAO are as high as any well-known applied mathematical journal in the world. We trust that by publishing quality research and creative work, the possibility of more collaborations between researchers would be provided. We invite all applied mathematicians especially in the fields of numerical analysis and optimization to join us by submitting their original work to the Iranian Journal of Numerical Analysis and Optimization.

Mohammad Hadi Farahi

Contents

High order second derivative methods with Runge–Kutta stability for the numerical solution of stiff ODEs 1
A. Abdi and G. Hojjati

The block LSMR algorithm for solving linear systems with multiple right-hand sides 11
F. Toutounian and M. Mojarrab

A practical review of the Adomian decomposition method: computer implementation aspects 29
A. Molabahrami

An adaptive meshless method of line based on radial basis functions 45
J. Biazar and M. Hosami

Application of modified simple equation method to Burgers, Huxley and Burgers-Huxley equations 59
Z. Ayati, M. Moradi and M. Mirzazadeh

On convergence and stability conditions of homotopy perturbation method for an inverse heat conduction problem 75
Q. Jannati and A. Zakeri

An adaptive nonmonotone trust region method for unconstrained optimization problems based on a simple subproblem 95
Z. Saeidian and M.R. Peyghami

Iranian Journal of Numerical Analysis and Optimization
Vol. 5, No. 2, (2015), pp 1-10

High order second derivative methods with Runge–Kutta stability for the numerical solution of stiff ODEs

A. Abdi* and G. Hojjati

Abstract

We describe the construction of second derivative general linear methods (SGLMs) of orders five and six. We will aim for methods which are A–stable and have Runge–Kutta stability property. Some numerical results are given to show the efficiency of the constructed methods in solving stiff initial value problems.

Keywords: Ordinary differential equation; General linear methods; Runge–Kutta stability; A–stability; Second derivative methods.

1 Introduction

In many fields such as control theory, chemical kinetics, biology and the movement of stars in galaxies, dynamic behavior is modeled by systems of ordinary differential equations (ODEs). We consider the autonomous ODEs in the form

$$\begin{aligned} &y'(x) = f(y(x)), \qquad x \in [x_0, \overline{x}], \\ &y(x_0) = y_0, \end{aligned} \tag{1}$$

where $f : \mathbb{R}^m \to \mathbb{R}^m$ and m is the dimensionality of the system. We restrict our attention to autonomous systems because non-autonomous systems can be made autonomous by adding an extra equation to the system.

For system (1), let $g := f_y f$. For problems in which g can be calculated along with f, at a moderate additional cost, second derivative methods become feasible. General linear methods (GLMs) [6, 7, 12] as a unifying

*Corresponding author
Received 15 October 2014; revised 15 February 2015; accepted 21 February 2015
A. Abdi
Faculty of Mathematical Sciences, University of Tabriz, Tabriz, Iran.
e-mail: a_abdi@tabrizu.ac.ir

G. Hojjati
Faculty of Mathematical Sciences, University of Tabriz, Tabriz, Iran.
email: ghojjati@tabrizu.ac.ir

framework for the traditional methods, like Runge–Kutta methods, linear multistep methods, predictor–corrector methods and hybrid methods, have been extended to the second derivative general linear methods (SGLMs) by Butcher and Hojjati [8]. These methods which are s-stage and r-value, for the numerical solution of (1) are given by

$$\begin{aligned} Y^{[n]} &= h(A\otimes I_m)F(Y^{[n]}) + h^2(\overline{A}\otimes I_m)G(Y^{[n]}) + (U\otimes I_m)y^{[n-1]},\\ y^{[n]} &= h(B\otimes I_m)F(Y^{[n]}) + h^2(\overline{B}\otimes I_m)G(Y^{[n]}) + (V\otimes I_m)y^{[n-1]}, \end{aligned} \tag{2}$$

where h is the stepsize, A, $\overline{A}\in\mathbb{R}^{s\times s}$, $U\in\mathbb{R}^{s\times r}$, B, $\overline{B}\in\mathbb{R}^{r\times s}$ and $V\in\mathbb{R}^{r\times r}$ and notation $\otimes$ is the Kronecker product. Here, $Y^{[n]}=[Y_i^{[n]}]_{i=1}^s$ is an approximation of stage order q to the vector $y(x_{n-1}+ch)=[y(x_{n-1}+c_ih)]_{i=1}^s$, i.e.

$$Y_i^{[n]} = \sum_{k=0}^{q}\frac{c_i^k}{k!}h^k y^{(k)}(x_{n-1}) + O(h^{q+1}),\quad i=1,2,\ldots,s, \tag{3}$$

$F(Y^{[n]}):=[f(Y_i^{[n]})]_{i=1}^s$ and $G(Y^{[n]}):=[g(Y_i^{[n]})]_{i=1}^s$ where the vector $c=[c_1\ \ c_2\ \ \cdots\ \ c_s]^T$ is the abscissa vector. Also the vectors $y^{[n-1]}=[y_i^{[n-1]}]_{i=1}^r$ and $y^{[n]}=[y_i^{[n]}]_{i=1}^r$ are the input and output vectors at the step number n, respectively, which for a method of order p take the following forms

$$y_i^{[n-1]} = \sum_{k=0}^{p}\alpha_{ik}h^k y^{(k)}(x_{n-1}) + O(h^{p+1}),\quad i=1,2,\ldots,r, \tag{4}$$

and

$$y_i^{[n]} = \sum_{k=0}^{p}\alpha_{ik}h^k y^{(k)}(x_{n}) + O(h^{p+1}),\qquad i=1,2,\ldots,r, \tag{5}$$

for some $\alpha_{ik}\in\mathbb{R}$ associated with the method.

The main features of SGLMs including pre-consistency, consistency, zero-stability and types of these methods have been discussed in [3]. It has been shown in [4] that the SGLM (2) with the input vector (4) has order p and stage order $q=p$ iff

$$e^{cz} = zAe^{cz} + z^2\overline{A}e^{cz} + Uw(z) + O(z^{p+1}), \tag{6}$$

$$e^{z}w(z) = zBe^{cz} + z^2\overline{B}e^{cz} + Vw(z) + O(z^{p+1}). \tag{7}$$

where

$$e^{cz} = \left[e^{c_1z}\ \ e^{c_2z}\ \ \cdots\ \ e^{c_sz}\right]^T,$$

and $w(z)$ is a vector with elements given by

$$w_i(z) = \sum_{k=0}^{p}\alpha_{ik}z^k,\qquad i=1,2,\cdots,r.$$

In the special SGLMs with $p = q = r = s$, $U = I_s$ and $V\mathrm{e} = \mathrm{e}$, $\mathrm{e} = [1, 1, \ldots, 1]^T \in \mathbb{R}^s$, an equivalent condition for order conditions has been found in [5] as

$$B = B_0 - AB_1 - \overline{A}B_2 - VB_3 - (\overline{B} - V\overline{A})B_4 + VA,$$

where the (i, j) elements of B_0, B_1, B_2, B_3, and B_4 are given respectively by

$$\frac{\int_0^{1+c_i} \phi_j(x)dx}{\phi_j(c_j)}, \qquad \frac{\phi_j(1+c_i)}{\phi_j(c_j)}, \qquad \frac{\phi_j'(1+c_i)}{\phi_j(c_j)}, \qquad \frac{\int_0^{c_i} \phi_j(x)dx}{\phi_j(c_j)}, \qquad \frac{\phi_j'(c_i)}{\phi_j(c_j)}.$$

Here,

$$\phi_i(x) = \prod_{j=1, j\neq i}^{s} (x - c_j), \qquad i = 1, 2, \cdots, s.$$

Construction of SGLMs which are also suitable for the numerical solution of differential algebraic equations (DAEs) has been discussed in [10]. Some obtained order barries for different types of SGLMs, found in [3,4,10], are useful in construction of these methods. These barriers have been also confirmed by means of order arrows by Abdi and Butcher [1, 2]. Recently, efficiency of these methods in solving stiff ODEs arising from chemical reactions has been shown in [11].

In continuation of studying on SGLMs, in this paper we construct A–stable methods of orders five and six with $r = s = 3$ and Runge–Kutta stability property.

Next sections of this paper are organized as follows: In Sec. 2, we discuss about stability behaviour of Runge–Kutta stable three-stage methods. Sec. 3 is devoted to construction of SGLMs of orders five and six with A–stability property. Some numerical experiments are given in Sec. 4 to demonstrate the efficiency of the constructed methods.

2 RKS three-stage methods

We first recall that the stability matrix for SGLMs can be obtained by applying the methods to the standard test problem of Dahlquist [9] $y' = \zeta y$, where ζ is a complex number, which it is

$$M(z) = V + \left(zB + z^2\overline{B}\right)\left(I - zA - z^2\overline{A}\right)^{-1}U,$$

where $z = h\zeta$. Thus, we are interested in stable behavior of powers of $M(z)$. If $M(z)$ has only a single non-zero eigenvalue, $R(z)$, then the method is said to possess Runge-Kutta stability (RKS) property. For RKS methods, the stability behaviour is related to $R(z)$.
For the methods in which coefficient matrices A and $\overline{A}$ are lower triangular

with the same elements λ and μ on the diagonal, respectively, $R(z)$ takes the form

$$R(z) = \frac{N(z)}{(1-\lambda z - \mu z^2)^s}, \tag{8}$$

where $\deg(N) \leq 2s$. For the methods of order five and six with three stages that will be discussed in Sec. 3, the polynomial N defined in (8) satisfies

$$N(z) = (1-\lambda z - \mu z^2)^3 e^z - C_5 z^6 + O(z^7),$$

and

$$N(z) = (1-\lambda z - \mu z^2)^3 e^z + O(z^7),$$

respectively, for an arbitrary C_5 as the error constant of the method. For the method of order six, the error constant is

$$C_6 = \frac{1}{5040} - \frac{1}{240}\lambda - \frac{1}{40}\mu + \frac{1}{4}\lambda\mu + \big(\frac{1}{40} - \frac{1}{2}\mu\big)\lambda^2 + \big(\frac{1}{2} - \frac{3}{2}\lambda\big)\mu^2 - \frac{1}{24}\lambda^3 - \mu^3.$$

For these methods to be A–stable, using E-polynomial theorem [7], it is necessary and sufficient that $\lambda > 0$, $\mu < 0$, and so that the $E(y)$ is non-negative for y real where the E-polynomial is defined by

$$E(y) = |1 - \lambda \mathbf{i} y + \mu y^2|^6 - |N(\mathbf{i}y)|^2,$$

where $\mathbf{i}$ is the imaginary unit. The boundary of the regions of A-stable choices of (λ, μ) for the methods of order five (with different values of C_5) and order six are plotted in Figure 1 and Figure 2.

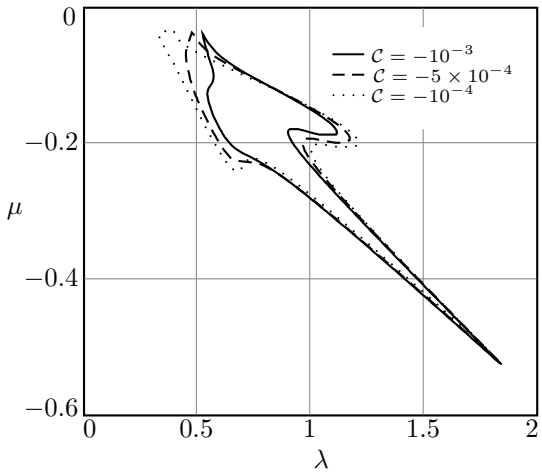

Figure 1: The boundary of the regions of A–stable choices of (λ, μ) for $s = 3$, $p = 5$ corresponding to $\mathcal{C} = -10^{-3}, -5 \times 10^{-4}, -10^{-4}$

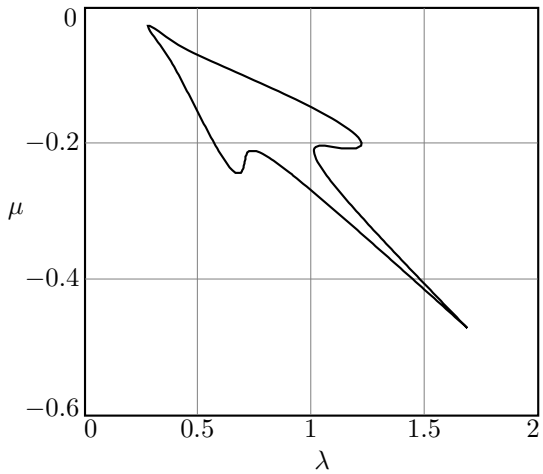

Figure 2: The boundary of the region of A–stable choices of (λ, μ) for $s = 3$, $p = 6$

3 *A*–stable RKS methods of orders 5 and 6

Construction SGLMs of orders $p = q \leq 4$ has been discussed in [3–5, 10]. In this section, we construct A–stable three-stage methods of orders five and six with RKS property. Throughout the construction of these methods, we will consider $U = I_s$ and $V = \mathrm{e}v^T$ where $v \in \mathbb{R}^r$ and $v^T\mathrm{e} = 1$. The later guarantees zero-stability of the methods [3].

3.1 Order 5 methods

Choosing $c = [0 \ \frac{1}{2} \ 1]^T$, $(\lambda, \mu) = (0.6, -0.1)$ from the intersection of the regions in Figure1 and solving the order conditions and the nonlinear RKS conditions, the coefficients matrices of the method take the following forms

$$A = \begin{bmatrix} 0.6000000000 & 0 & 0 \\ 0.4538633794 & 0.6000000000 & 0 \\ 0.8442059328 & 0.8999163314 & 0.6000000000 \end{bmatrix},$$

$$\overline{A} = \begin{bmatrix} -0.1000000000 & 0 & 0 \\ -0.1450566118 & -0.1000000000 & 0 \\ -0.9847293116 & -0.1278647721 & -0.1000000000 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.3902646263 & 0.4639576064 & 0.2524239604 \\ -0.3312778090 & 1.1306242731 & 0.3534363496 \\ 5.0478598121 & -4.1644469839 & -0.5208888994 \end{bmatrix},$$

$$\overline{B} = \begin{bmatrix} -0.2677332867 & -0.3732899225 & -0.0223237563 \\ -0.4095181371 & -0.6362626571 & -0.0357186615 \\ 0.5750983052 & 1.6053219094 & 0.0622616286 \end{bmatrix},$$

$$v = \begin{bmatrix} 1.2203054517 & -0.3423946125 & 0.1220891608 \end{bmatrix}^T.$$

This method is A–stable with the error constant $C_5 \approx -3.50 \times 10^{-4}$.

3.2 Order 6 methods

Choosing $c = [0 \;\; c_1 \;\; 1]^T$, c_1 as a free parameter, and solving the order conditions and the nonlinear RKS conditions, we get $c_1 = -1.4989329045$ and the coefficients matrices of the method take the following forms

$$A = \begin{bmatrix} 0.4007120047 & 0 & 0 \\ 0.5574459850 & 0.4007120047 & 0 \\ 0.7281456081 & 0.0121320319 & 0.4007120047 \end{bmatrix},$$

$$\overline{A} = \begin{bmatrix} -0.0612701047 & 0 & 0 \\ -0.0145743957 & -0.0612701047 & 0 \\ 0.3881180321 & 0.1117302066 & -0.0612701047 \end{bmatrix},$$

$$B = \begin{bmatrix} 1.1371686053 & 0.2249968367 & 0.0903218055 \\ -0.0512895056 & 0.1078326109 & -0.6604347472 \\ 1.5642870990 & 0.3929237249 & -0.2450012162 \end{bmatrix},$$

$$\overline{B} = \begin{bmatrix} -0.0425486219 & 0.0078897842 & -0.0128566928 \\ 0.1945434509 & -0.0296649869 & 0.0449770864 \\ 0.3584398092 & 0.0701030286 & -0.0116769898 \end{bmatrix},$$

$$v = \begin{bmatrix} 0.8572479903 & 0.2113738061 & -0.0686217964 \end{bmatrix}^T.$$

The obtained value for (λ, μ) is the interior of the region of A–stable choices presented in Figure 2. The error constant for this A–stable method is $C_6 \approx 2.56 \times 10^{-5}$.

4 Numerical verifications

In this section we present some numerical results by applying the constructed methods of orders five and six in Sec. 3, in order to demonstrate the theoretical expectations. Computational experiments are carried out by applying the methods to the following two stiff problems.

S1– The non-linear stiff test problem

$$\begin{cases} y_1'(x) = -1002y_1(x) + 1000y_2^2(x),\ y_1(0) = 1, \\ y_2'(x) = y_1(x) - y_2(x)(1 + y_2(x)),\ y_2(0) = 1. \end{cases}$$

The exact solution is $y_1(x) = \exp(-2x)$ and $y_2(x) = \exp(-x)$ and $x \in [0, 1]$.

S2– The stiff initial value problem arose from a chemistry problem

$$\begin{cases} y_1'(x) = -0.013y_2 - 1000y_1y_2 - 2500y_1y_3,\ y_1(0) = 0, \\ y_2'(x) = -0.013y_2 - 1000y_1y_2, \qquad\qquad\quad y_2(0) = 1, \\ y_3'(x) = -2500y_1y_3, \qquad\qquad\qquad\quad y_3(0) = 1. \end{cases}$$

The reference solution at $x = 2$ is

$$\begin{aligned} y_1(2) &= -0.3616933169289 \times 10^{-5}, \\ y_2(2) &= 0.9815029948230, \\ y_3(2) &= 1.018493388244. \end{aligned}$$

Numerical results for the Problem S1, reported in Table 1, illustrate accuracy of the methods of order 5 and 6. These results are obtained with fixed stepsizes $h = 1/2^k$ with several integer values for k. In this table, we have listed norm of error $\|e_h(x)\|$ at the endpoint of integration $x = 1$. Also, in this table, the rows p refer to the numerical estimates to the order of

convergence, computed by the formula $p = \log_2(\|e_h(x)\|/\|e_{h/2}(x)\|)$ where $e_h(x)$ and $e_{h/2}(x)$ are errors corresponding to stepsizes h and $h/2$.

Table 1: The global error at the end of the interval of integration $[0, 1]$ for problem S1

h	2^{-2}	2^{-3}	2^{-4}	2^{-5}
Order 5 method	2.25×10^{-7}	5.61×10^{-9}	1.51×10^{-10}	4.34×10^{-12}
p		5.33	5.22	5.12
Order 6 method	6.92×10^{-8}	2.94×10^{-10}	2.45×10^{-12}	5.03×10^{-14}
p		7.88	6.91	5.61

Numerical results for the Problem S2 are given in Table 2 with stepsize $h = 0.001$. Comparing the obtained results by the methods with the reference solution shows the efficiency of the methods for solving stiff non-linear problems.

Table 2: Numerical results for problem S2 solved by the methods of orders five and six

x	y	Order 5 method	Order 6 method
	y_1	$-0.3616933169478728 \times 10^{-5}$	$-0.3616933215630078 \times 10^{-5}$
2	y_2	0.9815029948594308	0.9815030036954803
	y_3	1.018493388207507	1.018493379371295

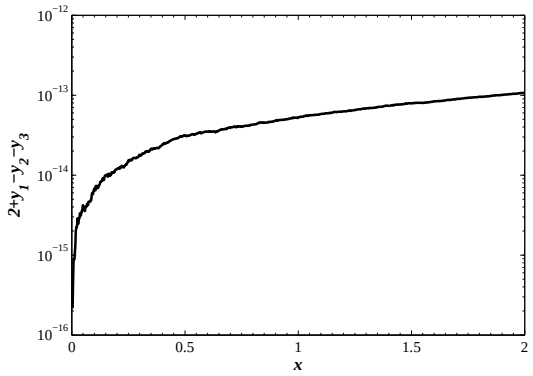

Figure 3: Variation of $2 + y_1 - y_2 - y_3$ versus x which y_1, y_2 and y_3 are the numerical solutions obtained by the method of order 5

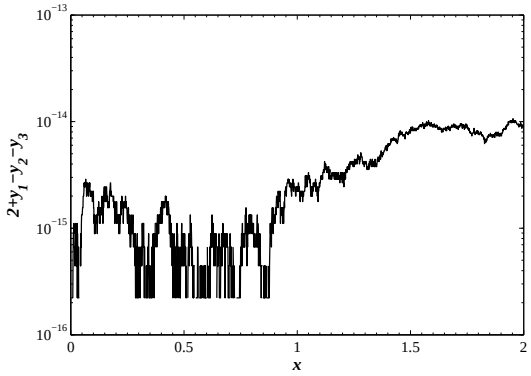

Figure 4: Variation of $2 + y_1 - y_2 - y_3$ versus x which y_1, y_2 and y_3 are the numerical solutions obtained by the method of order 6

The differential equations in Problem S2 satisfy a linear conservation law

$$2 + y_1(x) - y_2(x) - y_3(x) = 0, \tag{9}$$

for all x. In Figure 3 and Figure 4, we have plotted the graph of $2+y_1-y_2-y_3$ versus x. We observe that for both methods of orders five and six equation (9) for the obtained numerical solutions holds approximately with high accuracy which demonstrate the accuracy of the applied methods.

5 Conclusion

For methods of higher orders ($p \geq 5$) with $p = q = r = s$, it is no longer possible to solve the nonlinear systems of equations for satisfying RKS property by symbolic manipulation packages [5]. It seems that this difficulty does not appear for methods with fewer stages. In this paper we constructed RKS methods of orders $p = 5$ and $p = 6$ with $r = s = 3$.

References

1. Abdi, A. and Butcher, J. C. *Order bounds for second derivative approcimations*, BIT, 52 (2012) 273–281.

2. Abdi, A. and Butcher, J. C. *Applications of order arrows*, Appl. Numer. Math., 62 (2012) 556–566.

3. Abdi, A. and Hojjati, G. *An extension of general linear methods*, Numer. Algor., 57 (2011) 149–167.

4. Abdi, A. and Hojjati, G. *Maximal order for second derivative general linear methods with Runge–Kutta stability*, Appl. Numer. Math., 61 (2011) 1046–1058.

5. Abdi, A., Braȁs, M. and Hojjati, G. *On the construction of second derivative diagonally implicit multistage integration methods*, Appl. Numer. Math., 76 (2014) 1–18.

6. Butcher, J. C. *On the convergence of numerical solutions to ordinary differential equations*, Math. Comp., 20 (1966) 1–10.

7. Butcher, J. C. *Numerical Methods for Ordinary Differential Equations*, Wiley, New York, 2008.

8. Butcher, J. C. and Hojjati, G. *Second derivative methods with RK stability*, Numer. Algor., 40 (2005) 415–429.

9. Dahlquist, G. *A special stability problem for linear multistep methods*, BIT, 3 (1963) 27–43.

10. Ezzeddine, A. K., Hojjati, G. and Abdi, A. *Sequential second derivative general linear methods for stiff systems*, Bull. Iranian Math. Soc., 40 (2014) 83–100.

11. Hojjati, G., Abdi, A., Mirzaee, F. and Bimesl, S. *Numerical solution of stiff systems of differential equations arising from chemical reactions*, Iran. J. Numer. Anal. Optim., 4 (2014) 25–39.

12. Jackiewicz, Z. *General linear methods for ordinary differential equations*, Wiley, New Jersey, 2009.

Iranian Journal of Numerical Analysis and Optimization
Vol. 5, No. 2, (2015), pp 11-28

The block LSMR algorithm for solving linear systems with multiple right-hand sides

F. Toutounian* and M. Mojarrab

Abstract

LSMR (Least Squares Minimal Residual) is an iterative method for the solution of the linear system of equations and least-squares problems. This paper presents a block version of the LSMR algorithm for solving linear systems with multiple right-hand sides. The new algorithm is based on the block bidiagonalization and derived by minimizing the Frobenius norm of the residual matrix of normal equations. In addition, the convergence of the proposed algorithm is discussed. In practice, it is also observed that the Frobenius norm of the residual matrix decreases monotonically. Finally, numerical experiments from real applications are employed to verify the effectiveness of the presented method.

Keywords: LSMR method; Bidiagonalization; Block methods; Iterative methods; Multiple right-hand sides.

1 Introduction

This paper is concerned with the solution of linear system of the form

$$AX = B, \quad A \in \mathbb{R}^{n\times n}, \quad B \in \mathbb{R}^{n\times s}, \quad s \ll n. \tag{1}$$

If A is large and sparse or sometimes not readily available, then iterative solvers may become the only choice. These solvers are categorized to the following three classes:

*Corresponding author
Received 23 April 2014; revised 6 August 2014; accepted 18 March 2015
F. Toutounian
Department of Applied Mathematics, Faculty of Mathematical Sciences, Ferdowsi University of Mashhad, Iran.
The Center of Excellence on Modelling and Control Systems, Ferdowsi University of Mashhad, Iran. e-mail: toutouni@math.um.ac.ir

M. Mojarrab
Department of Applied Mathematics, Faculty of Mathematical Sciences, Ferdowsi University of Mashhad, Iran.
Department of Mathematics, University of Sistan and Baluchestan, Zahedan, Iran. email: mojarrab@stu-mail.um.ac.ir

The first class is the global methods. The term global is due to Saad [34] and has been further expanded by Jbilou et al. [21] with the global FOM and GMRES algorithms for matrix equations. These methods are based on the use of a global projection process onto a matrix Krylov subspace. References on this class include [2, 7, 8, 12, 13, 13, 21–23, 25–27, 32, 33].

The second class is the seed methods. The main idea of this kind of methods is briefed below. We first select a single system as the seed system and generate the corresponding Krylov subspace. Then we project all the residuals of the other linear systems onto the same Krylov subspace to find new approximate solutions as initial approximations. See [3, 5, 7, 18, 20, 30, 35] for details.

The last class is the block methods which are more suitable for dense systems with preconditioner. The first block solvers are the block conjugate gradient (Bl-CG) algorithm and the block biconjugate gradient (Bl-BCG) algorithm proposed in [28]. Variable Bl-CG algorithms for symmetric positive definite problems are implemented on parallel computers [19, 29]. If the matrix is symmetric, an adaptive block Lanczos algorithm and a block version of Minres method are devised in [17]. For nonsymmetric problems, the Bl-BCG algorithm [6, 28], the block generalized minimal residual (Bl-GMRES) algorithm [1, 1, 4, 7, 9–11, 36, 37], the block quasi minimum residual (Bl-QMR) algorithm [14], the block BiCGStab (Bl-BICGSTAB) algorithm [31], the block Lanczos method [34] and the block least squares (Bl-LSQR) algorithm [15] have been developed.

In this paper, we present a block version of LSMR algorithm [4] for solving the problem (1). Our algorithm is based on the block bidiagonalization [9]. We construct a simple recurrence formula for generating the sequences of approximations $\{X_k\}$ such that the Frobenius norm of $A^T R_k$ decreases monotonically, where $R_k = B - AX_k$.

Throughout this paper, we use the following notations. For two $n \times s$ matrices X and Y, we define the following inner product: $\langle X, Y\rangle = \mathrm{tr}(X^T Y)$, where $\mathrm{tr}(Z)$ denoted the trace of the square matrix Z. The associated norm is the Frobenius norm denoted by $\|\cdot\|_F$. We will use the notation $\langle\cdot,\cdot\rangle_2$ for the usual inner product in $\mathbb{R}^n$ and the associated norm denoted by $\|\cdot\|_2$. Finally, 0_s and I_s will denote the zero and the identity matrices in $\mathbb{R}^{s\times s}$.

The remainder of this paper is organized as follows. In Section 2, we give a sketch of the LSMR method and its properties. In Section 3, we present the block version of the LSMR algorithm. In Section 4, the convergence of the presented algorithm is considered. In Section 5, some numerical experiments on test matrices from the University of Florida Sparse Matrix Collection(Davis [7]) are presented to show the efficiency of the method. Finally, we make some concluding remarks in Section 6.

2 The LSMR algorithm

In this section, we present a brief of the LSMR algorithm [4], which is an iterative method for solving real linear system of the form

$$Ax = b,$$

where A is a matrix of order n and $x, b \in \mathbb{R}^n$.

LSMR algorithm uses an algorithm of Golub and Kahan [10], which is stated as procedure Bidiag 1 in [32] to reduce the augmented matrix $[b\ A]$ to the upper-diagonal form $[\beta_1 e_1\ B_k]$, where e_1 denotes the first column of the identity matrix. The procedure Bidiag 1 can be described as follows.

Bidiag 1 (Starting vector b; reduction to lower bidiagonal form)

$$\left.\begin{array}{l} \beta_1 u_1 = b, \quad \alpha_1 v_1 = A^T u_1, \\ \beta_{i+1} u_{i+1} = A v_i - \alpha_i u_i, \\ \alpha_{i+1} v_{i+1} = A^T u_{i+1} - \beta_{i+1} v_i, \end{array}\right\} \qquad i = 1, 2, \ldots \tag{2}$$

The scalars $\alpha_i \geq 0$ and $\beta_i \geq 0$ are chosen so that $\|u_i\|_2 = \|v_i\|_2 = 1$. With the definitions

$$U_k \equiv [u_1, u_2, \ \ldots\ u_k], \quad V_k \equiv [v_1, v_2, \ \ldots\ , v_k], \quad B_k \equiv \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \ddots & \ddots & \\ & & \beta_k & \alpha_k \\ & & & \beta_{k+1} \end{bmatrix},$$

$$L_{k+1} = [B_k \ \ \alpha_{k+1} e_{k+1}], \quad V_{k+1} = \begin{bmatrix} V_k & v_{k+1} \end{bmatrix},$$

the recurrence relations (2) may be rewritten as

$$\begin{aligned} &U_{k+1}(\beta_1 e_1) = b, \\ &AV_k = U_{k+1} B_k, \\ &A^T U_{k+1} = V_k B_k^T + \alpha_{k+1} v_{k+1} e_{k+1}^T = V_{k+1} L_{k+1}^T. \\ &A^T A V_k = A^T U_{k+1} B_k = V_{k+1} L_{k+1}^T B_k = V_{k+1} \begin{bmatrix} B_k^T \\ \alpha_{k+1} e_{k+1}^T \end{bmatrix} B_k, \\ &\qquad\qquad = V_{k+1} \begin{bmatrix} B_k^T B_k \\ \alpha_{k+1} \beta_{k+1} e_k^T \end{bmatrix}. \end{aligned}$$

This is equivalent to what would be generated by the symmetric Lanczos process with matrix $A^T A$ and starting vector $A^T b$. As we observe the procedure Bidiag1 will be stop if $Av_i - \alpha_i u_i = 0$ or $A^T u_{i+1} - \beta_{i+1} v_i = 0$, for some i. In exact arithmetic, we have $U_{k+1}^T U_{k+1} = I$ and $V_k^T V_k = I$, where I is the identity matrix.

Hence using procedure Bidiag 1 the LSMR method constructs an approximation solution of the form $x_k = V_k y_k$ which solves the least-squares problem $\min_{y_k} \|A^T r_k\|$, where $r_k = b - Ax_k$. The main steps of the LSMR algorithm can be summarized as follows.

Algorithm 1 LSMR algorithm

Set $\beta_1 u_1 = b$, $\alpha_1 v_1 = A^T u_1$, $\overline{\alpha}_1 = \alpha_1$, $\overline{\zeta}_1 = \alpha_1\beta_1$, $\rho_0 = 1$, $\overline{\rho}_0 = 1$, $\overline{c}_0 = 1$, $\overline{s}_0 = 0$, $h_1 = v_1$, $\overline{\mathrm{h}}_0 = 0$, $x_0 = 0$,
For $k = 1, 2, \ldots$, until convergence Do:
$\beta_{k+1}u_{k+1} = Av_k - \alpha_k u_k$,
$\alpha_{k+1}v_{k+1} = A^T u_{k+1} - \beta_{k+1}v_k$,
$\rho_k = (\overline{\alpha}_k^2 + \beta_{k+1}^2)^{\frac{1}{2}}$,
$c_k = \overline{\alpha}_k/\rho_k$,
$s_k = \beta_{k+1}/\rho_k$,
$\theta_{k+1} = s_k\alpha_{k+1}$,
$\overline{\alpha}_{k+1} = c_k\alpha_{k+1}$,
$\overline{\theta}_k = \overline{s}_{k-1}\rho_k$,
$\overline{\rho}_k = ((\overline{c}_{k-1}\rho_k)^2 + \theta_{k+1}^2)^{\frac{1}{2}}$,
$\overline{c}_k = \overline{c}_{k-1}\rho_k/\overline{\rho}_k$,
$\overline{s}_k = \theta_{k+1}/\overline{\rho}_k$,
$\zeta_k = \overline{c}_k\overline{\zeta}_k$,
$\overline{\zeta}_{k+1} = -\overline{s}_k\overline{\zeta}_k$,
$\overline{h}_k = h_k - (\overline{\theta}_k\rho_k/(\rho_{k-1}\overline{\rho}_{k-1}))\overline{h}_{k-1}$,
$x_k = x_{k-1} + (\zeta_k/(\rho_k\overline{\rho}_k))\overline{h}_k$,
$h_{k+1} = v_{k+1} - (\theta_{k+1}/\rho_k)h_k$,
If $|\overline{\zeta}_{k+1}|$ is small enough then stop,
End Do.

More details about the LSMR algorithm can be found in [4].

3 The block LSMR method

We first recall the block Bidiag 1 algorithm [9]. This algorithm is the basis for our block LSMR method.

The block Bidiag 1 procedure constructs the sets of the $n \times s$ block vectors $V_1, V_2, \ldots$ and $U_1, U_2, \ldots$ such that $V_i^T V_j = 0_s$, $U_i^T U_j = 0_s$, for $i \neq j$, and $V_i^T V_i = I_s$, $U_i^T U_i = I_s$; and they form the orthonormal basis of $\mathbb{R}^{n\times ks}$.

