Iranian Journal of Numerical Analysis and Optimization

Volume 5 , Number 1

Winter 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 1, Winter 2015

ISSN: 1735-7144

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:** +98-51-38828604

E-mail: mjms@um.ac.ir

Website: http://jm.um.ac.ir/index.php/math

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 1, Winter 2015

Ferdowsi University of Mashhad - Iran

Iranian Journal of Numerical Analysis and Optimization

Editor in Charge

H. R. Tareghian*

Editor in Chief

M. H. Farahi

Managing Editor

M. Gachpazan

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.

I would like to express my appreciation to Dr H. Taghizadeh Kakhki who has just left the Editorial Board for his valuable assistance to IJNAO. Also, I would like to welcome Professor A.V. Kamyad, from Ferdowsi University of Mashhad, and Dr A. Emrouznejad, Reader of Aston University, Birmingham, U.K., who recently joined the Editorial Board.

Mohammad Hadi Farahi

Contents

Iranian Journal of Numerical Analysis and Optimization
Vol 5, No. 1, (2015), pp 1-12

Strong approximation for Itô stochastic differential equations

M. Namjoo*

Abstract

In this paper, a class of semi-implicit two-stage stochastic Runge-Kutta methods (SRKs) of strong global order one, with minimum principal error constants are given. These methods are applied to solve Itô stochastic differential equations (SDEs) with a Wiener process. The efficiency of this method with respect to explicit two-stage Itô Runge-Kutta methods (IRKs), Itô method, Milstien method, semi-implicit and implicit two-stage Stratonovich Runge-Kutta methods are demonstrated by presenting some numerical results.

Keywords: Stochastic differential equations; Strong approximation; Runge-Kutta methods.

1 Introduction

In recent years, a great deal of concern has been raised regarding the study of SDEs as an important area of research. Many phenomena in science and engineering have been modeled by deterministic ordinary differential equations (DODEs). However, some of the parameters and initial data are not known with complete certainty due to lack of information. Therefore, to represent a more accurate model of the behavior of such phenomena they usually should be modeled by SDEs. Some areas where SDEs have been used extensively in modeling phenomena include chemistry, physics, engineering, mathematical biology and finance (see, for example, [5], [7]). Since explicit solutions are known only for a few equations, the study of numerical methods have become more important and these must be designed to be implemented with a certain order of accuracy. Consider the autonomous Itô SDE given by

$$dy(t) = g_0(y(t))\mathrm{d}t + g_1(y(t))\mathrm{d}W(t), \qquad y(t_0) = y_0, \qquad t \in [t_0, t_f], \tag{1}$$

*Corresponding author

Received 6 April 2014; revised 21 July 2014; accepted 11 August 2014

M. Namjoo
Department of Mathematics, School of Mathematical Sciences, Vali-e-Asr University of Rafsanjan, Rafsanjan, Iran. e-mail: namjoo@vru.ac.ir

where g_0 and g_1 are real-valued functions which are called the drift coefficient and the diffusion coefficient, respectively, and $W(t)$ is a one-dimensional standard Wiener process, whose increment $\Delta W(t) = W(t+h) - W(t)$ has a Gaussian distribution with mean 0 and variance h, i.e. $W(t+h) - W(t) \sim N(0,h) = \sqrt{h}\ N(0,1)$, and the solution $y(t)$ is an Itô process. A Wiener process (named after N. Wiener) is sometimes called Brownian Motion, which is a term used to describe the phenomenon of the erratic behaviour of a particle in a liquid, acted on by random impulses, in the absence of friction. Equation (1) can also be written as a stochastic integral equation

$$y(t) = y_0 + \int_{t_0}^{t} g_0(y(s))\mathrm{d}s + \int_{t_0}^{t} g_1(y(s))\mathrm{d}W(s),$$

where the first integral is a mean square Riemann-Stieltjes integral and the second integral is a stochastic integral which can be interpreted in many ways (see [10]). The two most studied interpretations are due to Itô and Stratonovich that depend on the points of the partitioning in which the integrand is evaluated. If the lower end point t_n is chosen, it leads to Itô integral and if midpoint $(t_n + t_{n+1})/2$ is chosen, it leads to Stratonovich integral. The Stratonovich interpretation follows the common rules of integral calculus, while the Itô formulation has the advantage of preserving the martingale property of Wiener process. It is always possible to switch from one interpretation to the other, because an Itô SDE can be converted to a Stratonovich SDE (and vice versa) by means of the following formula (see [5])

$$\bar{g}_0(y) = g_0(y) - \frac{1}{2} g_1'(y) g_1(y),$$

where equation (1) is in the Stratonovich form when $\bar{g}_0$ is used in place of g_0. There are different numerical methods for solving these kinds of differential equations (see, for example, [1], [6], [8]). Numerical methods for SDEs are recursive methods where trajectories, in other words, the sample paths of solution are computed at discrete time steps. These methods are classified to strong and weak. Only strong convergence will be considered in this paper. Strong convergence is required, when each trajectory of the numerical method must be closed to the exact solution. Formally, if y_N is the numerical approximation to $y(t_N)$ after N steps with constant stepsize $h = (t_f - t_0)/N$, then y_N is said to converge strongly to $y(t_N)$ with strong order p if there exists $C > 0$ (independent of h) and $\delta > 0$ such that

$$\mathbb{E}(|y_N - y(t_N)|) \leq Ch^p, \qquad h \in (0, \delta).$$

An outline of this paper is as follows: In Section 2, the semi-implicit SRKs for SDEs are introduced, moreover order conditions for a class of SRKs with order one are stated. In particular, the new class of semi-implicit two-stage SRKs for SDEs with minimum principal error constants is constructed and the fixed point iteration algorithm will be used to improve the semi-implicit method. In Section 3 we have some numerical results which show the efficiency of this method.

2 The semi-implicit Itô Runge-Kutta methods for SDEs

The most famous numerical method that can be obtained from a stochastic Taylor expansion is Milstein method. This method for the SDE problem (1) is given by

$$y_{n+1} = y_n + hg_0(y_n) + J_1 g_1(y_n) + \frac{1}{2}({J_1}^2 - h)g_1'(y_n)g_1(y_n),$$

where $J_1 = W(t_n+h) - W(t_n)$ with $h = (t_f - t_0)/N$ for some integer N. This method converges with strong order one as long as $\mathbb{E}(y_0^2) < \infty$, and g_0, g_0', g_1, g_1' and g_1'' satisfy a uniform Lipschitz condition. Higher order numerical methods can be obtained by truncating farther terms of the stochastic Taylor expansion. This technique involves considerable complexities in implementation because of the approximation of higher order stochastic integrals and the evaluation of high order derivatives of both the drift and diffusion coefficients. Thus, it is important to be able to derive derivatives free numerical methods and this leads to SRKs. For the SDE (1) SRKs is given by (see [2]):

$$\begin{aligned} Y_i &= y_n + \sum_{j=1}^{s} Z_{ij}^{(0)} g_0(Y_j) + \sum_{j=1}^{s} Z_{ij}^{(1)} g_1(Y_j), \qquad i = 1, 2, \ldots, s, \\ y_{n+1} &= y_n + \sum_{j=1}^{s} z_j^{(0)} g_0(Y_j) + \sum_{j=1}^{s} z_j^{(1)} g_1(Y_j), \end{aligned} \tag{2}$$

which can be represented in tableau form as

$$\begin{array}{|c|c} Z^{(0)} & Z^{(1)} \\ \hline {z^{(0)}}^T & {z^{(1)}}^T \end{array}\ ,$$

where $Z^{(k)} = (Z_{ij}^{(k)})$ for $i, j = 1, 2, \ldots, s$ and ${z^{(k)}}^T = (z_1^{(k)}, \ldots, z_s^{(k)})$ represents for $k = 0, 1$. Here $Y_1, \ldots, Y_s$ represent the internal stage of the method, and y_{n+1} is the update of the numerical solution at the end of the current step. Since (2) is a generalization of the class of Runge-Kutta methods in deterministic case, for consistency the stepsize will be included in the parameter matrix associated with the deterministic components, so $Z^{(0)} = hA$ and ${z^{(0)}}^T = h\alpha^T$, while $Z^{(1)}$ and ${z^{(1)}}^T$ have elements that are arbitrary random variables. In order to derive methods with strong global order one, the existence of stochastic Taylor series expansion of the SRK method in the Itô case and the Itô Taylor series expansion of the exact solution is necessary. By comparing these two expansions, the local truncation error over one step with an exact initial value can be written as (see [3]):

$$L(t_0+h) = y(t_0+h) - Y(t_0+h) = \sum_{t\in T^\star} e(t)\ F(t)y_0,$$

where $e(t)$ and $F(t)y_0$ are called the local truncation error coefficients and the elementary differential for tree t, respectively and $T^\star$ is the set of bi-coloured rooted trees . Assuming certain conditions on the cofficients of the method and satisfying Lipschitz condition for the drift and diffusion cofficients SDE, a method will have strong global convergence of order one if it has strong local order one and mean local order one (see [3]). In [9] the order one conditions for a class of IRKs in the form

$$\begin{gathered} Z^{(0)} = hA, \qquad {z^{(0)}}^T = h\alpha^T, \qquad Z^{(1)} = \sqrt{h}B^{(1)} + J_1 B^{(2)}, \\ {z^{(1)}}^T = \sqrt{h}{\gamma^{(1)}}^T + J_1 {\gamma^{(2)}}^T, \end{gathered} \tag{3}$$

are given, where A, $B^{(1)}$ and $B^{(2)}$ are $s\times s$ real matrices, and $\alpha^T = (\alpha_1,\dots,\alpha_s)$, ${\gamma^{(1)}}^T = (\gamma_1^{(1)},\gamma_2^{(1)},\dots,\gamma_s^{(1)})$ and ${\gamma^{(2)}}^T = (\gamma_1^{(2)},\gamma_2^{(2)},\dots,\gamma_s^{(2)})$ are row s-dimensional vectors. In fact a SRK method of the form (3) will have strong global order one if (see [9])

$$\begin{cases} \alpha^T e = 1, \\ {\gamma^{(1)}}^T e = 0, \\ {\gamma^{(2)}}^T e = 1, \\ {\gamma^{(1)}}^T B^{(1)} e = -\frac{1}{2}, \\ {\gamma^{(1)}}^T B^{(2)} e + {\gamma^{(2)}}^T B^{(1)} e = 0, \\ {\gamma^{(2)}}^T B^{(2)} e = \frac{1}{2}, \\ \alpha^T B^{(1)} e = 0, \\ {\gamma^{(1)}}^T A e = 0, \\ {\gamma^{(1)}}^T (B^{(1)}e)^2 + {\gamma^{(1)}}^T (B^{(2)}e)^2 + 2{\gamma^{(2)}}^T (B^{(1)}e)(B^{(2)}e) = 0, \\ {\gamma^{(1)}}^T {B^{(1)}}^2 e + {\gamma^{(1)}}^T {B^{(2)}}^2 e + {\gamma^{(2)}}^T (B^{(1)}B^{(2)}e + B^{(2)}B^{(1)}e) = 0. \end{cases} \tag{4}$$

Here $e = (1,\dots,1)^T \in \mathbb{R}^s$ and multiplication of vectors are componentwise. If the matrices A, $B^{(1)}$ and $B^{(2)}$ are strictly lower triangular, then the method (3) is said to be explicit, while if A, $B^{(1)}$ and $B^{(2)}$ are lower triangular, then the method (3) is said to be semi-implicit. A family of two-stage explicit SRKs of the form (3) with minimum principal error terms can be presented by the following tableau (see [9]):

$$\begin{array}{c|cc|cc} & 0 & 0 & 0 & 0 \\ & 0 & 0 & -\frac{1}{2}(\sqrt{h}-J_1) & 0 \\ \hline & h & 0 & -\sqrt{h} & \sqrt{h}+J_1 \end{array}$$

which is called '$EM1$', and has the principal error constants

$$\frac{1}{3}h^3,\ \frac{1}{3}h^3,\ \frac{1}{6}h^3,\ \frac{1}{16}h^3,$$

and the other family of two-stage explicit methods satisfying (4) with minimum principal error constants can be presented by (see [9]):

$$\begin{array}{c|cc|cc} & 0 & 0 & 0 & 0 \\ & 0 & 0 & \frac{1}{2}(\sqrt{h}+J_1) & 0 \\ \hline & h & 0 & \sqrt{h} & -\sqrt{h}+J_1 \end{array}$$

which is called '$EM2$', and has the principal error constants

$$\frac{1}{3}h^3,\ \frac{1}{3}h^3,\ \frac{1}{6}h^3,\ \frac{1}{16}h^3.$$

Also the Itô method (see [2]) that is a derivative free version of the Milstein method with strong global order one, can be presented by the following tableau:

$$\begin{array}{c|cc|cc} & 0 & 0 & 0 & 0 \\ & 0 & 0 & \sqrt{h} & 0 \\ \hline & h & 0 & J_1-\frac{\sqrt{h}}{2}((\frac{J_1}{\sqrt{h}})^2-1) & \frac{\sqrt{h}}{2}((\frac{J_1}{\sqrt{h}})^2-1) \end{array}$$

This method is called 'IRK' and has the principal error constants

$$\frac{1}{3}h^3,\ \frac{1}{3}h^3,\ \frac{1}{6}h^3,\ \frac{3}{8}h^3.$$

In [1] a class of semi-implicit and implicit Stratonovich Runge-Kutta methods of strong order one with minimum principal error constants for SDEs is constructed. More precisely, this class of semi-implicit and implicit two-stage Stratonovich Runge-Kutta methods with minimum principal error cofficients can be presented, by the following tableau (see [1])

$$\begin{array}{c|cc|cc} & \frac{3+\sqrt{3}}{6}h & 0 & \frac{3+\sqrt{3}}{6}J_1 & 0 \\ & -\frac{\sqrt{3}}{3}h & \frac{3+\sqrt{3}}{6}h & -\frac{\sqrt{3}}{3}J_1 & \frac{3+\sqrt{3}}{6}J_1 \\ \hline & \frac{1}{2}h & \frac{1}{2}h & \frac{1}{2}J_1 & \frac{1}{2}J_1 \end{array}$$

and

$$\begin{array}{c|cc|cc} & \frac{1}{4}h & \frac{3-2\sqrt{3}}{12}h & \frac{1}{4}J_1 & \frac{3-2\sqrt{3}}{12}J_1 \\ & \frac{3-2\sqrt{3}}{12}h & \frac{1}{4}h & \frac{3-2\sqrt{3}}{12}J_1 & \frac{1}{4}J_1 \\ \hline & \frac{1}{2}h & \frac{1}{2}h & \frac{1}{2}J_1 & \frac{1}{2}J_1 \end{array}$$

which are called '$SIM1$' and 'IM', respectively. In order to generalize the above explicit IRKs to semi-implicit case, consider $s=2$, hence the matrices $A,\ B^{(1)}$ and $B^{(2)}$ will have the following forms:

$$A=\begin{pmatrix} a_{11} & 0\\ a_{21} & a_{22}\end{pmatrix},\quad B^{(1)}=\begin{pmatrix} b^{(1)}_{11} & 0\\ b^{(1)}_{21} & b^{(1)}_{22}\end{pmatrix},\quad B^{(2)}=\begin{pmatrix} b^{(2)}_{11} & 0\\ b^{(2)}_{21} & b^{(2)}_{22}\end{pmatrix}.$$

Now by system equations (4) and according to the structure of matrices A, $B^{(1)}$ and $B^{(2)}$ of the above form and by MAPLE, we have the following system equations of ten equations with fifteen unknowns:

$$\begin{cases}
\alpha_1+\alpha_2=1,\\
\gamma^{(1)}_1+\gamma^{(1)}_2=0,\\
\gamma^{(2)}_1+\gamma^{(2)}_2=1,\\
\gamma^{(1)}_1 b^{(1)}_{11}+\gamma^{(1)}_2 b^{(1)}_{21}+\gamma^{(1)}_2 b^{(1)}_{22}=-\frac{1}{2},\\
\gamma^{(1)}_1 b^{(2)}_{11}+\gamma^{(1)}_2 b^{(2)}_{21}+\gamma^{(1)}_2 b^{(2)}_{22}+\gamma^{(2)}_1 b^{(1)}_{11}+\gamma^{(2)}_2 b^{(1)}_{21}+\gamma^{(2)}_2 b^{(1)}_{22}=0,\\
\gamma^{(2)}_1 b^{(2)}_{11}+\gamma^{(2)}_2 b^{(2)}_{21}+\gamma^{(2)}_2 b^{(2)}_{22}=\frac{1}{2},\\
\alpha_1 b^{(1)}_{11}+\alpha_2 b^{(1)}_{21}+\alpha_2 b^{(1)}_{22}=0,\\
\gamma^{(1)}_1 a_{11}+\gamma^{(1)}_2 a_{21}+\gamma^{(1)}_2 a_{22}=0,\\
\gamma^{(1)}_1 {b^{(1)}_{11}}^2+\gamma^{(1)}_2\left(b^{(1)}_{21}+b^{(1)}_{22}\right)^2+\gamma^{(1)}_1 {b^{(2)}_{11}}^2+\gamma^{(1)}_2\left(b^{(2)}_{21}+b^{(2)}_{22}\right)^2+2\gamma^{(2)}_1 b^{(1)}_{11} b^{(2)}_{11}\\
+2\gamma^{(2)}_2\left(b^{(2)}_{21}+b^{(2)}_{22}\right)\left(b^{(1)}_{21}+b^{(1)}_{22}\right)=0,\\
2\gamma^{(2)}_1 b^{(1)}_{11} b^{(2)}_{11}+\gamma^{(2)}_2\left(b^{(1)}_{21}b^{(2)}_{11}+b^{(1)}_{22}b^{(2)}_{21}+2b^{(1)}_{22}b^{(2)}_{22}+b^{(2)}_{21}b^{(1)}_{11}+b^{(2)}_{22}b^{(1)}_{21}\right)\\
+\gamma^{(1)}_1 {b^{(1)}_{11}}^2+\gamma^{(1)}_2\left(b^{(1)}_{21}b^{(1)}_{11}+b^{(1)}_{22}b^{(1)}_{21}\right)+\gamma^{(1)}_2 {b^{(1)}_{22}}^2+\gamma^{(1)}_1 {b^{(2)}_{11}}^2\\
+\gamma^{(1)}_2\left(b^{(2)}_{21}b^{(2)}_{11}+b^{(2)}_{22}b^{(2)}_{21}\right)+\gamma^{(1)}_2 {b^{(2)}_{22}}^2=0.
\end{cases}\tag{5}$$

Moreover, by system equations (4), since $\alpha^T B^{(1)}e=0$ and ${\gamma^{(1)}}^T Ae=0$, hence we can minimize the error constants corresponding to trees $[\tau_1]_0$ and $[\tau_0]_1$, that are given by

$$\begin{cases}
E[I_{10}-{z^{(0)}}^T Z^{(1)}e]^2=\left(\frac{1}{3}-\left(\alpha^T B^{(2)}e\right)+\left(\alpha^T B^{(2)}e\right)^2+\left(\alpha^T B^{(1)}e\right)^2\right)h^3\\
\qquad\qquad\qquad\qquad=\left(\frac{1}{3}-\left(\alpha^T B^{(2)}e\right)+\left(\alpha^T B^{(2)}e\right)^2\right)h^3,\\
E[I_{01}-{z^{(1)}}^T Z^{(0)}e]^2=\left(\frac{1}{3}-\left({\gamma^{(2)}}^T Ae\right)+\left({\gamma^{(2)}}^T Ae\right)^2+\left({\gamma^{(1)}}^T Ae\right)^2\right)h^3\\
\qquad\qquad\qquad\qquad=\left(\frac{1}{3}-\left({\gamma^{(2)}}^T Ae\right)+\left({\gamma^{(2)}}^T Ae\right)^2\right)h^3.
\end{cases}$$

These error constants are minimized with the minimum value $\frac{1}{12}$ if

$$\alpha^T B^{(2)}e=\frac{1}{2},\qquad {\gamma^{(2)}}^T Ae=\frac{1}{2},$$

or equivalently, if

$$\begin{cases}
\alpha_1 b^{(2)}_{11}+\alpha_2 b^{(2)}_{21}+\alpha_2 b^{(2)}_{22}=\frac{1}{2},\\
\gamma^{(2)}_1 a_{11}+\gamma^{(2)}_2 a_{21}+\gamma^{(2)}_2 a_{22}=\frac{1}{2}.
\end{cases}\tag{6}$$

By augmenting equations (6) to system (5) and solving the new system by MAPLE it is observed that the new system has a three parameters solution that are given by

$$\begin{cases} a_{11} = \frac{1}{2}, \quad a_{21} = \frac{1}{2} - a_{22}, \quad \alpha_1 = \alpha_2 = \frac{1}{2}, \\ b_{11}^{(1)} = \frac{1}{4\gamma_2^{(1)}}, \quad b_{21}^{(1)} = \frac{2b_{22}^{(2)}-1}{4\gamma_2^{(1)}}, \quad b_{22}^{(1)} = -\frac{b_{22}^{(2)}}{2\gamma_2^{(1)}}, \quad \gamma_1^{(1)} = -\gamma_2^{(1)}, \\ b_{11}^{(2)} = \frac{1}{2}, \quad b_{21}^{(2)} = \frac{1}{2} - b_{22}^{(2)}, \quad \gamma_1^{(2)} = \gamma_2^{(2)} = \frac{1}{2}, \quad \gamma_2^{(1)} \neq 0. \end{cases} \tag{7}$$

In order to determine the free parameter of the deterministic part, i.e. a_{22}, we choose the deterministic part of SRK method (2) to be the Runge –Kutta method given by

$$\begin{array}{c|cc} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

that is, it has order 2 (see [4]). This ensures that the semi-implicit method works well in the case of small stochastic influence. From (7) we can assume $A = B^{(2)}$, and consequently for $\gamma_2^{(1)} \neq 0$, a one-parameter solution can be represented by the following tableau

$$\begin{array}{c|cc|cc} & \frac{1}{2}h & 0 & \frac{\sqrt{h}}{4\gamma_2^{(1)}} + \frac{1}{2}J_1 & 0 \\ & \frac{1}{4}h & \frac{1}{4}h & -\frac{\sqrt{h}}{8\gamma_2^{(1)}} + \frac{1}{4}J_1 & -\frac{\sqrt{h}}{8\gamma_2^{(1)}} + \frac{1}{4}J_1 \\ \hline & \frac{1}{2}h & \frac{1}{2}h & -\gamma_2^{(1)}\sqrt{h} + \frac{1}{2}J_1 & \gamma_2^{(1)}\sqrt{h} + \frac{1}{2}J_1 \end{array}$$

In order to choose $\gamma_2^{(1)}$, one can use the minimum of the error constants corresponding to trees $[[\tau_1]_1]_1$ and $[\tau_1, \tau_1]_1$, that are given by

$$\begin{cases} E[I_{111} - {z^{(1)}}^T {Z^{(1)}}^2 e]^2 = \left(\frac{3+96{\gamma_2^{(1)}}^2+560{\gamma_2^{(1)}}^4}{3072{\gamma_2^{(1)}}^4} \right) h^3, \\ E[I_{111} + \frac{1}{2}I_{01} - \frac{1}{2}{z^{(1)}}^T \left(Z^{(1)}e\right)^2 - \frac{1}{2}{z^{(1)}}^T Z^{(0)} e]^2 = \left(\frac{1+8{\gamma_2^{(1)}}^2+48{\gamma_2^{(1)}}^4}{1024{\gamma_2^{(1)}}^4} \right) h^3. \end{cases}$$

By introducing two functions f and g in the following form

$$f(\lambda) = \frac{3 + 96\lambda^2 + 560\lambda^4}{3072\lambda^4}, \qquad g(\lambda) = \frac{1 + 8\lambda^2 + 48\lambda^4}{1024\lambda^4}$$

it can be shown that these are decreasing functions on the interval $(0, +\infty)$, and moreover

$$\lim_{\lambda \to +\infty} f(\lambda) = \frac{35}{192}, \qquad \lim_{\lambda \to +\infty} g(\lambda) = \frac{3}{64}.$$

Now by choosing $\gamma_2^{(1)} = 3$, this class of methods can be represented by the following tableau

$$\begin{array}{c|cc|cc} & \frac{1}{2}h & 0 & \frac{\sqrt{h}}{12}+\frac{1}{2}J_1 & 0 \\ & \frac{1}{4}h & \frac{1}{4}h & -\frac{\sqrt{h}}{24}+\frac{1}{4}J_1 & -\frac{\sqrt{h}}{24}+\frac{1}{4}J_1 \\ \hline & \frac{1}{2}h & \frac{1}{2}h & -3\sqrt{h}+\frac{1}{2}J_1 & 3\sqrt{h}+\frac{1}{2}J_1 \end{array}$$

which is named '$SIM2$', and has principal error constants

$$\frac{1}{12}h^3, \ \frac{1}{12}h^3, \ \frac{15409}{82944}h^3, \ \frac{3961}{82944}h^3.$$

Note that the principal error coefficients corresponding to trees $[[\tau_1]_1]_1$ and $[\tau_1, \tau_1]_1$, are very close to the limits of f and g as $\lambda \to +\infty$, respectively. Since f and g are even functions, the above analysis shows that the choice $\gamma_2^{(1)} = 3$ is suitable. If we use the 1-norm to estimate the contribution of all error terms to the principal error term, then, Table1 represents the following values for methods 'IRK' , '$EM1$' , '$EM2$' and '$SIM2$'.