Block Bidiag 1 (Starting matrix B; reduction to block lower bidiagonal form)

$$\left.\begin{array}{l} U_1B_1 = B, \quad V_1A_1 = A^TU_1, \\ U_{i+1}B_{i+1} = AV_i - U_iA_i^T, \\ V_{i+1}A_{i+1} = A^TU_{i+1} - V_iB_{i+1}^T, \end{array}\right\} \qquad i = 1, 2, \ldots, k, \tag{3}$$

where $U_i, V_i \in \mathbb{R}^{n\times s}; B_i, A_i \in \mathbb{R}^{s\times s}$, and $U_1B_1, V_1A_1, U_{i+1}B_{i+1}$, $V_{i+1}A_{i+1}$ are thin QR decompositions of the matrices $B, A^TU_1, AV_i - U_iA_i^T$, $A^TU_{i+1} - V_iB_{i+1}^T$, respectively. With the definitions

$$\overline{U}_k \equiv [U_1,\ U_2,\ \ldots\ ,\ U_k]\,,\ \overline{V}_k \equiv [V_1,\ V_2,\ \ldots\ ,\ V_k]\,,\ T_k \equiv \begin{bmatrix} A_1^T & & & \\ B_2 & A_2^T & & \\ & \ddots & \ddots & \\ & & B_k & A_k^T \\ & & & B_{k+1} \end{bmatrix},$$

the recurrence relations (3) may be rewritten as:

$$\begin{aligned} &\overline{U}_{k+1}E_1B_1 = B, \\ &A\overline{V}_k = \overline{U}_{k+1}T_k, \\ &A^T\overline{U}_{k+1} = \overline{V}_kT_k^T + V_{k+1}A_{k+1}E_{k+1}^T, \end{aligned}$$

where E_i is the $(k+1)s \times s$ matrix which is zero except for the rows i to $i+s$, which are the $s \times s$ identity matrix. We have also $\overline{V}_k^T\overline{V}_k = I_{ks}$ and $\overline{U}_{k+1}^T\overline{U}_{k+1} = I_{(k+1)s}$, where I_l is the $l \times l$ identity matrix. We define

$$\overline{L}_{k+1} \equiv \left[\, T_k \ E_{k+1}A_{k+1}^T \,\right],$$

then

$$\begin{aligned} &A^T\overline{U}_{k+1} = \overline{V}_{k+1}\overline{L}_{k+1}^T, \\ &A^TA\overline{V}_k = A^T\overline{U}_{k+1}T_k = \overline{V}_{k+1}\overline{L}_{k+1}^TT_k = \overline{V}_{k+1}\begin{bmatrix} T_k^T \\ A_{k+1}E_{k+1}^T \end{bmatrix}T_k \\ &\qquad\qquad = \overline{V}_{k+1}\begin{bmatrix} T_k^TT_k \\ A_{k+1}E_{k+1}^TT_k \end{bmatrix}. \end{aligned} \tag{4}$$

At iteration k we seek an approximate solution X_k of the form

$$X_k = \overline{V}_kY_k, \tag{5}$$

where Y_k is an $ks \times s$ matrix. Let $\overline{B}_k \equiv A_kB_k$ for all k. Since

$$\begin{aligned} A^TR_k &= A^TB - A^TAX_k \\ &= V_1A_1B_1 - A^TA\overline{V}_kY_k, \end{aligned}$$

we have

$$A^T R_k = V_1 \overline{B}_1 - \overline{V}_{k+1} \begin{bmatrix} T_k^T T_k \\ A_{k+1} E_{k+1}^T T_k \end{bmatrix} Y_k$$
$$= \overline{V}_{k+1} (E_1 \overline{B}_1 - \begin{bmatrix} T_k^T T_k \\ \overline{B}_{k+1} \overline{E}_k^T \end{bmatrix} Y_k), \tag{6}$$

where $\overline{E}_k$ is the $ks \times s$ matrix, which is zero except for kth s rows, which are the $s \times s$ identity matrix.

In the block LSMR algorithm, we would like to choose $Y_k \in \mathbb{R}^{ks \times s}$ which minimizes the Frobenius norm of $A^T R_k$. From (6), $A^T R_k$ can be written as

$$A^T R_k = \overline{V}_{k+1} \begin{bmatrix} \widetilde{E}_1 \overline{B}_1 - T_k^T T_k Y_k \\ -\overline{B}_{k+1} \overline{E}_k^T Y_k \end{bmatrix}, \tag{7}$$

where $\widetilde{E}_1$ is the matrix obtained from E_1 by deleting its last block row. But since the columns of the matrix $\overline{V}_{k+1}$ are orthonormal, it follows that:

$$\|A^T R_k\|_F^2 = \left\| \begin{bmatrix} \widetilde{E}_1 \overline{B}_1 - T_k^T T_k Y_k \\ -\overline{B}_{k+1} \overline{E}_k^T Y_k \end{bmatrix} \right\|_F^2 = \|\widetilde{E}_1 \overline{B}_1 - T_k^T T_k Y_k\|_F^2 + \|\overline{B}_{k+1} \overline{E}_k^T Y_k\|_F^2. \tag{8}$$

We now define the linear operators χ_k and ψ_k as follows.

For $Y \in \mathbb{R}^{ks \times s}$

$\chi_k(Y) = T_k^T T_k Y,$

and

$\psi_k(Y) = \overline{B}_{k+1} \overline{E}_k^T Y.$

Then the relation (8) can be expressed as

$$\|A^T R_k\|_F^2 = \|\chi_k(Y_k) - \widetilde{E}_1 \overline{B}_1\|_F^2 + \|\psi_k(Y_k)\|_F^2. \tag{9}$$

Therefore, Y_k minimizes the Frobenius norm of the quantity $A^T R_k$ if and only if it satisfies the following linear matrix equation

$$\chi_k^T(\chi_k(Y_k) - \widetilde{E}_1 \overline{B}_1) + \psi_k^T(\psi_k(Y_k)) = 0_s, \tag{10}$$

where the linear operators χ_k^T and ψ_k^T are the transpose of the operators χ_k and ψ_k, respectively. Therefore, (10) is also written as the following

$$(T_k^T T_k)^T (T_k^T T_k Y_k - \widetilde{E}_1 \overline{B}_1) + (\overline{B}_{k+1} \overline{E}_k^T)^T (\overline{B}_{k+1} \overline{E}_k^T Y_k) = 0_s. \tag{11}$$

Hence, Y_k is given by

$$Y_k = \widehat{T}_k^{-1} F_k,$$

where

$$\widehat{T}_k = (T_k^T T_k)^2 + \overline{E}_k \overline{B}_{k+1}^T \overline{B}_{k+1} \overline{E}_k^T, \qquad F_k = T_k^T T_k \widetilde{E}_1 \overline{B}_1. \tag{12}$$

We define the matrix $\overline{T}_k$ as follows:

$$\overline{T}_k = \begin{bmatrix} T_k^T T_k \\ \overline{B}_{k+1} \overline{E}_k^T \end{bmatrix} = \begin{bmatrix} \overline{A}_1 & \overline{B}_2^T & & \\ \overline{B}_2 & \overline{A}_2 & \ddots & \\ & \ddots & \ddots & \overline{B}_k^T \\ & & \overline{B}_k & \overline{A}_k \\ & & & \overline{B}_{k+1} \end{bmatrix},$$

where $\overline{A}_i = A_i A_i^T + B_{i+1}^T B_{i+1}$, for $i = 1, 2, \ldots, k$. Therefore

$$\widehat{T}_k = \overline{T}_k^T \overline{T}_k, \quad F_k = [(\overline{A}_1 \overline{B}_1)^T (\overline{B}_2 \overline{B}_1)^T 0_s \ldots 0_s]^T, \tag{13}$$

and the approximate solution of the system (1) is given by

$$X_k = \overline{V}_k \widehat{T}_k^{-1} F_k.$$

Suppose that using the QR decomposition [11], we obtain a unitary matrix $\overline{Q}_k$ such that

$$\overline{T}_k = \overline{Q}_k \begin{bmatrix} \overline{R}_k \\ 0_{s \times ks} \end{bmatrix}, \qquad \overline{R}_k = \begin{bmatrix} \overline{\alpha}_1 & \overline{\beta}_2 & \overline{\theta}_3 & & & \\ & \overline{\alpha}_2 & \overline{\beta}_3 & \overline{\theta}_4 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \overline{\alpha}_{k-2} & \overline{\beta}_{k-1} & \overline{\theta}_k \\ & & & & \overline{\alpha}_{k-1} & \overline{\beta}_k \\ & & & & & \overline{\alpha}_k \end{bmatrix}, \tag{14}$$

where $\overline{R}_k$ is upper triangular as shown and $\overline{\alpha}_i$, $\overline{\beta}_i$, $\overline{\theta}_i$ are the $s \times s$ matrices. So,

$$X_k = \overline{V}_k (\overline{R}_k^T \overline{R}_k)^{-1} F_k.$$

By setting

$$\overline{P}_k = \overline{V}_k \overline{R}_k^{-1} \equiv \left[P_1 \; P_2 \; \ldots \; P_k \right],$$

and

$$\overline{F}_k = \overline{R}_k^{-T} F_k \equiv \left[\varphi_1^T \; \varphi_2^T \; \ldots \; \varphi_k^T \right]^T,$$

we have

$$\begin{aligned} P_k &= (V_k - P_{k-2} \overline{\theta}_k - P_{k-1} \overline{\beta}_k) \overline{\alpha}_k^{-1}, \\ X_k &= X_{k-1} + P_k \varphi_k. \end{aligned} \tag{15}$$

From (15) the residual R_k is given by

$$R_k = R_{k-1} - AP_k\varphi_k, \tag{16}$$

where AP_k can be computed from the previous AP_k's and AV_k by the simple update

$$AP_k = (AV_k - AP_{k-2}\overline{\theta}_k - AP_{k-1}\overline{\beta}_k)\overline{\alpha}_k^{-1}.$$

In addition, as [4], we show that the $\|R_k\|_F$ can be estimated by a simple formula. By transforming T_k to block upper-bidiagonal form using a QR factorization: $\begin{bmatrix} \widehat{R}_k \\ 0 \end{bmatrix} = \widehat{Q}_{k+1}T_k$ with $\widehat{Q}_{k+1} = \widehat{P}_k \dots \widehat{P}_1$, we have

$$\begin{aligned} R_k &= B - AX_k \\ &= U_1B_1 - A\overline{V}_kY_k \\ &= \overline{U}_{k+1}(E_1B_1 - T_kY_k) \\ &= \check{U}_{k+1}\widehat{Q}_{k+1}^T(\widehat{Q}_{k+1}E_1B_1 - \begin{bmatrix} \widehat{R}_k \\ 0 \end{bmatrix} Y_k). \end{aligned}$$

Since the columns of the matrices $\widehat{Q}_{k+1}$ and $\overline{U}_{k+1}$ are orthonormal, we have

$$\|R_k\|_F = \|\widehat{Q}_{k+1}E_1B_1 - \begin{bmatrix} \widehat{R}_k \\ 0 \end{bmatrix} Y_k\|_F. \tag{17}$$

With definitions

$$\widehat{Q}_{k+1}E_1B_1 = [\widetilde{\beta}_1^T \ \dots \ \widetilde{\beta}_{k-1}^T \ \dot{\beta}_k^T \ \ddot{\beta}_{k+1}^T]^T, \quad \widehat{R}_kY = [\widetilde{\tau}_1^T \ \dots \ \widetilde{\tau}_{k-1}^T \ \dot{\tau}_k^T]^T, \tag{18}$$

the following Lemma shows that we can estimate $\|R_k\|_F$ from just the last two blocks of $\widehat{Q}_{k+1}E_1B_1$ and the last block of $\widehat{R}_kY_k$.

Lemma 1. *In* (17) *and* (18), $\widetilde{\beta}_i = \widetilde{\tau}_i$ *for* $i = 1, 2, \dots, k-1$.

Proof. The proof is similar to that of Lemma 3.1 in [4] (see [28]). □

For the Frobenius norm of A^TR_k, by using Theorem 1 (in section 4), we can also obtain the following simple formula:

$$\|A^TR_k\|_F^2 = \|A^TR_{k-1}\|_F^2 - \|\varphi_k\|_F^2, \qquad \text{with} \quad \|A^TR_0\|_F = \|\overline{B}_1\|_F = \|\varphi_0\|_F.$$

Now we can summarize the above descriptions as the following algorithm.

Algorithm 2 Algorithm (Bl-LSMR)

Set $X_0 = 0_{n\times s}$,
Set $\overline{a}_0 = 0_s$, $\overline{b}_{-1} = 0_s$, $\overline{b}_0 = I_s$, $\overline{c}_0 = 0_s$, $\overline{d}_{-1} = 0_s$, $\overline{d}_0 = I_s$,
Set $P_{-1} = P_0 = 0_{n\times s}$,
Compute $U_1B_1 = B$, $V_1A_1 = A^TU_1$ (QR decomposition of B and A^TU_1),
Set $\overline{B}_1 = A_1B_1$,
Set $\varphi_{-1} = 0_s$, $\varphi_0 = -\overline{B}_1$,
Set $\|A^TR_0\|_F = \|\varphi_0\|_F$,
For $k = 1, 2, \ldots$, until convergence Do:
$\overline{W}_k = AV_k - U_kA_k^T$,
$U_{k+1}B_{k+1} = \overline{W}_k$ (QR decomposition of $\overline{W}_k$),
$\overline{A}_k = A_kA_k^T + B_{k+1}^TB_{k+1}$,
$\overline{S}_k = A^TU_{k+1} - V_kB_{k+1}^T$,
$V_{k+1}A_{k+1} = \overline{S}_k$ (QR decomposition of $\overline{S}_k$),
$\overline{B}_{k+1} = A_{k+1}B_{k+1}$,
$\dot{\beta}_k = \overline{d}_{k-2}\overline{B}_k^T$,
$\dot{\alpha}_k = \overline{c}_{k-1}\dot{\beta}_k + \overline{d}_{k-1}\overline{A}_k$,
$\overline{\beta}_k = \overline{a}_{k-1}\dot{\beta}_k + \overline{b}_{k-1}\overline{A}_k$,
$\overline{\theta}_k = \overline{b}_{k-2}\overline{B}_k^T$,
Compute an unitary matrix $\overline{Q}(\overline{a}_k, \overline{b}_k, \overline{c}_k, \overline{d}_k)$ such that
$\begin{bmatrix} \overline{a}_k & \overline{b}_k \\ \overline{c}_k & \overline{d}_k \end{bmatrix} \begin{bmatrix} \dot{\alpha}_k \\ \overline{B}_{k+1} \end{bmatrix} = \begin{bmatrix} \overline{\alpha}_k \\ 0 \end{bmatrix}$,
$\varphi_k = -\overline{\alpha}_k^{-T}(\overline{\theta}_k^T\varphi_{k-2} + \overline{\beta}_k^T\varphi_{k-1})$,
$P_k = (V_k - P_{k-2}\overline{\theta}_k - P_{k-1}\overline{\beta}_k)\overline{\alpha}_k^{-1}$,
$X_k = X_{k-1} + P_k\varphi_k$,
$R_k = R_{k-1} - AP_k\varphi_k$,
$\|A^TR_k\|_F^2 = \|A^TR_{k-1}\|_F^2 - \|\varphi_k\|_F^2$,
If $\|A^TR_k\|_F$ is small enough then stop,
End Do.

The Bl-LSMR algorithm will be break down at step k, if $\overline{\alpha}_k$ is singular. This happens when the matrix $\begin{bmatrix} \dot{\alpha}_k \\ \overline{B}_{k+1} \end{bmatrix}$ is not full rank. So the Bl-LSMR algorithm will not break down at step k, if $\overline{B}_{k+1}$ is nonsingular. We will not treat the problem of breakdown in this paper and we also assume that the matrices $\overline{B}_k$'s produced by the Bl-LSMR algorithm are nonsingular.

We mention that, we can use the Bl-LSMR algorithm for computing a matrix solution X to the problem

$$\text{minimize}\|\mathrm{AX} - \mathrm{B}\|_{\mathrm{F}}, \quad \mathrm{A} \in \mathbb{R}^{\mathrm{m}\times\mathrm{n}}, \quad \mathrm{B} \in \mathbb{R}^{\mathrm{m}\times\mathrm{s}}, \quad \mathrm{s} \ll \min\{\mathrm{m}, \mathrm{n}\},$$

where $m \geq n$ or $m \leq n$. In Section 5, we present the results of the Bl-LSMR algorithm for this kind of problems.

4 The convergence of the Bl-LSMR algorithm

In this section, we aim at studying the convergence behavior of the Bl-LSMR method. We first give the following lemmas.

Lemma 2. *Let $P_i, i = 1, 2, \ldots, k$, be the $n \times s$ auxiliary matrices produced by the Bl-LSMR algorithm and R_k be the residual matrix associated with the approximate solution X_k of the matrix equation*(1)*. Then, we have*

$$(A^T A P_k)^T A^T R_k = 0_s.$$

Proof. Using $\overline{P}_k = \overline{V}_k \overline{R}_k^{-1}$ and equation(4), we have

$$\begin{aligned} A^T A P_k &= A^T A \overline{P}_k \overline{E}_k \\ &= A^T A \overline{V}_k \overline{R}_k^{-1} \overline{E}_k \\ &= \overline{V}_{k+1} \begin{bmatrix} T_k^T T_k \\ \overline{B}_{k+1} \overline{E}_k^T \end{bmatrix} \overline{R}_k^{-1} \overline{E}_k. \end{aligned} \tag{19}$$

From (19), and (7), we have

$$\begin{aligned} (A^T A P_k)^T (A^T R_k) &= \overline{E}_k^T \overline{R}_k^{-T} \left[T_k^T T_k, (\overline{B}_{k+1} \overline{E}_k^T)^T \right] \overline{V}_{k+1}^T \overline{V}_{k+1} \begin{bmatrix} \widetilde{E}_1 \overline{B}_1 - T_k^T T_k Y_k \\ -\overline{B}_{k+1} \overline{E}_k^T Y_k \end{bmatrix} \\ &= \overline{E}_k^T \overline{R}_k^{-T} (T_k^T T_k (\widetilde{E}_1 \overline{B}_1 - T_k^T T_k Y_k) - (\overline{B}_{k+1} \overline{E}_k^T)^T \overline{B}_{k+1} \overline{E}_k^T Y_k) \\ &= 0_s. \qquad \text{(from (11))} \end{aligned}$$

We note that $\overline{V}_{k+1}$ is orthonormal, thus $\overline{V}_{k+1}^T \overline{V}_{k+1} = I_{(k+1)s}$. □

Lemma 3. *Let $P_i, i = 1, 2, \ldots, k$, be the $n \times s$ auxiliary matrices produced by the Bl-LSMR algorithm. Then we have the following property*

$$P_i^T A^T A A^T A P_i = I_s.$$

Proof. Using (19), (12), (13) and (14), we have

$$
\begin{aligned}
(A^TAP_i)^T(A^TAP_i) &= (\overline{V}_{i+1}\begin{bmatrix} T_i^TT_i \\ \overline{B}_{i+1}\overline{E}_i^T \end{bmatrix}\overline{R}_i^{-1}\overline{E}_i)^T(\overline{V}_{i+1}\begin{bmatrix} T_i^TT_i \\ \overline{B}_{i+1}\overline{E}_i^T \end{bmatrix}\overline{R}_i^{-1}\overline{E}_i) \\
&= \overline{E}_i^T\overline{R}_i^{-T}\begin{bmatrix} T_i^TT_i & \overline{B}_{i+1}^T\overline{E}_i \end{bmatrix}\begin{bmatrix} T_i^TT_i \\ \overline{B}_{i+1}\overline{E}_i^T \end{bmatrix}\overline{R}_i^{-1}\overline{E}_i \\
&= \overline{E}_i^T\overline{R}_i^{-T}\overline{T}_i^T\overline{T}_i\overline{R}_i^{-T}\overline{E}_i \\
&= \overline{E}_i^T\overline{R}_i^{-T}\begin{bmatrix} \overline{R}_i^T & 0_{ks\times s} \end{bmatrix}\overline{Q}_i^T\overline{Q}_i\begin{bmatrix} \overline{R}_i \\ 0_{s\times ks} \end{bmatrix}\overline{R}_i^{-1}\overline{E}_i \\
&= \overline{E}_i^T\begin{bmatrix} I_{ks} & 0_{ks\times s} \end{bmatrix}\begin{bmatrix} I_{ks} \\ 0_{s\times ks} \end{bmatrix}\overline{E}_i \\
&= \overline{E}_i^T\overline{E}_i = I_s.
\end{aligned}
$$

□

Theorem 1. *Let X_k be the approximate solution of* (1)*, obtained from the Bl-LSMR algorithm. Then*

$$\|A^TR_k\|_F \leq \|A^TR_{k-1}\|_F,$$

where $R_k = B - AX_k$.

Proof. From(16), we have

$$A^TR_{k-1} = A^TR_k + A^TAP_k\varphi_k.$$

Using Lemma 2, since A^TR_k and A^TAP_k are orthogonal, we have

$$\|A^TR_{k-1}\|_F^2 = \|A^TR_k\|_F^2 + \|A^TAP_k\varphi_k\|_F^2.$$

Thus

$$\|A^TR_k\|_F^2 = \|A^TR_{k-1}\|_F^2 - \|A^TAP_k\varphi_k\|_F^2.$$

Using Lemma 3, we have

$$
\begin{aligned}
&\|A^TR_k\|_F^2 = \|A^TR_{k-1}\|_F^2 - \|\varphi_k\|_F^2, \\
&\|A^TR_k\|_F \leq \|A^TR_{k-1}\|_F.
\end{aligned}
$$

□

Theorem 1 is helpful in showing that if $\|\varphi_k\|_F$ is not very small in each iteration of the Bl-LSMR algorithm, then the Bl-LSMR algorithm will be stopped after a finite number of iterations. Otherwise, it is possible to occur stagnation. In this case, we can apply a reliable preconditioner for the block linear system of equations (1).

5 Numerical examples

In this section, we consider the system $AX = B$, where $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{m\times s}$, $X \in \mathbb{R}^{n\times s}$, and we present numerical results for several matrices taken from the University of Florida Sparse Matrix Collection (Davis [7]). These matrices with their properties are shown in Table 1. Our implementation is done on MATLAB version 07 on a PC machine with 4 GB RAM. Moreover, for the initial guess $X_0 = 0_{n\times s}$ and $B = \text{rand}(m, s)$, where the function rand creates an $m \times s$ random matrix with the coefficients uniformly distributed in $[0, 1]$. The stopping criteria is set to $\|A^T R_k\|_F/\|R_k\|_F \leq 10^{-10} \times \|A\|_F$.

Diagonal scaling was applied to the columns of $[A,\ B]$ to give a scaled problem $AX = B$, in which the columns of $[A,\ B]$ have unit 2-norm. By scaling, the number of iterations of Bl-LSMR for convergence reduced satisfactorily.

In Table 2, we give the ratio $t(s)/t(1)$, for $s = 5, 10, 20$, and 30, where $t(s)$ is the CPU time for Bl-LSMR algorithm and $t(1)$ is the CPU time obtained when applying LSMR for one right-hand side linear system. Note that the time obtained by LSMR for one right-hand side depends on which right-hand was used. So, $t(1)$ is the average of the times needed for the s right-hand sides using LSMR. The results of Table 2 show that the Bl-LSMR algorithm is effective and less expensive than the LSMR algorithm, because the indicator $t(s)/t(1)$ is less than s.

To show that the Frobenius norm of residual matrix decreases monotonically, we display the convergence history in Figure 1 for the systems corresponding to the matrices of Table 2 and Bl-LSMR algorithm. In this figure, the vertical axis and horizontal axis are the logarithm in base 10 of the Frobenius norm of residual matrix and the number of iterations to convergence, respectively. We observe that for all matrices the Frobenius norm of residual matrix decreases monotonically.

We display the convergence history of Bl-LSMR and Bl-LSQR in Figure 2 for the system corresponding to the matrix LPnetlib/lp_pilot. Figure 3 (left and right) shows both solvers reducing $\|A^T R_k\|_F/\|R_k\|_F$ and $\|R_k\|_F$ monotonically and similarly.

Table 1: Test problems information

Matrix\Property	rows	columns	sym	nnz	id	Discipline
Hamm/add32	4960	4960	no	19848	540	Electronic circuit design
Simon/appu	14000	14000	no	1853104	811	Random sparse matrix used in the APP BENCHMARK
HB/fs6801	680	680	no	2184	149	Chemical kinetics
HB/gre115	115	115	no	421	161	Simulation studies in computer systems
HB/gr-30-30	900	900	yes	7744	159	Partial differential equations
LPnetlib/lpadlittle	56	138	no	424	596	Linear programming problem
LPnetlib/lp_maros	846	1966	no	10137	642	Linear programming problem
LPnetlib/lp_pilot	1441	4860	no	44375	654	Linear programming problem
LPnetlib/lp_sc205	205	317	no	665	665	Linear programming problem
Bai/pde2961	2961	2961	no	14585	324	Partial differential equations
Bai/pde900	900	900	no	4380	325	Partial differential equations
Bai/rdb3200l	3200	3200	no	18880	1633	Chemical engineering
HB/sherman4	1104	1104	no	3786	245	Oil reservoir modeling

Table 2: Effectiveness of Bl-LSMR algorithm measured $t(s)/t(1)$

Matrix\s	5	10	20	30
Hamm/add32	0.47	0.95	3.07	5.39
Simon/appu	1.24	1.89	3.21	5.13
HB/fs6801	0.27	0.38	0.97	1.19
HB/gre115	0.99	0.51	3.41	8.57
HB/gr-30-30	1.55	1.72	2.05	2.53
LPnetlib/lpadlittle	0.37	0.42	1.63	12.54
LPnetlib/lp_maros	2.92	3.75	6.79	12.36
LPnetlib/lp_pilot	2.40	4.95	15.90	22.92
LPnetlib/lp_sc205	0.70	1.30	2.11	4.70
Bai/pde2961	0.33	0.52	0.98	1.14
Bai/pde900	0.49	0.72	1.10	1.47
Bai/rdb3200l	0.30	0.39	0.38	0.76
HB/sherman4	0.37	0.50	0.54	1.03

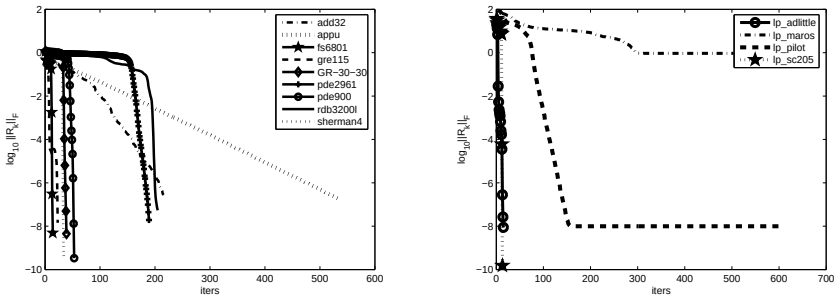

Figure 1: Convergence history of the Bl-LSMR algorithm with s=20

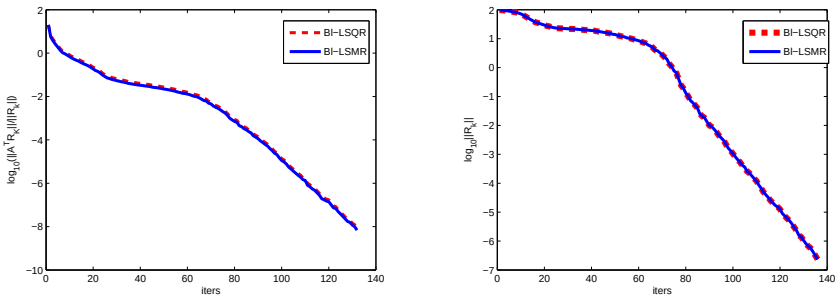

Figure 2: Bl-LSMR and Bl-LSQR solving a linear system AX = B with s = 20: problem LPnetlib/lp_pilot

6 Conclusion

In this paper, we have presented a block version of LSMR algorithm for solving linear systems with multiple right-hand sides. We derived a simple recurrence formula for generating the sequence of approximate solutions $\{X_k\}$ such that the Frobenius norm of the quantity $A^T R_k$ decreases monotonically. In addition, we studied the convergence of the presented method. Besides, we showed that in absence of the break down condition, the presented algorithm always converges. Numerical results have shown that the new algorithm obtains the results which are effective and less expensive than the LSMR algorithm applied to each right-hand side.

Acknowledgements
We would like to thank the referees for their valuable remarks and helpful suggestions.

References

1. Abdel-Rehim, A. M., Morgan, R. B. and Wilcox, W. *Improved seed methods for symmetric positive definite linear equations with multiple right-hand sides*, 2008, Arxiv preprint arXiv:0810.0330v1.
2. Bellalij, M., Jbilou, K. and Sadok, H.*New convergence results on the global GMRES method for diagonalizable matrices*, J. Comput. Appl. Math. 219 (2008) 350-358.
3. Chan,T. F. and Wang, W.*Analysis of projection methods for solving linear systems with multiple right-hand sides*, SIAM J. Sci. Comput. 18 (1997) 1698-1721.
4. Chin-Lung Fong, D. and Saunders, M. *LSMR: An iterative algorithm for sparse least-squares problems*, SIAM J. Sci. Comput. 33 (2011) 2950-2971.
5. Dai, H. *Two algorithms for symmetric linear systems with multiple right-hand sides*, Numer. Math. J. Chin. Univ. (Engl. Ser.) 9 (2000) 91-110.
6. Darnell, D., Morgan, R. B. and Wilcox, W. *Deflated GMRES for systems with multiple shifts and multiple right-hand sides*, Linear Algebra Appl. 429 (2008) 2415-2434.
7. Davis, T. A. *University of Florida Sparse Matrix Collection*, http://www.cise.ufl.edu/research/sparse / matrices.
8. Freund, R. and Malhotra, M. *A Block-QMR algorithm for non-hermitian linear systems with multiple right-hand sides*, Linear Algebra Appl. 254 (1997) 119-157.

9. Golub, G. H., Luk, F. T. and Overton, M. L. *A Block Lanczos method for computing the singular values and corresponding singular vectors of the matrix*, ACM Trans. Math. Software 7 (1981) 149-169.

10. Golub, G. H. and Kahan, W. *Calculating the singular values and pseudoinverse of a matrix*, SIAM J. Numer. Anal, 2 (1965) 205-224.

11. Golub, G. H. and Van Loan, C. F. *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, 1983.

12. Gu, G. and Cao, Z.*A block GMRES method augmented with eigenvectors*, Appl. Math. Comput. 121 (2001) 271-289.

13. Gu, G. and Qian, H. *Skew-symmetric methods for solving nonsymmetric linear systems with multiple right-hand sides*, J. Comput. Appl. Math. 223 (2009) 567-577.

14. Gu, C. and Yang, Z. *Global SCD algorithm for real positive definite linear systems with multiple right-hand sides*, Appl. Math. Comput. 189 (2007) 59-67.

15. Guennouni, A. El., Jbilou, K. and Sadok, H. *A block version of BICGSTAB for linear systems with multiple right-hand sides*, Elec. Trans. Numer. Anal. 16 (2003) 129-142.

16. Guennouni, A. El., Jbilou, K. and Sadok, H. *The block Lanczos method for linear systems with multiple right-hand sides*, Appl. Numer. Math. 51 (2004) 243-256.

17. Gutknecht, M. H. *Block Krylov space methods for linear systems with multiple right-hand sides: an introduction, in: A. H. Siddiqi, I. S. Duff, O. Christensen (Eds.), Modern Mathematical Models, Methods and Algorithms for Real Word Systems*, Anamaya Publishers, New Delhi, India, 2007, 420-447.

18. Haase, G. and Reitzinger, S. *Cache issues of algebraic multigrid methods for linear systems with multiple right-hand sides*, SIAM J. Sci. Comput. 27 (2005) 1-18.

19. Heyouni, M.*The global Hessenberg and global CMRH methods for linear systems with multiple right-hand sides*, Numer. Algorithms 26 (2001) 317-332.

20. Heyouni, M. and Essai, A. *Matrix Krylov subspace methods for linear systems with multiple right-hand sides*, Numer. Algorithms 40 (2005) 137-156.

21. Jbilou, K., Messaoudi, A. and Sadok, H. *Global FOM and GMRES algorithms for matrix equations*, Appl. Numer. Math. 31 (1999) 49-63.

22. Jbilou, K. and Sadok, H. *Global Lanczos-based methods with applications*, Technical Report LMA 42, Universiti du Littoral, Calais, France, 1997.

23. Jbilou, K, Sadok, H. and Tinzefte, A. *Oblique projection methods for linear systems with multiple right-hand sides*, Elec. Trans. Numer. Anal. 20 (2005)119-138.

24. Joly, P. *Resolution de Systems Lineaires Avec Plusieurs Second Members par la Methode du Gradient Conjugue*, Tech. Rep. R-91012, Publications du Laboratire d'Analyse Numerique, Universite Pierre et Marie Curie, Paris, 1991.

25. Karimi, S. and Toutounian, F. *The block least squares method for solving nonsymmetric linear systems with multiple right-hand sides*, Appl. Math. Comput. 177 (2006) 852-862.

26. Lin, Y. *Implicitly restarted global FOM and GMRES for nonsymmetric matrix equations and Sylvester equations*, Appl. Math. Comput. 167 (2005) 1004-1025.

27. Liu, H. and Zhong, B. *Simpler block GMRES for nonsymmetric systems with multiple right-hand sides*, Elec. Trans. Numer. Anal. 30 (2008) 1-9.

28. Mojarrab, M. *The Block least square methods for matrix equations*, PhD thesis, Ferdowsi University of Mashhad, Iran, 2014.

29. Morgan, R. B. *Restarted block-GMRES with deflation of eignvalues*, Appl. Numer. Math. 54 (2005) 222-236.

30. Nikishin, A. and Yeremin, A. *Variable block CG algorithms for solving large sparse symmetric positive definite linear systems on parallel computers I: general iterative scheme*, SIAM J. Matrix Anal. 16 (1995) 1135-1153.

31. ÓLeary, D. *The block conjugate gradient algorithm and related methods*, Linear Algebra Appl. 29 (1980) 293-322.

32. Paige, C. C. and Saunders, M. A. *LSQR: an algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software 8 (1982) 43-71.

33. Robbe, M. and Sadkane, M. *Exact and inexact breakdowns in the block GMRES method*, Linear Algebra Appl. 419 (2006) 265-285.

34. Saad, Y. *Iterative methods for sparse linear systems*, SIAM, 2nd edn, 2003.

35. Saad, Y. *On the Lanczos method for solving symmetric linear systems with several right-hand sides*, Math. Comput. 48 (1987) 651-662.