Table 1: 1-norm of principal error coefficients

	IRK	$EM1$	$EM2$	$SIM2$
$\Vert principal \ \ error\Vert_1$	1.2083	0.89583	0.89583	0.40019

From Table1, it follows that the 1-norm principal error of the method '$SIM2$' is less than the 1-norm principal error of '$EM1$' and '$EM2$' methods. In order to improve the results of employing the '$SIM2$' method at each step, we can solve the system for stage-variables Y_1 and Y_2 by the fixed-point iteration scheme with starting values for these variables coming from the '$EM1$' or '$EM2$' methods. In fact, for the stage-variable Y_1 in the '$SIM2$' method let

$$G_1(Y_1) \equiv y_n + \frac{1}{2}h \ g_0(Y_1) + \frac{1}{12}(\sqrt{h} + 6J_1) \ g_1(Y_1),$$

and hence the fixed-point iteration for solving Y_1 is given by

$$Y_1{}^{[s+1]} = G_1(Y_1{}^{[s]}), \qquad s = 0, 1, 2, \ldots, \tag{8}$$

with stopping criteria

$$|Y_1^{[s+1]} - Y_1^{[s]}| < \epsilon, \tag{9}$$

where ϵ is a positive known tolerance value. In order to consider the convergence property of fixed point iterations (8), it is sufficient to have

$$\mid G_1'(Y) \mid = \mid \frac{1}{2}h \ g_0'(Y) + \frac{1}{12}(\sqrt{h} + 6J_1) \ g_1'(Y) \mid < 1.$$

Also for the stage-variable Y_2, let

$$G_2(Y_2) \equiv y_n + \frac{1}{4}h\ (g_0({Y_1}^{[s+1]}) + g_0(Y_2)) + \frac{1}{24}(-\sqrt{h} + 6J_1)\ (g_1({Y_1}^{[s+1]}) + g_1(Y_2)),$$

such that $Y_1^{[s+1]}$ satisfy condition (9). Consequently the fixed-point iteration for solving Y_2 is given by

$${Y_2}^{[t+1]} = G_2({Y_2}^{[t]}), \qquad t = 0, 1, 2, \ldots, \tag{10}$$

with stopping criteria

$$|Y_2^{[t+1]} - Y_2^{[t]}| < \epsilon. \tag{11}$$

Note that iterations (10) is convergent if

$$\mid G_2'(Y) \mid = \mid \frac{1}{4}h\ g_0'(Y) + \frac{1}{24}(-\sqrt{h} + 6J_1)g_1'(Y) \mid < 1.$$

Finally y_{n+1} for the '$SIM2$' method will be evaluated by

$$y_{n+1} = y_n + \frac{1}{2}h\left(g_0\left(Y_1^{[s+1]}\right) + g_0\left(Y_2^{[t+1]}\right)\right) + \left(-3\sqrt{h} + \frac{1}{2}J_1\right)g_1\left(Y_1^{[s+1]}\right) + \left(3\sqrt{h} + \frac{1}{2}J_1\right)g_1\left(Y_2^{[t+1]}\right),$$

where $Y_1^{[s+1]}$ and $Y_2^{[t+1]}$ satisfy conditions (9) and (11).

3 Numerical results and conclusion

In this section, the numerical results from the implementation of the above seven methods are compared. These methods are 'IRK', '$Milstein$', '$EM1$', '$EM2$', '$SIM1$', 'IM' and '$SIM2$'. They will be implemented with constant stepsize on two problems taken from [5], for which the exact solution in terms of a Wiener process is known. Since $J_1 \sim N(0, h)$, hence for generating the Wiener increments J_1 in MATLAB environment of random numbers generator `randn` $(\#traj, \#step)$ is used, such that each call to `randn` $(\#traj, \#step)$ creates a $\#traj \times \#step$ matrix of independent $N(0, 1)$ samples. When these methods are simulated, the same sequence of random numbers for the Wiener increment J_1 are used for the stepsize under consideration. The average error for each stepsize at the end of the interval of integration is defined by

$$AE = \frac{1}{K}\sum_{i=1}^{K} \mid y_N^{(i)} - y^{(i)}(t_N) \mid,$$

where $y_N^{(i)}$ is the numerical approximation and $y^{(i)}(t_N)$ is the exact solution of SDE at t_N in the i-th simulation over all K simulations. All of the numerical results are based on 1000 simulated trajectories. The results appear in Tables 2-4.

Test problem 1. Consider

$$dy(t) = -a^2 y(t)(1 - y^2(t))dt + a(1 - y^2(t))dW(t), \quad y(0) = 0, \quad t \in [0, 1]$$

with the exact solution

$$y(t) = \tan h(aW(t) + \arctan h(y_0)).$$

This problem is solved numerically with the choice of parameter $a = 1$.

Table 2: Global errors for Test problem 1, with $a = 1$, $K = 1000$ and $\epsilon = 0.001$

h	$\frac{1}{25}$	$\frac{1}{50}$	$\frac{1}{100}$	$\frac{1}{200}$	$\frac{1}{400}$
IRK	0.21400e-1	0.10299e-1	0.51948e-2	0.24299e-2	0.12254e-2
$Milstein$	0.16276e-1	0.82454e-2	0.42156e-2	0.19930e-2	0.10127e-2
$EM1$	0.12121e-1	0.59344e-2	0.30475e-2	0.14587e-2	0.70585e-3
$EM2$	0.12043e-1	0.57056e-2	0.29270e-2	0.13901e-2	0.71060e-3
$SIM1$	0.55857e-2	0.21190e-2	0.96207e-3	0.45136e-3	0.22157e-3
IM	0.13035e-3	0.64121e-4	0.34962e-4	0.17710e-4	0.81462e-5
$SIM2$	0.80715e-4	0.44013e-4	0.21736e-4	0.10551e-4	0.51995e-5

Test problem 2. Consider

$$dy(t) = -(\alpha+\beta^2 y(t))(1-y^2(t))dt+\beta(1-y^2(t))dW(t), \quad y(0) = 0.5, \quad t \in [0, 1]$$

with the exact solution

$$y(t) = \frac{(1 + y_0)\exp(-2\alpha t + 2\beta W(t)) + y_0 - 1}{(1 + y_0)\exp(-2\alpha t + 2\beta W(t)) + 1 - y_0}.$$

This problem is solved numerically with $\alpha = -1$ and for $\beta = 1$ and 0.01. Comparing the numerical results in Tables 2-4, it follows that the '$SIM2$' method is more accurate than the '$EM1$', '$EM2$', '$SIM1$' and 'IM' methods. Also for problems in which the deterministic term dominates (Test problem 2 with $\beta = 0.01$) the improvement of the '$SIM2$' method becomes noticeable as the stepsize is reduced. This is because the deterministic component of the '$SIM2$' method is the second order Runge-Kutta method. On the other hand, for problems in which deterministic term dominates (Test problem 2 with $\beta = 0.01$) the global errors for two-stage explicit methods are the same. This is because these methods the deterministic components are the same. The future work should be based on the construction of implicit IRKs for SDEs with two or more Wiener processes.

Table 3: Global errors for test problem 2, with $\alpha = -1,\ \beta = 1,\ K = 1000$ and $\epsilon = 0.001$.

h	$\frac{1}{25}$	$\frac{1}{50}$	$\frac{1}{100}$	$\frac{1}{200}$	$\frac{1}{400}$
IRK	0.12763e-1	0.58682e-2	0.29961e-2	0.15034e-2	0.74495e-3
$Milstein$	0.11513e-1	0.51633e-2	0.27770e-2	0.13806e-2	0.68995e-3
$EM1$	0.96413e-2	0.41781e-2	0.21225e-2	0.10660e-2	0.54324e-3
$EM2$	0.93988e-2	0.42298e-2	0.20985e-2	0.10210e-2	0.52317e-3
$SIM1$	0.65238-3	0.32108e-3	0.15186-3	0.65537e-4	0.30367e-4
IM	0.79517e-4	0.42130e-4	0.21167e-4	0.10561e-4	0.51995e-5
$SIM2$	0.57845e-4	0.30499e-4	0.15806e-4	0.79504e-5	0.37761e-5

Table 4: Global errors for test problem 2, with $\alpha = -1,\ \beta = 0.01,\ K = 1000$ and $\epsilon = 0.001$

h	$\frac{1}{25}$	$\frac{1}{50}$	$\frac{1}{100}$	$\frac{1}{200}$	$\frac{1}{400}$
IRK	0.50778e-2	0.25193e-2	0.12544e-2	0.62592e-3	0.31264e-3
$Milstein$	0.50778e-2	0.25193e-2	0.12544e-2	0.62592e-3	0.31264e-3
$EM1$	0.50778e-2	0.25193e-2	0.12544e-2	0.62592e-3	0.31264e-3
$EM2$	0.50778e-2	0.25193e-2	0.12544e-2	0.62592e-3	0.31264e-3
$SIM1$	0.70238e-5	0.34182e-5	0.17423e-5	0.52895e-6	0.25624e-6
IM	0.55102e-5	0.26426e-5	0.13193e-5	0.58242e-6	0.29121e-6
$SIM2$	0.62103e-6	0.15741e-6	0.40441e-7	0.10595e-7	0.28838e-8

References

1. Barid Loghmani, G and Mohseni, M. *On the implicit and semi-implicit Runge-Kutta methods for stochastic ordinary differential equations*, Italian Journal of Pure and Applied Mathematics, 13 (2003), 51–64.

2. Burrage, P.M. *Runge-Kutta methods for stochastic differential equations*, Ph.D. Thesis, Department of Mathematics, University of Queensland, Australia, 1992.

3. Burrage, K. and Burrage, P.M. *Order conditions of stochastic Runge-Kutta methods by B-series*, SIAM Journal on Numerical Analysis 38 (2000), 1626–1646.

4. Butcher, J.C. *Numerical methods for ordinary differential equations,* John Wiley, England, 2003.

5. Kloeden, P.E. and Platen, E. *Numerical solution of stochastic differential equations,* Springer, Berlin, 1995.

6. Kloeden, P.E., Platen, E. and Schurz, H. *Numerical sloution of stochastic differential equations through computer experiments*, Springer, Berlin, 1994.