36. Salkuyeh, D. K. *CG-type algorithms to solve symmetric matrix equations*, Appl. Math. Comput. 172 (2006) 985-999.

37. Simoncini, V. *A stabilized QMR version of block BICG*, SIAM J. Matrix Anal. Appl. 18 (1997) 419-434.

38. Simoncini, V. and Gallopoulos, E. *Convergence properties of block GMRES and matrix polynomials*, Linear Algebra Appl. 247 (1996) 97-119.

39. Simoncini, V. and Gallopoulos, E. *An iterative method for nonsymmetric systems with multiple right-hand sides*, SIAM J. Sci. Comput. 16 (1995) 917-933.

40. Smith, C., Peterson, A. and Mittra, R. *A conjugate gradient algorithm for treatment of multiple incident electromagnetic fields*, IEEE Trans. Antennas Propagation 37 (1989) 1490-1493.

41. Toutounian, F. and Karimi, S. *Global least squares method (Gl-LSQR) for solving general linear systems with several right-hand sides*, Appl. Math. Comput. 178 (2006) 452-460.

42. Van Der Vorst, H. *An iterative solution method for solving f(A) = b, using Krylov subspace information obtained for the symmetric positive definite matrix A*, J. Comput. Appl. 18 (1987) 249-263.

43. Vital, B. *Etude de Quelques Méthodes de Résolution de Probléms Linéaires de Grande Taille sur Multiprocesseur*, Ph.D. Thesis, Univérsité de Rennes, Rennes, France, 1990.

44. Zhang, J. and Dai, H. *Global CGS algorithm for linear systems with multiple right-hand sides*, Numer. Math. J. Chin. Univ. 30 (2008) 390-399 (in Chinese).

45. Zhang, J., Dai, H. and Zhao, J. *A new family of global methods for linear systems with multiple right-hand sides*, J. Comput. Appl. Math. 236 (2011) 1562-1575.

46. Zhang, J., Dai, H. and Zhao, J. *Generalized global conjugate gradient squared algorithm*, Appl. Math. Comput. 216 (2010) 3694-3706.

Iranian Journal of Numerical Analysis and Optimization
Vol. 5, No. 2, (2015), pp 29-43

A practical review of the Adomian decomposition method: computer implementation aspects

A. Molabahrami

Abstract

In this paper, a practical review of the Adomian decomposition method, to extend the procedure to handle the strongly nonlinear problems under the mixed conditions, is given and the convergence of the algorithm is proved. For this respect, a new and simple way to generate the Adomian polynomials, for a general nonlinear function, is proposed. The proposed procedure, provides an explicit formula to calculate the Adomian polynomials of a nonlinear function. The efficiency of the approach will be shown by applying the procedure on several interesting integro-differential problems. The Mathematica programs generating the Adomian polynomials and Adomian solutions based on the procedures in this paper are designed.

Keywords: Adomian decomposition method; Adomian polynomials; Nonlinear integro-differential problems; Series solution; Strongly nonlinear problems; Explicit machine computation and programs.

1 Introduction

To construct series pattern solution for a problem with strong nonlinearity, it is necessary to construct nonlinear terms of the governing equation in the form of a series by using the components of the solution series. To the end, one of the best and suitable way is to use the Adomian polynomials. The so-called Adomian polynomials are used to deduce the recursive relation during the implementation of the Adomian decomposition method (ADM) while solving nonlinear problems. The main aim of the present paper is to provide a simple and new method to handle a strongly nonlinear problem by using ADM in the frame of a symbolic computer program so that by giving linear operator, it generates: initial guess, integral inverse of the linear operator, recursive relation and the terms of solution series automatically. To achieve this purpose, we first propose an explicit formula to calculate the

Received 7 October 2014; revised 20 December 2014; accepted 18 March 2015
A. Molabahrami
Department of Mathematics, Ilam University, PO Box 69315516, Ilam, Iran e-mail: bahrami@iust.ac.ir

Adomian polynomials and the Adomian series of a general nonlinear function and implement the proposed algorithms in Mathematica. In this respect, we first outline the modifications of some definitions as already given in [8]. Let u be a function of the parameter λ, whose Maclaurin series is given by

$$u(\lambda) = \sum_{n=0}^{+\infty} u_n \lambda^n. \tag{1}$$

This series is called the parametric series of u. Let ϕ be a function of the parameter λ, the mth-order parametric derivative of ϕ is

$$\mathbb{D}_m[\phi] = \frac{1}{m!} \frac{\partial^m \phi}{\partial \lambda^m}\bigg|_{\lambda=0}, \tag{2}$$

where $m \geq 0$ is an integer. The mth-order Adomian polynomial of ϕ is

$$A_m(\phi(u)) = \mathbb{D}_m \left[\phi(u(\lambda))\right], \tag{3}$$

where $m \geq 0$ is an integer and $A_m(\phi(u)) = A_m(\phi(u); u_0, u_1, ..., u_m)$.

Remark 1. For the case $0 \leq \lambda \leq 1$, the parametric series (1) and parametric derivative (2) reduce to homotopy series and homotopy derivative respectively [8].

Several algorithms [3,5,12,13] for symbolic programming have since been devised to efficiently generate the Adomian polynomials quickly to high orders, for example, a convenient formula for the Adomian polynomials is the rule of Rach, which reads (see Page 16 in [1] and Page 51 in [2])

$$A_n(f(u)) = \sum_{k=1}^{n} f^{(k)}(u_0) C(k,n), \ n \geq 1, \tag{4}$$

where the $C(k,n)$ are the sums of all possible products of k components $u_i, i = 1, 2, ..., n-k+1$, whose subscripts sum to n, divided by the factorial of the number of repeated subscripts. An equivalent expression of Equation (4) is

$$A_n(f(u)) = \sum_{p_1+2p_2+...+np_n=n} f^{(p_1+p_2+...+p_n)}(u_0) \prod_{s=1}^{n} \frac{u_s^{p_s}}{p_s!}, \ n \geq 1. \tag{5}$$

In the present paper, we propose an explicit formula to calculate the $C(k,n)$ in (4) and we show that for rapid computer-generation of the Adomian polynomials there is no need to use the (5).

2 Adomian polynomials for a general function

In this section, we first outline two theorems as already given in [7,10]. Then using them, we propose a new theorem which provides a new and simple way to calculate the Adomian polynomials for a general smooth function. Here, we mention them with a minor modification. For simplicity, we use the following notation

$$\widehat{u}_{m,n} = \sum_{i=n}^{m} u_i \lambda^i.$$

Theorem 1. *For function $f(u) = u^k$, the corresponding mth-order Adomian polynomial is given by*

$$A_m(u^k) = \sum_{r_1=0}^{m} u_{m-r_1} \sum_{r_2=0}^{r_1} u_{r_1-r_2} \sum_{r_3=0}^{r_2} u_{r_2-r_3} \cdots \sum_{r_{k-2}=0}^{r_{k-3}} u_{r_{k-3}-r_{k-2}} \sum_{r_{k-1}=0}^{r_{k-2}} u_{r_{k-2}-r_{k-1}} u_{r_{k-1}}, \qquad (6)$$

where $m \geq 0$ and $k \geq 0$ are positive integers.

Proof. Considering the definition (2), refer to [10]. □

Theorem 2. *For parametric series (1), it holds*

$$\mathbb{D}_m \left[f(u(\lambda)) \right] = \mathbb{D}_m \left[f(\widehat{u}_{m,0}) \right],$$

where f is a smooth function.

Proof. Refer to [7]. □

Corollary 1. *From Theorem 1, we find*

$$u^k(\lambda) = \left(\sum_{n=0}^{+\infty} u_n \lambda^n \right)^k = u_0^k + \sum_{m=1}^{+\infty} A_m(u^k) \lambda^m, \qquad (7)$$

where the Adomian polynomials $A_m(u^k)$ are given by (6) .

Remark 2. It is clear that $A_m(u^k)$ in Theorem 1 can easily be calculated by a simple code by using a symbolic software such as Mathematica. For this respect, the **Code. 1**, reported in Appendix, can be used in Mathematica. For instance, by ADPforPowerLaw[4,6], the $A_4(u^6)$ is calculated as follows

$$A_4(u^6) = 15u_0^2u_1^4 + 60u_0^3u_1^2u_2 + 15u_0^4u_2^2 + 30u_0^4u_1u_3 + 6u_0^5u_4.$$

Corollary 2. *From Theorem 2, for $m \geq n$, we find*

$$\mathbb{D}_m\left[(f(\widehat{u}_{\infty,n})\right] = \mathbb{D}_m\left[f(\widehat{u}_{m,n})\right],$$

where f is a smooth function.

Corollary 3. *From Corollary 2 and Theorem 1, for $m \geq k$, we find*

$$\mathbb{D}_m\left[\left(\hat{u}_{m,1}\right)^k\right] = \sum_{r_1=0}^{m-1} \sum_{\substack{r_2=0 \\ r_2\neq r_1}}^{r_1} \cdots \sum_{\substack{r_{k-2}=0 \\ r_{k-2}\neq r_{k-3}}}^{r_{k-3}} \sum_{\substack{r_{k-1}=0 \\ r_{k-1}\neq r_{k-2}}}^{r_{k-2}} u_{r_{k-1}} \prod_{j=0}^{k-2} u_{r_j - r_{j+1}}, \quad (8)$$

where $r_0 = m$.

Corollary 4. *It is easy to see that*

$$\mathbb{D}_m\left[\left(\hat{u}_{m,1}\right)^k\right] = \begin{cases} 0, & m < k, \\ u_1^k, & m = k. \end{cases} \quad (9)$$

Corollary 5. *Let $kn \geq m+1$ and $n \geq 1$, we find*

$$\mathbb{D}_m\left[\left(\widehat{u}_{\infty,n}\right)^k\right] = 0.$$

Remark 3. A sample Mathematica program for $\mathbb{D}_m\left[\left(\hat{u}_{m,1}\right)^k\right]$ in (8) is given by the **Code. 2** reported in Appendix. An alternative way is to use the Theorem 1 by taking $u_0 = 0$. To achieve this purpose, In Code 1, in the last command, the $Expand[D_{k,m}]$ is replaced by $Expand[D_{k,m}]/.u_0 \to 0$.

The following theorem provides a suitable and simple way to construct a recurrent relation for a general smooth function appeared within a structure with nonlinear terms in the equations.

Theorem 3. *Assume that $f(u)$ has the Taylor expansion with respect to u_0, then*

$$A_m\left(f\left(u\right)\right) = \sum_{k=1}^{m} \frac{f^{(k)}(u_0)}{k!} \mathbb{D}_m\left[\left(\widehat{u}_{m,1}\right)^k\right]. \quad (10)$$

Proof. Expanding $f(u)$ in Taylor series with respect to u_0, one has

$$f(u) = f(u_0) + \sum_{k=1}^{+\infty} \frac{f^{(k)}(u_0)}{k!} \left(u - u_0\right)^k. \quad (11)$$

From the (11), we have

$$A_m\left[f\left(u\right)\right] = \mathbb{D}_m\left[\sum_{k=1}^{\infty} \frac{f^{(k)}(u_0)}{k!} \left(u(\lambda) - u_0\right)^k\right],$$

recalling the Corollaries 5 and 2, we find

$$A_m\left(f\left(u\right)\right)=\sum_{k=1}^{m}\frac{f^{(k)}(u_0)}{k!}\mathbb{D}_m\left[\left(u(\lambda)-u_0\right)^k\right]=\sum_{k=1}^{m}\frac{f^{(k)}(u_0)}{k!}\mathbb{D}_m\left[\left(\widehat{u}_{m,1}\right)^k\right].$$

This ends the proof. □

Remark 4. The expansion (11) is rigorously guaranteed by the Cauchy-Kovaleski theorem. In fact, by using this theorem the Taylor expansion about the function $u_0(x)$ can be seen as an expansion in Banach space [4].

Remark 5. The expression $\mathbb{D}_m\left[\left(\widehat{u}_{m,1}\right)^k\right]$ in (10) can easily be calculated by (8) or Theorem 1 with taking $u_0=0$.

Corollary 5. *From Theorem 3, we find*

$$f(u(\lambda))=f(u_0)+\sum_{m=1}^{+\infty}w_m\lambda^m, \tag{12}$$

where $w_m=A_m(f(u))$ can be calculated by (10).

Remark 6. To calculate $A_m(f(u))$, the Code. 3, which is given in the Appendix, can be used in Mathematica. Using the $AdomianPolynomial[f[u],m]$, for $m=1,2,3,4$, the corresponding Adomian polynomials are given as follows

$A_1(f(u))=u_1f'(u_0)$,

$A_2(f(u))=u_2f'(u_0)+\frac{1}{2}u_1^2f''(u_0)$,

$A_3(f(u))=u_3f'(u_0)+u_1u_2f''(u_0)+\frac{1}{6}u_1^3f'''(u_0)$,

$A_4(f(u))=u_4f'(u_0)+\frac{1}{48}\left(24u_2^2+48u_1u_3\right)f''(u_0)+\frac{1}{2}u_1^2u_2f'''(u_0)+\frac{1}{24}u_1^4f^{(4)}(u_0)$.

Also, the Code. 4, reported in Appendix, shows an alternative way to calculate the Adomian polynomials by using the (3) and Corollary 2.

2.1 An improvement for the Rach's rule

According to (4) and Theorem 3, we find

$$C(k,n)=\frac{1}{n!}\mathbb{D}_n\left[\left(\widehat{u}_{n,1}\right)^k\right],$$

thus, the Rach's rule gets its explicit presentation.

3 A practical review of the Adomian decomposition method

In this section, we propose a practical review of the ADM and implementation it as an automatic program in the frame of a symbolic software such as Mathematica.

Adomian decomposition method [1], was first proposed by Adomian in 1988 and was further developed and improved by Adomian [2,3]. To illustrate the ADM for solving a general nonlinear problem, consider the following nonlinear problem

$$\begin{cases} \mathcal{N}(u) = f, \\ \mathbb{B}(u) = f_0, \end{cases} \tag{13}$$

where $\mathcal{N}$ is a general nonlinear operator, $\mathbb{B}$ is a linear initial/boundary operator, u is the unknown function that will be determined and f and g are given functions. An especial case of (13), $f = 0$, was given in [9]. The ADM consists in looking for the solution of Equation (13) in the series form

$$u = u_0 + \sum_{n=1}^{+\infty} u_n, \tag{14}$$

where u_0 is an initial guess and has the property

$$\mathbb{B}(u_0) = f_0, \tag{15}$$

and the other components of the solution series (14) have the property

$$\mathbb{B}(u_n) = 0, n \geq 1. \tag{16}$$

Thus, from (15) and (16) the solution series given by (14), satisfies the conditions of (13). For the nonlinear conditions, the ADM needs an improvement.

To construct series pattern solution, (14) by ADM, in the first step the nonlinear operator $\mathcal{N}$ is decomposed as

$$\mathcal{N}(u) = L(u) + N(u), \tag{17}$$

where L is a linear operator and $N = \mathcal{N} - L$. To continue in ADM, the nonlinear operator N is decomposed as

$$N(u) = \sum_{n=0}^{+\infty} A_n, \tag{18}$$

where A_n is the nth-order Adomian polynomial of N. Using (17), the Equation (13) becomes

$$\begin{cases} L(u) + N(u) = f, \\ \mathbb{B}(u) = f_0, \end{cases} \tag{19}$$

by ignoring the condition $\mathbb{B}(u) = f_0$, the solution of Equation (19) satisfies

$$u = u_g + L^{-1}[f] - L^{-1}[N(u)], \tag{20}$$

where u_g is the general solution of the linear equation $L(u) = 0$ and L^{-1} is the integral inverse of L. Now, by substituting (14) in (20) and considering (18), we get

$$\begin{cases} u_0 = u_g, \\ u_n = (1 - \chi_n)L^{-1}[f] - L^{-1}[A_{n-1}],\ n \geq 1. \end{cases} \tag{21}$$

where

$$\chi_n = \begin{cases} 0,\ n \leq 1, \\ 1,\ n \geq 2. \end{cases}$$

Recalling the condition $\mathbb{B}(u) = f_0$, we have

$$\begin{cases} u_0 = \mathbb{B}(u_g) = u_g^*, \\ \mathbb{B}(u_n) = 0,\ n \geq 1. \end{cases} \tag{22}$$

Thus, the recurrent relation to find a series pattern solution of Equation (13) by means of ADM is

$$\begin{cases} u_0 = u_g^*, \\ u_n = (1 - \chi_n)\, L^{-1}[f] - L^{-1}[A_{n-1}],\ \mathbb{B}(u_n) = 0, \end{cases} \tag{23}$$

let $uParticular = L^{-1}[-A_{n-1}] + (1 - \chi_n)\, L^{-1}[f]$, then, for $n \geq 1$, we find

$$\begin{cases} u_0 = u_g^*, \\ \\ u_n = uParticular + uParticularStar, \end{cases} \tag{24}$$

where $uParticularStar = \mathbb{B}\left(L^{-1}[A_{n-1}]\right) - (1 - \chi_n)\left(\mathbb{B}(L^{-1}[f])\right)$. It is important to notice that the definition of the integral inverse L^{-1} in (23), depends on the condition (16). L^{-1} in (24) can be defined such that the $L^{-1}[*]$ gives a particular solution of the equation $L(u) = *$.

Remark 7. It should be emphasized that the main step of the ADM is to choose a proper linear operator. According to (24), the linear operator should be defined so that

1. The following problem has a solution

$$\begin{cases} L(u_0) = 0, \\ \mathbb{B}(u_0) = f_0. \end{cases} \tag{25}$$

According to the Equations (20) and (21), the problem (25) gives the initial guess.

2. The following equation has a solution

$$\mathbb{B}\left(L^{-1}[A_{n-1}]\right) - (1-\chi(n))\,\mathbb{B}\left(L^{-1}[f]\right) = \mathbb{B}(u_g), \tag{26}$$

where $n \geq 1$. This equation helps to define the integral inverse of L.

3. The solution series, given by (14), is convergent. For this respect, some conditions have been given in [11].

3.1 Decomposition series and convergence analysis

An important hypothesis in ADM is the composition of the nonlinear operator N as shown in (18). There are some serious questions about (18) such as, from where does the series $\sum_{n=0}^{+\infty} A_n$ derive? Does it always converge? Is its sum really N? What are the other series that we could use instead of $\sum_{n=0}^{+\infty} A_n$? In order to answer these questions, and also, in order to explain some other issues, Gabet proposed a theory which explains and justifies the practical method [6]. In this subsection, we answer to the above mentioned questions by using the results of the section 2.

To derive (18), from (12), under the assumptions of the Theorem 3, we have

$$N(u(\lambda)) = N(u_0) + \sum_{n=1}^{+\infty} A_n \lambda^n, \tag{27}$$

where A_n is the nth-order Adomian polynomial of N. Let the parametric series (1) is convergent at $\lambda = 1$ and $s = \sum_{n=0}^{+\infty} u_n$, then

$$N(s) = A_0 + \sum_{n=1}^{+\infty} A_n. \tag{28}$$

According to the (23), it is clear that if the solution series (14) is convergent to s, then the decomposition series $\sum_{n=0}^{+\infty} A_n$ converges to $f - L(s)$. On the other hand, according to the (23), the decomposition series $\sum_{n=0}^{+\infty} A_n$ converges to $N(s)$. Therefore, if the solution series given by ADM, of the form (14) generated by (24), is convergent, then it must be an exact solution of the considered nonlinear problem, denoted by (13). Bearing in mind the above discussion, we can express the following theorem on the convergency of the ADM.

Theorem 4. *Assume that the ADM solution series given by (14) and (24) is convergent, then, under the assumptions of the Theorem 3, it must be an exact solution of the (13).*

4 Test examples

To show the efficiency of Theorem 4, described in the previous section, some examples are presented. We consider the mth-order nonlinear integro-differential equation with variable coefficients

$$\sum_{r=0}^{m} f_r(x)u^{(r)}(x) + \lambda \int_a^{\beta} K(x,t)F(u(t))dt = g(x), \;\; x \in [a,b], \tag{29}$$

where $F(u(x))$ is a function of $u(x)$, $K(x,t)$ is the kernel of the integro-differential equation, λ is a parameter, $g(x)$ is the data function, $f_r(x)$ is a function with respect to x, $u(x)$ is the unknown function that will be determined. For the Fredholm and Volterra kinds we take $\beta = b$ and $\beta = x$ respectively. We consider Equation (29) under the mixed conditions

$$\sum_{r=0}^{m-1} \Big(a_{rj}u^{(r)}(a) + b_{rj}u^{(r)}(b) + c_{rj}u^{(r)}(c) \Big) = \mu_j, \;\; j = 0,1,2,...,m-1, \tag{30}$$

where a_{rj}, b_{rj}, c_{rj} and μ_j are constants, and c, $a < c < b$, is a constant. For the linear case, it is assumed that $F(u(x)) = u(x)$. The computations will be performed using the program ***ADMforIDEs*** reported in the appendix. The program ***ADMforIDEs*** has designed in a general manner so that, for a given problem and its related linear operator, it calculates the integral inverse of the linear operator, initial guess, recursive relation and the terms of solution series automatically.

For Examples 1-4, we choose the linear operator as follows

$$L(u(x)) = \frac{d^2}{dx^2}u(x), \tag{31}$$

therefore, from the (22) and (31), we find

$$u_0(x) = x. \tag{32}$$

We emphasize that no attempt has been made here to obtain all solutions of a given problem.

Example 1. Consider the following nonlinear problem

$$\begin{cases} u''(x) + xu'(x) - u(x) + x\int_0^{\frac{\pi}{2}} t\sin\left(u(t)\right)dt = x \\ u(\frac{1}{2}) + u'(0) = \frac{3}{2}, \;\; 2u(\frac{1}{2}) + u'(1) = 2. \end{cases} \tag{33}$$

An exact solution is $u(x) = x$. Starting by (32), the recurrent relation (24) gives

$$u_n(x) = 0,\ n \geq 1,$$

therefore, using the ADM, the exact solution of Equation (33) is obtained as follows

$$u(x) = u_0(x) + \sum_{n=1}^{+\infty} u_n(x) = x. \tag{34}$$

Computations have been carried out by program ***ADMforIDEs***. The list of commands is as follows
$conditions[u\_] := \{(u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 0) - \frac{3}{2},$
$2 * (u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 1) - 2\};$
$Lcoefficients = \{0, 0, 1\};$
$Problemcoefficients = \{-1, x, 1\};$
ADMforIDEs$[\mathrm{Sin}[u], x, x * t, 0, \frac{\pi}{2}, 1, 2, 5]$

Example 2. Consider the following nonlinear problem

$$\begin{cases} u''(x) + xu'(x) - u(x) + 2x \int_0^x te^{-u^2(t)}dt = x - xe^{-x^2}, \\ u(\frac{1}{2}) + u'(0) = \frac{3}{2},\ 2u(\frac{1}{2}) + u'(1) = 2. \end{cases} \tag{35}$$

The exact solution is $u(x) = x$. Using the recurrent relation (24) with initial guess (32), we find

$$u_n(x) = 0,\ n \geq 1,$$

therefore, the ADM gives the exact solution of Equation (35) as follows

$$u(x) = u_0(x) + \sum_{n=1}^{+\infty} u_n(x) = x. \tag{36}$$

Computations have been carried out with the help of the program ***ADMforIDEs***. The list of commands is as follows
$conditions[u\_] := \{(u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 0) - \frac{3}{2},$
$2 * (u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 1) - 2\};$
$Lcoefficients = \{0, 0, 1\};$
$Problemcoefficients = \{-1, x, 1\};$
ADMforIDEs$[\mathrm{Exp}[-u^2], x - x * \mathrm{Exp}[-x^2], 2 * x * t, 0, x, 1, 2, 5]$

Example 3. Consider the following nonlinear problem

$$\begin{cases} u''(x) + xu'(x) - u(x) + (2m+2)x \int_0^1 tu^{2m}(t)dt = x, \\ u(\frac{1}{2}) + u'(0) = \frac{3}{2},\ 2u(\frac{1}{2}) + u'(1) = 2, \end{cases} \tag{37}$$

where m is a positive integer. An exact solution is $u(x) = x$. Recalling the recurrent relation (24) and initial guess (32), we obtain

$$u_n(x) = 0,\ n \geq 1,$$

therefore, the ADM obtains the exact solution of Equation (37) as follows

$$u(x) = u_0(x) + \sum_{n=1}^{+\infty} u_n(x) = x. \tag{38}$$

To achieve the computations by program ***ADMforIDEs***, the list of commands is as follows
$conditions[u\_] := \{(u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 0) - \frac{3}{2},$
$2 * (u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 1) - 2\};$
$Lcoefficients = \{0, 0, 1\};$
$Problemcoefficients = \{-1, x, 1\};$
$\mathrm{Assumptions} = m > 0 \&\& \mathrm{Element}[m, \mathrm{Integers}];$
$\mathrm{Refine}[\mathbf{ADMforIDEs}[u^{2m}, x, (2m+2) * x * t, 0, 1, 1, 2, 5]]$

Example 4. Consider the following nonlinear problem

$$\begin{cases} u''(x) + xu'(x) - u(x) + \frac{4m+1}{2m} x \int_0^1 t \sqrt[2m]{u(t)}dt = x, \\ u(\frac{1}{2}) + u'(0) = \frac{3}{2},\ 2u(\frac{1}{2}) + u'(1) = 2, \end{cases} \tag{39}$$

where m is a positive integer. An exact solution is $u(x) = x$. Putting to use the recurrent relation (24) and initial guess (32), we get

$$u_n(x) = 0,\ n \geq 1,$$

therefore, the ADM leads to the exact solution of Equation (39) as follows

$$u(x) = u_0(x) + \sum_{n=1}^{+\infty} u_n(x) = x. \tag{40}$$

For performing the computations with the help of the program ***ADMforIDEs***, the list of commands is as follows
$conditions[u\_] := \{(u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 0) - \frac{3}{2},$
$2 * (u[x]/.x \to \frac{1}{2}) + (\mathrm{D}[u[x], x]/.x \to 1) - 2\};$
$Lcoefficients = \{0, 0, 1\};$
$Problemcoefficients = \{-1, x, 1\};$
$\mathrm{Assumptions} = m > 0 \&\& \mathrm{Element}[m, \mathrm{Integers}];$
$\mathrm{Refine}[\mathbf{ADMforIDEs}\ [\sqrt[2m]{u}, x, \frac{4m+1}{2m} * x * t, 0, 1, 1, 2, 5]]$

Remark 8. In Examples 1-4, the first-order iteration gives the exact solution.

Example 5. Consider the following problem

$$\begin{cases} u'(x) + u(x) - \int_0^1 u(t)dt = \frac{1}{2}\left(\frac{1}{e^2} - 1\right), \\ u(0) = 1, \end{cases} \tag{41}$$

we choose the linear operator as follows

$$L(u(x)) = \frac{d}{dx}u(x) + u(x), \tag{42}$$

therefore, from the (22) and (42), we find $u_0(x) = e^{-x}$. Putting to use the recurrent relation (24) and obtained initial guess, we have

$$u_n(x) = \frac{1}{2}(e-1)^2(e^x - 1)e^{-x-n-1},\ n \geq 1,$$

therefore, the ADM leads to the exact solution of Equation (39) as follows

$$u(x) = u_0(x) + \sum_{n=1}^{+\infty} u_n(x) = \frac{1}{2}e^{-x-1}\left(-e^x + e^{x+1} + 1 + e\right). \tag{43}$$

For performing the computations with the help of the program ***ADMforIDEs***, the list of commands is as follows

$conditions[u\_] := \{(u[x]/.x \to 0) - 1\};$
$Lcoefficients = \{1, 1\};$
$Problemcoefficients = \{1, 1\};$
ADMforIDEs $\left[u, \frac{1}{2}\left(\frac{1}{e^2} - 1\right), 1, 0, 1, -1, 1, 5\right]$

Remark 9. The program ***ADMforIDEs***, reported in the Appendix, monitors some terms of the solution series, the number of the terms is given by **SolutionOrder**. To generate the general term of the solution series (if possible) and the closed form of the solution series (if possible), the user must be added the following commands to the end of the list of commands . It is important to emphasis that the following commands must be used in Mathematica 7 or later.

m1 =Input["By evaluating the obtained terms solution series, please input the index of the term in which it is possible to generate the general term of the solution series?"];
SolutionTerms = Join[Table[$u[m]$, $\{m$, m1, SolutionOrder$\}$]];
SolutionCoefficents = FindSequenceFunction[SolutionTerms, n];
Print ["$u_n(x)$=", TraditionalForm[SolutionCoefficents]] ;
FinalSolution = Sum[SolutionCoefficents, $\{n$, m1, Infinity$\}$,GenerateConditions → True];
Sol1 = Simplify $\left[\sum_{n=0}^{\text{m1}-1} u[n] + \text{FinalSolution}\right]$;
Print $\left["u(x)=\sum_{n=0}^{+\infty} u_n(x)=", \text{TraditionalForm[Sol1]}\right]$

5 Concluding remarks

By calculating the parametric derivative for a function of a more general type, described in Theorem 3, the Adomian decomposition method becomes a powerful analytic approach for obtaining convergent series solutions of strongly nonlinear problems governing physical models in applied science and engineering. Using the parametric derivative, some lemmas and theorems provided in ADM papers in the literature have been unified and modified here via some new theorems and corollaries. Also, by means of the (24) the Adomian decomposition method was extended to handel the nonlinear problems with the mixed conditions.

Acknowledgements

The author would like to thank the anonymous referees especially the second referee for his/her constructive comments and suggestions. Many thanks are due to financial support from the Ilam University of Iran.

Appendix (Mathematica programs)

Code. 1(A sample Mathematica program for $A_m(u^k)$ given by (6))

$\mathbf{ADPforPowerLaw}[m\_, k\_] := \text{Module}[\{\}, D_{1,j\_} := u_j; D_{i\_,0} := u_0^i;$
$D_{2,j\_} := \sum_{r=0}^{j} u_{j-r}u_r;\ D_{i\_,order\_} := \sum_{r=0}^{order} u_{order-r}D_{i-1,r};$
$\text{Print}[''A''_m,''\,(\,'',u^k,''\,)\,='', \text{Expand}[\text{TraditionalForm}[D_{k,m}]]]];$

Code. 2(A sample Mathematica program for $\mathbb{D}_m\left[(\hat{u}_{m,1})^k\right]$ given by (8))

$\mathbf{DnonU0}[m\_, k\_] := \text{Module}[\{\}, D_{1,j\_} := \text{If}[j > 0, u_j, 0]; D_{i\_,0} := 0; D_{2,j\_} := \sum_{r=1}^{j-1} u_{j-r}u_r;$
$D_{i\_,order\_} := \sum_{r=2}^{order-1} u_{order-r}D_{i-1,r}; \text{Print}[''\mathbb{D}''_m,''\,('',\hat{u}^k_{m,''1''},''\,)\,='',$
$\text{Expand}[\text{TraditionalForm}[D_{k,m}]]];$

Code. 3(A sample Mathematica program for $A_m(f(u))$ given by (10))

$\mathbf{AdomianPolynomial}[function\_, m\_] := \text{Module}[\{\}, D_{1,j\_,x\_} := \text{If}[j > 0, x_j, 0];$
$D_{i\_,0,x\_} := 0; D_{2,j\_,x\_} := \sum_{r=1}^{j-1} x_{j-r}x_r;\ D_{i\_,order\_,x\_} := \sum_{r=2}^{order-1} x_{order-r}D_{i-1,r,x};$
$HD_{order\_} := \text{If}[m == 0, function/.u \to u_0,$

$\sum_{n=1}^{order}\left((\mathrm{D}[function,\{u,n\}]/.u\to u_0)/n!*D_{n,order,u}\right]);$
$\mathrm{Print}[''A''_m,''('',function,'')='',\mathrm{Expand}[\mathrm{TraditionalForm}[HD_m]]]];$

Code. 4(A sample Mathematica program for $A_m(f(u))$ by using the (3) & Corollary 2)

AdomianPolynomial1$[function_-,m_-]:=\mathrm{Module}[\{\},g[u_-]:=function;$
$A[m1_-]:=\frac{1}{\mathrm{Factorial}[m1]}(\mathrm{D}[g[u]/.u\to\mathrm{Sum}[u_n*p^n,\{n,0,m1\}],\{p,m1\}]/.p\to 0);$
$\mathrm{Print}[''A''_m,''('',function,'')='',\mathrm{Expand}[\mathrm{TraditionalForm}[A[m]]]];$

ADMforIDEs(A sample Mathematica program of ADM for solving IDEs given by (29))

ADMforIDEs$[functionF_-,functiong_-,kernel_-,a_-,\beta_-,\lambda_-,ProblemOrder_-,SolutionOrder_-]$
$:=\mathrm{Module}[\{\},F[u_-]:=functionF;g[x_-]:=functiong;k[x_-,t_-]:=kernel;$
$(*----$**Definition of the linear operator**$----------*)$
$L[f_-]:=\mathrm{Module}[\{\},\mathrm{Expand}[\sum_{i=0}^{ProblemOrder}(Lcoefficients[[i+1]]*\mathrm{D}[f,\{x,i\}])]];$
$(*----$**Definition of the integral inverse of linear operator**$-------*)$
$Linverse[f_-]:=\mathrm{Module}[\{Linv,u,solution\},Linv=\mathrm{DSolve}[L[u[x]]==f,u[x],x];$
$solution=Linv[[1,1,2]]/.C[\_]\to 0;\mathrm{Expand}[solution]];$
$Linverse[p\_Plus]:=\mathrm{Map}[Linverse,p];Linverse[c_-*f_-]:=c*Linverse[f]/;\mathrm{FreeQ}[c,x];$
$(*----$**Definition of the Adomian polynomials**$----------*)$
$HDN[horder_-]:=\mathrm{Module}[\{\},D_{1,j\_}:=\mathrm{If}[j>0,u[j],0];D_{i\_,0}:=0;$
$D_{2,j\_}:=\mathrm{If}[j>0,\sum_{r=1}^{j-1}u[j-r]*u[r]];D_{i\_,m\_}:=\sum_{r=2}^{m-1}u[m-r]*D_{i-1,r};$
$DH_{m\_,function\_}:=\mathrm{If}[horder==0,function/.u\to u[0],$
$\mathrm{Expand}[\sum_{n=1}^{m}((\mathrm{D}[function,\{u,n\}]/.u\to u[0])/(n!)*D_{n,m})]]];$
$(*----$**Definition of the initial guess**$----------*)$
$ugStar=\mathrm{DSolve}[L[u[x]]==0,conditions[u]==0,u[x],x];$
$u[0]=\ \mathrm{First}[u[x]/.ugStar];$
$(*----$**Definition of the $\mathbf{u_g}$**$----------*)$
$ug=\mathrm{DSolve}[L[u[x]]==0,u[x],x];$
$ug1=\mathrm{First}[u[x]/.ug];$
$(*----$**Some needed commands**$----------*)$
$X[m_-]:=\mathrm{If}[m\le 1,0,1];C1=\mathrm{Arrary}[C,ProblemOrder];w[x\_]:=ug1;$
$\mathrm{Which}[ProblemOrder==1,constants[u\_]:=\mathrm{First}[conditions[u]]/.\_[\_]->0,$
$ProblemOrder>1,constants[u\_]:=conditions[u]/.\_[\_]->0];$
$f11=constants[u1]-conditions[w];$
$(*----$**Main block**$----------*)$
$\mathrm{For}[m=1,m<SolutionOrder,m++,HDN[m-1];$
$\{A[m-1]=\sum_{i=0}^{ProblemOrder}(Problemcoefficients[[i+1]]-Lcoefficients[[i+1]])$
$*\mathrm{D}[u[m-1],\{x,i\}]+\lambda*\mathrm{Integrate}[k[x,t]*((DH_{m-1,F[u]})/.x\to t),\{t,a,\beta\}],$
$uParticular=Linverse[-A[m-1]+(1-X[m])*g[x]],w1[x\_]:=uParticular,$
$f12=conditions[w1]-constants[u1],ug2=\mathrm{Solve}[\{f11==f12\},C1],$
$uParticularStar=\mathrm{First}[ug1//.ug2],$

$u[m] = uParticular + uParticularStar,$
$\text{Print}[''u'',_m\ ,'' ='', \text{TraditionalForm}[\text{Collect}[u[m], x]]]\}]];$

References

1. Adomian, G. *Nonlinear Stochastic Systems Theory and Applications to Physics*, Kluwer Academic, Dordrecht, 1989.
2. Adomian, G. *Solving Frontier Problems of Physics: The Decomposition Method*, Kluwer Academic, Dordrecht, 1994.
3. Adomian, G. and Rach, R. *On composite nonlinearities and the decomposition method*, J. Math. Anal. Appl. 113 (1986) 504-509.
4. Cherruault, Y., Adomian, G., Abbaoui, K. and Rach R. *Further remarks on convergence of decomposition method*, International Journal of Bio-Medical Computing 38 (1995) 89-93.
5. Duan, J., Rach, R., Baleanu, D. and Wazwaz, A. *A review of the Adomian decomposition method and its applications to fractional differential equations*, Commun. Frac. Calc. 3 (2) (2012) 73-99.
6. Gabet, L. *The Theoretical Foundation of the Adomian Method*, Computers Math. Applic. 27 (1994) 41-52.
7. Ghorbani, A. *Beyond Adomian polynomails: He polynomails*, Chaos, Soliton and Fractals 39 (2009) 1486-1492.
8. Liao, S. J. *Notes on the homotopy analysis method: Some definitions and theorems*, Commun Nonlinear Sci Numer Simul 14 (2009) 983–997.
9. Molabahrami, A. *Integral mean value method for solving a general nonlinear Fredholm integro-differential equation under the mixed conditions*, Communications in Numerical Analysis 2013 (2013) 1-15. http://dx.doi.org/10.5899/2013/cna-00146.
10. Molabahrami, A. and Khani, F. *The homotopy analysis method to solve the Burgers-Huxley equation*, Nonlinear Anal-Real 10 (2009) 589-600.
11. Ngarhasta, N., Some, B., Abbaoui, K. and Cherruault, Y. *New numerical study of adomian method applied to a diffusion model*, Kybernetes 31 (2002) 61-75.
12. Rach, R. *A convenient computational form for the Adomian polynomials*, J. Math. Anal. Appl. 102 (1984) 415-419.
13. Rach, R. *A new definition of the Adomian polynomials*, Kybernetes 37 (2008) 910-955.

Iranian Journal of Numerical Analysis and Optimization
Vol. 5, No. 2, (2015), pp 45-58

An adaptive meshless method of line based on radial basis functions

J. Biazar* and M. Hosami

Abstract

In this paper, an adaptive meshless method of line is applied to distribute the nodes in the spatial domain. In many cases in meshless methods, it is also necessary for the chosen nodes to have certain smoothness properties. The set of nodes is also required to satisfy certain constraints. In this paper, one of these constraints is investigated. The aim of this manuscript is the implementation of an algorithm for selection of the nodes satisfying a given constraint, in the meshless method of line. This algorithm is applied to some illustrative examples to show the efficiency of the algorithm and its ability to increase the accuracy.

Keywords: Adaptive Meshless Methods; Meshless Method of Line; Radial Basis Functions.

1 Introduction

In the last decade, application of radial basis functions (RBFs) in the meshless methods, for numerical solution of various types of partial differential equations (PDEs) has been developed [9–11]. One of the main advantages of this method is the mesh-free property. Meshless methods do not typically need a mesh. They need some scattered nodes in the domain that can be selected uniformly or randomly. This is one of the important properties of the meshless methods. An alternative meshless method is an approach that uses a mesh to obtain a good set of nodes based on the problem options (such as the form of equation, initial or boundary conditions). These methods are known as adaptive meshless methods. Early researchers have incorporated

*Corresponding author
Received 2 June 2014; revised 17 November 2014; accepted 6 May 2015
J. Biazar
Department of Applied Mathematics, Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran. e-mail: biazar@guilan.ac.ir

M. Hosami
Department of Applied Mathematics, Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran. email: mhosami@phd.guilan.ac.ir

the adaptive methods in several schemes [1, 28, 29, 34, 36]. In this paper an adaptive method known as Equidistribution [7,14] is introduced for selecting a set of nodes under a specified criterion on the set. The criterion is that in the set of nodes, the ratio of the largest distance to the smallest distance must be smaller than a given parameter k. Kautsky and Nichols introduced an algorithm to enforce this criterion in the Equidistribution algorithm [7]. In this research, this algorithm is applied in meshless method of line to improve the accuracy of the method. This paper is presented as follows. In Section 2, radial basis functions are introduced. In Section 3, an adaptive method is described for selecting a set of nodes and an algorithm is introduced based on the given criterion. Section 4, is devoted to presenting some illustrative examples, and comparing the numerical results of uniform and adaptive meshes.

2 Radial basis functions to approximate a function

In this section some essential points about radial basis functions (RBFs), are introduced. For more details, interested readers are referred to [1,9–11,19,37]. Suppose that a real function $u = u(x),\ x \in \mathbb{R}^d$, should be approximated. An approximation to u, by radial basis functions, will be defined as the following

$$u^*(x) = \sum_{j=1}^{N} \lambda_j \, \varphi(\ \|x - x_j\|\ ) \qquad \lambda_j \in \mathbb{R}.$$

Where $x, x_j \in \mathbb{R}^d$, and norm is the Euclidean norm, and φ is a RBF on $\mathbb{R}^d$. An RBF is a real valued function which is only dependent on the distance r, between x and a point $x_j \in \mathbb{R}^d (r = \ ||x - x_i||)$. Some of important RBFs are:

$\varphi(r) = \sqrt{1 + \varepsilon^2 r^2}$ Multiquadrics (MQ),

$\varphi(r) = 1/(1 + \varepsilon^2 r^2)$ Inverse Quadratics (IQ),

$\varphi(r) = 1/\sqrt{1 + \varepsilon^2 r^2}$ Inverse Multiquadrics (IMQ),

$\varphi(r) = e^{-\varepsilon^2 r^2}$ Gaussian (GA),

where ε is called the shape parameter. N distinct nodes x_j are called central nodes. In matrix notation, the approximated function $u^*(x)$ is denoted as follows,

$$u^*(x) = \sum_{j=1}^{N} \lambda_j \, \varphi(r_j) = \Phi^t(r)\, \lambda, \tag{1}$$

where

$$\Phi(r) = [\varphi(r_1), \varphi(r_2), ..., \varphi(r_N)]^t, \quad \lambda = [\lambda_1, \lambda_2, ..., \lambda_N]^t, \quad \varphi(r_j) = \varphi(\|x - x_j\|),$$

λ, is the vector of coefficients, that will be determined. By considering $u^*(x_i) = u_i$, equation (1) can be presented as a system of equations $A\lambda = U$, where, $U = [u_1, u_2, ..., u_N]^t$, and by considering $\varphi(r_{ij}) = \varphi(\|x_i - x_j\|)$,

$$A = [\Phi^t(r_1), \Phi^t(r_2), ..., \Phi^t(r_N)]^t,$$

where $\Phi^t(r_i) = [\varphi(r_{i1}), \varphi(r_{i2}), ..., \varphi(r_{iN})]$. By solving the system of equations $A\lambda = U$, the unknown vector λ will be determined. There are several factors affecting the RBF interpolation process, such as central nodes distribution, shape parameter, etc. In this paper our focus is on the central nodes distribution.

3 An adaptive meshless method

3.1 Meshless method of line

Method of line (MOL) is a general method for solving a PDE. In this method, two sequential strategies will be followed: discretizing all directions except one (usually the time direction for time-dependent PDEs) and integrating the semi-discrete problem as a system of ODEs. By choosing RBF collocation method (Kansa Method) [9,10] as integrator system, the method is called the meshless method of line (MMOL). MMOL involves the following main steps:

1- Partitioning the spatial domain (In meshless method of line, this step is reduced to choosing some center nodes x_i in the spatial domain).

2- Discretizing of the problem in one direction (Usually, time direction in time-dependent PDEs).

3- Approximating the solution $u(x, t_n)$ in each step of time by RBF-approximation as follows

$$u(x, t_n) = \sum_{j=1}^{N} \lambda_j \, \varphi(r_j) = \Phi^t(r)\lambda \qquad \lambda_j \in \mathbb{R}. \tag{2}$$

4- Substituting (2) in the governing equation and collocating x_i. This leads to a system of ordinary differential equation.

5- Solving the system of ODEs by suitable method, such as RK4 (In each step of RK4, the solution of the problem in each time step is obtained).

This method is well-addressed in [4, 15].

3.2 Adaptive meshless method of line

In each step of RK4 in MMOL, the center nodes x_i can be selected by an adaptive mesh. Adaptivity is a well-known concept in mesh generation. The purpose of the adaption is to change the center nodes, so that to achieve greater accuracy. As an example, if the problem was approximating a function with a rapid change in some areas of its domain, concentrating the center nodes in these areas could improve the accuracy of the approximation. There are several adaptive algorithms for choosing central nodes in the domain. In this research, methods based on Equidistribution are investigated.

Definition 1. (Equidistribution). Let M is a non-negative piecewise continuous function on $[a, b]$, and c is a constant, such that $n = (1/c)\int_a^b M(x)\,dx$ is an integer. The mesh

$$\Pi : \ a = x_1, x_2, ..., x_n = b,$$

is called equi-distributing (e.d.) on $[a, b]$ with respect to M and c if

$$\int_{x_{i-1}}^{x_i} M(x)\,dx = c \ , \ \ j = 2...n,$$

and is called subequi-distributing (s.e.d.) on $[a, b]$, with respect to M and c if, for $nc \geq \int_a^b M$,

$$\int_{x_{i-1}}^{x_i} M(x)\,dx \leq c, \quad j = 2\,...\,n.$$

A suitable algorithm to produce an e.d. mesh is given in [7]. In the definition 1, the function M, often called a monitor, is dependent on the function u. A well-known monitor function is arc-length monitor. The arc-length monitor is defined as the following

$$M(x) = \sqrt{1 + u_x^2}.$$

To find more details about the monitors and implementation of the algorithm, interested readers are referred to [6, 7, 17].

In [31], Sarra introduced an adaptive algorithm which was developed to RBF methods for interpolation problems and PDEs. He applied the method for time dependent PDEs. The method is a combination of the meshless Method of Line and an Equidistribution algorithm for producing a set of center nodes, in each step. The algorithm is an e.d. one with arc-length monitor. The method is summarized as follows:
In the adaptive algorithm, we start at time t^0 with uniform nodes. To advance the PDE in time with the adaptive grid algorithm the method is implemented

as follows. Assume that $s_j^n,\ j = 1..N$, is approximate solution at time t^n at distinct nodes $x_j^n,\ j = 1..N$. Then, the MMOL is used on these central nodes to obtain approximations $\bar{s}_j^{n+1},\ j = 1..N$, at time t^{n+1}. Next, by an Equidistribution based algorithm, a new set of nodes is obtained based on the properties of $\bar{s}^{n+1}$. To obtain new central nodes, the points $(x_j^n, \bar{s}_j^{n+1})$ are joined by straight lines and the length of the resulting polygon is computed (Figure 1-a, 1-b). Then N equally spaced points on the polygon are found which divide its total length into N equal parts (Figure 1-c). The new nodes $x_j^{n+1},\ j = 1..N$, are found as the projection of these N equally spaced points on the polygon to the x-axis (Figure 1-d). Finally, s_j^{n+1} is obtained by interpolating the values $(x_j^n, \bar{s}_j^{n+1})$. Applying this algorithm, distribute the nodes on the spatial domain based on the approximated solution at each time step, i.e. in step one, the nodes are distributed based on initial condition. If there are regions of steep gradients, it is obvious that the algorithm concentrate the nodes over these regions. In these regions, the nodes will be near together and this fact leads to an ill-conditioned problem. Since condition number of RBF matrix becomes very large or sometimes even close to singular. Thus, based on the Equidistribution mesh without constraint, there is not any guarantee to well-conditioning of the problem. Thus imposing some constraints can be useful to overcome this deficiency. One of these constraints to control the distribution of the nodes in the domain, is as follows

$$\frac{h_{\max}}{h_{\min}} < k, \tag{3}$$

where $h_i = x_i - x_{i-1}$. On the other hand, the introduced algorithm does not work if the constraint be applied. To apply the Equidistribution algorithm subject to this constraint, some modifications must be done. In addition to the investigated constraint, there are some other constraints, such as a constraint introduced by Kautsky and Nichols which is; the ratio of the length of successive subintervals must be less than a parameter k. In this study we investigate the constraint (3). In the following, an algorithm due to Kautsky and Nichols [7] will be introduced to distribute a set of nodes for which the constraint (3) is satisfied.

3.3 An algorithm for the adaptive nodes with constraint

Suppose that $(x_j, s_j),\ j = 1, 2, .., N$ are some data points. Our goal is to gain a set of nodes based on the Equidistribution algorithm that satisfy the constraint (3). Thus, an s.e.d. mesh is produced, with respect to M and c.

Theorem 1. *If* $\Pi : \{a = x_1, x_2, ..., x_n = b\}$ *is an e.d. mesh on* $[a, b]$ *with respect to* g *and* d*, where*

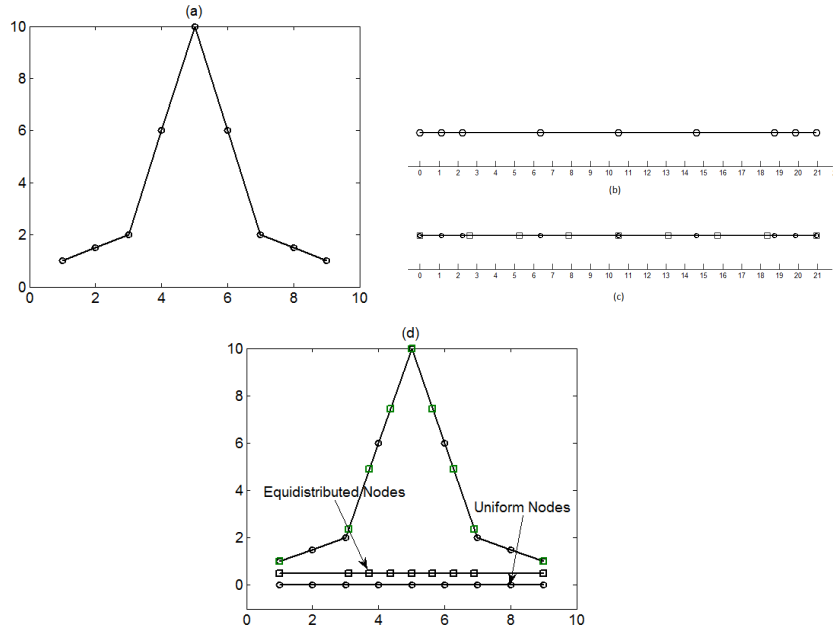

Figure 1: Geometrical interpretation of the Equidistribution procedure

$$g(t) = \max(M(t), p),$$

with

$$p = (1/k) \max_{t\in[a,b]} M(t),$$

and $d = (1/c)\int_a^b g(x)\,dx$ *(and* n *is equal to the smallest integer such that* $nc \geq \int_a^b g$*), then* Π *is a s.e.d. on* $[a, b]$ *with respect to* M *and* c*, and satisfies in* (3).

Proof. For proof and more details about the implementation of the algorithm, see [7].

Figure 2, illustrates the effect of the constraint in the distributing of nodes. The figure also shows the uniform adaptive nodes without constraint, and adaptive nodes with constraint. It is obvious that the constraint omits the huge concentration in a region.

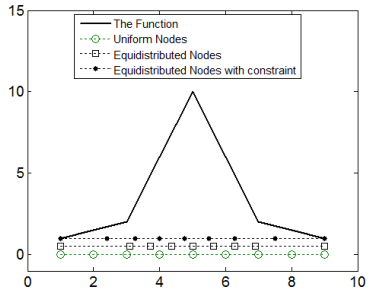

Figure 2: The comparison of three types of distribution for a test function

4 Numerical experiments

In this section, the algorithm is implemented on two time-dependent partial differential equations. The method is a combination of the algorithm which is introduced in 3.1 and Equidistribution algorithm (introduced in 3.3), regarding the constraint (3). In fact, the e.d. algorithm is implemented in each step of time in meshless method of line to produce adaptive central nodes which satisfy the constraint (3).

Example 1. Consider the Burger equation

$$u_t + u\,u_x = \upsilon\,u_{xx}, \tag{4}$$

on the interval [-1,1]. The exact solution is $u(x,t) = \frac{0.1\,e^a + 0.5\,e^b + e^c}{e^a + e^b + e^c}$, where $a = -(x + 0.5 + 4.95t)/(20\upsilon)$, $b = -(x + 0.5 + 0.75t)/(4\upsilon)$, and $c = -(x + 0.625)/(2\upsilon)$. The initial condition $u(x,0)$ and the boundary conditions $u(-1,t)$, $u(1,t)$ are specified. By choosing $\upsilon = 0.0035$, the equation is solved by uniform and adaptive nodes. Meshless method of line combined with adaptive algorithm is applied on equation (4). By choosing N center nodes $\{x_1, x_2, ..., x_N\}$ in the domain [-1,1], at a constant time t, the solution $u(x,t)$ can be expressed in RBF-approximation as follows

$$u(x,t) = \sum_{j=1}^{N} \lambda_j\,\varphi(r_j) = \Phi^t(r)\lambda. \tag{5}$$

Collocating (5) by $\{x_1, x_2, ..., x_N\}$, leads us to the following system of equation

$$A\lambda = u, \tag{6}$$

where $u = [u(x_1,t), u(x_2,t), ..., u(x_N,t)]$. By substituting $\lambda = A^{-1}u$ into (5), we have

$$u(x,t) = \Phi^t(r)A^{-1}u = V(x)u(t), \tag{7}$$

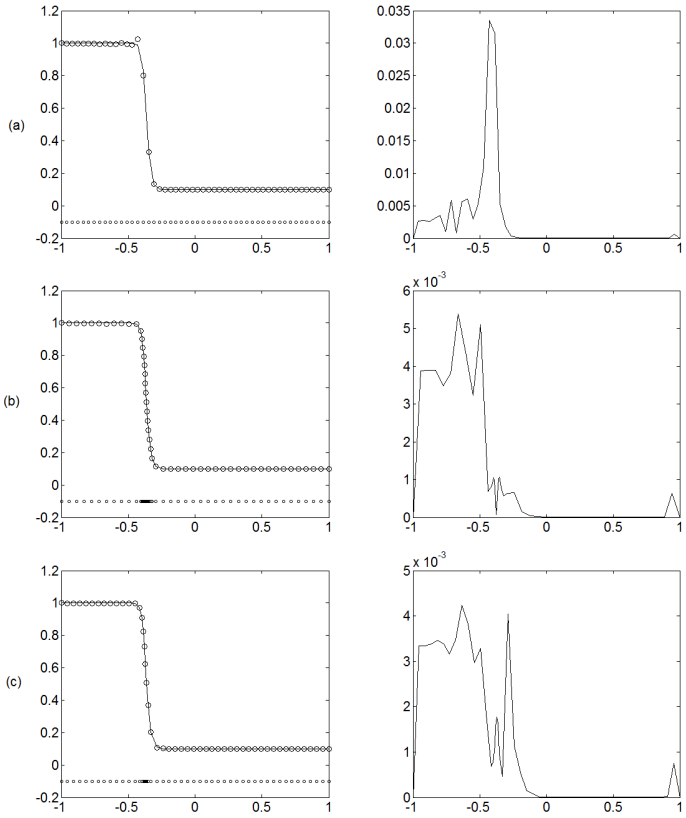

Figure 3: Plots of the approximate solution and absolute error of equation (4) at t=0.5 using 50 uniform nodes (a), adaptive nodes without constraint (b), and adaptive nodes with constraint (c)

where $V(x) = \Phi^t(x)A^{-1} = [V_1(x), ..., V_N(x)]$. By substituting (7) into the Burger equation (4), and collocating the center nodes x_i, we obtain

$$\frac{du_i}{dt} + u_i \left(V_x(x_i)\, u\right) = \upsilon \, \left(V_{xx}(x_i)u\right), \qquad i = 1, 2, ..., N.$$

This equation can be written as a system of ordinary differential equations as

$$\frac{du}{dt} = -u \otimes \left(V_x(x_i)\, u\right) + \upsilon \, \left(V_{xx}(x_i)u\right), \tag{8}$$

where $\otimes$ denote component by component multiplication of two vectors. Equation (8), is rewritten as

$$\frac{du}{dt} = F(u), \tag{9}$$

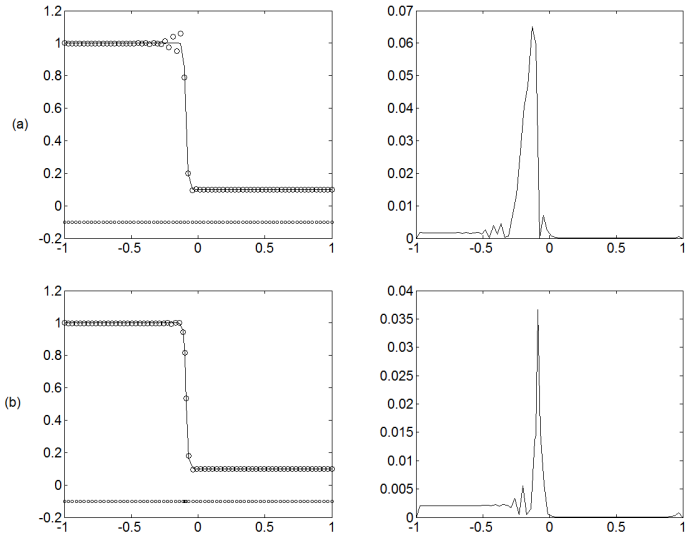

Figure 4: Plots of the approximate solution and absolute error of equation (4) at t=1 using 70 uniform nodes (a), and adaptive nodes with constraint (b)

where $F(u) = -u \otimes (V_x(x_i)\, u) + \upsilon\ (V_{xx}(x_i)u)$. The system of ordinary differential equations (9) can be solved by RK4 method. In the nth step of RK4, $u(x, t_n)$ is approximated. As mentioned before, the center nodes $\{x_1, x_2, ..., x_N\}$ in each step can be selected adaptively. We solve the Burger equation (4), by adaptive meshless method of line by three different distribution of center nodes; uniformly distributed nodes, adaptive nodes without constraints, and adaptive nodes with the constraint (3). Figure 3 shows the approximate solution at t=0.5 with different center nodes. The approximate solution by uniform nodes demonstrates that, it has the minimum accuracy in the sharpest region of the solution. Furthermore Figure 3-b, and 3-c show the same accuracy for two adaptive center nodes. It is important that without constraint (3), the condition number of the RBF matrix may be very large (close to singular) or singular, and RBF interpolation can't work exactly. Due to this fact, in this example at time 1, by 70 adaptive nodes without constraint, the method is failed to obtain a solution (Table 1). The results of using uniform nodes and adaptive nodes with constraint are shown in Figure 4. Table 1 illustrates the accuracy of the adaptive algorithm. It is known that the value of the parameter k influence the concentration of the nodes. Thus to illustrate the impact of the parameter k in distributing the adaptive nodes, and in the accuracy of the results, the error norm by different values of this parameter are investigated in Table 1.

Table 1: The error norms of the approximate solution of Example 1

t	N	Distribution of nodes	k	ε	Max error	RMS error	Figure
0.5	50	Uniform	-	31	0.0335	0.0070	Figure 3-a
		Adaptive without constraint	-	31	0.0054	0.0018	Figure 3-b
		Adaptive with constraint	2	31	0.0307	0.0064	-
			3	31	0.0154	0.0036	-
			6	31	0.0042	0.0018	Figure 3-c
0.5	70	Uniform	-	31	0.0059	0.0013	-
		Adaptive without constraint	-	31	0.0027	0.0011	-
		Adaptive with constraint	2	31	0.0023	9.65e-4	-
			3	31	0.0019	8.81e-4	-
			6	31	0.0022	0.0010	-
1	70	Uniform	-	31	0.0650	0.0135	Figure 4-a
			-	51	0.0942	0.0166	-
		Adaptive without constraint	-	31	NaN	NaN	-
			-	51	NaN	NaN	-
		Adaptive with constraint	3	31	0.3548	0.0509	-
			3	51	0.0367	0.0055	Figure 4-b
			6	31	0.0321	0.0067	-
			6	51	0.0435	0.0087	-

Example 2. Consider the KdV equation

$$u_t + \varepsilon u\, u_x + \mu\, u_{xxx} = 0, \tag{10}$$

with $\varepsilon = 6$, and $\mu = 1$. The initial condition is

$$u(x, 0) = 2\, \sec h^2(x).$$

The exact solution is

$$u(x, t) = 2\, \sec h^2(x - 4t).$$

The computational domain is$[-10, 40]$. The boundary conditions $u(-10, t)$ and $u(40, t)$ are determined. This problem is solved by the same method as the example 1. Figure 5 shows the solution of the equation (10), with uniform and adaptive center nodes. It is obvious that, by 110 center nodes, at t=0.5, the approximate solutions using adaptive nodes have better accuracy. The RMS error and Max-error of the results (Table 2), shown that the adaptive nodes with constraint result in better accuracy. If the number of central nodes increased up to 150, the solutions by adaptive nodes have the same accuracy. It is predictable, because when the number of nodes is large, the e.d. algorithm leads to nearly uniform distribution of nodes and consequently, the errors of approximate solutions are close. Table 2, demonstrate the impact of N, k, and shape parameter ε, in the accuracy of the results.

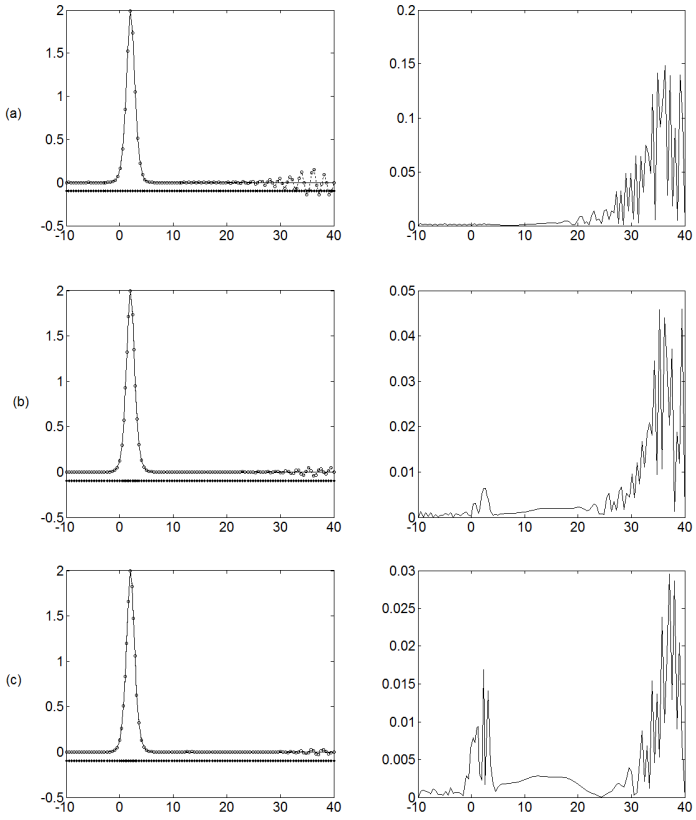

Figure 5: Plots of the approximate solution and absolute error of equation (10) at t=0.5 using 110 uniform nodes (a), adaptive nodes without constraint (b), and adaptive nodes with constraint (c)

Table 2: The error norms of the approximate solution of Example 2

t	N	Distribution of nodes	k	ε	Max error	RMS error	Figure
0.5	110	Uniform	-	0.8	0.1485	0.0389	Figure 5-a
		Adaptive without constraint	-	0.9	0.0460	0.0110	Figure 5-b
		Adaptive with constraint	2	1.2	0.0295	0.0069	Figure 5-c
			3	1.2	0.0295	0.0069	-
			6	1.2	0.0295	0.0069	-
0.5	150	Uniform	-	0.8	0.0205	0.0065	-
		Adaptive without constraint	-	0.8	0.0046	0.0014	-
		Adaptive with constraint	2	0.8	0.0033	0.0010	-
			3	0.8	0.0033	0.0010	-
			6	0.8	0.0033	0.0010	-
1	150	Uniform	-	0.8	0.0197	0.0094	-
			-	1.1	0.0074	0.0034	-
		Adaptive without constraint	-	0.8	0.0045	0.0021	-
			-	1.1	0.0054	0.0025	-
		Adaptive with constraint	2	0.8	0.0033	0.0016	-
			2	1.1	0.0030	0.0014	-
			3	0.8	0.0033	0.0016	-
			3	1.1	0.0030	0.0014	-

5 Conclusion

In this paper, an Equidistribution algorithm has been applied to distribute the central nodes in adaptive modes to RBF methods. To have some smoothness properties, by the e.d. algorithm, the central nodes satisfying in a given constraint are obtained. This method was applied to two nonlinear time-dependent partial differential equations by MMOL. In numerical examples, the results obtained by uniform nodes, and adaptive nodes with, and without the constraint have been compared. The numerical results in Example 1, reveal that with adaptive nodes, a more accurate approximate solution has been obtained. Our numerical experience shows that, in this example, to achieve the accuracy as good as adaptive nodes, at least 150 uniform nodes must be applied. Also in Example 2, with 110 uniform nodes, the obtained results by adaptive nodes with constraint have better accuracy. With 150 center nodes a good accuracy has been obtained by three distributions. The numerical results in the examples illustrate the efficiency of adaptive nodes to solving some nonlinear PDEs with MMOL. The results show that applying the adaptive central nodes is more accurate in the problems with speed gradient functions.

Acknowledgement The Authors are grateful to reviewers for their constructive and helpful comments, which helped to improve the paper.

References

1. Behrens, J and Iske, A. *Grid-free adaptive semi-Lagrangian advection using radial basis functions*, Computers & Mathematics with Applications, 43 (3–5) (2002) 319–327.

2. Belytschko, T., Krongauze, Y., Organ, D., Fleming, M. and Krysl, P *Meshless methods: An overview and recent developments*, Computer Methods in Applied Mechanics and Engineering, 139 (1996) 3–47 (special issue on Meshless Methods).

3. Bozzini, M., Lenarduzzi, L. and Schaback, R. *Adaptive interpolation by scaled multiquadrics*, Advances in Computational Mathematics, 16(4) (2002) 375-387.

4. Cao, W., Huang, W. and Russell, R. D. *A study of monitor functions for two dimensional adaptive mesh generation*, SIAM Journal on Scientific Computing, 20(6)(1999) 1978–1994.

5. Carey, G. F. and Dinh, H. T. *Grading functions and mesh redistribution*, SIAM Journalon Numerical Analysis , 22(5)(1985) 1028–1040.

6. Fasshauer, G. E. *Mesh free Approximation Methods with MATLAB.* World Scientific Co. Pte. Ltd., Singapore, 2007.

7. Ferreira, A. J. M., Kansa, E. J., Fasshauer, G. E. and Leitao, V. M. A. *Progress on Meshless Methods*, Computational Methods in Applied Sciences, Springer 2009.

8. Hon, Y. C. *Multiquadric collocation method with adaptive technique for problems with boundary layer*, International Journal of Applied Science and Computations, 6(3) (1999) 173–184.

9. Hon, Y. C., Chen, C. S. and Schaback, R. *Scientific Computing with Radial Basis Functions.* Draft version 0.0, Cambridge, 2003.

10. Hon, Y. C, Schaback, R. and Zhou, X. *An adaptive greedy algorithm for solving large RBF collocation problems*, Numerical Algorithms, 32(1) (2003)13–25.

11. Kansa, E. J. *Multiquadrics-A scattered data approximation scheme with applications to computational fuid-dynamics-I surface approximations and partial derivative estimates*, Computer and Mathematics with Appllications, 19 (1990) 127–145.

12. Kansa, E. J. *Multiquadrics-A scattered data approximation scheme with applications to computational fuid dynamics- II. Solution to parabolic, hyperbolic and elliptic partial differential equations*, Computer and Mathematics with Appllications, 19 (1990) 147–161.

13. Kautsky, J. and Nichols, N. K. *Equi-distributing meshes with constraints*, SIAM Journal on Scientific and Statistical Computing, 1(4) (1980) 449-511.

14. Quan, S. *A meshless method of lines for the numerical solution of KdV equation using radial basis functions*, Engineering Analysis with Boundary Elements, 33 (2009) 1171–1180.