7. Oksendal, B. *Stochastic differential equations: An introduction with applications*, Springer, Berlin, 1998.

8. Soheili, A.R. and Namjoo, M. *Strong approximation of stochastic differential equations with Runge-Kutta methods*, World Journal of Modeling and Simulation 4 (2008), 83–93.

9. Soheili, A.R. and Namjoo, M. *Strong Runge-Kutta methods with order one for numerical solution of Ito stochastic differential equations*, Applied Mathematics Research eXpress. (2007), Article ID abm003, 17 pages.

10. Soong, T.T. *Random differential equations in science and engineering*, Academic Press, New York, 1973.

Iranian Journal of Numerical Analysis and Optimization
Vol 5, No. 1, (2015), pp 13-36

A two-phase variable neighborhood search for solving nonlinear optimal control problems

R. Ghanbari*, A. Heydari and S. Nezhadhosein

Abstract

In this paper, a two-phase algorithm, namely IVNS, is proposed for solving nonlinear optimal control problems. In each phase of the algorithm, we use a variable neighborhood search (VNS), which performs a uniform distribution in the shaking step and the successive quadratic programming, as the local search step. In the first phase, VNS starts with a completely random initial solution of control input values. To increase the accuracy of the solution obtained from the phase 1, some new time nodes are added and the values of the new control inputs are estimated by spline interpolation. Next, in the second phase, VNS restarts by the solution constructed by the phase 1. The proposed algorithm is implemented on more than 20 well-known benchmarks and real world problems, then the results are compared with some recently proposed algorithms. The numerical results show that IVNS can find the best solution on 84% of test problems. Also, to compare the IVNS with a common VNS (when the number of time nodes is same in both phases), a computational study is done. This study shows that IVNS needs less computational time with respect to common VNS, when the quality of solutions are not different significantly.

Keywords: Nonlinear optimal control problem; Variable neighborhood search; Successive quadratic programming.

*Corresponding author

Received 19 April 2014; revised 29 December 2014; accepted 6 January 2014

R. Ghanbari
Department of Applied Mathematics, Faculty of Mathematical Science, Ferdowsi University of Mashhad, Mashhad, Iran. e-mail: rghanbari@um.ac.ir

A. Heydari
Department of Applied Mathematics, Payame Noor University, Mashhad, Iran. email: a_heidari@pnu.ac.ir

S. Nezhadhosein
Department of Applied Mathematics, Payame Noor University, Tehran, Iran. email: s_nejhadhosein@pnu.ac.ir

1 Introduction

Nonlinear optimal control problems (NOCP) are dynamic optimization problems with many applications in process systems engineering, including the design of trajectories for the optimal operation of batch and semi-batch reactors, economic systems, plasma physics, etc. [7].

Providing high-quality solutions with minimum computational time is the main issue for solving NOCPs. The numerical methods, direct [29] or indirect [46], usually have two main deficiencies, including low accuracy and convergence to a poor local solution. In direct methods, the quality of solutions depend on discretization resolution. These methods use control parametrization to convert continuous problems to discrete problems, so they may have less accuracy. However, the adaptive strategies [8, 43] can overcome these defects, but they may be trapped by a local optimal, yet. In the indirect approach, the problem using Pontryagins minimum principle (PMP) is converted to two boundary value problems (TBVP) and then it can be solved by numerical methods such as shooting method [29]. These methods need the good initial guesses that lie within the domain of convergence. Therefore, numerical methods are not usually suitable for solving NOCPs, especially for large-scale and multimodal models.

Metaheuristics as the global optimization methods can overcome these problems, but they usually need more computational time, though they don't really need good initial guesses and deterministic rules. Several researchers have used metaheuristics to solve optimal control problems. For instance, Michalewicz et al. [34] applied floating-point Genetic algorithms (GA) to solve discrete time optimal control problems, Yamashita and Shima [52] used the classical GAs to solve the free final time optimal control problems with terminal constraints. Abo-Hammour et al. [1] used continuous GA for solving NOCPs. Recently, Sun et al. [47] proposed a hybrid improved GA, for solving NOCPs and applied it for chemical processes. Moreover, the other usages of GA for optimal control problems can be found in [44, 45]. Modares and Naghibi-Sistani [37], proposed a hybrid algorithm by integrating an improved Particle Swarm Optimization (PSO) with a successive quadratic programming (SQP), for solving NOCPs. Lopez-Cruz et al. [14], applied Differential Evolution (DE) algorithms for solving the multimodal optimal control problems. Recently, Ghosh et al. [22] developed an ecologically inspired optimization technique, called Invasive Weed Optimization (IWO), for solving optimal control problems. The other well-known metaheuristic algorithms which are used for solving NOCPs are Genetic Programming (GP) [30], PSO [3, 4], Ant Colony Optimization (ACO) [48] and DE [31, 50].

Based on the success of the metaheuristics for solving NOCPs mentioned above, we propose an algorithm that use a well-known metaheuristic namely VNS (variable neighbourhood search) to solve NOCPs. Also, achieving a global optimal solution for NOCPs is another motivation for us to use a VNS [35]. VNS is an intelligent and metaheuristic method for solving a set of combinatorial optimization and global optimization problems which uses neighborhood changes and uniform distributions in search procedure. Unlike many other metaheuristics, it is simple and requires few parameters [32]. Mladenović et al. [36] proposed a general VNS for solving continuous optimization. Moreover, VNS was used for solving several optimization problem [25] such as mixed integer programming [26], vertex weighted $k-$cardinality tree problem [10], and scheduling problem [13].

In this paper, VNS uses a uniform distribution in the shaking step and the SQP [39], as the local search step (similar to [37]). SQP is an iterative algorithm for solving NLP, which uses gradient information. Furthermore, SQP is used for solving NOCPs alone [6, 18].

For performing VNS to solve an NOCP, the time interval is uniformly divided by using a constant number of time nodes. Next, in each of these time nodes, the control variable is approximated by a scalar matrix of control input values. Thus, an infinite dimensional NOCP is changed to a finite dimensional nonlinear programming (NLP). Now, we encounter two conflict situations: the quality of the global solution and the needed computational time. In other words, when the number of time nodes is increased then we expect the quality of the global solution to increase but we know that in this situation the computational time is increased dramatically. In other situation, we consider less number of time nodes to reduce the computational but we may find a poor local solution. To conquer these problems, IVNS, performs VNS in two phases. In the first phase of IVNS (exploration phase), to decrease the computational time and to find a promising solution in the search space, VNS uses a less number of time nodes. Next to increase the quality of the solution obtained from Phase 1, the number of time nodes is increased. Using the obtained solution in Phase 1, the values of the new control inputs are estimated by spline interpolation. Next, in the second phase of IVNS (exploitation phase), VNS uses the solution constructed by the above procedure, as an initial solution. A computational study in our numerical experiments shows that there is a significant difference between the computational time of IVNS and a common VNS, that uses all time nodes from the beginning.

The rest of the paper is organized as follow: in Section 2, NOCPs are briefly introduced. In Section 3, IVNS is described. In Section 4, we provide more than 20 NOCPs to examine the numerical behaviour of the proposed algorithm. Results are compared with some numerical and metaheuristic methods. A computational study is carried out in Section 5 to show the effect of the second phase. We conclude in Section 6.

2 Problem formulation

NOCPs are formulated as optimization problems by the performance index as the objective function and differentiate equations as constraints that called dynamic optimizations. There are several types of these problems e.g. tracking problem, terminal control problem and time minimization problem [29]. We consider nonlinear bounded continuous-time control problems in which a vector of control functions, u, is exerted over the planning horizon $[t_0, t_f]$. The particular problem considered is that of finding the control input vector $u(t) \in \mathbb{R}^m$ that minimizes the performance index:

$$min \quad J = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} g(x(t), u(t), t)dt \tag{1}$$

subject to:

$$\begin{align} \dot{x}(t) &= f(x(t), u(t), t), \tag{2}\\ c(x(t), u(t), t) &= 0, \tag{3}\\ d(x(t), u(t), t) &\leq 0, \tag{4}\\ \psi(x(t_f), t_f) &= 0, \tag{5}\\ x(t_0) &= x_0, \qquad t \in [t_0, t_f]. \tag{6} \end{align}$$

where $x(t) \in \mathbb{R}^n$ denotes the state vector for the system and $x_0 \in \mathbb{R}^n$ is the initial state. The functions $f : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}^n$, $g : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}$, $c : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}^{n_c}$, $d : \mathbb{R}^{n+m} \times \mathbb{R} \to \mathbb{R}^{n_d}$, $\psi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^{n_\psi}$ and $\phi : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ are assumed to be sufficiently smooth on appropriate open sets. The cost function (1) must be minimized subject to dynamic (2), control and state equality constraints (3), control and state inequality constraints (4), the initial condition (6) and the final state constraints (5).

3 Proposed algorithm

Here, we propose IVNS for solving NOCPs. Before providing a description of IVNS, we introduce VNS.

3.1 VNS algorithm

VNS where introduced by Mladenović and Hansen in 1997 [35] is a metaheuristic algorithm which uses neighborhood changes systemically idea, both in the descent to local minima and in the escape from valleys which contain

local minima. It explores distant neighborhoods of the current incumbent solution, and moves from there to a new one if and only if an improvement is necessary. Local search method is applied repeatedly to get in the neighborhood to local optima [36]. Here, the implemented VNS in each phase has the following steps:
Initialization: The time interval is divided into $N_t - 1$ subintervals using time nodes $t_0, \dots, t_{N_t-1}$ and then control input values are computed (or selected randomly) as control points. This can be done by the following stages:

1. Let $t_k = t_0 + kh$, where $h = \frac{t_f - t_0}{N_t - 1}, k = 0, 1, \dots, N_t - 1$, be time nodes, where t_0 and t_f are the initial and final times, respectively.

2. The corresponding control input value at each time node, $t_k, k = 0, \dots, N_t - 1$ is an $m \times 1$ vector, $u_k = [u_1^{(k)}, \dots, u_m^{(k)}]^T$, having the following components:

$$u_i^{(k)} = u_{left,i} + (u_{right,i} - u_{left,i}).r_i, \quad i = 1, 2, \dots, m \tag{7}$$

   where r_i is a random number in $[0, 1]$ with uniform distribution and $u_{left},\ u_{right} \in \mathbb{R}^m$ are the lower and the upper bound vectors of control input values, which can be given by the problem's definition or the user (e.g. see the NOCPs No. 7 and 8 in Appendix, respectively). $u = [u_k]_{k=0}^{N_t-1}$ is called control input matrix.