15. Sanz-Serna, J. and Christie, I. *A simple adaptive technique for nonlinear wave problems*, Journal of Computational Physics, 67 (1986) 348-360.

16. Sarra, S. A. *Adaptive radial basis function methods for time dependent partial differential equations*, Applied Numerical Mathematics, 54 (1) (2005) 7994.

17. Schaback, R. and Wendland, H. *Adaptive greedy techniques for approximate solution of large RBF systems*, Numerical Algorithms, 24(3) (2000)239–254.

18. Schiesser, W. E. *The numerical method of lines: integration of partial differential equations*, San Diego, California: Academic Press; 1991.

Iranian Journal of Numerical Analysis and Optimization
Vol. 5, No. 2, (2015), pp 59-73

Application of modified simple equation method to Burgers, Huxley and Burgers-Huxley equations

Z. Ayati*, M. Moradi and M. Mirzazadeh

Abstract

In this paper, modified simple equation method has been applied to obtain generalized solutions of Burgers, Huxley equations and combined forms of these equations. The new exact solutions of these equations have been obtained. It has been shown that the proposed method provides a very effective, and powerful mathematical tool for solving nonlinear partial differential equations.

Keywords: Modified simple equation method; Burgers equation; Huxley equation; Burger-Huxley equation.

1 Introduction

Mathematical modeling of many real phenomena leads to a non-linear partial differential equations in various fields of sciences and engineering. Many powerful methods have been presented for solving PDEs so far, such as tanh-function method [19] and [28], sine-cosine method [29], Homotopy Analysis method [17], Homotopy perturbation method [6], variational iteration method [9] and [10], Adomian decomposition method [1], Exp-function method [1], [11], [36] and [37], simplest equation method [7] and [4], and many others. Most recently, a modification of simplest equation method (MSE method) has been developed to obtain solutions of various nonlinear

*Corresponding author
Received 3 June 2014; revised 7 April 2015; accepted 17 June 2015
Z. Ayati
Department of Engineering Sciences, Faculty of Technology and Engineering East of Guilan, University of Guilan, Rudsar-Vajargah, Iran. e-mail: zainab.ayati@guilan.ac.ir ; Ayati.zainab@gmail.com

M. Moradi
Department of Engineering Sciences, Faculty of Technology and Engineering East of Guilan, University of Guilan, Rudsar-Vajargah, Iran. email:

M. Mirzazadeh
Department of Engineering Sciences, Faculty of Technology and Engineering East of Guilan, University of Guilan, Rudsar-Vajargah, Iran. email:

evolution equations [14], [15], [21], [31], [32], [33] and [34]. The present paper is motivated by the desire to extend the MSE method to obtain generalized solutions of Burgers, Huxley, and Burgers-Huxley. The procedure of this method, by the help of Matlab, Maple or any mathematical package, is of utter simplicity.

2 The MSE method

Consider a nonlinear partial equation in two independent variables, say x and t , in the form of

$$P(u, u_t, u_x, u_{tt}, u_{xx}, ...) = 0, \tag{1}$$

where $u = u(x, t)$ an unknown function, P is a polynomial in $u = u(x, t)$ and its various partial derivatives, in which the highest order derivatives and nonlinear terms are involved. This method consists of the following steps.

Step 1. Using the transformation

$$\xi = x + wt, \tag{2}$$

where w is constant, we can rewrite equation (1) as a following nonlinear ODE:

$$Q(u, u', u'', ...) = 0. \tag{3}$$

Where the superscripts denote the derivatives with respect to ξ .

Step 2. Suppose that the solution of equation (2) can be expressed as follows

$$u(\xi) = \sum_{i=0}^{m} a_i (\frac{F'(\xi)}{F(\xi)})^i. \tag{4}$$

Where a_i are constants to be determined later, with $a_m \neq 0$ and $F(\xi)$ is an unknown function to be determined later.

Step 3. The positive integer m can be determined by considering the homogeneous balance of the highest order derivatives and highest order nonlinear appearing in equation (2).

Step 4. Calculating all necessary derivatives $u', u'', u''', ...$, and substituting equation (3) into equation (2) yields a polynomial of $\frac{F'(\xi)}{F(\xi)}$ and its derivatives. Equating the coefficients of same power of $F^{-i}(\xi)$ to zero gives a system of equations which can be used to solve for determining unknown constants,$F(\xi)$

and $F'(\xi)$. By substituting obtained results into equation (3), solutions of the equation (1) can be obtained.

3 Application of the MSE method

In this section, the modified simple equation method has been applied to obtain generalized solutions of Burgers, Huxley, and Burgers-Huxley.

3.1 Application MSE method to Burgers equation

The Burgers equation is a nonlinear partial differential equation of second order of the form

$$u_t + uu_x = \nu u_{xx}. \tag{5}$$

Where ν is the viscosity coefficient [2], [22], [23] and [27]. Many problems can be modeled by the Burgers‘ equation. This equation is one of the very few nonlinear partial differential equations which can be solved exactly for the restricted set of initial function. The study of the general properties of the equation has drawn considerable attention due to its place of application in some fields such as gas dynamics, heat conduction, elastically, etc.
To apply MSE method on equation (4), lets introduce a variable ξ , defined as

$$\xi = x - wt. \tag{6}$$

So, equation (4) turns to the following system of ordinary different equation,

$$-wu' + uu' = \nu u''. \tag{7}$$

Where w is constant to be determined. Integrating (7) and considering the integral constant to be zero, we obtain

$$-wu + \frac{1}{2}u^2 = \nu u'. \tag{8}$$

Suppose that the solution of ODE equation (8) can be expressed by a polynomial in $\frac{F'}{F}$ as shown in (3). Balancing the terms u^2 and u' in equation (8), yields to $m = 1$. So we can write (3) as the following simple form

$$u(\xi) = a_1 \frac{F(\xi)}{F(\xi)} + a_0, a_1 \neq 0. \tag{9}$$

So

$$u' = a_1(\frac{F''}{F} - (\frac{F'}{F})^2). \tag{10}$$

Substituting (9) and (10) into equation (8) and equating each coefficient of $F^{-i}(\xi)$, $(i = 0, 1, 2)$ to zero, we derive

$$-wa_0 + \frac{1}{2}a_0^2 = 0, \tag{11}$$

$$(-w + a_0)F' - \nu F'' = 0, \tag{12}$$

$$(\frac{1}{2}a_1^2 + \nu a_1)(F')^2 = 0. \tag{13}$$

By solving equations (11) and (13), the following results will be obtained

$$a_0 = 0, 2w, a_1 = -2\nu.$$

Case 1. when equation (12) turns to

$$wF' + \nu F'' = 0.$$

So

$$F' = Ae^{\frac{-w}{\nu}\xi}. \tag{14}$$

Where A is a arbitrary constant. Integrating (13) with respect ξ, $F(\xi)$ will be obtained as fallows

$$F = \frac{-A\nu}{w}e^{\frac{-w}{\nu}\xi} + B,$$

where B is a constant of integration. Now, the exact solution of equation (4) has the form

$$u_1(x,t) = \frac{-2\nu Ae^{\frac{-w}{\nu}(x-wt)}}{\frac{-A\nu}{w}e^{\frac{-w}{\nu}(x-wt)} + B}.$$

Case 2. when $a_0 = 2w$, equation (12) yields to

$$wF' - \nu F'' = 0.$$

So

$$F' = Ae^{\frac{w}{\nu}\xi},$$

and

$$F = \frac{A\nu}{w}e^{\frac{w}{\nu}\xi} + B.$$

Now, the exact solution of equation (4) has the form

$$u_2(x,t) = 2w - \frac{2w\nu A e^{\frac{w}{\nu}(x-wt)}}{A\nu e^{\frac{w}{\nu}(x-wt)} + wB} = \frac{2w^2 B}{A\nu e^{\frac{w}{\nu}(x-wt)} + wB}.$$

Note that all obtained solutions have been checked with maple 13 by putting into the original equation and found correct.

3.2 Application MSE method to Huxley equation

Now we will bring to bear the MSE method to obtain exact solution to the Huxley equation [3], [7], [8], [12], [13], [18], [25] and [26] in the following form

$$u_t = u_{xx} + u(k-u)(u-1). \tag{15}$$

The Huxley equation is an evolution equation that describes the nerve propagation in biology from which molecular CB properties can be calculated. It also gives a phenomenological description of the behaviour of the myosin heads II. This equation has many fascinating phenomena such as bursting oscillation [3], interspike [18], bifurcation, and chaos [35]. A generalized exact solution can gain an insight into these phenomena. There is no universal method for nonlinear equations. In this part, the exact solution will be obtained by the MSE method.

By considering (6), equation (15) turns to the following ordinary differential equation,

$$wu' + u'' + u(k-u)(u-1) = 0. \tag{16}$$

Balancing the terms u'' and u^3 in equation (16), yields to $m = 1$. So we can rewrite (3) as the following simple form

$$u(\xi) = a_1 \frac{F(\xi)}{F(\xi)} + a_0, a_1 \neq 0. \tag{17}$$

Now by substituting 19) into equation (16) and equating each coefficient of $F^{-i}(\xi)$,$(i = 0, 1, 2, 3)$ to zero, the following result will be obtained

$$a_0^3 - (k+1)a_0^2 + ka_0 = 0, \tag{18}$$

$$a_1 F''' + wa_1 F'' + (2(k+1)a_0 a_1 - 3a_0^2 a_1 - ka_1)F' = 0, \tag{19}$$

$$-3a_1 F' F'' + ((k+1)a_1 - wa_1 - 3a_0 a_1^2)(F')^2 = 0, \tag{20}$$

$$(2a_1 - a_1^3)(F')^3 = 0. \tag{21}$$

Solving equations (15) and (22), we drive

$$a_0 = 0, 1, k,$$

$$a_1 = \pm\sqrt{2}.$$

Case 1. when $a_0 = 0$, equations (16) and (17) yield

$$F''' + wF'' - kF' = 0, \tag{22}$$

$$3F'' + (w - k - 1)F' = 0. \tag{23}$$

By substituting equation (23) into (22), we obtain

$$F''' + (w + \frac{3k}{w - k - 1})F'' = 0.$$

So

$$F'' = Ae^{-\alpha\xi}. \tag{24}$$

Where $\alpha = w + \frac{3k}{w-k-1}$ and A is a arbitrary constant. Therefore, we have

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi} + B. \tag{25}$$

where A and B are arbitrary constants. By substituting (25) into equations (22) and (23), we get

$$w = \frac{k+1}{4} \pm \frac{3}{4}\sqrt{k^2 - 6k + 1}, B = 0.$$

Thus, (25) can be rewritten as follows

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi}. \tag{26}$$

Integrating (26) with respect ξ , $F(\xi)$ will be obtained as follows

$$F = \frac{A}{\alpha^2}e^{-\alpha\xi} + C,$$

where C is a constant of integration. Substituting the value of F and F' into equation (19), the following exact solution of equation (16) has been obtained

$$u_1(x,t) = \frac{\pm\sqrt{2}A\alpha e^{-\alpha\xi}}{Ae^{-\alpha\xi} + \alpha^2 C}.$$

Case 2. when $a_0 = 1$, equations (16) and (17) turns to

$$F''' + wF'' + (k - 1)F' = 0, \tag{27}$$

$$3F'' + ((w - k - 1 \pm 3\sqrt{2})F' = 0. \tag{28}$$

By substituting equation (8) into (27), we obtain

$$F''' + (w - \frac{3(k-1)}{w-k-1 \pm 3\sqrt{2}})F'' = 0.$$

So

$$F'' = Ae^{-\alpha\xi}, \tag{29}$$

where $\alpha = w - \frac{3(k-1)}{w-k-1\pm 3\sqrt{2}}$ and A is a arbitrary constant. Therefore, we have

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi} + B, \tag{30}$$

where A and B are arbitrary constants. By substituting (30) into equations (27) and (28), we get

$$w = \frac{k+1}{4} - \frac{3}{4}\sqrt{2} \pm \frac{3}{4}\sqrt{k^2 - 6k - 6k\sqrt{2} + 27 - 6\sqrt{2}}, B = 0,$$

or

$$w = \frac{k+1}{4} + \frac{3}{4}\sqrt{2} \pm \frac{3}{4}\sqrt{k^2 - 6k + 6k\sqrt{2} + 27 + 6\sqrt{2}}, B = 0.$$

Thus, (30) can be rewritten as follows

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi}. \tag{31}$$

Integrating (31) with respect , will be obtained as follows

$$F = \frac{A}{\alpha^2}e^{-\alpha\xi} + C,$$

where C is a constant of integration. Substituting the value of F and F' into equation (19), the following exact solution of equation (16) has been obtained

$$u_2(x,t) = 1 + \frac{\pm\sqrt{2}A\alpha e^{-\alpha\xi}}{Ae^{-\alpha\xi} + \alpha^2 C}.$$

Case 3. when $a_0 = k$, equations (16) and (17) turn to

$$F''' + wF'' + k(1-k)F' = 0,$$

$$3F'' + ((w - 1 + k(-1 \pm 3\sqrt{2}))F' = 0.$$

By using the same method applied in case 1, the following solution will be obtained

$$u_3(x,t) = k + \frac{\pm\sqrt{2}A\alpha e^{-\alpha\xi}}{Ae^{-\alpha\xi} + \alpha^2 C},$$

where

$$\alpha = w - \frac{3k(1-k)}{w-1+k(-1\pm 3\sqrt{2})},$$

and

$$w = \frac{1-k(-1\pm 3\sqrt{2})}{4} \pm \frac{3}{4}\sqrt{1-2k(-1\pm 3\sqrt{2})+k(-1\pm 3\sqrt{2})^2-8k^2+8k}.$$

3.3 Application MSE method to Burgers-Huxley equation

The analysis presented in this part is based on the generalized nonlinear Burgers-Huxley equation,

$$u_t = u_{xx} + uu_x + u(k-u)(u-1), \tag{32}$$

which models the interaction between reaction mechanisms, convection effects and diffusion transports [20], and some special cases of the equation, which usually appear in mathematical modelling of some real world phenomena. It also gives a phenomenological description of the behaviour of the myosin heads II [30] and Fitzhugh-Nagoma equation, an important nonlinear reaction-diffusion equation used in circuit theory, biology and population genetics [5].

By considering (6), equation (32) turns to the following ordinary differential equation,

$$wu' + u'' + uu' + u(k-u)(u-1) = 0. \tag{33}$$

Balancing the terms u'' and u^3 in Eq. (33), yields to $m = 1$. So we can rewrite (3) as the following simple form

$$u(\xi) = a_1(\frac{F'}{F}) + a_0, a_1 \neq 0. \tag{34}$$

Now by substituting (34) into equation (33) and equating each coefficient of $F^{-i}(\xi)$, $(i = 0, 1, 2, 3)$ to zero, the following result will be obtained

$$a_0^3 - (k+1)a_0^2 + ka_0 = 0, \tag{35}$$

$$a_1F''' + (wa_1 + a_0a_1)F'' + (2(k+1)a_0a_1 - 3a_0^2a_1 - ka_1)F' = 0, \tag{36}$$

$$(-3a_1 + a_1^2)F'F'' + ((k+1)a_1 - wa_1 - a_0a_1 - 3a_0a_1^2)(F')^2 = 0, \tag{37}$$

$$(2a_1 - a_1^2 - a_1^3)(F')^3 = 0. \tag{38}$$

Solving equation (35) and (38), we get

$$a_0 = 0, 1, k,$$

$$a_1 = 1, -2.$$

Case 1. when $a_0 = 0$ and $a_1 = 1$ equations (36) and (37) yield

$$F''' + wF'' - kF' = 0, \tag{39}$$

$$2F'' + (w - k - 1)F' = 0. \tag{40}$$

By substituting equation (40) into (39), we obtain

$$F''' + (w + \frac{2k}{w - k - 1})F'' = 0.$$

So

$$F'' = Ae^{-\alpha\xi} \tag{41}$$

where $\alpha = w + \frac{2k)}{w-k-1}$ and A is a arbitrary constant. Therefore, we have

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi} + B, \tag{42}$$

where A and B are arbitrary constants. By substituting (42) into equations (39) and (40), we get

$$w = \pm(k - 1), B = 0,$$

So

$$\alpha = -1, -k.$$

Thus, (42) can be rewritten as follows

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi}. \tag{43}$$

Integrating (43) with respect ξ , $F(\xi)$ will be obtained as follows

$$F' = \frac{A}{\alpha^2}e^{-\alpha\xi} + C,$$

where C is a constant of integration. Substituting the value of F and F' into equation (34), the following exact solutions of equation (33) has been obtained

$$u_1(x,t) = \frac{Ae^{x-(k-1)t}}{Ae^{x-(k-1)t} + C},$$

$$u_2(x,t) = \frac{AKe^{k(x+(k-1)t)}}{Ae^{k(x+(k-1)t)} + k^2C}.$$

Case 2. when $a_0 = 0$, and $a_1 = -2$ equations (36) and (37) yield

$$F''' + wF'' - kF' = 0, \tag{44}$$

$$-5F'' + (w - k - 1)F' = 0. \tag{45}$$

By substituting equation (45) into (44), we obtain

$$F''' + (w + \frac{5k)}{w - k - 1})F'' = 0.$$

By using the same method applied in case 1, the following solution will be obtained

$$u_{3,4}(x,t) = \frac{2A\alpha e^{-\alpha\xi}}{Ae^{-\alpha\xi} + \alpha^2 C},$$

where

$$\alpha = w + \frac{5k}{w - k - 1},$$

and

$$w = \frac{3}{8}(k+1) \pm \frac{5}{8}\sqrt{k^2 - 14k + 1}.$$

Case 3. when $a_0 = 1$ and $a_1 = 1$, equations (31) and (32) turn to

$$F''' + (w+1)F'' + (k-1)F' = 0, \tag{46}$$

$$2F'' + (w + 3 - k)F' = 0. \tag{47}$$

By substituting equation (47) into (46), we obtain

$$F''' + (w + 1 - \frac{2(k-1)}{w+3-k})F'' = 0.$$

So

$$F'' = Ae^{-\alpha\xi},$$

where $\alpha = w + 1 - \frac{2(k-1)}{w+3-k}$ and A is an arbitrary constant. Therefore, we have

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi} + B, \tag{48}$$

where A and B are arbitrary constants.
By substituting (48) into equations (46) and (47), we get

$$w = -1 \pm k, B = 0.$$

Thus, the following exact solutions of equation (33) has been obtained.

$$u_5(x,t) = 1 - \frac{Ae^{-x+(1-k)t}}{Ae^{-x+(1-k)t} + C},$$

$$u_6(x,t) = 1 - \frac{A(1-k)e^{-(1-k)(x+(1+k)t)}}{Ae^{-(1-k)(x+(1+k)t)} + (1-k)^2C}.$$

Case 4. when $a_0 = 1$ and $a_1 = -2$, by using the same method applied in case 1, the following solution will be obtained.

$$u_{7,8}(x,t) = 1 + \frac{2A\alpha e^{-\alpha\xi}}{Ae^{-\alpha\xi} + \alpha^2 C}.$$

Where

$$\alpha = w + 1 - \frac{5(k-1)}{w-6-k},$$

and

$$w = \frac{13+3k}{8}(k+1) \pm \frac{5}{8}\sqrt{k^2+30k+33}.$$

Case 5. when $a_0 = k$ and $a_1 = 1$, equations (31) and (32) turn to

$$F''' + (w+k)F'' + k(1-k)F' = 0, \tag{49}$$

$$2F'' + (w+3k-1)F' = 0. \tag{50}$$

By substituting equation (50) into (49), we obtain

$$F''' + (w + k - \frac{2k(1-k)}{w+3k-1})F'' = 0.$$

So

$$F'' = Ae^{-\alpha\xi},$$

where $\alpha = w + k - \frac{2k(1-k)}{w+3k-1}$ and A is an arbitrary constant. Therefore, we have

$$F' = -\frac{A}{\alpha}e^{-\alpha\xi} + B, \tag{51}$$

where A and B are arbitrary constants. By substituting (51) into equations (49) and (50), we get

$$w = -k \pm 1, B = 0.$$

Thus, the following exact solutions of equation (32) has been obtained

$$u_9(x,t) = k - \frac{Ake^{-k(x+(k-1)t)}}{Ae^{-k(x+(k-1)t)} + k^2C},$$

$$u_{10}(x,t) = k - \frac{A(k-1)e^{-(k-1)(x+(k+1)t)}}{Ae^{-(k-1)(x+(k+1)t)} + (k-1)^2 C}.$$

Case 6. when $a_0 = k$ and $a_1 = -2$, by using the same method applied in case 5, the following solution will be obtained.

$$u_{11,12}(x,t) = k + \frac{2A\alpha e^{-\alpha\xi}}{Ae^{-\alpha\xi} + \alpha^2 C}.$$

Where

$$\alpha = w + k - \frac{5k(1-k)}{w - 1 - 6k},$$

and

$$w = \frac{13+3k}{8}(k+1) \pm \frac{5}{8}\sqrt{33k^2 + 30k + 1}.$$

4 Conclusion

In this paper, modified simple equation method has been applied to obtain the generalized solutions of some nonlinear partial differential equation. The results show that modified simple equation method is a powerful tool for obtaining the exact solutions of nonlinear differential equations. It may be concluded that, the method can be easily extended to all kinds of nonlinear equations. The advantage of this method over other methods is that in most methods applied for the exact solution of partial differential equations such as Exp-function method, $\frac{G'}{G}$ - expansion method, tanh-function method, and so on, the solution is presented in terms of some pre-defined functions, but in the MSE method, $F(\xi)$ is not pre-defined or not a solution of any pre-defined equation. Therefore, some new solutions might be found by this method.The computations associated in this work were performed by Maple 13.

References

1. Biazar, J., Babolian, E., Nouri, A. and Islam, R. *An alternate algorithm for computing Adomian Decomposition method in special cases*, Applied Mathematics and Computation 38 (2003)523–529.

2. Burgers, M. *The Nonlinear Diffusion Equation*, Reidel, Dordtrecht, 1974.

3. Duan, L. X. and Lu, Q. S. *ursting oscillations near codimension-two bifurcations in the Chay Neuron model*, International Journal of Nonlinear Sciences and Numerical Simulation 7(1) (2006) 59–64.

4. Ebaid, A. *Exact solitary wave solutions for some nonlinear evolution equations via Exp-function method*, Phys.Lett. A 365 (2007) 213–219.

5. Hariharan, G. and Kannan, K. *Haar wavelet method for solving FitzHugh-Nagumo equation*, International Journal of Mathematical and Statistical Sciences 2 (2) (2010) 59–63.

6. He, J.H. *Application of homotopy perturbation method to nonlinear wave equations*, Chaos Solitons Fractals 26 (2005) 695–700.

7. He, J. H. *Modified Hodgkin-Huxley model*, Chaos, Solitons,and Fractals 29(2) (2006) 303–306.

8. He, J. H. *Resistance in cell membrane and nerve fiber*, Neuroscience Letters 373(1) (2005) 48–50.

9. He, J. H. *Variational iteration method for autonomous ordinary differential systems*, Appl. Math. Comput. 114 (2000) 115–123.

10. He, J. H. *Variational iteration method - a kind of non-linear analytical technique: some examples*, Int. J. Nonlinear Mech. 34 (1999) 699–708.

11. He, J. H. and Wu, X. H. *Exp-function method for nonlinear wave equations*, Chaos Solitons Fractals, 30 (2006) 700–708.

12. He, J. H. and Wu, X. H. *Modified Morris-Lecar model for interacting ion channels*, Neurocomputing 64 (2005) 543–545.

13. Hodgkin, A. L. and Huxley, A. F. *A quantitative description of membrane current and its application to conduction and excitation in nerve*, The Journal of Physiology 117(4) (1952) 500–544.

14. Jawad, A. J. M., Petkovic, M. D. and Biswas, A. *Modified simple equation method for nonlinear evolution equations*, Appl. Math. Comput. 217 (2010) 869–877.

15. khan, K. and Akbar, M. A. *Exact and solitary wave solutions for the Tzitzeica-Dodd-Bullough and the modified Kdv-Zakharov-Kuznetsov equations using the modified simple equation method*, Ain Shams Engineering Journal 4(4) (2013)903–909.

16. Kudryashov, N. A. *Nadejda B. Loguinova, Extended simplest equation method for nonlinear differential equations*, Applied Mathematics and Computation 205 (2008) 396–402.

17. Liao, S. J. *The proposed homotopy analysis techniques for the solution of nonlinear problems*, Ph.D. dissertation. Shanghai: Shanghai Jiao Tong,University; 1992 [in English].

18. Liu, S. Q., Fan, T. and Lu, Q. S. *The spike order of the winnerless competition -WLC-model and its application to the inhibition neural system*, International Journal of Nonlinear Sciences and Numerical Simulation 6(2) (2005) 133–138.

19. Malfliet, W. and Hereman, W. *The tanh method: I. Exact solutions of nonlinear evolution and wave equations*, Phys Scr 54 (1996) 563–568.

20. Satsuma, J. *Topics in soliton theory and exactly solvable nonlinear equations*, Singapore:World Scientific (1987).

21. Taghizadeh, N., Mirzazadeh, M., Samiei Paghaleh, A. and Vahidi, J. *Exact solutions of nonlinear evolution equations by using the modified simple equation method*, Ain Shams Engineering Journal 3 (2012) 321–325.

22. Veksler, A. and Zarmi, Y. *Wave interactions and the analysis of the perturbed Burgers equation*, Physica D 211 (2005) 57–73.

23. Veksler, A. and Zarmi, Y. *Freedom in the expansion and obstacles to integrability in multiple-soliton solutions of the perturbed KdV equation*, Physica D 217 (2006) 77–87.

24. Vitanov N. K. *Modified method of simplest equation:powerful tool for obtaining exact and approximate traveling-wave solutions of nonlinear PDEs*, Commun Nonlinear Sci Numer Simulat 16 (2011) 1176–85.

25. Wang, J., Chen, L. and Fei, X. *Analysis and control of the bifurcation of Hodgkin-Huxley model*, haos, Solitons, and Fractals 31(1) (2007) 247–256.

26. Wang, J., Chen, L. and Fei, X. *Bifurcation control of the Hodgkin-Huxley equations*, Chaos, Solitons and Fractals 33(1) (2007) 217–224.

27. Wazwaz, A. M. *Analytic study on Burgers, Fisher, Huxley equations and combined forms of these equations*, Applied Mathematics and Computation 195 (2008) 754–761.

28. Wazwaz, A. M. *The tanh method: Exact solutions of the Sine-Gordon and Sinh-Gordon equations*, Appl. Math. Comput. 167 (2005) 1196–1210.

29. Wazwaz A. M. *The tanh and the sine-cosine methods for a reliable treatment of the modi.ed equal width equation and its variants*, Commun Nonlinear Sci Numer Simul 11 (2006) 148–60.

30. Wazwaz A. M. *Travelling wave solutions of generalized forms of Burgers, Burgers-KdV and Burgers-Huxley equations*, Applied Mathematics and Computation 169 (2005) 639–656.

31. Zayed, E. M. E. *A note on the modified simple equation method applied to Sharma-Tasso-Olver equation*, Appl. Math. Comput. 218 (2011) 3962–3964.

32. Zayed, E. M. E. *The modified simple equation method for two nonlinear PDEs with power law and Kerr law nonlinearity, Pan American Mathematical Journal*, International Publications USA 24(1) (2014) 65–74.

33. Zayed, E. M. E. *The modified simple equation method applied to nonlinear two models of diffusion-reaction equations*, Journal of Mathematical Research and Applications 2(2) (2014) 5–13.

34. Zayed, E. M. E. and Ibrahim, S. A. H. *Exact solutions of nonlinear evolution equations in mathematical physics using the modified simple equation method*, Chinese Phys. Lett. 29 (6) (2012) p. 060201.

35. Zhang, G. J., Xu, J. X., Yao, H. and Wei, R. *Mechanism of bifurcation-dependent coherence resonance of an excitable neuron model*, International Journal of Nonlinear Sciences and Numerical Simulation 7(4) (2006) 447–450.

36. Zhang, S. *Application of Exp-function method to a KdV equation with variable coefficients*, Phys. Lett. A 365(2007) 448–453.

37. Zhu, S. D. *Exp-function method for the discrete mKdV lattice*, Int. J. Nonlinear Sci. Numer. Simul. 8 (2007) 465–469.

Iranian Journal of Numerical Analysis and Optimization
Vol. 5, No. 2, (2015), pp 75-93

On convergence and stability conditions of homotopy perturbation method for an inverse heat conduction problem

Q. Jannati and A. Zakeri*

Abstract

In this paper, we investigate the application of the Homotopy Perturbation Method (HPM) for solving a one-dimensional nonlinear inverse heat conduction problem. In this problem the thermal conductivity term is a linear function with respect to unknown heat temperature in bounded interval. Furthermore, the temperature histories are unknown at the end point of the interval. This problem is ill-posed. So, using the finite difference scheme and discretizing the time interval, the partial differential equation is reduced into a System of Nonlinear Ordinary Differential Equations (SNODE's). Then, using HPM, the approximated solution of the obtained Ordinary Differential Equation (ODE) system is determined. In the sequel, the stability and convergence conditions of the proposed method are investigated. Finally, an upper bound of the error is provided.

Keywords: Homotopy perturbation method; Diffusion equation; Discretizing method; Inverse problem.

1 Introduction

Inverse heat conduction problems are used to describe many important phenomena in physics, chemistry, mechanics, etc. There has been a great amount of investigation to solving inverse heat conduction problems in one and multi dimensional spaces. Many effective methods have been provided. However, lots of inverse heat transfer problems, which arise in natural phenomena, such as radiational heat transfer, modelling of case hardening, gravimetry,

*Corresponding author
Received 26 October 2014; revised 8 February 2015; accepted 25 July 2015
Q. Jannati
Faculty of Mathematics, K. N. Toosi University of Technology, Tehran, Iran. e-mail: jannati@dena.kntu.ac.ir

A. Zakeri
Faculty of Mathematics, K. N. Toosi University of Technology, Tehran, Iran. e-mail: azakeri@kntu.ac.ir

and etc., have nonlinear forms and so they are not solvable with analytical methods, but unfortunately, most of presented methods are useful just for solving linear forms.

Usually these problems are ill-posed in the sense of Hadamard. Therefore, the regularization method is a successive technique for solving ill-posed problems and it may be applied to entire class of problems which arise from physical observations.

Beck et al. have investigated an inverse problem in one-dimensional space with two general procedures, function specification and regularization methods, and a method of combining these, trial function method, and have implemented all of these methods in a sequential manner [3]. Lesnic et al. in [15] and [19] have considered a special case of distributed (identification) parameter problems in one-dimensional spaces. They have shown that for a one-dimensional quasi-heterogeneous material with square-root harmonic conductivity, a single measurement of the conductivity and the flux on the boundary is sufficient to determine uniquely the unknown physical parameters and the solution function. Alivanov considered the solution of inverse problems by analytical approaches [2]. Qu and Dou [20], Lewandowski [18], and Jia et al. [16], have studied the nonlinear diffusion equation and provided some numerical techniques. Shidfar and Zakeri in [21] - [23] have investigated the existence and uniqueness of a solution for a two-dimensional nonlinear inverse diffusion problem. Also, Zakeri et al. have begun their research by a Cauchy inverse problem and found a solution by HMP method [26] and in continuation they have gone on by an inverse heat conduction problem and solved it by the HPM again [27]. Also, they have applied an approach which contained a difference method and the HPM together, and solved the problem with a reliable accuracy [28].

In continuation of above researches our intend is investigation of sufficiently condition for HPM for solving inverse heat conduction problems.

In next section, the HPM is introduced shortly and in Section 3, an approximated solution for the inverse heat conduction problem is obtained via HPM . Then, the stability and convergence of the above mentioned method are studied in Section 3 and Section 4, respectively . Some numerical results are illustrated by some tables and figures in Section 6. Finally conclusions and some suggestions for more research are given in last section.

2 Basic concepts of HPM

In this section we introduce the basic concepts of the HPM , according to [26], in brief.

Consider the following nonlinear equation

$$A(u) - f(r) = 0, \quad r \in \Omega, \tag{1}$$

with the boundary conditions

$$B(u, \partial u/\partial n) = 0, \quad r \in \Gamma,$$

where A is a general differential operator, B is a boundary operator, $f(r)$ is a known analytic function and Γ is the boundary of the domain Ω. The operator A can be generally divided into two parts L and N, where they are the linear and nonlinear parts of A, respectively. So, Eq. (1) converts into the following form

$$L(u) + N(u) - f(r) = 0. \tag{2}$$

In [10], He constructed a homotopy $H : \Omega \times [0, 1] \longrightarrow \mathbb{R}$ which satisfies:

$$H(v, p) = (1 - p)[L(v) - L(v_0)] + p[A(v) - f(r)] = 0, \tag{3}$$

or

$$H(v, p) = L(v) - L(v_0) + pL(v_0) + p[N(v) - f(r)] = 0, \tag{4}$$

where $r \in \Omega$, and $p \in [0, 1]$ is called the homotopy parameter, and v_0, is an initial approximation for the solution of Eq. (1) which satisfies the boundary conditions. Consequently

$$\begin{cases} H(v, 0) = L(v) - L(v_0) = 0, \\ H(v, 1) = A(v) - f(r) = 0. \end{cases}$$

Now, when p varies from 0 to 1, the homotopy $H(v, p)$, changes from $L(v) - L(v_0)$ to $A(v) - f(r)$.

Applying the perturbation technique due to the fact that $0 \leq \ p \ \leq 1$ is considered as a small parameter, we can assume that the solution of Eq. (3) or Eq.(4) can be expressed as a series in the form

$$v = v_0 + pv_1 + p^2 v_2 + \dots \ .$$

When $p \longrightarrow 1$, Eq.(3) or Eq.(4) corresponds to Eq.(2) and so v becomes the approximate solution of Eq. (2) i.e;

$$u = \lim_{p \to 1} v = v_0 + v_1 + v_2 + \dots \ . \tag{5}$$

The series (7) is convergent for most cases and the rate of convergence depends on $A(v)$ [11, 12].

In the next section, a nonlinear inverse heat conduction problem is considered. We discretize the time interval by means of backward finite difference method and apply the HPM. Then, the approximate solution of the problem is yield.