Evaluation: The corresponding state matrix with the control input matrix, u, is an $n \times N_t$ matrix, $x = [x_k]_{k=0}^{N_t-1}$, where x_k, $k = 0, 1, \dots, N_t - 1$, is an $n \times 1$ vector as the $(k+1)$-th column of state matrix, and can approximately be computed by the forth Runge-Kutta method on dynamic system (2) with the initial condition (6). Without loss of generality, assume $m = 1$ (for general case it can be extended easily). So, the evaluation procedure is as follows:

$$x_k = x_{k-1} + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4), \quad k = 1, 2, \dots, N_t - 1 \tag{8}$$

where

$$l_1 = hf(x_k, u_k, t_k), \qquad l_2 = hf(x_k + \frac{l_1}{2}, u_k + \frac{h}{2}, t_k)$$
$$l_3 = hf(x_k + \frac{l_2}{2}, u_k + \frac{h}{2}, t_k), \qquad l_4 = hf(x_k + l_3, u_k + h, t_k)$$

where $u_k = u(t_k)$ and $x_k = x(t_k)$, with initial condition $x(t_0) = x_0$. To approximate the performance index, the composite Simpson's method [5], is used. Then, the performance index in (1), J, is approximated by $\tilde{J}$ as follows:

$$J \simeq \tilde{J} = \phi(x_{N_t-1}, t_{N_t-1}) + \frac{h}{3}(f_0 + 4\sum_{i=1}^{[\frac{N_t}{2}]-1} f_{2i+1} + 2\sum_{i=0}^{[\frac{N_t}{2}]-1} f_{2i} + f_{N_t-1}) \tag{9}$$

where $f_k = f(x_k, u_k, t_k)$, $k = 0, 1, \dots N_t - 1$. If NOCP includes equality or inequality constraints e.g. (3) or (4), or has final state constraints, given by (5), then we add some penalties to the corresponding fitness value of the solution. Finally, we assign $I(u)$ to u as the fitness value as follows:

$$I(u) = \tilde{J} + \sum_{l=1}^{n_d} \sum_{j=0}^{N_t-1} M_{1l} max\{0, d_l(x_j, u_j, t_j)\} + \sum_{h=1}^{n_c} \sum_{j=0}^{N_t-1} M_{2h} c_h^2(x_j, u_j, t_j) + \sum_{p=1}^{n_\psi} M_{3p} \psi_p^2(x_{N_t-1}, t_{N_t-1}) \tag{10}$$

where $M_1 = [M_{11}, \dots, M_{1n_d}]^T$, $M_2 = [M_{21}, \dots, M_{2n_c}]^T$ and $M_3 = [M_{31}, \dots, M_{3n_\psi}]^T$ are big numbers, as the penalty coefficients, for $c_h(.,.,.)$, $h = 1, 2, \dots, n_c$, $d_l(.,.,.)$, $l = 1, 2, \dots, n_d$, and $\psi_p(.,.)$, $p = 1, 2, \dots, n_\psi$ defined in (3), (4) and (5), respectively.
The fitness value in (10), can be viewed as a nonlinear objective function with the decision variable as $u = [u_0, u_1, \dots, u_{N_t-1}]$. This cost function with upper and lower bounds of input signals construct a finite dimensional NLP problem as follows:

$$\begin{aligned} &\min \quad I(u) = I(u_0, u_1, \dots, u_{N_t-1}) \\ &s.t \\ &u_{left} \leq u_j \leq u_{right}, \quad j = 0, 1, \dots, N_t - 1 \end{aligned} \tag{11}$$

Neighborhood: VNS uses at most k_{max} neighborhoods, $V_{r_1}, \dots, V_{r_{k_{max}}}$, in which $r_i, i = 1, \dots, k_{max}$ is the radii of i-th neighborhood, V_i, of the control input matrix u.
Shaking: In this stage, using a uniform distribution, a random direction matrix $d \in [-1, 1]^{m \times N_t}$ is firstly generated and then a random solution, $\bar{u}$, is selected in the k-th neighborhood, V_k, by the following equation:

$$\bar{u} = u + d.\alpha.(r + k - 1) \tag{12}$$

where $r \in [0, 1]$ is a random number, k is the index of neighborhood and α is the parameter of radii.
Local search: In this stage, SQP algorithm [9, 39] is performed on the NLP (11), using $\bar{u}^0 = \bar{u}$, constructed in (12), as the initial solution when the maximum number of iteration is *sqpmaxiter*.

SQP, is an effective and iterative algorithm for the numerical solution of the constrained NLP problem. This technique is based on finding a solution to the system of nonlinear equations that arise from the first-order necessary conditions for an extremum of the NLP problem. Using an initial solution of NLP, $\bar{u}^k$, $k = 0, 1, \dots$, a sequence of solutions as $\bar{u}^{k+1} = \bar{u}^k + d^k$ is constructed, which d^k is the optimal solution of the constructed quadratic programming (QP) that approximates NLP in the iteration k based on $\bar{u}^k$, as the search direction in the line search procedure. For the NLP (11), the principal idea is the formulation of a QP subproblem based on a quadratic approximation of the Lagrangian function as $L(u, \lambda) = I(u) + \lambda^T h(u)$, where the vector λ is Lagrangian multiplier and $h(u)$ return the vector of, inequality constraints evaluated at u. The QP is obtained by linearizing the nonlinear functions as follows:

$$\begin{aligned}&\min \frac{1}{2} d^T H(\bar{u}^k) d + \nabla I(\bar{u}^k)^T d \\ &\nabla h(\bar{u}^k)^T d + h(\bar{u}^k) \leq 0\end{aligned}$$

Similar to [18], here a finite difference approximation is applied to compute the gradient of the cost function and the constraints, with the following components

$$\frac{\partial I}{\partial u_j} = \frac{I(...u_j + \delta...) - I(u_j)}{\delta}, \quad j = 0, 1, \ldots, N_t - 1 \tag{13}$$

where δ is the double precision of machine. So, the gradient vector is $\nabla I = [\frac{\partial I}{\partial u_0}, \ldots, \frac{\partial I}{\partial u_{N_t - 1}}]^T$. Also, at each major iteration a positive definite quasi-Newton approximation of the Hessian of the Lagrangian function, H, is calculated using the BFGS method [39], where $\lambda_i, \ i = 1, ..., m$, is an estimated of the Lagrange multipliers. The general procedure of SQP, for NLP (11), is as follows:

1. Given an initial solution $\bar{u}^0$. Let $k = 0$.

2. Construct the QP subproblem (13), based on $\bar{u}^0$, using the approximations of the gradient and the Hessian of the the Lagrangian function.

3. Compute the new point as $\bar{u}^{k+1} = \bar{u}^k + d^k$, where d^k is the optimal solution of the current QP.

4. Let k=k+1 and go to step 2.

Here, in IVNS, SQP is used as the local search step, and we use the maximum number of iterations as the main criterion for stopping SQP. In other words, we terminate SQP when it converges either to local solution or the maximum number of SQP's iterations is reached.

Terminal conditions: The algorithm is terminated when the number of neighborhoods reached to k_{max} or the difference between cost functions in two consecutive iterations is less than ε (a given number).

VNS algorithm is given in Algorithm 1.

Algorithm 1 VNS algorithm

```
{Initialization} Input the number of time nodes $N_t$, the maximum number of iteration for SQP, $sqpmaxiter$, a maximum number of neighborhood, $k_{max}$, the parameter of radii, $\alpha$ defined in (12), the lower and the upper bound vectors of control input values $u_{left}$, $u_{right}$, an initial solution, $u^*$, and $\varepsilon$. Let $k = 1$.
{Evaluation} Evaluate the fitness of the initial solution, $u^*$ and let $I^* = I(u^*)$, where $I(.)$ is defined in (10).
repeat
    {Shaking} Using (12), select $u$ in $k$-th neighborhood of $u^*$.
    {Local search} Perform SQP algorithm on the NLP (11), using $u$ as the initial solution when the maximum number of iteration is $sqpmaxiter$. Let $\bar{u}$ be the obtained solution, $\bar{I} = I(\bar{u})$ and $e = |\bar{I} - I^*|$.
    if $\bar{I} < I^*$ then
        Let $u^* = \bar{u}$, $I^* = \bar{I}$ and $k = 1$.
    else
        Let $k = k + 1$
    end if
until $k > k_{max}$ or $e < \varepsilon$
Return $u^*$ as the approximate solution, $x^*$ as the corresponding state and the corresponding fitness $I^*$.
```

3.2 IVNS

We now give a new algorithm, IVNS, which is a two-phase direct metaheuristic approach. The main idea of IVNS is to find promising solution of the search space using the computational time as few as possible.

IVNS has two main phases (as discussed in Section 1). In the first phase, we perform VNS (Algorithm 1) with a completely random initial solution constructed by (7). Since the main goal in this phase is to find the promising solution in the search space, we use a few number of time nodes.

Next, to maintain the property of the solution given in Phase 1 and to increase the accurately of this solution, we add some additional time nodes. Thus, we increase time nodes from N_{t_1} in the Phase 1 to N_{t_2} in the Phase 2. To use the information of the obtained solution from Phase 1 in the construction of the initial solution for Phase 2, we use Spline interpolation to estimate the values of the control inputs based on the curve obtained from the Phase 1. In the second phase, VNS restarts with this solution. Finally, IVNS is given in Algorithm 2.

Remark 3.1. *As we know, there are no general theorems on convergence of metaheuristics algorithm exist [28, 38]. Also, a specific theory on convergence of VNS does not exist, but a simple framework for global convergence of VNS based on attraction probabilities concept, can be found in [11]. However, we*

Algorithm 2 IVNS

```
Initialization Input $u_{left}$ and $u_{right}$.
{Phase 1} Perform VNS (Algorithm 1) with a random initial solution and using the parameters $N_{t_1}$, $sqpmaxiter$, $k_{max}, \alpha$ and $\varepsilon$. (see Algorithm 1)
{Constructing an initial solution for Phase 2} Increase time nodes uniformly to $N_{t_2}$ and estimate the corresponding control input values by using Spline interpolation on the obtained solution from Phase 1.
{Phase 2} Restart VNS (Algorithm 1) with the constructed initial solution and using $N_{t_2}$, $sqpmaxiter$, $k_{max}$, $\alpha$ and $\varepsilon$. (see Algorithm 1)
```

mentioned that all metaheuristics are practical algorithms that are interesting for their numerical behaviour, [16].

4 Numerical experiments

In this Section, to investigate the efficiency of IVNS, more than 20 well-known and real world NOCPs, as benchmark problems, are considered. These problems are selected with single control signal and multi control signals.

The numerical behaviour of the algorithms can be studied from two points of view: the performance index and the final state constraints. Let J be the value of the performance index and $\psi = [\psi_1, \dots, \psi_{n_\psi}]^T$, defined in (5), and $\phi_f = \|\psi\|_2$ be the vector of final state constraints and the error of ψ, respectively. Now, the absolute errors for J and ϕ_f, are defined as follows:

$$E_J = |J - J^*|, \quad E_\psi = |\phi_f - \phi_f^*| \tag{14}$$

where J^* and $\phi_f^* = \|\psi^*\|_2$ are the best obtained solutions among the methods, or the exact solutions (when exist). To control the accuracy study, we now define a new criterion, called factor, to compare the numerical behaviour of the algorithms as follows:

$$K_\psi = E_J + E_\psi \tag{15}$$

where E_J and E_ψ are defined in (14). Note that K_ψ shows the summation of two important errors. Thus, based on K_ψ we can study the behaviour of algorithms on the quality and feasibility of given solutions, simultaneously.

To solve any NOCP described in the Appendix, we must know IVNS's parameters including N_{t_1}, N_{t_2}, k_{max}, α, ε and *sqpmaxiter* (see Algorithm 1), and the problem's parameters including u_{left}, u_{right} and M_i, $i = 1, 2, 3$, in (10). To estimate the best value of these parameters, for each problem, we run the proposed algorithm with different values of the parameters and then select the best. In all NOCPs, we consider the parameters $sqpmaxiter = 30$, $\alpha = 10^{-3}$ and $k_{max} = 10$. The other parameters are given in the associated subsection or in Table 2. Because of the stochastic nature of the proposed algorithm, 12 different runs were done, for each NOCP, and the best result are reported in Table 1. The best value of each column is highlighted in the bold. The reported numerical results for each algorithm included the value of performance index, J, the absolute error of J and E_J, are defined in (14). The final state constraints, $\psi = [\psi_1, \dots, \psi_{n_\psi}]^T$, the two-norm or error of the final state constraints, ϕ_f, the absolute error of ϕ_f and E_ψ, are defined in (14), and the factor K_ψ is defined in (15).

The algorithm was implemented in Matlab R2011a environment on a Notebook with Windows 7 Ultimate, CPU 2.53 GHz and 4.00 GB RAM. Also, to implement SQP in the proposed algorithm, as the local search, we used 'fmincon' in Matlab when the 'Algorithm' was set to 'SQP'.

In Subsection 4.1, the numerical results of IVNS are compared with exact solutions. Also, for comparing IVNS with metaheuristics and numerical algorithms in two Subsections 4.2 and 4.3, we consider 22 NOCPs. Their models are described in the Appendix, which are presented in terms of equations (1)-(6). The numerical results are summarized in Table 1. Details of these comparisons are given in the following subsection.

4.1 Comparison with the exact solution

Consider the nonlinear system state equations [24]

$$\dot{x}_1 = x_2^3,$$
$$\dot{x}_2 = u$$

The cost functional to be minimized, starting from the initial states $x_1(0) = 0$ and $x_2(0) = 1$, is

$$J = 4x_1(2) + x_2(2) + 4\int_0^2 u^2(t)dt$$

The exact trajectories of the problem, from PMP, are $x_1^*(t) = \frac{2}{5} - \frac{64}{5(t+2)^5}$ and $x_2^*(t) = \frac{4}{(t+2)^2}$, with the exact control signal $u^*(t) = \frac{-8}{(t+2)^3}$. Also the exact value of the performance index is $J^* = 3.35$. For the proposed algorithm, IVNS's parameters are set as $N_{t_1} = 15$, $N_{t_2} = 21$, $\varepsilon = 10^{-6}$ and the problem's parameters are set as $u_{left} = -1$ and $u_{right} = -\frac{1}{4}$. The IVNS's solution for the problem is $J = 3.3418$, thus, $E_J = K_\psi = 0.0082$.

Figure 1 shows the graphs of the exact and the obtained trajectories, for x_1 and x_2, and Figure 2 shows the graphs of the exact and the obtained control signals.

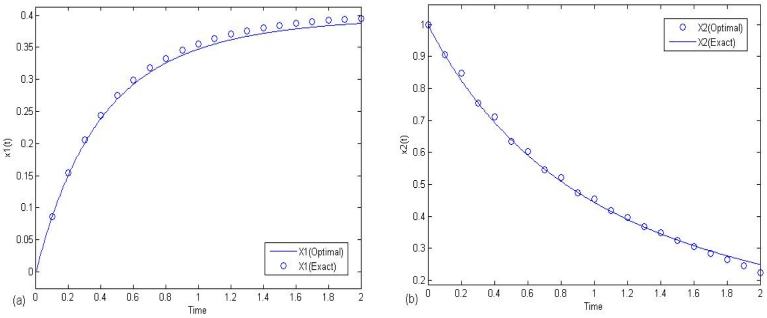


Figure 1: The exact and the obtained trajectories of (a) x_1 and (b) x_2, for the NOCP in subsection 4.1

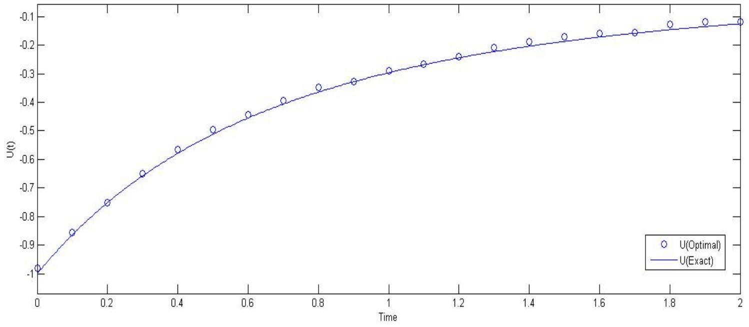


Figure 2: The exact and the obtained control signals for the NOCP in subsection 4.1

4.2 Comparison with metaheuristic algorithms

Here, six NOCPs are considered, NOCPs No. 1-6 in Appendix. The numerical results for the first NOCP is compared with hybrid improved GA, HIGA, proposed in [47]. The NOCPs No. 2-4, in the Appendix are compared with a metaheuristic, continuous GA and CGA, proposed in [1], which gave better solutions than shooting method and gradient algorithm (as the indirect methods) [29, 12], and SUMT (as the direct methods) [18]. For NOCPs No. 5 and 6 the results are compared with another metaheuristic, called IPSO, proposed in [37]. It has been shown that, for these NOCPs, IPSO was more accurate than some metaheuristic algorithms such as GA [42], DE [14], PSO [27] and some numerical methods [21, 23].

TCCR problem [47]

The first NOCP in the Appendix is a chemical process of Temperature Control for Consecutive Reaction, TCCR, which is an unconstrained two-state variable mathematical system. The objective is to obtain the optimal temperature profile that maximizes the yield of the temperature product B at the end of operation in a batch reactor, where the reaction $A \rightarrow B \rightarrow C$ is occurred. The state variables, x_1 and x_2 are the concentration of A and B, respectively, and the control variable u is the temperature. The problem solved by HIGA [47], which was more accurate than ACO [40] and iterative ACO [53]. From Table 1, we can see that the numerical behaviour of IVNS is better than HIGA.

VDP problem [1, 17]

The second NOCP in the Appendix is Van Der Pol, VDP, problem which has two state variables and one control variable. VDP problem has a final state constraint, which is $\psi = x_1(t_f) - x_2(t_f) + 1 = 0$. The problem solved by CGA [1] and IVNS. From [1], the norm of final state constraint for the CGA equals $\phi_f^* = 2.67 \times 10^{-11}$, however, this value for IVNS equals $\phi_f = 3.04 \times 10^{-9}$. So, the factor K_ψ for these methods can be seen in the sixth column of the Table 1. Note that the K_ψ of IVNS, 3.01×10^{-9}, is less than CGA's K_ψ, 3.0×10^{-4}. From Table 1, it is seem that IVNS can achieved more suitable solution than CGA.

CRP problem [1, 29]

The third NOCP in the Appendix is a mathematical model of Chemical Reactor Problem, CRP, which has two state variables and one control variable. The control variable is the flow of a coolant through a coil inserted in the reactor that controls the first-order irreversible exothermic reaction taking place in the reactor. The state variables, x_1 and x_2, are the deviations from the steady-state temperature and concentration, respectively. The numerical results of IVNS and CGA are shown in the third row of Table 1. CRP problem has two final state constraints, $\psi = [x_1, x_2]^T$. From [1], the norm of final state constraints for CGA, equals $\phi_f^* = 7.57 \times 10^{-10}$, when IVNS's norm of final state constraints is $\phi_f = 2.50 \times 10^{-8}$. But, the corresponding K_ψ of two methods shows that IVNS could achieve more accurate solutions than CGA.

FFRP problem [1, 18]

The fourth NOCP in the Appendix is Free Floating Robot Problem, FFRP, which has six state variables and four control variables. It was solved by CGA [1]. FFRP problem has six final state constraints, $\psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5, x_6]^T$. The norm of final state constraints for IVNS is $\phi_f^* = 4.61 \times 10^{-4}$, however, this value, from [1], for CGA is $\phi_f = 4.65 \times 10^{-3}$. From Table 1, we can see the numerical behaviour of IVNS is better than CGA, also it is clear that the obtained values of J, E_J, ϕ_f, E_ψ and K_ψ from IVNS are better than CGA.

CSTCR problem [37]

The fifth NOCP in the Appendix is a model of a nonlinear Continuous Stirred-tank Chemical Reactor, CSTCR. It has two state variables $x_1(t)$ and $x_2(t)$, as the deviation from the steady-state temperature and concentration, and one control variable $u(t)$, which represents the effect of the flow rate of cooling fluid on chemical reactor. The objective is to maintain the temperature and concentration close to steady-state values without expending large amount of control effort. Also, this is a benchmark problem in the handbook of test problems in local and global optimization [20], which is a multimodal optimal control problem [2]. It involves two different local minima. The values of the performance indices, for these solutions, equal 0.244 and 0.133. The numerical results of IVNS, with the parameters in Table 2, are compared with IPSO [37], and numerical methods in [2, 14]. From the results of the fifth row of Table 1, we can see that IVNS is the best.

MSNIC problem [37]

In the sixth NOCP in the Appendix, a Mathematical System with Nonlinear Inequality Constraint, MSNIC, is considered. It includes an inequality constraint, $d(x,t) = x_2(t) + 0.5 - 8(t - 0.5)^2 \leq 0$. From the sixth row of Table 1, we can see that the obtained value of the performance index, for IVNS is $J^* = 0.1720$, which is better than IPSO's, 0.1727, and other numerical methods given in [23, 33].

4.3 Comparison with numerical algorithms

In this subsection, for NOCPs no. 7-22, the results of IVNS are compared with some numerical methods such that SQP [18], SUMT [18], Bézier [21], HPM [15], DTM [41] and ADM [19]. Usually, for these methods the final state constraints are not reported. But these values are reported for IVNS in Table 1.

Comparison with Bézier [21]

The NOCP No. 7, in the Appendix, has exact solution, i.e. the exact value of performance index equals $J^* = -5.5285$ [49]. This problem has an inequality constraint as $d(x,t) = -6 - x_1(t) \leq 0$. It has been solved by a numerical method, proposed in [21], called Bézier, and the proposed algorithm, IVNS, with the parameters in Table 2. From seventh row of Table 1, the obtained value of the performance index from IVNS is better and more accurate than Bézier method.

Comparison with HPM [15], DTM [41] and ADM [19]

In this subsection, the results of IVNS with the parameters given in Table 2, are compared with HPM [15], DTM [41] and ADM [19]. For NOCP No. 8 in the Appendix, which is a constraint nonlinear model, the numerical results are compared with HPM. This NOCP has a final state constraint as

$$\psi = x - 0.5 = 0.$$

From [15], the norm of final state constraint for HPM is $\phi_f = 4.2 \times 10^{-6}$, however, this value for IVNS equals $\phi_f^* = 6.83 \times 10^{-11}$. From Table 1, it is clear that the obtained values of the performance index, the norm of final state constraint and K_ψ from IVNS are better than HPM's.

The problem No. 9 in the Appendix is a linear quadratic optimal control which has been solved by two numerical methods, DTM [41] and ADM [19]. Using the approximate values of $k(t)$, which is used to achieve the optimal control signal by linear feedback control as $u(t) = -k(t)x(t)$, the performance index could be calculated. The exact solution, from PMP, equals $J^* = 0.1929$. From Table 1, the values of E_J and K_ψ, for IVNS, with the same number of points, $N_{t_2} = 15$, equals 0.0052, which is less than DTM and ADM methods, (0.0087).

Comparison with SQP and SUMT

For NOCPs No. 10-22 in the Appendix, the numerical results of IVNS (the parameters are given in Table 2) are compared with SQP and SUMT methods. All these problems are described in [18]. For SQP and SUMT, the status of the final state constraints were not reported, so, we replaced the values of ϕ_f instead of E_ψ, in Table 1. Also, in computation of the factor, K_ψ, the values of E_ψ for SQP and SUMT methods are considered to be zero. The results (given in Table 1) show that IVNS could find more accurate results for performance index J, and the factor K_ψ, perspective.