3 Solution of nonlinear inverse heat conduction problem by HPM

Let $T > 0$, and consider the nonlinear parabolic partial differential equation

$$u_t - (D(u)u_x)_x = \Phi(x,t),\ (x,t) \in \Omega = (0,1) \times (0,T), \tag{6}$$

with initial condition

$$u(x,0) = s(x), \quad x \in [0,1], \tag{7}$$

and boundary conditions

$$u(0,t) = f(t), \quad t \in [0,T], \tag{8}$$
$$u(1,t) = h(t), \quad t \in [0,T], \tag{9}$$

where $D(u) = a(t)u + b(t) > 0$, and a, b, s and f are known functions such that, $b(t)$ is far from zero in $[0,T]$.

If h is given, then the problem (6)-(9) is a direct problem which is solvable by means of common numerical and approximation approaches for solving PDEs, such as finite difference method [7], finite element method [7], radial basis functions [4], homotopy perturbation method [13], Adomian decomposition method [1] and so on.

Now, suppose that h is unknown. Then the problem (6)-(9) becomes an inverse problem. Consequently, an overspecified condition, such as

$$u_x(0,t) = g(t), \quad t \in [0,T], \tag{10}$$

where g is a known function, is used.

For positive integer n, let $\triangle t = \frac{1}{k} = \frac{T}{n}$, $t_j = j \triangle t, j \in J_n = \{1,2,...,n\}$. Put $u_0(x) = u(x,0) = s(x)$, $a_j = a(t_j)$, $b_j = b(t_j)$, $\Phi_j(x) = \Phi(x,t_j)$, for any $j \in J_n$, such that they are given fixed nodes. Similarly, we consider the $u_j(x)$, as the approximated value of $u(x,t_j)$, $j \in J_n$.

Using the backward finite difference scheme for the term u_t in the form

$$u_t(x) \simeq\ k(\ u_j(x) - u_{j-1}(x)\ ), \qquad j \in J_n,$$

and substituting in Eq. (6), a system of second order ordinary differential equations with respect to x is obtained. We have

$$k(u_j(x) - u_{j-1}(x)) - \frac{d}{dx}\{(a(t_j)u_j(x) + b(t_j))\frac{d}{dx}u_j(x)\} = \Phi_j(x),\ 1 \le j \le n,$$

or

$$\frac{d^2}{dx^2}u_j(x)\ - \{\frac{k}{b(t_j)}(u_j(x) - u_{j-1}(x)) - \frac{a(t_j)}{b(t_j)}\ (\frac{d}{dx}\ (u_j(x)\frac{d}{dx}u_j(x)\ )\ )\ \}$$

$$= \frac{-1}{b(t_j)}\Phi_j(x). \tag{11}$$

For simplicity, define $\mathbf{u}(x) = (u_1(x), u_2(x), \ldots, u_n(x))^T$. Then we can write Eq. (6) as follows

$$A\mathbf{u} = L_x\mathbf{u} - N\mathbf{u} = \Psi(x,t),$$

where $\Psi(x,t) = (\frac{-\Phi_1(x)}{b(t_1)}, \ldots, \frac{-\Phi_n(x)}{b(t_n)})^T$. Moreover, L_x and N are the linear and nonlinear parts of the operator A, respectively, and are as follows:

$$L_x = \frac{d^2}{dx^2},$$

$$N\mathbf{u} = -\mathbf{M_2}\frac{d}{dx}\ (D(\mathbf{u},\mathbf{u})) + \mathbf{M_1}\mathbf{u} + \mathbf{m},$$

where

$$\mathbf{m} = (\tfrac{-k}{b(t_1)}\ s(x), 0, \ldots, 0)^T{}_{n\times 1}, \qquad \mathbf{M_1} = k \begin{pmatrix} \frac{1}{b(t_1)} & & & 0 \\ \frac{-1}{b(t_2)} & \frac{1}{b(t_2)} & & \\ & \ddots & \ddots & \\ 0 & & \frac{-1}{b(t_n)} & \frac{1}{b(t_n)} \end{pmatrix},$$

$$\mathbf{M_2} = diag\ (\ \tfrac{a(t_1)}{b(t_1)}, \tfrac{a(t_2)}{b(t_2)}, \ \ldots\ , \ \tfrac{a(t_n)}{b(t_n)}\ ),$$

and

$$D(\mathbf{u}(x), \mathbf{v}(x)) = (u_1(x)\tfrac{d}{dx}v_1(x), u_2(x)\tfrac{d}{dx}v_2(x), \ldots, u_n(x)\tfrac{d}{dx}v_n(x))^T.$$

After twice integration of Eq. (6) with respect to x, and applying the conditions (7)- (9), we obtain

$$\mathbf{u}(x) - x\mathbf{g} - \mathbf{f} - \int_0^x\int_0^x N\mathbf{u}\ (x)\ dxdx = \int_0^x\int_0^x \Psi(x)dxdx,$$

where $\mathbf{g} = (\ g(t_1),\ \ldots\ ,\ g(t_n)\ )^T$, and $\mathbf{f} = (\ f(t_1),\ \ldots\ ,\ f(t_n)\ )^T$.

Now, using HPM and [8, 26], we choose a convex homotopy such that

$$\mathbf{H}(\mathbf{v}(x), p) = \mathbf{v}(x) - \mathbf{h}(x) - p\int_0^x\int_0^x N\mathbf{v}(x)\ dxdx = 0, \tag{12}$$

and

$$\mathbf{F}(\mathbf{u}(x)) = \mathbf{u}(x) - \mathbf{h}(x) = 0,$$

where

$$\mathbf{h}(x) = x\mathbf{g} + \mathbf{f} + \int_0^x \int_0^x \Psi(x)\ dxdx,$$

and $\mathbf{v}(x) = (\ v(x,t_1),\ \dots\ v(x,t_n)\ )^T$. Furthermore, Eq. (12) gives

$$\mathbf{v}(x) = \mathbf{h}(x) + p\int_0^x \int_0^x N\mathbf{v}(x)\ dxdx. \tag{13}$$

Combining Eq.s (11) and (16), we obtain the following results

$$\mathbf{v}(x) = x\mathbf{g} + \mathbf{f} + \int_0^x \int_0^x \Psi(x)\ dxdx +$$
$$p\int_0^x \int_0^x \{\mathbf{M}_1\ (\ \mathbf{v}(x) - \mathbf{u}_s(x)\ ) - \mathbf{M}_2\ \frac{d}{dx}\ D(\mathbf{u}(x), \mathbf{u}(x))\}dxdx, \tag{14}$$

where $\mathbf{u}_s(x) = (s(x), u_1(x), \dots, u_{n-1}(x))^T$. Thus, it is concluded that

$$\mathbf{v}_0(x) = \mathbf{h}(x) = x\mathbf{g}+\mathbf{f}+\int_0^x \int_0^x \Psi(x)dxdx = (v_0(x,t_1), \cdots, v_0(x,t_n))^T, \tag{15}$$

and

$$\begin{aligned}\mathbf{v}_1(x) &= \int_0^x \int_0^x \left\{\mathbf{M}_1\ (\mathbf{v}_0(x) - \mathbf{u}_s(x)) - \mathbf{M}_2\ \frac{d}{dx}\ D(\mathbf{v}_0(x), \mathbf{v}_0(x))\ \right\}\ dxdx \\ &= (v_1(x,t_1),\ \cdots\ ,\ v_1(x,t_n))^T. \end{aligned} \tag{16}$$

The above relations are obtained by equating the terms with identical powers of p in Eq. (19). The approximate solution is

$$\mathbf{u}(x) \simeq \mathbf{v}_0(x) + \mathbf{v}_1(x) = \mathbf{v}(x). \tag{17}$$

4 Convergence and stability analysis

In this section, we use continuity of $u(x,t)$ on the compact domain Ω, and prove that $\mathbf{v}(x) \simeq \mathbf{v}_0(x) + \mathbf{v}_1(x)$ depends continuously on the data. Therefore, adding a perturbation term to a, b, f, g and Φ, an upper bound for errors of their solutions are found. In each case, we show that as the perturbation term tends to zero, the solutions errors vanish.

Lemma 1. *Let* $M(t) = \int_0^1 |\Phi(x,t)|dx > 0$ *is a bounded function such that* $M(t) \le M$ *for any* $0 \le t \le T$, *and* $\hat{\boldsymbol{v}}_0(x)$ *correspond to* $\boldsymbol{v}_0(x)$, *where* $b(t)$ *is perturbed by* $\delta b(t)$ *in Equation (6). Then we have*

$$|\hat{v}_0(x,t_j) - v_0(x,t_j)| = \frac{|\delta b(t_j)|}{|b(t_j)(b(t_j) + \delta b(t_j))|}M, \quad 0 \le j \le n, \quad x \in [0,1],$$

consequently, if $|\delta b(t)| \to 0$, *then* $|\hat{v}_0(x,t) - v_0(x,t)| \to 0$, *for any* $t = t_j$, $j = 1, \cdots, n$, *and* $0 \le x \le 1$.

Proof. Using the Equation (15), we have

$$v_0(x,t_j) = f(t_j) + xg(t_j) - \frac{1}{b(t_j)} \int_0^x \int_0^x \Phi_j(x) dx dx. \tag{18}$$

If $b(t_j)$ is perturbed by $\delta b(t_j)$, then

$$\hat{v}_0(x,t_j) = f(t_j) + xg(t_j) - \frac{1}{b(t_j) + \delta b(t_j)} \int_0^x \int_0^x \Phi_j(x) dx dx. \tag{19}$$

Now from Equations (18) and (19), we obtain

$$v_0(x,t_j) - \hat{v}_0(x,t_j) = \frac{-\delta b(t_j)}{b(t_j)(b(t_j) + \delta b(t_j))} \int_0^x \int_0^x \Phi_j(x) dx dx,$$

consequently

$$|v_0(x,t_j) - \hat{v}_0(x,t_j)| \le \frac{|\delta b(t_j)|}{|b(t_j)(b(t_j) + \delta b(t_j))|} \; M, \quad for\ any\ 0 \le j \le n,$$

and this completes the proof. □

Lemma 2. *Let* $M(t)$ *and* M *are as defined in lemma 1, and* $\hat{v}_1(x,t_j)$ *is the value of* $v_1(x,t_j)$ *for any* $t = t_j$, $j = 1, \cdots, n$, *when* $b(t_j)$ *is perturbed by* $b(t_j) + \delta b(t_j)$ *as in problem Equation (6), such that* $\delta b(t_0) = 0$. *If* $|\delta b(t_j)| \to 0$, *then* $|\hat{v}_1(x,t_j) - v_1(x,t_j)| \to 0$, *for any* $0 \le x \le 1$, *and* $1 \le j \le n$.

Proof. Similar to detailed proof presented for lemma 1, assume

$$v_1(x,t_j) = \int_0^x \int_0^x \{\frac{k}{b(t_j)}(v_0(x,t_j) - u_{j-1}(x)) \\ - \frac{a(t_j)}{b(t_j)}(\frac{d}{dx}\{v_0(x,t_j)\frac{d}{dx}v_0(x,t_j)\})\} dx dx.$$

Suppose that $b(t_j)$ is replaced by $b(t_j) + \delta b(t_j)$, for any $j = 1, \cdots, n$. Then we have

$$\hat{v}_1(x,t_j) = \int_0^x \int_0^x \{k\frac{\hat{v}_0(x,t_j) - \hat{u}_{j-1}(x)}{b(t_j) + \delta b(t_j)} \\ - \frac{a(t_j)}{b(t_j) + \delta b(t_j)}(\frac{d}{dx}\{\hat{v}_0(x,t_j)\frac{d}{dx}\hat{v}_0(x,t_j)\})\} dx dx,$$

or

$$
\begin{aligned}
|\hat{v}_1(x,t_j)-v_1(x,t_j)| \leq k\int_0^1\int_0^1 \{&\frac{|\hat{v}_0(x,t_j)-v_0(x,t_j)|}{|b(t_j)+\delta b(t_j)|}+\frac{|\hat{u}_{j-1}(x)-u_{j-1}(x)|}{|b(t_j)+\delta b(t_j)|}\\
&+\frac{|\delta b(t_j)|C}{|b(t_j)(b(t_j)+\delta b(t_j))|}\}dxdx\\
&+\frac{|a(t_j)||\delta b(t_j)|}{|b(t_j)(b(t_j)+\delta b(t_j))|}\,(\frac{M}{|b(t_j)(b(t_j)+\delta b(t_j))|}\\
&\times \max_{\substack{x\in[0,l],\\ j=1,\cdots,n.}}\{|\hat{v}_0(x,t_j)|,|v_0(x,t_j)|\}\\
&+\frac{|v_0^2(x,t_j)|+|f(t_j)^2|+|f(t_j)g(t_j)|}{2|b(t_j)|}),
\end{aligned}
$$

when $C\in\mathbb{R}$ is an upper bound for $|v_0(x,t_j)-u_{j-1}(x)|$. Therefore, for fixed k, from lemma 1, it is derived,

$$\lim_{|\delta b(t_j)|\longrightarrow 0}|\hat{v}_1(x,t_j)-v_1(x,t_j)|\longrightarrow 0,$$

for $0\leq x\leq 1$, and $1\leq j\leq n$. □

Remark 1. By an induction, it is shown that, when $|\delta b(t_j)|\longrightarrow 0$, the second term of the integral in the above inequality vanishes.

Theorem 1. *Suppose that $v(x,t_j)$, $v_0(x,t_j)$, $\hat{v}_0(x,t_j)$, $v_1(x,t_j)$, $\hat{v}_1(x,t_j)$, M, $M(t)$, $b(t_j)$ and $\delta b(t_j)$, are the same as defined in lemmas 1 and 2. Then $v(x,t_j)=v_0(x,t_j)+v_1(x,t_j)$ depends continuously on the data.*

Proof. Obviously, by considering lemmas 1 and 2, the statement of Theorem 1 is proved. □

Theorem 2. *Suppose that $\delta a(t_j)$ is the perturbation term that perturbs $a(t_j)$ to $a(t_j)+\delta a(t_j)$, and v_0, v_1, $\hat{v}_0$, $\hat{v}_1$, M, $M(t)$, $b(t_j)$ and $\delta b(t_j)$ satisfy assumptions of Theorem 1. Then $v(x,t_j)$ depends continuously on the data.*

Proof. The first part of $v(x,t)$ is independent of $a(t)$. Then, using Theorem 1, there is nothing to prove for $v_0(x,t_j)$, $1\leq j\leq n$. Now we just prove that $v_1(x,t_j)$ depends continuously on the data. We have

$$\hat{v}_1(x,t_j)=\int_0^x\int_0^x\{k\frac{\hat{v}_0(x,t_j)-\hat{u}_{j-1}(x)}{b(t_j)+\delta b(t_j)}-\frac{a(t_j)+\delta a(t_j)}{b(t_j)+\delta b(t_j)}(\frac{d}{dx}\{\hat{v}_0(x,t_j)\frac{d}{dx}\hat{v}_0(x,t_j)\})\}dxdx,$$

then, it is concluded that

$$
\begin{aligned}
|\hat{v}_1(x,t_j)-v_1(x,t_j)| \leq k\int_0^1\int_0^1 \{&\frac{|\hat{v}_0(x,t_j)-v_0(x,t_j)|}{|b(t_j)+\delta b(t_j)|}+\frac{|\hat{u}_{j-1}(x)-u_{j-1}(x)|}{|b(t_j)+\delta b(t_j)|}\\
&+\frac{|\delta b(t_j)|}{|b(t_j)(b(t_j)+\delta b(t_j))|}\}dxdx\\
&+\frac{|a(t_j)||\delta b(t_j)|}{|b(t_j)(b(t_j)+\delta b(t_j))|}\,(\frac{M}{|b(t_j)(b(t_j)+\delta b(t_j))|}
\end{aligned}
$$

$$\times \max_{\substack{x\in[0,1],\\ j=1,\cdots,n.}} \{|\hat{v}_0(x,t_j)|, |v_0(x,t_j)|\}$$
$$+ \frac{|v_0^2(x,t_j)| + |f(t_j)^2| + |f(t_j)g(t_j)|}{2|b(t_j)|})$$
$$+ |\delta a(t_j)|\frac{|\hat{v}_0^2(x,t_j)|}{2}.$$

□

Similarly, it is shown that, the approximate solution $\mathbf{v}(x)$ in (17), depends continuously on the data, when $f(t)$, $g(t)$ and $\Phi(x,t)$ are perturbed by small perturbation terms in their domains. So, we give the following theorem.

Theorem 3. *Let $f(t)$, $g(t)$ and $\Phi(x,t)$ be the same as defined in Equation (6). If*

$$f(t) \longmapsto f(t) + \delta f(t),$$
$$g(t) \longmapsto g(t) + \delta g(t),$$
$$\Phi(x,t) \longmapsto \Phi(x,t) + \delta\Phi(x,t),$$

and $v \longmapsto v + \delta v$, then $|\delta v(x,t)| \longmapsto 0$, when

$$\max_{\substack{0\le x\le l\\ 0\le t\le T}} \{|\delta f(t)| + |\delta g(t)| + |\delta\Phi(x,t)|\} \longrightarrow 0.$$

Furthermore, we have

$$|\hat{v}(x,t_j) - v(x,t_j)| \le |\delta f(t_j)| + |\delta g(t_j)| \ + \frac{\|\delta\Phi(x,t_j)\|}{2|b(t_j)|}$$
$$+ \frac{|k|}{|b(t_j)|}\int_0^1\int_0^1 \{|\hat{v}_0(x,t_j) - v_0(x,t_j)| + |\hat{u}_{j-1} - u_{j-1}|\}dxdx$$
$$+ \frac{|a(t_j)|}{2|b(t_j)|}|\hat{v}_0^2(x,t_j) - v_0^2(x,t_j)|.$$

Proof. We have

$$\hat{v}_0(x,t_j) = f(t_j) + \delta f(t_j) + xg(t_j) + x\delta g(t_j) - \frac{1}{b(t_j)}\int_0^x\int_0^x (\Phi(x,t_j) + \delta\Phi(x,t_j))dxdx,$$

and so

$$|\hat{v}_0(x,t_j) - v_0(x,t_j)| \le |\delta f(t_j)| + \ |\delta g(t_j)| + \frac{\|\delta\Phi(x,t_j)\|}{2|b(t_j)|}. \tag{20}$$

So, $|\hat{v}_0(x,t) - v_0(x,t)| \longrightarrow 0$, when

$$\max_{\substack{0\le x\le 1\\ 0\le t\le T}} \{|\delta f(t)| + |\delta g(t)| + |\delta\Phi(x,t)|\} \longrightarrow 0.$$

Similarly, putting

$$\hat{v}_1(x,t_j)=\int_0^x\int_0^x\left\{\frac{k}{b(t_j)}(\hat{v}_0(x,t_j)-\hat{u}_{j-1}(x))-\frac{a(t_j)}{b(t_j)}\frac{d}{dx}(\hat{v}_0(x,t_j)\frac{d}{dx}\hat{v}_0(x,t_j))\right\}dxdx;$$

which via lemma 2 and Equation (20), simplifies to the form

$$\begin{aligned}|\hat{v}_1(x,t_j)-v_1(x,t_j)|\leq{}&\frac{|k|}{|b(t_j)|}\int_0^1\int_0^1\{|\hat{v}_0(x,t_j)-v_0(x,t_j)|+|\hat{u}_{j-1}-u_{j-1}|\}\,dxdx\\&+\frac{|a(t_j)|}{2|b(t_j)|}|\hat{v}_0^2(x,t_j)-v_0^2(x,t_j)|.\end{aligned}\tag{21}$$

Now, $|\hat{v}_1(x,t)-v_1(x,t)|\longrightarrow 0$, whenever

$$\max_{\substack{0\leq x\leq l\\0\leq t\leq T}}\{|\delta f(t)|+|\delta g(t)|+|\delta\Phi(x,t)|\}\longrightarrow 0.$$

By adding the two sides of Equations (20) and (21), we obtain

$$\begin{aligned}|\hat{v}(x,t_j)-v(x,t_j)|\leq{}&|\delta f(t_j)|+|\delta g(t_j)|+\frac{\|\delta\Phi(x,t_j)\|}{2|b(t_j)|}\\&+\frac{|k|}{|b(t_j)|}\int_0^1\int_0^1\{|\hat{v}_0(x,t_j)-v_0(x,t_j)|+|\hat{u}_{j-1}-u_{j-1}|\}dxdx\\&+\frac{|a(t_j)|}{2|b(t_j)|}|\hat{v}_0^2(x,t_j)-v_0^2(x,t_j)|.\end{aligned}$$

Finally,$|\hat{v}(x,t)-v(x,t)|\longrightarrow 0$, whenever

$$\max_{\substack{0\leq x\leq 1\\0\leq t\leq T}}\{|\delta f(t)|+|\delta g(t)|+|\delta\Phi(x,t)|\}\longrightarrow 0.$$

□

In the next section, a necessary condition for convergence of the approximate solution, when the step size Δt, tends to zero is obtainsd.

5 Convergence conditions for the problem (6)-(9)

In this section, we apply the error term of finite difference method in the relations (15), (16) and (17) and then convergence condition of the solution will be investigated. So, first, we give the following theorem.

Theorem 4. *Let* $|\Delta t\ b(t)|>1$ *for any* $0\leq t\leq T$. *If* $|\Delta t|\longrightarrow 0$, *then* $u_j(x)\longrightarrow u(x,t)$, *for any* $0\leq j\leq n,\quad 0\leq x\leq 1$.

Proof. Using Taylor's series expansion, we have

$$v_j(x) \simeq v(x,t_j) + \delta\Phi,$$

where $\delta\Phi = \Delta t \frac{\partial u(x,\theta_j)}{\partial t}$, and $t_j < \theta_j < t_{j+1}$.
So, from (18) we obtain

$$v_0(x,t_j) = f(t_j) + xg(t_j) - \frac{1}{b(t_j)} \int_0^x \int_0^x \Phi_j(x) dxdx,$$

or

$$v_{0j} = f(t_j) + xg(t_j) - \frac{1}{b(t_j)} \int_0^x \int_0^x \Phi_j(x) dxdx + \delta\Phi_j.$$

Clearly, if $\delta\Phi_j \longrightarrow 0$, then $v_{0j} \longrightarrow v_0(x,t_j)$.

Again, by (2) we have

$$v_1(x,t_j) = \int_0^x \int_0^x \frac{k}{b(t_j)}(v_0(x,t_j) - u_{j-1}(x)) - \frac{a(t_j)}{b(t_j)}(\frac{d}{dx}\{v_0(x,t_j)\frac{d}{dx}v_0(x,t_j)\})dxdx.$$

thus

$$\begin{aligned} v_{1j} = & \int_0^x \int_0^x \frac{k}{b(t_j)}\{(v_{0j} - \delta\Phi_j) - (u_{j-1}(x) - \delta\Phi_{j-1})\} \\ & - \frac{a(t_j)}{b(t_j)}(\frac{d}{dx}\{(v_{0j} - \delta\Phi_j)\frac{d}{dx}(v_{0j} - \delta\Phi_j)\})dxdx \\ & + \delta\Phi_j, \end{aligned}$$

and

$$\begin{aligned} v_{1j} = & v_1(x,t_j) - \frac{k}{b(t_j)} \int_0^x \int_0^x (\delta\Phi_j - \delta\Phi_{j-1})dxdx \\ & + \frac{a(t_j)}{b(t_j)}\{\frac{\delta\Phi_j^2}{2} - \int_0^x v_{0j}\frac{d}{dx}\delta\Phi_j dx - \int_0^x \delta\Phi_j \frac{d}{dx}v_{0j}dx\}. \end{aligned}$$

That means if $\delta\Phi_j \longrightarrow 0$, then $v_{1j} \longrightarrow v_1(x,t_j)$, and $v_j \longrightarrow v(x,t_j)$.

□

6 Numerical results

In this section, we give a numerical example.
Let

$$u_t - \frac{\partial}{\partial x}\{(\frac{1}{6}\ e^{-t}u + (t+5)\ e^{-t})\frac{\partial u}{\partial x}\} = -\frac{7}{3}t - 9, \qquad (x,t) \in [0,1] \times [0,1],$$

$$u(x,0) = x^2, \quad 0 \le x \le 1,$$

$$\begin{aligned} u(0,t) &= t, \qquad 0 \le t \le 1, \\ u_x(0,t) &= 0, \qquad 0 \le t \le 1. \end{aligned} \tag{22}$$

Obviously
$\Phi(x,t) = -\frac{7}{3}t - 9, \qquad a(t) = \frac{1}{6}\ e^{-t}, \qquad b(t) = (t+5)\ e^{-t}.$
The exact solution is $u(x,t) = x^2 e^t + t$. We obtain the approximate solution by applying equations (15), (16) and (17), at $x = 0.1, 0.2, \ldots, 1$, where $t = 0.25, 0.5, 0.75, 1$, and we assume that $\triangle t = 0.25$. Consequently the solution will be constructed in the form

$$\mathbf{v}_{0_j}(x) = h(x,t_j) = t_j - \frac{1}{(t_j+5)e^{-t_j}}\ (-\frac{7}{3}t - 9)\frac{x^2}{2},$$
$$\mathbf{v}_{1_j}(x) = \int_0^x \int_0^x \left\{ \frac{4e^{t_j}}{(t_j+5)}(\mathbf{v}_{0_j}(x) - u(x,t_{j-1})) - \frac{\frac{1}{6}}{(t_j+5)}\frac{d}{dx}\left(\mathbf{v}_{0_j}(x)\frac{d}{dx}\mathbf{v}_{0_j}(x)\right)\right\} dxdx,$$

for $j = 1,2,3,4$.
The exact solution, approximate solution and relative error for the above problem are given in Tables $2-5$ and $6-9$ at $t = t_j = j\triangle t, \quad j = 1,2,3,4$.
To illustrate stability, according to Table 1, we enter some noise terms into data functions in Eq. (22).

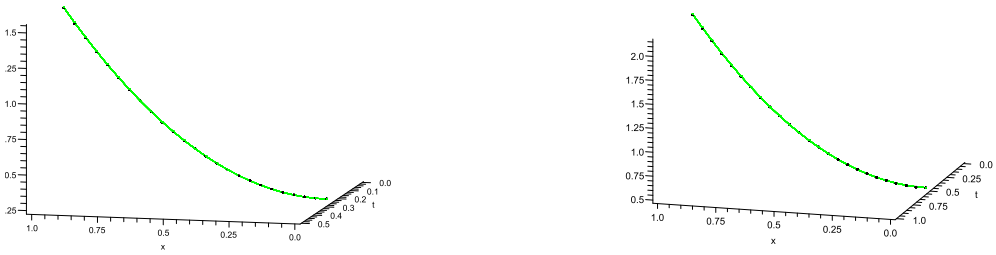

Figure 1: Approximate $(\cdots)$ and exact solution of $u(x,t_j)$ in $t = 0.25$ and $t = 0.5$.

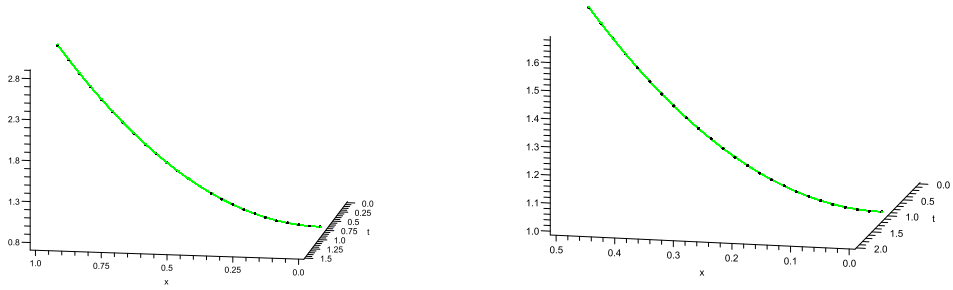

Figure 2: Approximate $(\cdots)$ and exact solution of $u(x,t_j)$ in $t=0.75$ and $t=1$.

Table 1: Perturbation terms in problem (22)

j	a_j	b_j	f_j	g_j	Φ_j
1	-0.006486161986	-0.001002685512	0.001219370604	-0.001002685512	0.001219370604
2	0.01848017099	-0.0008600955762	-0.0002264198307	-0.0008600955762	-0.0002264198307
3	-0.005506853028	0.002919491298	-0.0004964518833	0.002919491298	-0.0004964518833
4	-0.006487155975	-0.001056710212	-0.0004964988898	-0.001056710212	-0.0004964988898

Table 2: Exact and approximate solution of $u_j(x)$ at $t_j=0.25$ with and without perturbation terms in a and b

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	0.2628402542	0.2628483725	3.08×10^{-5}	0.2628562616	6.09×10^{-5}
0.2	0.3013610167	0.3013841491	7.67×10^{-5}	0.3014170267	1.85×10^{-4}
0.3	0.3655622875	0.3655793088	4.65×10^{-5}	0.3656582382	2.62×10^{-4}
0.4	0.4554440667	0.4553871491	1.24×10^{-4}	0.4555397988	2.10×10^{-4}
0.5	0.5710063542	0.5707422861	4.62×10^{-4}	0.5710055740	1.36×10^{-6}
0.6	0.7122491501	0.7115606556	9.66×10^{-4}	0.7119833904	3.37×10^{-4}
0.7	0.8791724543	0.8777395127	1.62×10^{-3}	0.8783850345	8.95×10^{-4}
0.8	1.071776267	1.069157431	2.44×10^{-3}	1.070106259	1.55×10^{-3}
0.9	1.290060588	1.285674303	3.40×10^{-3}	1.287026770	2.35×10^{-3}
1	1.534025417	1.527131339	4.49×10^{-3}	1.529010242	3.26×10^{-3}

Table 3: Exact and approximate solution of $u_j(x)$ at $t_j = 0.5$ with and without perturbation terms in a and b

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	0.5164872127	0.5165050073	3.44×10^{-5}	0.5164663253	4.04×10^{-5}
0.2	0.5659488508	0.5660064728	1.01×10^{-4}	0.5658439475	1.85×10^{-4}
0.3	0.6483849144	0.6484638498	1.21×10^{-4}	0.6480689075	4.87×10^{-4}
0.4	0.7637954034	0.7638099282	1.90×10^{-5}	0.7630349271	9.95×10^{-4}
0.5	0.9121803178	0.9119513987	2.50×10^{-4}	0.9105938906	1.73×10^{-3}
0.6	1.093539658	1.092769640	7.04×10^{-4}	1.090556514	2.72×10^{-3}
0.7	1.307873423	1.306121720	1.33×10^{-3}	1.302693219	3.96×10^{-3}
0.8	1.555181613	1.551841629	1.09×10^{-3}	1.546735182	5.43×10^{-3}
0.9	1.835464230	1.829741743	3.11×10^{-3}	1.822375591	7.13×10^{-3}
1	2.148721271	2.139614496	4.23×10^{-3}	2.129271087	9.05×10^{-3}

Table 4: Exact and approximate solution of $u_j(x)$ at $t_j = 0.75$ with and without perturbation terms in a and b

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	0.7711700002	0.7711983689	3.36×10^{-5}	0.7712065247	4.73×10^{-5}
0.2	0.8346800007	0.8347737522	1.12×10^{-4}	0.8348115589	1.57×10^{-4}
0.3	0.9405300015	0.9406671824	1.45×10^{-4}	0.9407717933	2.57×10^{-4}
0.4	1.088720003	1.088781041	5.60×10^{-5}	1.089016095	2.71×10^{-4}
0.5	1.279250004	1.278980053	2.11×10^{-4}	1.279447063	1.54×10^{-4}
0.6	1.512120006	1.511092655	6.79×10^{-4}	1.511943216	1.16×10^{-4}
0.7	1.787330008	1.784912741	1.35×10^{-3}	1.786361766	5.41×10^{-4}
0.8	2.104880011	2.100201772	2.22×10^{-3}	2.102541999	1.11×10^{-3}
0.9	2.464770014	2.456691211	3.27×10^{-3}	2.460309231	1.80×10^{-3}
1	2.867000017	2.854085288	4.50×10^{-3}	2.859479316	2.62×10^{-3}

Table 5: Exact and approximate solution of $u_j(x)$ at $t_j = 1$ with and without perturbation terms in a and b

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	1.027182818	1.027222369	3.85×10^{-5}	1.027311557	1.25×10^{-4}
0.2	1.108731273	1.108860738	1.16×10^{-4}	1.109229081	4.48×10^{-4}
0.3	1.244645364	1.244829262	1.47×10^{-4}	1.245701389	8.48×10^{-4}
0.4	1.434925092	1.434986046	2.42×10^{-5}	1.436644053	1.19×10^{-3}
0.5	1.679570457	1.679134916	2.59×10^{-4}	1.681940678	1.41×10^{-3}
0.6	1.978581458	1.977027872	7.85×10^{-4}	1.981444677	1.44×10^{-3}
0.7	2.331958096	2.328368191	1.53×10^{-3}	2.334981455	1.29×10^{-3}
0.8	2.739700370	2.732814161	2.51×10^{-3}	2.742351004	9.67×10^{-4}
0.9	3.201808281	3.189983398	3.69×10^{-3}	3.203330804	4.75×10^{-4}
1	3.718281828	3.699457703	5.06×10^{-3}	3.717678990	1.62×10^{-4}

Table 6: Exact and approximate solution of $u_j(x)$ at $t_j = 0.25$ with and without perturbation terms in f, g and Φ

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	0.2628402542	0.2628483725	3.08×10^{-5}	0.2630645568	8.53×10^{-4}
0.2	0.3013610167	0.3013841491	7.67×10^{-5}	0.3015988237	7.89×10^{-4}
0.3	0.3655622875	0.3655793088	4.65×10^{-5}	0.3657914404	6.26×10^{-4}
0.4	0.4554440667	0.4553871491	1.24×10^{-4}	0.4555956649	3.32×10^{-4}
0.5	0.5710063542	0.5707422861	4.62×10^{-4}	0.5709460580	1.05×10^{-4}
0.6	0.7122491501	0.7115606556	9.66×10^{-4}	0.7117584843	6.88×10^{-4}
0.7	0.8791724543	0.8777395127	1.62×10^{-3}	0.8779301112	1.41×10^{-3}
0.8	1.071776267	1.069157431	2.44×10^{-3}	1.069339409	2.27×10^{-3}
0.9	1.290060588	1.285674303	3.40×10^{-3}	1.285846152	3.26×10^{-3}
1	1.534025417	1.527131339	4.49×10^{-3}	1.527291415	4.38×10^{-3}

Table 7: Exact and approximate solution of $u_j(x)$ at $t_j = 0.5$ with and without perturbation terms in f, g and Φ

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	0.5164872127	0.5165050073	3.44×10^{-5}	0.5154115152	2.08×10^{-3}
0.2	0.5659488508	0.5660064728	1.01×10^{-4}	0.5648920604	1.86×10^{-3}
0.3	0.6483849144	0.6484638498	1.21×10^{-4}	0.6473145948	1.65×10^{-3}
0.4	0.7637954034	0.7638099282	1.90×10^{-5}	0.7626119527	1.54×10^{-3}
0.5	0.9121803178	0.9119513987	2.50×10^{-4}	0.9106908844	1.63×10^{-3}
0.6	1.093539658	1.092769640	7.04×10^{-4}	1.091432842	1.92×10^{-3}
0.7	1.307873423	1.306121720	1.33×10^{-3}	1.304694990	1.30×10^{-3}
0.8	1.555181613	1.551841629	1.09×10^{-3}	1.550311436	3.13×10^{-3}
0.9	1.835464230	1.829741743	3.11×10^{-3}	1.828094690	4.01×10^{-3}
1	2.148721271	2.139614496	4.23×10^{-3}	2.137837344	5.06×10^{-3}

Table 8: Exact and approximate solution of $u_j(x)$ at $t_j = 0.75$ with and without perturbation terms in f, g and Φ

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	0.7711700002	0.7711983689	3.36×10^{-5}	0.7736467637	3.21×10^{-3}
0.2	0.8346800007	0.8347737522	1.12×10^{-4}	0.8372983250	3.13×10^{-3}
0.3	0.9405300015	0.9406671824	1.45×10^{-4}	0.9433190828	2.96×10^{-3}
0.4	1.088720003	1.088781041	5.60×10^{-5}	1.091611970	2.65×10^{-3}
0.5	1.279250004	1.278980053	2.11×10^{-4}	1.282042479	2.18×10^{-3}
0.6	1.512120006	1.511092655	6.79×10^{-4}	1.514440029	1.53×10^{-3}
0.7	1.787330008	1.784912741	1.35×10^{-3}	1.788599719	7.10×10^{-4}
0.8	2.104880011	2.100201772	2.22×10^{-3}	2.104284426	2.82×10^{-4}
0.9	2.464770014	2.456691211	3.27×10^{-3}	2.461227246	1.43×10^{-3}
1	2.867000017	2.854085288	4.50×10^{-3}	2.859134249	2.74×10^{-3}

Table 9: Exact and approximate solution of $u_j(x)$ at $t_j = 1$ with and without perturbation terms in f, g and Φ

x	exact solution	approximate solution	relative error	perturbed solution	relative error
0.1	1.027182818	1.027222369	3.85×10^{-5}	1.025635295	1.50×10^{-3}
0.2	1.108731273	1.108860738	1.16×10^{-4}	1.107171622	1.40×10^{-3}
0.3	1.244645364	1.244829262	1.47×10^{-4}	1.242968576	1.34×10^{-3}
0.4	1.434925092	1.434986046	2.42×10^{-5}	1.432882007	1.42×10^{-3}
0.5	1.679570457	1.679134916	2.59×10^{-4}	1.676712572	1.70×10^{-3}
0.6	1.978581458	1.977027872	7.85×10^{-4}	1.974208173	2.21×10^{-3}
0.7	2.331958096	2.328368191	1.53×10^{-3}	2.325067038	2.95×10^{-3}
0.8	2.739700370	2.732814161	2.51×10^{-3}	2.728941436	3.92×10^{-3}
0.9	3.201808281	3.189983398	3.69×10^{-3}	3.185441974	5.11×10^{-3}
1	3.718281828	3.699457703	5.06×10^{-3}	3.694142422	6.49×10^{-3}

7 Conclusions

The HPM for the one-dimensional inverse problems has been presented. The method described is mathematically simple and computationally effective. As we see in Tables $2 - 9$, small errors in the data make small errors in the solution, so that, the solution depends continuously on the data. In this paper, the noise terms that are shown in Table 1, are made randomized and have standard normal distributions. *Maple 16 packages* have been used to compute the solution before and after adding noise terms. Rapidity , accuracy and stability are advantages of this formulation.