Table 1: The best of numerical results for 12 different runs of NOCPs described in Appendix

Problem	Algorithm	J	E_J	E_ψ	K_ψ
TCCR	HIGA[47]	0.61046	2.0×10^{-5}	—	2.0×10^{-5}
	IVNS	**0.61048**	**0**	—	**0**
VDP	CGA [1]	1.7404	3.0×10^{-4}	**0**	3.0×10^{-4}
	IVNS	**1.7401**	**0**	3.01×10^{-9}	$\mathbf{3.01 \times 10^{-9}}$
CRP	CGA[1]	0.0163	4.0×10^{-4}	**0**	4.0×10^{-4}
	IVNS	**0.0159**	**0**	2.42×10^{-8}	$\mathbf{2.42 \times 10^{-8}}$
FFRP	CGA[1]	83.63	17.72	0.0042	17.7242
	IVNS	**65.91**	**0**	**0**	**0**
CSTCR	IPSO [37]	0.1354	0.0024	—	0.0024
	[2]	$J \in [0.135, 0.245]$	0.0020	—	0.0020
	[14]	$J \in [0.1358, 0.1449]$	0.0028	—	0.0028
	IVNS	**0.1328**	$\mathbf{2.0 \times 10^{-4}}$	—	$\mathbf{2.0 \times 10^{-4}}$
MSNIC	IPSO [37]	0.1727	0.0007	—	0.0007
	[23]	0.1816	0.0096	—	0.0096
	[33]	0.1769	0.0049	—	0.0049
	IVNS	**0.1720**	**0**	—	**0**
NOCP no. 7	Bézier [21]	-5.3898	0.1387	—	0.1387
	IVNS	$\mathbf{-5.5082}$	**0.0203**	—	**0.0203**
NOCP no. 8	HPM [15]	0.2353	0.0338	4.20×10^{-6}	0.0338
	IVNS	**0.2015**	**0**	**0**	**0**
NOCP no. 9	DTM [41]	0.2016	0.0087	—	0.0087
	ADM [19]	0.2016	0.0087	—	0.0087
	IVNS	**0.1877**	**0.0052**	—	**0.0052**
NOCP no. 10[b]	SUMT [18]	5.15×10^{-6}	5.14×10^{-6}	—	5.14×10^{-6}
	SQP [18]	6.57×10^{-6}	6.56×10^{-6}	—	6.56×10^{-6}
	IVNS	$\mathbf{6.57 \times 10^{-11}}$	**0**	—	**0**
NOCP no. 11[b]	SUMT [18]	1.7980	0.0791	—	0.0791
	SQP [18]	1.7950	0.0761	—	0.0761
	IVNS	**1.7189**	**0**	—	**0**
NOCP no. 12[b]	SUMT [18]	0.1703	0.0223	—	0.0223
	SQP [18]	0.2163	0.0683	—	0.0683
	IVNS	**0.1480**	**0**	—	**0**
NOCP no. 13[b]	SUMT [18]	3.2500	0.3507	NR[a]	0.3507
	SQP [18]	3.2500	0.3507	NR	0.3507
	IVNS	**2.8993**	**0**	7.49×10^{-10}	$\mathbf{7.49 \times 10^{-10}}$
NOCP no. 14[b]	SUMT [18]	-0.2490	0.001	NR	0.001
	SQP [18]	-0.2490	0.001	NR	0.001
	IVNS	$\mathbf{-0.2500}$	**0**	2.6×10^{-10}	$\mathbf{2.6 \times 10^{-10}}$
NOCP no. 15[b]	SUMT [18]	0.0167	6.0×10^{-4}	NR	6.0×10^{-4}
	SQP [18]	0.0168	7.0×10^{-4}	NR	7.0×10^{-4}
	IVNS	**0.0161**	**0**	3.42×10^{-9}	$\mathbf{3.42 \times 10^{-9}}$
NOCP no. 16[b]	SUMT [18]	3.7700	0.4648	NR	0.4648
	SQP [18]	3.7220	0.4168	NR	0.4168
	IVNS	**3.3052**	**0**	3.35×10^{-8}	$\mathbf{3.35 \times 10^{-8}}$
NOCP no. 17[b]	SUMT [18]	9.29×10^{-4}	3.0×10^{-6}	NR	3.0×10^{-6}
	SQP [18]	1.01×10^{-3}	8.4×10^{-5}	NR	8.4×10^{-5}
	IVNS	$\mathbf{9.26 \times 10^{-4}}$	**0**	6.66×10^{-10}	$\mathbf{6.66 \times 10^{-10}}$
NOCP no. 18[b]	SUMT [18]	2.2080	0.2079	NR	0.2079
	SQP [18]	2.2120	0.2119	NR	0.2119
	IVNS	**2.0001**	**0**	5.01×10^{-11}	$\mathbf{5.01 \times 10^{-11}}$
NOCP no. 19[b]	SUMT [18]	-8.8690	0.0002	NR	0.0002
	SQP [18]	-8.8690	0.0002	NR	0.0002
	IVNS	$\mathbf{-8.8692}$	**0**	6.89×10^{-9}	$\mathbf{6.89 \times 10^{-9}}$
NOCP no. 20[b]	SUMT [18]	0.0368	0.0042	—	0.0042
	SQP [18]	0.0368	0.0042	—	0.0042
	IVNS	**0.0326**	**0**	—	**0**
NOCP no. 21[b]	SUMT [18]	76.83	12.11	NR	12.11
	SQP [18]	77.52	12.80	NR	12.80
	IVNS	**64.72**	**0**	1.46×10^{-4}	$\mathbf{1.46 \times 10^{-4}}$
NOCP no. 22[b]	SUMT [18]	0.3428	0.0670	NR	0.0670
	SQP [18]	0.3439	0.0681	NR	0.0681
	IVNS	**0.2758**	**0**	0.0021	**0.0021**

[a] Not Reported.

[b] We here consider, $E_\psi = \phi_f$ for IVNS, and for SQP and SUMT methods, $E_\psi = 0$ (since the values were not reported, we consider the best possible situation for SQP and SUMT).

Table 1 shows that IVNS was 100 percent successful in point of view the performance index, numerically. The associated values of E_J for IVNS are zero for all test problems. It shows that IVNS provides robust results with respect to the other methods.

To have a more careful comparison, we computed the Gap between the performance index's value of the algorithms and the best obtained performance index's value. In other words, let J be the obtained value of the performance index of an algorithm. Now, similar to [51], we define the Gap as follows:

$$Gap(J) = |\frac{J - J^*}{J^*}| \tag{16}$$

From Table 1, the mean values of Gap for IVNS, SQP and SUMT, on NOCPs No. 10-22, are 0, $7.69e+3$ and $6.02e+3$, respectively. Thus it is obvious that, IVNS gave more better solution in comparison with SQP and SUMT. We believe that this is due to the fact that IVNS tries to find the global solution but SQP and SUMT didn't escape from a local minimum.

To compare with the CGA (as a global search algorithm), from Table 1, we see that the mean values of the Gap for CGA is 0.0981. Thus, we can see IVNS is 100 percent better than CGA from Gap perspective. This result shows that IVNS's estimations of global minimal is better than CGA's estimation. Therefore, based on these numerical study, we can conclude that IVNS outperforms than CGA.

The mean values of violation of the norm of the final state constraints, ϕ_f, for IVNS is 1.16×10^{-4}. Therefore, it is evident that IVNS is more robust. Also, the mean value of ϕ_f for IVNS and CGA are 1.53×10^{-4} and 1.55×10^{-3}, respectively, on NOCPs no. 2-4. Thus, we can say that the feasibility of the solutions given by IVNS and CGA are competitive. Therefore, it is seen that IVNS could provide very suitable solutions with respect to the optimality and feasibility criteria. Also, the mean of the factor, K_ψ, for IVNS equals 1.28×10^{-3}. For NOCPs No. 10-22 the mean of factor for IVNS, SQP and SUMT equals 1.76×10^{-4}, 1.0768 and 1.0272, respectively. Therefore, we can say that IVNS outperform well-known numerical methods. Since, the computational times of the most algorithms were not reported thus we didn't give the computational times of IVNS in Table 1. But, the details of the computational time of IVNS is given in Table 3 that will be discussed in Section 5.

5 Comparison with a common VNS

The main idea for proposing a two-phase algorithm is to decrease the required computational time in solving NOCPs. So, we focus on investigating of the influence of the second phase in IVNS. To compare the IVNS with a common VNS, the number of time nodes are selected same in both phases. In common VNS, only the first phase of IVNS, which the number of time node equal

Table 2: The parameters of IVNS for NOCPs described in the Appendix

Problem	u_{left}	u_{right}	N_{t_1}	N_{t_2}	ε	M_i
TCCR	298	398	11	15	10^{-6}	—
VDP	-0.5	2	31	151	10^{-6}	7
CRP	-1.5	2	21	51	10^{-8}	$[1,1]^T$
FFRP	-15	10	31	61	10^{-3}	$[70,\dots,70]^T_{6\times1}$
CSTCR	0	5	31	51	10^{-9}	—
MSNIC	-20	20	21	51	10^{-3}	1
no. 7	-2	2	21	131	10^{-9}	1
no. 8	-2	2	31	91	10^{-6}	1
no. 9	-2	3	11	15	10^{-6}	—
no. 10	-3	3	21	51	10^{-6}	—
no. 11	-2	2	31	91	10^{-5}	1
no. 12	-20	20	31	51	10^{-8}	1
no. 13	-4	3	31	75	10^{-6}	$[100,100]^T$
no. 14	-1	1	31	71	10^{-6}	1000
no. 15	-2	2	21	41	10^{-6}	$[100,100]^T$
no. 16	$-\pi$	π	31	51	10^{-9}	$[100,100]^T$
no. 17	-1	1	21	35	10^{-6}	$[10,10]^T$
no. 18	-5	5	31	151	10^{-6}	$[10,10]^T$
no. 19	-30	30	31	171	10^{-6}	$[100,100]^T$
no. 20	-1	1	31	171	10^{-6}	—
no. 21	-15	10	31	71	10^{-6}	$[70,\dots,70]^T_{6\times1}$
no. 22	-15	10	21	91	10^{-6}	$[10,\dots,10]^T_{6\times1}$

N_{t_2}, is applied. For these methods, 35 different runs, for each NOCP in the Appendix, were made with the same parameters. The influence of these methods investigated for these NOCPs on the dependent outputs consist of performance index, J, the factor, ϕ_f and required computational time, $Time$. The results are given in Table 3.

From Table 3, we observe that the two-phase method has no significant effect on J, ϕ_f. But the two-phase method, IVNS, needs less computational time than the common VNS, significantly (except NOCP No. 16). Therefore, based on this computational study, we can conclude that the usage of two-phase VNS can decrease the computational time, significantly, without loss of quality of solution.

6 Conclusion

In this paper, a two-phase algorithm, namely IVNS, was proposed for solving NOCPs. In each phase of the algorithm, we used a VNS, which performed

Table 3: The best numerical results for NOCPs in Appendix, using IVNS and common VNS

	IVNS			VNS		
Problem	J	ϕ_f	*Time*	J	ϕ_f	*Time*
TCCR	0.6105	—	4.1496	0.6107	—	4.2482
VDP	1.7401	3.04×10^{-9}	375.69	1.7513	1.42×10^{-9}	413.28
CRP	0.0159	2.50×10^{-8}	78.09	0.0164	3.12×10^{-9}	112.05
FFRP	65.91	4.61×10^{-4}	264.62	50.31	8.17×10^{-3}	285.13
CSTCR	0.1328	—	48.82	0.1116	—	52.83
MSNIC	0.1720	—	10.49	0.1725	—	29.82
no. 7	-5.5082	—	42.27	-5.5012	—	65.81
no. 8	0.2015	6.83×10^{-11}	11.18	0.2012	4.21×10^{-10}	12.24
no. 9	0.1877	—	3.1278	0.1899	—	5.6636
no. 10	6.57×10^{-11}	—	3.7440	2.88×10^{-11}	—	3.9624
no. 11	1.7189	—	119.94	1.7152	—	139.55
no. 12	0.1480	—	41.38	0.1486	—	54.35
no. 13	2.8993	7.49×10^{-10}	39.04	2.8935	3.41×10^{-9}	38.36
no. 14	-0.2500	2.60×10^{-10}	52.61	-0.2498	1.52×10^{-8}	93.10
no. 15	0.0161	3.42×10^{-9}	124.02	0.0162	4.03×10^{-10}	154.65
no. 16	3.3052	3.35×10^{-8}	137.85	3.3051	1.03×10^{-10}	111.07
no. 17	9.26×10^{-4}	6.66×10^{-10}	144.16	9.81×10^{-4}	8.35×10^{-8}	178.23
no. 18	2.0001	5.01×10^{-11}	35.10	2.0001	2.13×10^{-12}	120.07
no. 19	-8.8692	6.89×10^{-9}	114.30	-8.8692	7.13×10^{-9}	129.02
no. 20	0.0326	—	42.69	0.0326	—	64.23
no. 21	64.72	1.46×10^{-4}	145.68	56.54	4.74×10^{-3}	148.01
no. 22	0.2758	0.0021	135.25	0.2765	0.0038	217.65

a uniform distribution in the shaking step and the SQP, as the local search step. In the first phase, VNS started with a completely random initial solution of control input values. To increase the accuracy of the solution obtained from Phase 1, the some new time nodes were added and the values of the new control inputs were estimated by Spline interpolation. Next, in the second phase, VNS restarted by the solution constructed by Phase 1. Finally, we implemented the proposed algorithm on more than 20 well-known benchmarks and real world problems, then the results were compared with some recently proposed algorithms. The numerical results showed that IVNS could found mostly better solution than other proposed algorithms. Also, to compare of IVNS with a common VNS a computational study was done that showed that IVNS needed less computational time with respect to a common VNS.