References

1. Adomian, G. *A new approach to the heat equation-An application of the decomposition method*, J. Math. Anal. Appl. 113 (1986) 202-209.

2. Alifonov, O.M. *Inverse heat transfer problems*, Springer-Verlag, 1994.

3. Beck, J.V., Blackwell, B. and St. Clair. Ch. J. *Inverse heat conduction, ill-posed problems*, John Willey and Sons, Inc, 1985.

4. Buhmann Martin, D. *Radial Basis Functions: Theory and Implementations*, Cambridge University Press, 2003.

5. Cannon, J.R. *The one dimensional heat equation*, Addison-Welsey , New york, 1984.

6. Dehghan, M. and Shakeri, F. *Solution of a partial differential equation subject to temperature overspecification by He's homotopy perturbation method*, Phys. Scr., 75 (2007) 778-787.

7. Douglas, J.J. and Russell, T.F. *Numerical Methods for Convection-Dominated Diffusion Problems Based on Combining the Method of Characteristics with Finite Element or Finite Difference Procedures*, SIAM J. Numer. Anal., 19 (5) (2006) 871-885

8. Ghasemi, M., Tavassoli Khanjani, M. and Davari, A. *Numerical solution of two -dimensional nonlinear differential equation by homotopy perturbation method*, Appl. Math. Comput. 189 (2007) 341-345.

9. He, J. H. *Some asymptotic methods for strongly nonlinear equations*, International Journal of Modern Physics B, 20 (2006) 1141-1199.

10. He, J. H. *An elementary introduction to recently developed asymptotic methods and nanomechanics in textile engineering*, International Journal of Modern Physics B, 22 (2008) 3487-3578.

11. He, J. H. *Homotopy perturbation technique*, Comput. Methods. Appl. Mech. Engrg., 178 (1999) 257-262.

12. He, J. H. *A coupling method of a homotopy technique and a perturbation technique for non-linear problems*, Int. J. Non-Linear Mechanics, 35 (2000) 37-43.

13. He, J. H. *The homotopy perturbation method for nonlinear oscilators with discontinuities*, Appl. Math. Comput., 5 (2006) 287-292.

14. He, J. H. *Homotopy perturbation method for solving boundary value problems*, Phys. lett. A., 35 (2006) 87-88.

15. Johansson, B.T. and Lesnic. D. *A method of fundamental solutions for transient heat conduction*, Engineering Analysis with Boundary Elements, 32 (2008) 697-703.

16. Jia, H., Xu, W., Zhao, X. and Li, Z. *Separation of variables and exact solutions to nonlinear diffusion equations with $x-$dependent convection and absorption*, J. Math. Anal. Appl., 339 (2008) 982-995.

17. Johansson, B.T. and Lesnic, D. *Determination of spacewise depedent heat suorce*, Journal of computational and Applied Mathematics, 209 (2007) 66-88.

18. Lewandowski, J.L.V. *Marker method for the solution of nonlinear diffusion equations*, JCAM., 196 (2006) 523-539.

19. Onyango, T.T.M., Ingham, D.B., Lesnic, D. and Soldicka, M. *Determination of a time-dependent heat transfer cofficient from non-standard boundary measurements*, Mathematics and Computers in Simulation, 79 (2009) 1577-1548

20. Qu, C., Ji, L. and Dou, J. *Exact solution and generalized conditional symmetries to* $(n+1)-$*dimensional nonlinear diffusion equations with source term*, Physics Letters A, 343 (2005) 139-147.

21. Shidfar, A. and Zakeri, A. *A numerical method for backward inverse heat conduction problem with two unknown functions*, International Journal of Engineering science, 16 (2008) 71-74.

22. Shidfar, A. and Zakeri, A. *Asymptotic solution for an inverse parabolic problem*, Mathematica Balkanica, 18 (2004) 475-483.

23. Shidfar, A. and Zakeri, A. *A two-dimensional inverse heat conduction problem for estimating heat source*, International Journal of Applied Mathematics and Mathematical Science, 10 (2005) 1633-1641.

24. Siddiqui, A.M., Mahmood, R. and Ghori, Q.K. *Homotopy perturbation method for thin film of a fourth grade fluid down a vertical cylinder*, Phys. lett. A., 352 (2006) 404-410.

25. Vogel, C.R. *Computational methods for inverse problems*, SIAM, 2002.

26. Zakeri, A., Aminataei, A. and Jannati, Q. *Application of He's homotopy perturbation method for Cauchy problem of ill-posed nonlinear diffusion equation*, Discrete Dynamics in Nature and Society, Vol. 2010, ID 780207.

27. Zakeri, A. and Jannati, Q. *An inverse problem for parabolic partial differential equations with nonlinear conductivity term in one dimensional space*, Scholarly Research Exchange, 2009, ID 468570.

28. Zakeri, A. and Jannati, Q. *Stability of homotopy perturbtion technique for an inverse diffusion problem*, The proceeding of fourty fourth Annual Iranian Mathematics Conference, pp. 682-685, Rafsanjan, Aug . 2014.

Iranian Journal of Numerical Analysis and Optimization
Vol. 5, No. 2, (2015), pp 95-117

An adaptive nonmonotone trust region method for unconstrained optimization problems based on a simple subproblem

Z. Saeidian and M.R. Peyghami*

Abstract

Using a simple quadratic model in the trust region subproblem, a new adaptive nonmonotone trust region method is proposed for solving unconstrained optimization problems. In our method, based on a slight modification of the proposed approach in (J. Optim. Theory Appl. 158(2):626-635, 2013), a new scalar approximation of the Hessian at the current point is provided. Our new proposed method is equipped with a new adaptive rule for updating the radius and an appropriate nonmonotone technique. Under some suitable and standard assumptions, the local and global convergence properties of the new algorithm as well as its convergence rate are investigated. Finally, the practical performance of the new proposed algorithm is verified on some test problems and compared with some existing algorithms in the literature.

Keywords: Trust region methods; Adaptive radius; Nonmonotone technique; Scalar approximation of the Hessian; Global convergence.

1 Introduction

In this paper, we deal with the following unconstrained optimization problem:

$$\min_{x\in\mathbb{R}^n} f(x) \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a twice continuously differentiable function. Two popular classes of optimization techniques for solving (1) are line search and trust

*Corresponding author
Received 20 December 2014; revised 18 April 2015; accepted 27 July 2015
Z. Saeidian
Faculty of Mathematics, K.N. Toosi University of Technology, Tehran, Iran. E-mail : z.saeedian@dena.kntu.ac.ir

M.R. Peyghami
Scientific Computations in OPtimization and Systems Engineering (SCOPE), K.N. Toosi University of Technology, Tehran, Iran. E-mail: peyghami@kntu.ac.ir

region methods; see, e.g., [9,17,18]. Line search methods refer to a procedure in which one moves along a (descent) direction as long as a sufficient reduction in the objective is achieved. On the other hand, in the classical trust region methods, a trial step is computed by minimizing a (quadratic) model of the objective function at the current point over a region around this point. Then, using the so-called trust region ratio, the trial step is accepted/rejected and the new point as well as the radius is updated accordingly. It has been shown that trust region methods have appropriate global and local convergence properties. These methods have been widely studied in the literature; see, e.g., [9,12,17,19,24,25].

Here, let us briefly describe one step of the classical trust region method. Given x_k, the trial step d_k is computed by solving the following subproblem:

$$\min q_k(d) = g_k^T d + \frac{1}{2} d^T B_k d \qquad s.t.\ \|d\| \leq \Delta_k, \tag{2}$$

where $g_k = \nabla f(x_k)$, B_k is a $n \times n$ symmetric matrix which is $\nabla^2 f(x_k)$ or its approximation, $\Delta_k > 0$ is the so-called trust region radius, and $\|.\|$ refers to the Euclidean norm. Due to the so-called trust region ratio

$$r_k = \frac{f(x_k) - f(x_k + d_k)}{q_k(0) - q_k(d_k)}, \tag{3}$$

one decides whether the trial step is accepted or rejected; given $\mu \in (0,1)$, if $r_k \geq \mu$, then the trial step is accepted and the new point is introduced by $x_{k+1} = x_k + d_k$. Otherwise, the trial step is rejected and the current point remains unchanged for the next iteration. In both cases, the trust region radius is updated appropriately.

In the monotone trust region methods, the sequence of the objective values is monotonically decreasing. This may cause slow convergence rate in some problems. In order to overcome this disadvantage, the concept of nonmonotone strategies have been introduced in the framework of trust region methods, see, e.g., [13, 14]. A nonmonotone line search method was first proposed by Chamberlain et al. in [8]. Grippo et al. in [13] introduced a nonmonotone technique for Newton's method and developed it for unconstrained optimization in [14]. Nevertheless many advantages of the Grippo's technique, it suffers from some drawbacks [2, 3, 27]. In order to overcome these difficulties, recently, Ahookhosh and Amini in [2] and Ahookhosh et al. in [3] proposed a new nonmonotone term as below:

$$R_k = \epsilon_k f_{\ell(k)} + (1 - \epsilon_k) f_k, \tag{4}$$

where $f_k = f(x_k)$, $\epsilon_k \in [\epsilon_{\min}, \epsilon_{\max}] \subset [0,1]$ and $f_{\ell(k)}$ is the Grippo's nonmonotone term which is defined by

$$f_{\ell(k)} = \max_{0 \leq j \leq M(k)} f_{k-j}, \tag{5}$$

where $M(0) = 0$ and, for $k \geq 1$, $M(k) = \min\{k, M\}$, for given positive integer M. They employed (4) in the trust region ratio (3) and suggested nonmonotone trust region methods which are globally convergent. The reported numerical results on test problems confirm the efficiency and robustness of these methods in practice too.

The radius updating strategy is a crucial point in trust region methods [1, 21, 28]. In the classical trust region methods, this parameter is simply enlarged, shrunk or stayed unchanged based on the magnitude of r_k. Several strategies have been introduced in the literature for radius updating and initial radius choosing; see e.g. [11, 21–23, 29]. Zhang et al. in [29] proposed the radius update according to $\Delta_k = c^p \|g_k\| \|\hat{B}_k^{-1}\|$, where $c \in (0, 1)$, p is a nonnegative integer and $\hat{B}_k = B_k + iI$ is a positive definite matrix, for some $i \in \mathbb{N}$. Although, Zhang's method uses more information of the objective function for updating the radius, it requires an estimation of $\|\hat{B}_k^{-1}\|$, which is costly. To reduce the computational cost of Zhang's updating rule, a simple adaptive rule was proposed by Shi and Wang in [23] according to $\Delta_k = c^p \frac{\|g_k\|^3}{g_k^T \hat{B}_k g_k}$, where $c \in (0, 1)$, $\hat{B}_k$ is a positive definite matrix and p is a nonnegative integer. Despite Zhang's method that only updates the radius based on the current point information, some updating rules based on the information of the last two iterates have been introduced; see, e.g., [15,29,30]. Among them, Li [15] proposed an adaptive trust region method in which the radius is updated according to $\Delta_k = \frac{\|d_{k-1}\|}{\|y_{k-1}\|} \|g_k\|$, where $y_{k-1} = g_k - g_{k-1}$ and $d_{k-1} = x_k - x_{k-1}$.

The advantages of nonmonotone and adaptive techniques have been simultaneously employed in the framework of trust region methods. Using the adaptive strategy proposed in [15], Sang et al. in [20] introduced a nonmonotone adaptive trust region method based on a simple subproblem for large-scale unconstrained optimization problems which makes full use of information in the last two iterates. The idea of simple subproblem is originated from the fact that solving the subproblem (2) is costly especially when B_k is a large-scale and dense matrix. Therefore, the skills of the quasi-Newton method is used for correcting B_k by a real diagonal matrix ΔB_{k-1} from B_{k-1}. Recently, Zhou et al. in [30] constructed a simple subproblem according to the modification of the secant condition of Wei in [26] and introduced a nonmonotone adaptive trust region method based on the simple subproblem. Later, Biglari and Solimanpur in [7] proposed another simple subproblem with some superior properties to that of [30] in which the approximation of the Hessian at the current point x_k is computed by

$$\hat{\gamma}_k := \gamma(x_k) = \frac{4(f_{k-1} - f_k) + 3g_k^T d_{k-1} + g_{k-1}^T d_{k-1}}{d_{k-1}^T d_{k-1}}. \tag{6}$$

In this paper, we proposed a new nonmonotone adaptive trust region method based on simple subproblem for unconstrained optimization problems. Our

approach is equipped with the nonmonotone technique as proposed in [2, 3], and uses a slight modification of the secant condition in [7] for constructing an approximation of the Hessian at the current point. Moreover, a modified version of the adaptive strategy in [20] is employed in the framework of the proposed algorithm. It is worth mentioning that the scalar approximation of the Hessian based on modified secant condition in [6] has superior to the standard Barzilai-Borwein method and its modifications. Under some standard assumptions, the global convergence property, as well as its superlinear convergence rate, is established. Numerical results show the efficiency of the proposed approach in practice comparing with some existing methods in the literature.
The rest of the paper is organized as follows: In Section 2, we present the structure of the new nonmonotone adaptive trust region method in details. The global convergence property, as well as its rate of convergence, is established in Section 3. Preliminary numerical results of applying the proposed algorithm on some test problems are given in Section 4. Finally, we end up the paper by some concluding remarks in Section 5.

2 The new algorithm

In this section, we propose a new adaptive nonmonotone trust region method for solving unconstrained optimization problems. Our algorithm combines the nonmonotone technique as proposed in [2] with an improved scalar approximation of the Hessian according to the modified secant equation as proposed in [6].
Let us describe one step of our new algorithm here: For given x_k, the trial step d_k is computed by (approximately) solving the following simple subproblem:

$$\min q_k(d) = g_k^T d + \frac{1}{2} d^T \gamma(x_k) d \qquad s.t.\ \|d\| \leq \Delta_k, \tag{7}$$

where $\gamma_k := \gamma(x_k)$ is a scalar approximation of the Hessian matrix. Since $\hat{\gamma}_k$, as defined by (6), may become negative in some iterations, we slightly modify (6) and define γ_k as below:

$$\gamma_k = \frac{4(f_{k-1} - f_k) + (3 + \eta_k) g_k^T d_{k-1} + g_{k-1}^T d_{k-1}}{d_{k-1}^T d_{k-1}}, \tag{8}$$

where η_k is computed by:

$$\eta_k = \begin{cases} \frac{4(f_k - f_{k-1}) - 3g_k^T d_{k-1} - g_{k-1}^T d_{k-1} + \delta}{{g_k}^T d_{k-1}}, & \text{if } \hat{\gamma}_k < 0, \\ 0, & \text{Otherwise,} \end{cases}$$

where δ is a small positive number. By this definition, it is obviously seen that $\gamma_k > 0$. Now, using d_k, the nonmonotone ratio is computed by:

$$r_k = \frac{R_k - f(x_k + d_k)}{Pred_k}, \tag{9}$$

where R_k is defined by (4) and $Pred_k = q_k(0) - q_k(d_k)$. For given $\mu \in (0, 1)$, the trial step is accepted whenever $r_k \geq \mu$; otherwise it is rejected. In both cases, the radius is adaptively updated according to $\Delta_k = \min\left\{\nu_k \frac{\|g_k\|}{\gamma_k}, \Delta_{\max}\right\}$, where $\Delta_{\max} > 0$ is a threshold value for the radii and ν_{k+1} is updated by:

$$\nu_{k+1} = \begin{cases} \sigma_0 \nu_k, & r_k < \mu_1, \\ \nu_k, & \mu_1 \leq r_k \leq \mu_2, \\ \min\{\sigma_1 \nu_k, \nu_{\max}\}, & r_k > \mu_2, \end{cases} \tag{10}$$

where $0 < \sigma_0 < 1 < \sigma_1$, $0 < \mu_1 < \mu_2 \leq 1$ and $\nu_{\max} > 0$ are given numbers. By the way, the new point is given by $x_{k+1} = x_k + d_k$ as long as $r_k \geq \mu$; otherwise, we set $x_{k+1} = x_k$.
The procedure of the new proposed nonmonotone trust region algorithm is outlined in Algorithm 1:

Algorithm 1: *A new nonmonotone adaptive trust region algorithm*

Input: $x_0 \in \mathbb{R}^n$, $0 < \mu < \mu_1 < \mu_2 \leq 1$, $0 < \sigma_0 < 1 < \sigma_1$, $0 < \epsilon_{\min} < \epsilon_{\max} < 1$, $\epsilon, \varepsilon, M, \nu_{\max}, \Delta_{\max} > 0$, $0 < \theta_1 < \theta_2$ and $\delta > 0$.

Step 0: Set $k = 0$, $\gamma_0 := \gamma(x_0) = 1$, $g_0 = g(x_0)$, $\nu_0 = 1$ and $\Delta_0 = \min\left\{\nu_0 \frac{\|g_0\|}{\gamma_0}, \Delta_{\max}\right\}$.

Step 1: **If** $\|g_k\| \leq \varepsilon$, **Then** Stop.

Step 2: Determine d_k by solving (7) and compute r_k using (9).

Step 3: **If** $r_k < \mu$, **Then** set $\Delta_k = \sigma_0 \Delta_k$, and goto Step 2.

Step 4: Set $x_{k+1} = x_k + d_k$.

Step 5: Compute γ_{k+1} using (8). **If** $\gamma_{k+1} \leq \epsilon$, **Then** set $\gamma_{k+1} = \theta_1$. **If** $\gamma_{k+1} \geq \frac{1}{\epsilon}$, **Then** set $\gamma_{k+1} = \theta_2$.

Step 6: Update ν_{k+1} using (10) and set $\Delta_{k+1} = \min\left\{\nu_{k+1} \frac{\|g_{k+1}\|}{\gamma_{k+1}}, \Delta_{\max}\right\}$. Set $k =: k + 1$ and goto Step 1.

Remark 1. Step 5 of Algorithm 1 implies that γ_k is a bounded positive number for all k. More precisely, we have $\min\{\epsilon, \theta_1\} \leq \gamma_k \leq \max\left\{\frac{1}{\epsilon}, \theta_2\right\}$.

Remark 2. The subproblem (7) can be easily solved by using the following procedure [20]: Let $\omega_k = \frac{g_k}{\gamma_k}$. If $\|\omega_k\| \leq \Delta_k$, then we set the trial step as $d_k = -\omega_k$. Otherwise, we choose $\alpha \in (0,1)$ so that $\|\alpha\omega_k\| = \Delta_k$. It can be easily verified that $\alpha = \frac{\Delta_k}{\|\omega_k\|}$. In this case, we set $d_k = -\alpha\omega_k = -\frac{\Delta_k}{\|\omega_k\|}\omega_k = -\frac{\Delta_k}{\|g_k\|}g_k$.

Remark 3. From Remark 2, one can easily see that, for all k, there exists a positive constant κ so that $\|d_k\| \leq \kappa\|g_k\|$.

3 Convergence analysis

In this section, our aim is to analyze the local and global convergence properties of Algorithm 1. For this purpose, the following assumption is imposed on the problem:

A1. The set $\Omega = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ is a closed and bounded set and $f(x)$ is a twice continuously differentiable function over Ω. Moreover, $\nabla f(x)$ is a Lipschitz continuous function over Ω.

Lemma 1. *Assume that d_k is a solution of the problem (7). Then, one has:*

$$Pred_k := q_k(0) - q_k(d_k) \geq \frac{1}{2}\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\}. \tag{11}$$

Proof. We proceed the proof in the following two possible cases for d_k:

Case I. $\|-\frac{g_k}{\gamma_k}\| \leq \Delta_k$, and therefore, $d_k = -\frac{g_k}{\gamma_k}$: In this case one can easily obtain the following relations:

$$\begin{aligned} q_k(0) - q_k(d_k) &= q_k(0) - q_k\left(-\frac{g_k}{\gamma_k}\right) \\ &= -g_k^T\left(-\frac{g_k}{\gamma_k}\right) - \frac{1}{2}\left(-\frac{g_k}{\gamma_k}\right)^T \gamma_k \left(-\frac{g_k}{\gamma_k}\right) \\ &= \frac{\|g_k\|^2}{\gamma_k} - \frac{1}{2}\frac{\|g_k\|^2}{\gamma_k} = \frac{\|g_k\|^2}{2\gamma_k} \geq \frac{1}{2}\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\}. \end{aligned}$$

Case II. $\|-\frac{g_k}{\gamma_k}\| > \Delta_k$, and therefore, $d_k = -\frac{\Delta_k}{\|g_k\|}g_k$: In this case, we have:

$$\begin{aligned} q_k(0) - q_k(d_k) &= q_k(0) - q_k\left(-\frac{\Delta_k}{\|g_k\|}g_k\right) \\ &= -g_k^T\left(-\frac{\Delta_k}{\|g_k\|}g_k\right) - \frac{1}{2}\left(-\frac{\Delta_k}{\|g_k\|}g_k\right)^T \gamma_k \left(-\frac{\Delta_k}{\|g_k\|}g_k\right) \\ &= \Delta_k\|g_k\| - \frac{1}{2}\gamma_k\Delta_k^2 > \Delta_k\|g_k\| - \frac{1}{2}\Delta_k\|g_k\| \\ &= \frac{1}{2}\Delta_k\|g_k\| \geq \frac{1}{2}\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\}, \end{aligned}$$

where the first inequality is obtained from the fact that $\gamma_k\Delta_k < \|g_k\|$.

Considering the above mentioned cases, the proof is completed. □

Lemma 2. *Let d_k be computed by the procedure as mentioned in Remark 2. Then, for all k, one has:*

$$|f(x_k) - f(x_k + d_k) - Pred_k| \leq O(\|d_k\|^2), \tag{12}$$

where $Pred_k$ is defined by (11).
Proof. Using Taylor's expansion and the fact that γ_k is bounded due to Remark 1, one can easily conclude the result. □
The following lemma states some appealing properties of the sequences $\{f_{\ell(k)}\}$ and $\{R_k\}$, which are defined by (5) and (4), respectively. One can find its proof in [2].

Lemma 3. *Suppose that Assumption A1 holds and the sequence $\{x_k\}$ is generated by Algorithm 1. Then, the following statements hold:*

i) *For all k, we have $f_k \leq R_k \leq f_{\ell(k)}$.*

ii) *The sequence $\{f_{\ell(k)}\}$ is a decreasing and convergent sequence.*

iii) $\lim_{k\to\infty} f_{\ell(k)} = \lim_{k\to\infty} f_k$.

iv) $\lim_{k\to\infty} R_k = \lim_{k\to\infty} f_k$.

Lemma 4. *Let Assumption A1 hold and the sequence $\{x_k\}$ be generated by Algorithm 1. Assume that there exists a constant $\zeta \in (0, 1)$ so that $\|g_k\| > \zeta$, for all k. Then, for any k, there exists a nonnegative integer p so that x_{k+p+1} is a successful iteration point, i.e., $r_{k+p+1} > \mu$.*
Proof. Suppose that, on the contrary, there exists an iteration k so that, for all nonnegative integer p, the point x_{k+p+1} is an unsuccessful iteration point, i.e.,

$$r_{k+p} < \mu, \qquad p = 0, 1, 2, \ldots. \tag{13}$$

In this case, from Step 3 of Algorithm 1, we have

$$\Delta_{k+p+1} \leq \sigma_0^{p+1} \Delta_k.$$

This inequality together with the definition of Δ_k imply that:

$$\lim_{p\to\infty} \Delta_{k+p+1} = 0. \tag{14}$$

Therefore, from Lemma 1, Remark 1 and (12), we have

$$\begin{aligned}\left|\frac{f(x_{k+p}) - f(x_{k+p} + d_{k+p})}{Pred_{k+p}} - 1\right| &= \left|\frac{f(x_{k+p}) - f(x_{k+p} + d_{k+p}) - Pred_{k+p}}{Pred_{k+p}}\right| \\ &\leq \frac{O(\|d_{k+p}\|^2)}{\frac{1}{2}\|g_{k+p}\| \min\{\Delta_{k+p}, \frac{\|g_{k+p}\|}{\gamma_{k+p}}\}} \\ &\leq \frac{O(\|\Delta_{k+p}\|^2)}{\frac{1}{2}\zeta \min\left\{\Delta_{k+p}, \frac{\zeta}{\max\{\frac{1}{\epsilon}, \theta_2\}}\right\}}.\end{aligned}$$

This implies that $\left|\frac{f(x_{k+p}) - f(x_{k+p} + d_{k+p})}{Pred_{k+p}} - 1\right| \to 0$, as $p \to \infty$. Thus, for sufficiently large p, using Lemma 3, we have

$$r_{k+p} = \frac{R_{k+p} - f(x_{k+p} + d_{k+p})}{Pred_{k+p}} \geq \frac{f(x_{k+p}) - f(x_{k+p} + d_{k+p})}{Pred_{k+p}} \to 1,$$

which contradicts $r_{k+p} < \mu$. This completes the proof of the lemma. □
Lemma 4 implies that the inner loop in Steps 2–3 of Algorithm 1 will be terminated after finite number of iterations, and therefore, Algorithm 1 is well-defined.
The following theorem provides the global convergence property of Algorithm 1 under some suitable and standard assumptions.

Theorem 1. *Suppose that Assumption A1 holds and $\{x_k\}$ is the sequence generated by Algorithm 1. Then, Algorithm 1 either stops at a stationary point or*

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{15}$$

Proof. Suppose that Algorithm 1 does not stop at a stationary point. We show that (15) holds for the infinite sequence $\{x_k\}$. Assume that, on the contrary, there exists a positive constant ζ so that

$$\|g_k\| > \zeta > 0, \qquad \forall k. \tag{16}$$

Using Lemma 4, Algorithm 1 is well-defined and the inner loop in Steps 2–3 is terminated after finite number of iterations. Therefore, we may assume that $r_k \geq \mu$. Now, from (9) and Lemma 1, we have

$$R_k - f_{k+1} \geq \mu Pred_k \geq \frac{1}{2}\mu\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\}$$
$$\geq \frac{1}{2}\mu\zeta \min\left\{\Delta_k, \frac{\zeta}{\max\left\{\frac{1}{\epsilon}, \theta_2\right\}}\right\} \geq 0. \tag{17}$$

By taking limit from both sides of this inequality, as $k \to \infty$, and using Lemma 3, we conclude that

$$\Delta_k = \nu_k \frac{\|g_k\|}{\gamma_k} \to 0. \tag{18}$$

Now, using Remark 1 and (16), (18) implies that

$$\nu_k \longrightarrow 0. \tag{19}$$

Therefore, from (16) and Lemmas 1 and 2, we have

$$\left|\frac{f(x_k) - f(x_k + d_k)}{Pred_k} - 1\right| = \left|\frac{f(x_k) - f(x_k + d_k) - Pred_k}{Pred_k}\right|$$
$$\leq \frac{O\left(\|d_k\|^2\right)}{\frac{1}{2}\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\}}$$
$$\leq \frac{O\left(\Delta_k^2\right)}{\frac{1}{2}\zeta \min\left\{\Delta_k, \frac{\zeta}{\max\left\{\frac{1}{\epsilon}, \theta_2\right\}}\right\}} \overset{k\to\infty}{\longrightarrow} 0,$$

which implies that

$$r_k = \frac{R_k - f(x_k + d_k)}{Pred_k} \geq \frac{f(x_k) - f(x_k + d_k)}{Pred_k} \longrightarrow 1. \tag{20}$$

This shows that, for sufficiently large k, we have successful iterations. Therefore, there exists a positive constant ν^* so that, for sufficiently large k, $\nu_k \geq \nu^*$. This contradicts (19). □

Under some extra assumptions on the problem and using the same proof line of Theorem 3.7 in [30], one can construct the superlinear convergence rate of the sequence $\{x_k\}$, generated by Algorithm 1, to its limit point x^*.

4 Numerical results

In this section, we focus on providing some computational results of applying Algorithm 1, denoted by FATRA, along with the following algorithms on some test problems in order to compare their performances:

- NATRM: Algorithm 2.1 in [30];
- NATRA: Algorithm 2.1 in [30] in which the nonmonotone term in computing the trust region ratio r_k is replaced by R_k, as given by (4);
- FATRM: Algorithm 1 in which the nonmonotone term in computing the trust region ratio r_k is replaced by $f_{\ell(k)}$, as given by (5);

All the algorithms are implemented in MATLAB 7.10.0 (R2010a) environment on a PC with CPU 2.0 GHz and 4GB RAM memory and double precision format. The following parameters are considered in the relevant algorithms:

$$\mu = 0.1, \mu_1 = 0.25, \mu_2 = 0.75, \epsilon_{\min} = 10^{-6}, \epsilon_{\max} = 10^{6}, \Delta_{\max} = 100, M = 10,$$
$$\sigma_0 = c_2 = 0.5, \sigma_1 = c_1 = 4, \nu_{\max} = \sigma_1^4, \nu_0 = 0.25, \varepsilon = \epsilon = 10^{-6}, \delta = 10^{-6}.$$

Moreover, in Step 5 of Algorithm 1, if $\gamma_{k+1} \leq \epsilon$, then we set $\theta_1 = \epsilon$; if $\gamma_{k+1} > \frac{1}{\epsilon}$, then we set $\theta_2 = \frac{1}{\epsilon}$. The simple subproblem at each iteration is solved by the procedure as mentioned in Remark 2. All the algorithms are being stopped either $\|g_k\| \leq 10^{-6}$, or the number of iterations and/or function evaluations exceeds 50000. In the latter case, we declare that the algorithm is failed. The considered test problems are those in [30] as well as some large-scale problems taken from [16] and [4]. We have also utilized the advantages of the performance profile of Dolan and Moré in [10] to compare the performances of considered algorithms.
Numerical results are given in Table 1. In this table, *Prob* stands for the problem name, and n_i, n_f and f_{opt} denote the number of iterations, the number of function evaluations and the optimum value of the objective function, respectively. It should be noted that the number of gradient evaluations are almost the same as n_i.

Figures 1 and 2 show the performance profiles of the results in Table 1 based on the number of iterations and function evaluations, respectively. At a glance to Figure 1, we can find out that, in terms of n_i, FATRA solves all the considered test problems successfully, while the other algorithms have at least one failure in their runs. Moreover, FATRA and FATRM algorithms solve roughly 67% and 61% of the problems at the lowest value of n_i, respectively. This percentage for NATRM and NATRA algorithms are 49% and 47%, respectively. Figure 2 is drawn based on n_f of the results in Table 1. From this figure, it is revealed that FATRA solves all the problems successfully while FATRM has one failure in its run. Moreover, NATRM and NATRA algorithms solve roughly 96% and 98% of the test problems successfully. On the other hand, FATRA and FATRM algorithms solve about 58% and 60% of test problems in the lowest value of n_f while these percentages for NATRM and NATRA algorithms are about 34% and 22%.

Besides the performance profiles of the considered algorithms based on n_i and n_f, we have stored the average CPU time in 20 runs for each algorithms

and drew the performance profile of the considered algorithms based on CPU time in Figure 3. The result shows that FATRA works well in this regard too. Based on the above mentioned arguments, one can easily realize that FATRA is competitive with FATRM, NATRM and NATRA algorithms in terms of n_i, n_f and CPU time. Moreover, the performance of FATRM is very close to FATRA.

Table 1: The numerical results

$Prob$	n	$NATRM$ $n_i/n_f/f_{opt}$	$NATRA$ $n_i/n_f/f_{opt}$	$FATRM$ $n_i/n_f/f_{opt}$	$FATRA$ $n_i/n_f/f_{opt}$
Almost Perturbed Quadratic [4]	1000	655/900/1.93e-013	637/892/9.27e-014	493/973/2.03e-013	503/1046/7.99e-014
	5000	1595/2286/2.09e-013	1299/1895/1.85e-014	1272/2605/2.33e-013	1141/2331/5.99e-014
	10000	3192/4675/2.07e-013	2710/4156/6.38e-014	2627/5713/4.47e-014	1993/4350/5.43e-022
BIGGSB1(CUTE) [4]	100	933/1286/1.28e-010	666/996/2.05e-010	930/1801/2.48e-010	908/1824/2.67e-012
	500	11877/17751/6.31e-009	4979/7945/6.15e-010	10565/23394/3.19e-010	7179/16265/3.39e-012
	1000	Failed	7920/13527/4.08e-011	Failed	18672/43134/9.51e-009
Diagonal 4 [4]	1000	9/12/6.96e-018	9/12/6.96e-018	8/8/8.94e-022	8/8/8.94e-022
	5000	9/12/3.48e-018	9/12/3.48e-018	8/9/1.60e-022	8/9/1.60e-022
	10000	9/12/6.96e-0187	9/12/6.96e-018	8/8/8.94e-022	8/8/8.94e-022
Diagonal 5 [4]	1000	6/6/6.93e+002	6/6/6.93e+002	6/6/6.93e+002	6/6/6.93e+002
	5000	6/6/3.46e+003	6/6/3.46e+003	6/6/3.46e+003	6/6/3.46e+003
	10000	6/6/6.93e+003	6/6/6.93e+003	6/6/6.93e+003	6/6/6.93e+003

Table 1: The numerical results (continued)

$Prob$	n	$NATRM$	$NATRA$	$FATRM$	$FATRA$
		$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$
Diagonal 7 [4]	1000	8/9/-8.16e+003	8/9/-8.16e+003	7/7/-8.16e+003	7/7/-8.16e+003
	5000	8/9/-4.08e+003	8/9/-4.08e+003	7/7/-4.08e+003	7/7/-4.08e+003
	10000	8/9/-8.16e+003	8/9/-8.16e+003	7/7/-8.16e+003	7/7/-8.16e+003
Diagonal 8 [4]	1000	6/7/-4.80e+003	6/7/-4.80e+003	6/6/-4.80e+003	6/6/-4.80e+003
	5000	6/7/-2.40e+003	6/7/-2.40e+003	6/6/-2.40e+003	6/6/-2.40e+003
	10000	6/7/-4.80e+003	6/7/-4.80e+003	6/6/-4.80e+003	6/6/-4.80e+003
DIXON3DQ(CUTE) [4]	100	1352/1912/2.92e-011	1107/1641/5.46e-011	1081/2123/2.46e-010	889/1782/4.24e-015
	500	13032/19407/5.56e-009	5386/8471/5.65e-011	10856/24068/5.36e-009	4994/11222/1.83e-009
	1000	30764/46139/4.49e-010	15248/25248/3.78e-014	Failed	15193/34094/1.73e-008
DQDRTIC(CUTE) [4]	1000	33/37/1.05e-016	33/37/1.05e-016	50/60/4.51e-019	47/58/1.88e-015
	5000	30/34/2.13e-015	30/34/2.13e-015	40/40/7.99e-017	40/40/7.99e-017
	10000	31/35/1.36e-016	31/35/1.36e-016	39/39/1.89e-016	36/41/6.72e-016

Table 1: The numerical results (continued)

$Prob$	n	$NATRM$ $n_i/n_f/f_{opt}$	$NATRA$ $n_i/n_f/f_{opt}$	$FATRM$ $n_i/n_f/f_{opt}$	$FATRA$ $n_i/n_f/f_{opt}$
Extended DENSCHNB [4]	1000	4/5/0	4/5/0	4/4/0	4/4/0
	5000	4/5/1.10e-027	4/5/1.10e-027	4/4/1.97e-027	4/4/1.97e-027
	10000	4/5/3.54e-026	4/5/3.54e-026	9/9/4.85e-022	9/9/4.85e-022
Extended Himmelblau [4]	1000	15/19/5.76e-018	15/19/5.76e-018	15/17/1.22e-024	15/17/1.22e-024
	5000	15/19/2.88e-018	15/19/2.88e-018	15/15/5.88e-016	15/15/5.88e-016
	10000	15/19/5.76e-018	15/19/5.76e-018	14/14/5.76e-018	14/14/5.76e-018
Extended PSC1 [4]	100	14/17/38.65	14/17/38.65	15/16/38.65	15/16/ 38.65
	500	14/17/1.93e+002	14/17/1.93e+002	14/15/1.93e+002	14/15/1.93e+002
	1000	14/17/3.86e+002	14/17/3.86e+002	15/16/3.86e+002	15/16/3.86e+002
Extended Tridiagonal 1 [4, 30]	1000	27/28/5.91e-009	27/28/5.91e-009	29/34/2.08e-009	29/34/2.08e-009
	5000	22/23/1.07e-008	22/23/1.07e-008	29/34/3.31e-009	28/34/3.39e-009
	10000	33/37/5.95e-009	33/37/5.95e-009	35/38/9.31e-009	35/38/9.31e-009

Table 1: The numerical results (continued)

Prob	n	$NATRM$ $n_i/n_f/f_{opt}$	$NATRA$ $n_i/n_f/f_{opt}$	$FATRM$ $n_i/n_f/f_{opt}$	$FATRA$ $n_i/n_f/f_{opt}$
Extended White and Holst [4, 30]	1000	81/126/9.10e-013	81/126/9.10e-013	51/74/1.29e-016	51/74/1.29e-016
	5000	82/127/2.27e-012	82/127/2.27e-012	51/64/1.77e-013	51/64/1.77e-013
	10000	84/129/4.09e-018	84/129/4.09e-018	71/98/1.41e-012	71/98/1.41e-012
Extended Wood [4]	1000	362/535/1.16e-014	362/535/1.16e-014	727/1486/2.71e-015	729/1477/1.14e-013
	5000	340/482/1.71e-014	340/482/1.71e-014	738/1347/5.33e-016	775/1477/9.67e-014
	10000	154/234/4.16e-013	163/247/2.71e-016	810/1494/1.21e-013	819/1667/1.26e-014
FLETCHCR [4]	100	1628/2261/5.15e-013	1342/1998/5.49e-013	1470/2969/3.70e-013	1349/2816/4.92e-013
	500	17074/25117/1.48e-011	10312/16396/9.85e-012	11449/24915/1.37e-011	9827/23269/1.001e-011
	1000	Failed	21632/36672/5.86e-011	Failed	Failed
Full Hessian FH2 [4]	100	958/1365/6.90e-013	1806/3034/5.98e-014	1185/2513/1.55e-013	1386/3161/9.65e-013
	500	10604/15314/9.06e-013	7234/11124/8.75e-014	8535/18335/4.49e-013	11448/26746/7.49e-013
	1000	31551/45869/7.05e-013	Failed	Failed	Failed

Table 1: The numerical results (continued)

Prob	n	$NATRM$ $n_i/n_f/f_{opt}$	$NATRA$ $n_i/n_f/f_{opt}$	$FATRM$ $n_i/n_f/f_{opt}$	$FATRA$ $n_i/n_f/f_{opt}$
Full Hessian FH3 [4]	1000	5/11/-0.24	5/11/-0.24	5/6/-0.24	5/6/-0.24
	5000	5/12/-0.24	5/12/-0.24	5/5/-0.24	5/5/-0.24
	10000	5/12/-0.24	5/12/-0.24	4/4/-0.24	4/4/-0.24
Generalized Quartic [4]	1000	12/14/2.46e-019	12/14/2.46e-019	12/13/4.31e-018	12/13/4.31e-018
	5000	11/13/6.55e-015	11/13/6.55e-015	14/14/8.49e-021	14/14/8.49e-021
	10000	10/12/9.61e-014	10/12/9.61e-014	10/10/5.74e-021	10/10/5.74e-021
Generalized Rosenbrock [4]	100	3866/5551/6.41e-013	3653/5351/5.45e-014	3675/7761/6.30e-013	3574/7681/9.96e-013
	500	13137/18512/9.78e-013	12942/18263/9.15e-013	12098/24912/2.10e-013	12031/24890/7.25e-013
	1000	24527/34507/9.87e-013	24550/34658/2.27e-013	23410/48550/2.62e-013	23352/48494/5.17e-014
Generalized Tridiagonal 1 [4]	100	29/31/97.21	29/31/97.21	29/30/97.21	29/30/97.21
	500	29/31/4.97e+002	29/31/4.97e+002	29/30/4.97e+002	29/30/4.97e+002
	1000	29/31/9.97e+002	29/31/9.97e+002	29/30/9.97e+002	29/30/9.97e+002

Table 1: The numerical results (continued)

Prob	n	$NATRM$	$NATRA$	$FATRM$	$FATRA$
		$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$
IE [16]	1000	9/10/ 1.54e-011	10/11/4.50e-15	9/9/ 1.26e-11	10/10/3.51e-15
	5000	10/11/2.25e-014	10/11/2.25e-014	10/10/1.75e-014	10/10/1.75e-014
	10000	10/11/ 9.00e-015	10/11/ 9.00e-015	10/10/7.02e-15	10/10/7.02e-15
LIARWHD [4]	1000	53/84/2.89e-018	53/84/2.89e-018	64/102/2.83e-018	64/102/2.83e-018
	5000	49/82/2.48e-021	49/82/2.48e-021	46/69/1.36e-014	46/69/1.36e-014
	10000	58/105/2.36e-019	58/105/2.36e-019	58/88/4.02e-017	58/88/4.02e-017
NONDIA [4]	100	22/30/1.59e-023	22/30/1.59e-023	19/22/4.53e-016	19/22/4.53e-016
	500	24/33/3.75e-024	24/33/3.75e-024	17/23/5.34e-021	17/23/5.34e-021
	1000	18/28/1.69e-013	18/28/1.69e-013	17/23/1.09e-018	17/23/1.09e-018
PEN1 [16]	100	309/555/9.02e-004	162/290/ 9.02e-04	35/39/ 9.02e-004	42/46/9.02e-04
	500	252/426/0.004	290/526/0.004	107/121/0.004	107/121/0.004
	1000	87/165/9.68e-3	204 /338/ 9.68e-3	229/249/0.9.68e-3	230/250/9.68e-3

Table 1: The numerical results (continued)

Prob	n	$NATRM$	$NATRA$	$FATRM$	$FATRA$
		$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$	$n_i/n_f/f_{opt}$
Perturbed Quadratic [4]	1000	420/555/2.23e-015	637/919/2.33e-013	513/1020/2.28e-013	538/1100/1.61e-013
	5000	1928/2826/2.42e-013	1675/2582/2.33e-013	1138/2379/1.77e-013	1046/2277/1.84e-013
	10000	2007/2927/4.42637l4e-014	2112/3291/1.32e-013	1816/3947/2.32e-013	1671/3790/2.45e-013
Perturbed quadratic diagonal [4]	1000	470/735/1.20e-011	1324/2497/2.03e-011	522/1238/1.68e-011	1103/2921/1.50e-011
	5000	1219/2003/2.17e-011	2789/5432/1.92e-011	1407/3528/1.98e-011	1152/3067/1.32e-011
	10000	1867/3122/5.17e-012	5785/11286/2.21e-011	1289/3250/8.13e-012	2687/7408/1.61e-011
Quadratic QF1 [4]	1000	451/587/-4.99e-004	515/723/-4.99e-004	576/1194/-4.99e-004	694/1470/-4.99e-004
	5000	2034/2931/-9.99e-005	1598/2446/-9.99e-005	2147/4663/-9.99e-005	1889/4172/-9.99e-005
	10000	2467/3604/-4.99e-005	2789/4392/-4.99e-005	2445/5195/-4.99e-005	1651/3483/-4.99e-005
QUARTC [4]	1000	2/3/0	2/3/0	2/2/0	2/2/0
	5000	2/3/0	2/3/0	2/2/0	2/2/0
	10000	2/3/0	2/3/0	2/2/0	2/2/0

Table 1: The numerical results

$Prob$	n	$NATRM$ $n_i/n_f/f_{opt}$	$NATRA$ $n_i/n_f/f_{opt}$	$FATRM$ $n_i/n_f/f_{opt}$	$FATRA$ $n_i/n_f/f_{opt}$
Raydan 1 [4,30]	1000	8/8/10000	8/8/10000	7/7/10000	7/7/10000
	5000	8/8/5000	8/8/5000	7/7/5000	7/7/5000
	10000	8/8/10000	8/8/10000	7/7/10000	7/7/10000
ROSEX [16]	1000	61/122 / 4.50e-021	64/100/5.18e-17	68/109/2.05e-014	68/109/2.05e-14
	5000	60/118 /2.73e-020	67/105 /2.93e-17	32/38/3.41e-016	32/38/3.41e-16
	10000	56/115/5.95e-016	71/151/4.66e-16	44/58/ 7.75e-016	44/58/3.23e-16
SINGX [16]	100	517/775/ 2.64e-009	7724 / 14030 / 4.01e-09	373/769/9.74e-008	491/1003/ 1.84e-09
	500	384/595/2.91e-009	7474/13674/2.91e-009	575/1178/5.88e-009	575/1178/5.88e-009
	1000	28640/46632/2.66e-006	3421 /6120/ 5.70e-09	350/479/1.42e-007	1045/2440/4.90e-09
TRIDIA [4]	1000	1984/2896/3.30e-013	2510/3880/3.42e-013	2657/5835/3.41e-013	2185/4915/4.29e-014
	5000	13693/20572/3.39e-013	11464/17956/6.38e-014	10928/24370/3.46e-013	9142/21255/3.40e-013
	10000	20721/31085/3.39e-013	13317/21063/1.53e-014	22037/48796/2.59e-013	18893/42973/4.81e-017

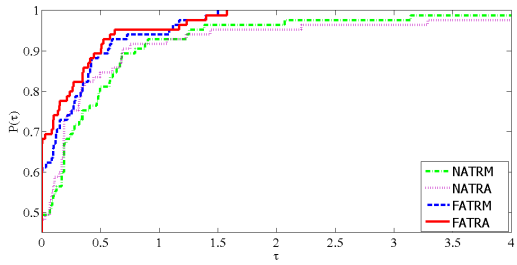

Figure 1: Performance profile of considered algorithms based on n_i

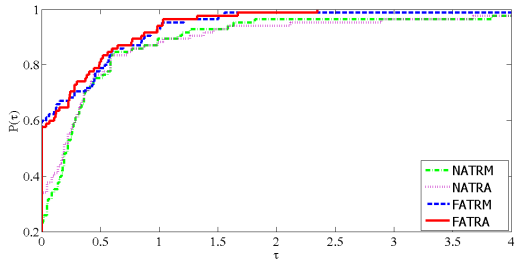

Figure 2: Performance profile of considered algorithms based on n_f

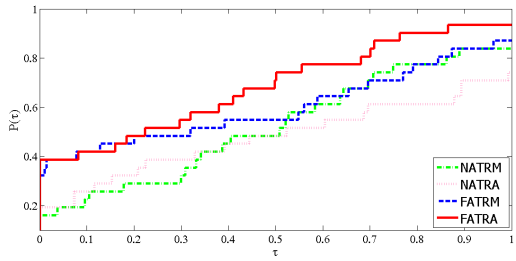

Figure 3: Performance profile of considered algorithms based on CPU time

5 Conclusion

In this paper, a new nonmonotone adaptive trust region method for solving unconstrained optimization problems based on a simple subproblem is presented. The new proposed algorithm uses the advantage of the adaptive trust region method, as proposed in [5], with the nonmonotone term, as suggested in [2]. The global convergence property of the new proposed method

is established under some standard assumptions. Numerical results on some large-scale test problems confirm the efficiency and effectiveness of the new proposed algorithm in comparison with some other existing algorithms in the literature.

Acknowledgment: The authors would like to thank the Research Council of K. N. Toosi University of Technology and the SCOPE research center for supporting this work. The authors also would like to appreciate D. Ataee Tarzanagh for his helpful and constructive comments.

References

1. Ahookhosh, M. and Amini, K. *A nonmonotone trust region method with adaptive radius for unconstrained optimization*, Comput. Math. Appl. 60(2010) 411–422.

2. Ahookhosh, M. and Amini, K. *An efficient nonmonotone trust-region method for unconstrained optimization*, Numer. Algorithms 59(2011) 523–540.

3. Ahookhosh, M., Amini, K. and Peyghami, M.R. *A nonmonotone trust-region line search method for large- scale unconstrained optimization*, Appl. Math. Model. 36(2012) 478–487.

4. Andrei, N. *An unconstrained optimization test functions collection*, Adv. Model. Optim. 10(1)(2008) 147–161.

5. Ataee Tarzanagh, D., Peyghami, M. R. and Mesgarani, H. *A new nonmonotone trust region method for unconstrained optimization equipped by an efficient adaptive radius*, Optim. Methods Softw. 29(4)(2014) 819–836.

6. Biglari, F., Hassan, M. and Leong, W. J. *New quasi-Newton methods via higher order tensor models*, J. Comput. Appl. Math. 235(2011) 2412–2422.

7. Biglari, F. and Solimanpur, M. *Scaling on the Spectral Gradient Method*, J. Optim. Theory Appl. 158(2)(2013) 626–635.

8. Chamberlain, R. M., Powell, M. J. D., Lemarechal, C. and Pedersen, H. C. *The watchdog technique for forcing convergence in algorithm for constrained optimization*, Math. Program. Stud. 16(1982) 1–17.

9. Conn, A., Gould, N. and Toint, Ph. L. *Trust Region Methods*, SIAM, Philadelphia, 2000.

10. Dolan, E. and Moré, J. J. *Benchmarking optimization software with performance profiles*, Math. Program. 91(2002) 201–213.

11. Fan, J. Y. and Yuan, Y. X. *A new trust region algorithm with trust region radius converging to zero*, Proceedings of the 5th International Conference on Optimization, Techniques and Applications, 2001.

12. Gertz, E. M. *A quasi-Newton trust-region method*, Math. Program. Ser. A. 100(3)(2004) 447–470.

13. Grippo, L., Lampariello, F. and Lucidi, S. *A nonmonotone line search technique for Newton's method*, SIAM J. Numer. Anal. 23(1986) 707–716.

14. Grippo, L., Lampariello, F. and Lucidi, S. *A truncated Newton method with nonmonotone line-search for unconstrained optimization*, J. Optim. Theory Appl. 60(1989) 401–419.

15. Li, G. D. *A trust region method with automatic determination of the trust region radius*, Chinese J. Engry. Math. 23(2006) 843–848.

16. Moré, J. J., Garbow, B. S. and Hilstron, K. E. *Testing unconstrained optimization software*, ACM Tran. Math. Software 7(1981) 17–41.

17. Nocedal, J. and Wright, S. J. *Numerical Optimization*, Springer, New York, 2006.

18. Nocedal, J. and Yuan, Y. *Combining trust region and line search techniques*, In: Y. Yuan (ed.), Advanced in Nonlinear Programming, Kluwer Academic, Dordrecht, 153–175, 1996.

19. Powell, M. J. D. *On the global convergence of trust region algorithms for unconstrained optimization*, Math. Program. 29(1984) 297–303.

20. Sang, Z. and Sun, Q. *A self-adaptive trust region method with line search based on a simple subproblem model*, J. Appl. Math. Comput. 232(2009) 514–522.

21. Sartenaer, A. *Automatic determination of an initial trust region in nonlinear programming*, SIAM J. Sci. Comput. 18(6)(1997) 1788–1803.

22. Shi, Z. and Guo, J. *A new trust region method for unconstrained optimization*, J. Comput. Appl. Math. 213(2008) 509–520.

23. Shi, Z. J. and Wang, H. Q. *A new self-adaptive trust region method for unconstrained optimization*, Technical Report, College of Operations Research and Management, Qufu Normal University, 2004.

24. Shultz, G. A., Schnabel, R. B. and Byrd, A. R. H. *A family of trust-region-based algorithm for unconstrained minimization with strong global convergence properties*, SIAM J. Numer. Anal. 22(1)(1985) 47–67.

25. Toint, Ph. L. *Non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints*, Math. Program. 77(1997) 69–94.

26. Wei, Z., Li, G. and Qi, L. *New quasi-Newton methods for unconstrained optimization problems*, Appl. Math. Comput. 175(2006) 1156–1188.

27. Zhang, H. C. and Hager, W. W. *A nonmonotone line search technique and its application to unconstrained optimization*, SIAM J. Optim. 14(2004) 1043–1056.

28. Zhang, X. S., Zhang, J. L. and Liao, L. Z. *A nonmonotone adaptive trust region method and its convergence*, Int. J. Comput. Math. Appl. 45(2003) 1469–1477.

29. Zhang, X. S., Zhang, J. L. and Liao, L. Z. *An adaptive trust region method and its convergence*, Sci. China 45(2002) 620–631.

30. Zhou, Q. and Zhang, C. *A new nonmonotone adaptive trust region method based on simple quadratic models*, J. Appl. Math. Comput. 40(2012) 111–123.

Persian Translation of Abstracts

روش‌های مشتق دوم مرتبه بالا با پایداری رانگ-کوتا برای حل عددی معادلات دیفرانسیل معمولی سخت

علی عبدی و غلامرضا حجتی

دانشگاه تبریز، دانشکده علوم ریاضی

چکیده : ساخت روش-های خطی عمومی مشتق دوم (SGLMs) از مراتب پنج و شش را بحث و توصیف می کنیم. در این بررسی، روش-های ساخته شده A - پایدار بوده و دارای خاصیت پایداری رانگ-کوتا هستند. برخی نتایج عددی برای نشان دادن کارایی روش-های ساخته شده برای حل مسائل مقدار اولیه سخت ارائه می شوند .

کلمات کلیدی : معادله دیفرانسیل معمولی؛ روش-های خطی عمومی؛ پایداری رانگ-کوتا؛ A - پایداری؛ روش-های مشتق دوم.

الگوریتم بلوکیLSMR برای حل دستگاه معادلات خطی با چند طرف ثانی

فائزه توتونیان [۲,۱] *و مریم محرب* [۳,۱]

[۱] دانشگاه فردوسی مشهد، دانشکده علوم ریاضی، گروه ریاضی کاربردی
[۲] دانشگاه فردوسی مشهد، قطب علمی مدلسازی و کنترل دستگاه ها
[۳] دانشگاه سیستان و بلوچستان، گروه ریاضی

چکیده : LSMR (مانده مینیمال کمترین توان‌های دوم) یک روش تکراری برای حل دستگاه معادلات خطی و مسائل کمترین توان‌های دوم می‌باشد. این مقاله یک نسخه بلوکی از الگوریتم LSMR را برای حل دستگاه‌های خطی با چند طرف ثانی ارائه می‌دهد. الگوریتم جدید مبتنی بر دوقطری سازی بلوکی است و از مینیمم سازی نرم فربینیوس معادلات نرمال ماتریس مانده نتیجه می‌شود. به‌علاوه، همگرایی الگوریتم پیشنهادی مورد بحث قرار می‌گیرد. همچنین، در عمل ملاحظه می‌شود که نرم فربینیوس ماتریس مانده به طور یکنواخت کاهش می‌یابد. در نهایت، آزمایش‌های عددی که بر روی مسائل کاربردی واقعی پیاده سازی شده‌اند، کارایی روش ارائه شده را تایید خواهند کرد.

کلمات کلیدی : روش LSMR ؛ دوقطری سازی؛ روش‌های بلوکی؛ روش‌های تکراری؛ چند طرف ثانی.

مروری عملی از روش تجزیه آدومیان: جنبه های پیاده سازی کامپیوتری

احمد ملابهرامی

دانشگاه ایلام، گروه ریاضی

چکیده : در این مقاله، مروی عملی از روش تجزیه آدومیان، جهت توسعه آن برای بررسی مسایل غیرخطی قوی تحت شرایط آمیخته، ارایه و همگرایی الگوریتم به صورت دقیق اثبات می شود. برای دست یابی به این هدف، یک روش جدید و ساده برای تولید چندجمله ای های آدومیان، برای فرم کلی یک تابع غیرخطی، ارایه می گردد. روش ارایه شده، یک فرمول صریح برای محاسبه چندجمله ای های آدومیان یک تابع غیرخطی فراهم می کند. کارایی رهیافت با به کارگیری آن روی چندین مسئله ی جالب انتگرال- دیفرانسیل نشان داده خواهد شد. برنامه های متمتیکای تولید چندجمله ای ها و جواب های آدومیان بر اساس رویه های این مقاله طرح یزی شده است.

کلمات کلیدی : روش تجزیه آدومیان؛ چندجمله ای های آدومیان؛ مسایل غیرخطی انتگرال- دیفرانسیل؛ جواب سری؛ مسایل غیرخطی قوی؛ محاسبات و برنامه های صریح ماشینی.

یک روش خط بدون شبکه سازگار بر اساس توابع پایه شعاعی

جعفر بی آزار و محمد هوسمی

دانشگاه گیلان، دانشکده علوم ریاضي، گروه ریاضي کاربردي

چکیده : در این مقاله، از یک روش خط بدون شبکه سازگار برای توزیع نقاط در دامنه فضایی استفاده می شود. در روش های بدون شبکه، در موارد بسیاری لازم است که نقاط انتخاب شده شرایط همواری خاصی داشته باشند و مجموعه نقاط در محدودیت هایی صدق کند. در این مقاله، یکی از این محدودیت ها بررسی می شود. هدف از این مطالعه، به کار بردن الگوریتمی در روش خط بدون شبکه است برای انتخاب نقاطی که در شرایطی مشخص، صدق کنند. برای نشان دادن کارایی الگوریتم و توانایی آن در افزایش دقت، آنرا در تعدادی مثال نمونه به کار می بریم.

کلمات کلیدی : روش های بدون شبکه سازگار؛ روش خط بدون شبکه؛ توابع پایه شعاعی.

کاربرد روش معادله ساده توسعه یافته برای معادلات برگرز ، هوکسلی و هوکسلی برگرز

زینب آیاتی، مجتبی مرادی و محمد میرزازاده

دانشگاه گیلان، دانشکده فنی و مهندسی شرق گیلان، گروه علوم مهندسی

چکیده : در این مقاله، روش معادله ساده توسعه یافته برای به دست آوردن جواب-های معادلات برگرز، هوکسلی و شکل ترکیب شده آنها به کار رفته است. جوابهای دقیق جدیدی از این معادلات به دست آمده است.نشان داده شده است که روش ارائه شده یک ابزار ریاضی قوی و بسیار موثر برای حل معادلات با مشتقات جزئی می باشد.

کلمات کلیدی : روش معادله ساده توسعه یافته؛ معادله برگرز؛ معادله هوکسلی؛ معادله برگرز-هوکسلی.

بررسی شرایط همگرایی و پایداری یک جواب تقریبی مسأله معکوس هدایت گرما توسط روش اختلال هموتوپی

قدسیه جنتی و علی ذاکری

دانشگاه خواجه نصیر الدین طوسی، دانشکده ریاضی

چکیده : در این مقاله کاربرد روش اختلال هموتوپی برای حل یک مسأله معکوس هدایت گرمایی غیرخطی در فضای یک بعدی مورد بررسی قرار می‌گیرد. در این مسأله رسانش به صورت تابع خطی از دمای ناشناخته در یک دامنه کران‌دار است. علاوه بر این مقدار دما در یک کران دامنه تعریف مجهول است. مسأله موردنظر یک مسأله بدخیم است. با به کار بستن روش تفاضلات متناهی و گسسته سازی متغیر زمانی، مسأله به یک دستگاه معادلات دیفرانسیل غیرخطی تبدیل می‌شود که جواب تقریبی آن با استفاده از روش اختلال هموتوپی تعیین می‌گردد. در ادامه شرایط همگرایی و پایداری روش پیشنهادی مورد بررسی قرار می‌گیرد. در آخر یک کران بالای خطا ارائه می‌گردد.

کلمات کلیدی : روش اختلال هموتوپی ؛ معادلات نفوذ؛ روش‌های گسسته- سازی؛ مسائل معکوس.

یک روش ناحیه اعتماد وفقی نایکنوا برای مسایل بهینه سازی نامقید بر اساس یک زیرمساله ساده

زینب سعیدیان طریی[۱] *و محمدرضا پیغامی*[۲,۱]

[۱] دانشگاه صنعتی خواجه نصیر الدین طوسی، دانشکده ریاضی
[۲] دانشگاه صنعتی خواجه نصیر الدین طوسی، مرکز پژوهشی محاسبات علمی در بهینه سازی و مهندسی سامانه

چکیده : با بکارگیری یک مدل درجه دوم ساده در زیرمساله ناحیه اعتماد، یک روش ناحیه اعتماد وفقی نایکنوا برای حل مسایل بهینه-سازی نامقید پیشنهاد می-شود. در این روش، با اعمال اصلاح جزیی روی روش پیشنهادی در (J. Optim. Theory Appl. ۱۵۸ (۲) ۶۲۶-۶۳۵ ، ۲۰۱۳) یک تقریب اسکالر جدید از ماتریس هسین در نقطه فعلی ارایه می--گردد. روش پیشنهادی به یک روش وفقی جدید برای بهنگام شعاع ناحیه اعتماد و یک تکنیک نایکنوا مجهز شده است. تحت برخی فرضیات استاندارد و مناسب، خواص همگرایی سراسری و موضعی الگوریتم پیشنهادی به همراه نرخ همگرایی آن بررسی می-شوند. در پایان، عملکرد عملی الگوریتم پیشنهادی روی برخی مسایل آزمونی مورد بررسی قرار گرفته و نتایج حاصل با برخی الگوریتم-های موجود در ادبیات موضوع مورد مقایسه قرار می-گیرند.

کلمات کلیدی : روش-های ناحیه اعتماد؛ شعاع وفقی؛ تکنیک نایکنوا؛ تقریب اسکالر ماتریس هسین؛ همگرایی سراسری.

Aims and scope

Iranian Journal of Numerical Analysis and Optimization (IJNAO) is published twice a year by the Department of Applied Mathematics, Faculty of Mathematical Sciences, Ferdowsi University of Mashhad. Papers dealing with different aspects of numerical analysis and optimization, theories and their applications in engineering and industry are considered for publication.

Journal Policy

After receiving an article, the editorial committee will assign referees. Refereeing process can be followed via the web site of the Journal.
The manuscripts are accepted for review with the understanding that the work has not been published and also it is not under consideration for publication by any other journal. All submissions should be accompanied by a written declaration signed by the author(s) that the paper has not been published before and has not been submitted for consideration elsewhere.

Instruction for Authors

The Journal publishes all papers in the fields of numerical analysis and optimization. Articles must be written in English.
All submitted papers will be refereed and the authors may be asked to revise their manuscripts according to the referee's reports. The Editorial Board of the Journal keeps the right to accept or reject the papers for publication.
The papers with more than one authors, should determine the corresponding author. The e-mail address of the corresponding author must appear at the end of the manuscript or as a footnote of the first page.
It is strongly recommended to set up the manuscript by Latex or Tex, using the template provided in the web site of the Journal. Manuscripts should be typed double-spaced with wide margins to provide enough room for editorial remarks.
References should be arranged in alphabetical order by the surname of the first author as examples below:
[1] Stoer, J. and Bulirsch, R. *Introduction to Numerical Analysis*, Springer-verlag, New York, 2002.
[2] Brunner, H. *A survey of recent advances in the numerical treatment of Volterra integral and integro-differential equations*, J. Comput. Appl. Math. 8 (1982), 213-229.

Submission of Manuscripts

Authors may submit their manuscripts by either of the following ways:

a) Online submission (pdf or dvi files) via the website of the Journal at:

http://ijnao.um.ac.ir

b) Via journal's email mjms@um.ac.ir

Copyright Agreement

Upon the acceptance of an article by the Journal, the corresponding author will be asked to sign a "Copyright Transfer Agreement" (see the web site) and send it to the Journal address. This will permit the publisher to publish and distribute the work.

Iranian Journal of Numerical Analysis and Optimization

Former : MRJMS

CONTENTS Vol. 5, No.2, pp 1-117, 2015

High order second derivative methods with Runge--Kutta stability for the numerical solution of stiff ODEs 1
A. Abdi and G. Hojjati

The block LSMR algorithm for solving linear systems with multiple right-hand sides 11
F. Toutounian and M. Mojarrab

A practical review of the Adomian decomposition method: computer implementation aspects 29
A. Molabahrami

An adaptive meshless method of line based on radial basis functions 45
J. Biazar and M. Hosami

Application of modified simple equation method to Burgers, Huxley and Burgers-Huxley equations 59
Z. Ayati, M. Moradi and M. Mirzazadeh

On convergence and stability conditions of homotopy perturbation method for an inverse heat conduction problem 75
Q. Jannati and A. Zakeri

An adaptive nonmonotone trust region method for unconstrained optimization problems based on a simple subproblem 95
Z. Saeidian and M.R. Peyghami

web site : http://ijnao.um.ac.ir

Email : mjms@um.ac.ir

ISSN : 2423-6977