Acknowledgements

Authors are grateful to there anonymous referees and editor for their constructive comments.

References

1. Abo-Hammour, Z.S., Asasfeh, A.G., Al-Smadi, A.M. and Alsmadi, O.M.K. *A novel continuous genetic algorithm for the solution of optimal control problems,* Optimal Control Applications and Methods, 32(4) (2011) 414–432.

2. Ali, M.M., Storey, C. and Törn, A. *Application of stochastic global optimization algorithms to practical problems,* Journal of Optimization Theory and Applications, 95(3) (1997) 545–563.

3. Arumugam, M.S., Murthy, G.R. and Loo, C. K. *On the optimal control of the steel annealing processes as a two stage hybrid systems via PSO algorithms,* International Journal Bio-Inspired Computing, 1(3) (2009) 198–209.

4. Arumugam, M.S. and Rao, M.V.C. *On the improved performances of the particle swarm optimization algorithms with adaptive parameters, crossover operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems,* Application Soft Computing, 8(1) (2008) 324–336.

5. Atkinson, K. and Han, W. *Theoretical Numerical Analysis: A Functional Analysis Framework*, Texts in Applied Mathematics, Springer, 2009.

6. Büskens, C. and Maurer, H. *SQP-methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real-time control*, Journal of Computational and Applied Mathematics, 120(1-2) (2000) 85–108.

7. Betts, J.T. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, Society for Industrial and Applied Mathematics, 2010.

8. Binder, T., Blank, L., Dahmen, W. and Marquardt, W. *Iterative algorithms for multiscale state estimation, part 1: Concepts*, Journal of Optimization Theory and Applications, 111(3) (2001) 501–527.

9. Bonnans, J.J.F., Gilbert, J.C., Lemaréchal, C. and Sagastizábal, C.A. *Numerical Optimization: Theoretical and Practical Aspects*, Springer London, Limited, 2006.

10. Brimberg, J., Uroevi, D. and Mladenović, N. *Variable neighborhood search for the vertex weighted k-cardinality tree problem*, European Journal of Operational Research, 171(1) (2006) 74 – 84.

11. Brimberg, J., Hansen, P. and Mladenović, N. *Attraction probabilities in variable neighborhood search*, 4OR, 8(2) (2010) 181–194.

12. Bryson, A.E. *Applied Optimal Control: Optimization, Estimation and Control*, Halsted Press book'. Taylor & Francis, 1975.

13. Costa, W., Goldbarg, M. and Goldbarg, E. *New VNS heuristic for total flowtime flowshop scheduling problem*, Expert Systems with Applications, 39(9) (2012) 8149–8161.

14. Lopez Cruz, I.L., Van Willigenburg, L.G. and Van Straten, G. *Efficient differential evolution algorithms for multimodal optimal control problems*, Applied Soft Computing, 3(2) (2003) 97–122.

15. Effati, S. and Saberi Nik, H. *Solving a class of linear and non-linear optimal control problems by homotopy perturbation method,* IMA Journal of Mathematical Control and Information, 28(4) (2011) 539–553.

16. Engelbrecht, A.P. *Computational Intelligence: An Introduction*, Wiley, 2007.

17. Fabien, B.C. *Numerical solution of constrained optimal control problems with parameters*, Applied Mathematics and Computation, 80(1) (1996) 43–62.

18. Fabien, B.C. *Some tools for the direct solution of optimal control problems*, Advances Engineering Software, 29(1) (1998) 45–61.

19. Fakharian, A., Beheshti, M.T.H. and Davari, A. *Solving the Hamilton -Jacobian-Bellman equation using adomian decomposition method*, International Journal of Computer Mathematics, 87(12) (2010) 2769–2785.

20. Floudas, C.A. and Pardalos, P.M. *Handbook of test problems in local and global optimization, Nonconvex optimization and its applications*, Kluwer Academic Publishers, 1999.

21. Ghomanjani, F., Farahi, M.H. and Gachpazan, M.*Bézier control points method to solve constrained quadratic optimal control of time varying linear systems*, Computational and Applied Mathematics, 31 (2012) 433–456.

22. Ghosh, A., Das, S., Chowdhury, A. and Giri, R. *An ecologically inspired direct search method for solving optimal control problems with Bézier parameterization*, Engineering Applications of Artificial Intelligence, 24(7) (2011) 1195–1203.

23. Goh, C.J. and Teo, K.L. *Control parametrization: A unified approach to optimal control problems with general constraints,* Automatica, 24(1) (1988) 13–18.

24. Gong, Q., Kang, W. and Ross, I.M. *A pseudospectral method for the optimal control of constrained feedback linearizable systems*, Automatic Control, IEEE Transactions on, 51(7) (2006) 1115–1129.

25. Hansen, P., Mladenović, N. and Pérez, J. M. *Variable neighbourhood search: methods and applications,* 4OR, 6(4) (2008) 319–360.

26. Hansen, P., Mladenović, N. and Urošević, D. *Variable neighborhood search and local branching,* Computers and Operations Research, 33(10) (2006) 3034–3045.

27. Herrera, F. and Zhang, J. *Optimal control of batch processes using particle swam optimisation with stacked neural network models,* Computers and Chemical Engineering, 33(10) (2009) 1593–1601.

28. Johnson, A.W. and Jacobson, S.H. *On the convergence of generalized hill climbing algorithms,* Discrete Applied Mathematics, 119(1-2) (2002) 37–57.

29. Kirk, D.E. *Optimal Control Theory: An Introduction*, Dover Publications, 2004.

30. Vincent Antony Kumar, A. and Balasubramaniam, P. *Optimal control for linear system using genetic programming,* Optimal Control Applications and Methods, 30(1) (2009) 47–60.

31. Lee, M.H., Han, C. and Chang, K.S. *Dynamic optimization of a continuous polymer reactor using a modified differential evolution algorithm,* Industrial and Engineering Chemistry Research, 38(12) (1999) 4825–4831.

32. Loudni, S., Boizumault, P. and Levasseur, N. *Advanced generic neighborhood heuristics for VNS*, Engineering Applications of Artificial Intelligence, 23(5) (2010) 736–764.

33. Mekarapiruk, W. and Luus, R. *Optimal control of inequality state constrained systems*, Industrial and Engineering Chemistry Research, 36(5) (1997) 1686–1694.

34. Michalewicz, Z., Janikow, C.Z. and Krawczyk, J.B. *A modified genetic algorithm for optimal control problems*, Computers & Mathematics with Applications, 23(12) (1992) 83–94.

35. Mladenović, N. and Hansen, P. *Variable neighborhood search*, Computers and Operations Research, 24(11) (1997) 1097–1100.

36. Mladenović, N., Dražić, M., Kovačevic-Vujčić, V. and Čangalović, M. *General variable neighborhood search for the continuous optimization*, European Journal of Operational Research, 191(3) (2008) 753–770.

37. Modares, H. and Naghibi-Sistani, M.B. *Solving nonlinear optimal control problems using a hybrid IPSO - SQP algorithm*, Engineering Applications of Artificial Intelligence, 24(3) (2011) 476–484.

38. Muhlenbein, H. and Zimmermann, J. *Size of neighborhood more important than temperature for stochastic local search*, In Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, volume 2, pages 1017–1024 vol.2, 2000.

39. Nocedal, J. and Wright, S.J. *Numerical Optimization*, Springer series in operations research and financial engineering, Springer, 1999.

40. Rajesh, J., Gupta, K., Kusumakar, H.S., Jayaraman, V.K. and Kulkarni, B.D. *Dynamic optimization of chemical processes using ant colony framework*, Computers & Chemistry, 25(6) (2001) 583–595.

41. Saberi Nik, H., Effati, S. and Yildirim, A. *Solution of linear optimal control systems by differential transform method,* Neural Computing and Applications, 23(5) (2013) 1311–1317.

42. Sarkar, D. and Modak, J.M. *Optimization of fed-batch bioreactors using genetic algorithm: multiple control variables,* Computers and Chemical Engineering, 28(5) (2009) 789–798.

43. Schlegel, M., Stockmann, K., Binder, T. and Marquardt, W. *Dynamic optimization using adaptive control vector parameterization,* Computers & Chemical Engineering, 29(8) (2005) 1731–1751.

44. Shi, X.H., Wan, L.M., Lee, H.P., Yang, X.W., Wang, L.M. and Liang, Y.C. *An improved genetic algorithm with variable population-size and a PSO-GA based hybrid evolutionary algorithm,* Machine Learning and Cybernetics, 2003 International Conference on, volume 3, pages 1735–1740, 2003.

45. Sim, Y.C., Leng, S.B. and Subramaniam, V. *A combined genetic algorithms-shooting method approach to solving optimal control problems,* International Journal of Systems Science, 31(1) (2000) 83–89.

46. Srinivasan, B., Palanki, S. and Bonvin, D. *Dynamic optimization of batch processes: I. characterization of the nominal solution,* Computers & Chemical Engineering, 27(1) (2003) 1 – 26.

47. Sun, F., Du, W., Qi, R., Qian, F. and Zhong, W. *A hybrid improved genetic algorithm and its application in dynamic optimization problems of chemical processes,* Chinese Journal of Chemical Engineering, 21(2) (2013) 144–154.

48. Marinus van Ast, J., Babuška, R. and De Schutter, B. *Novel ant colony optimization approach to optimal control,* International Journal of Intelligent Computing and Cybernetics, 2(3) (2009) 414–434.

49. Vlassenbroeck, J. *A chebyshev polynomial method for optimal control with state constraints,* Automatica, 24(4) (1988) 499–506.

50. Wang, F.S. and Chiou, J.P. *Optimal control and optimal time location problems of differential-algebraic systems by differential evolution,* Industrial and Engineering Chemistry Research, 36(12) (1997) 5348–5357.

51. Yaghini, M., Karimi, M. and Rahbar, M. *A hybrid metaheuristic approach for the capacitated p-median problem,* Applied Soft Computing, 13(9) (2013) 3922–3930.

52. Yamashita, Y. and Shima, M. *Numerical computational method using genetic algorithm for the optimal control problem with terminal constraints and free parameters,* Nonlinear Analysis: Theory, Methods & Applications, 30(4) (1997) 2285–2290.

53. Zhang, B., Chen, D. and Zhao, W. *Iterative ant-colony algorithm and its application to dynamic optimization of chemical process,* Computers & Chemical Engineering, 29(10) (2005) 2078–2086.

Appendix

The following NOCPs are described using eqns (1)-(6).

1. [47, 53, 40] (TCCR) $\phi = x_2,\ t_0 = 0,\ t_f = 1,\ f = [-4000exp(-2500/u) x_1^2, 4000exp(-2500/u)x_1^2 - 620000exp(-5000/u)x_2]^T,\ d = [298 - u, u - 398]^T,\ x_0 = [1, 0]^T$.

2. [1, 17] (VDP) $g = \frac{1}{2}(x_1^2 + x_2^2 + u^2), t_0 = 0, t_f = 5, f = [x_2, -x_2 + (1 - x_1^2)x_2 + u]^T, x_0 = [1, 0]^T, \psi = x_1 - x_2 + 1$.

3. [1, 29] (CRP) $g = \frac{1}{2}(x_1^2 + x_2^2 + 0.1u^2), t_0 = 0, t_f = 0.78, f = [x_1 - 2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)) - (x_1 + 0.25)u, 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T$.

4. [1, 18] (FFRP) $g = \frac{1}{2}(u_1^2 + u_2^2 + u_3^2 + u_4^2), t_0 = 0, t_f = 5, f = [x_2, ((u_1 + u_2)\cos x_5 - (u_2 + u_4)\sin x_5)/M, x_4, ((u_1 + u_3)\sin x_5 + (u_2 + u_4)\cos x_5)/M, x_6, (D(u_1 + u_3) - L_e(u_2 + u_4))/I]^T, x_0 = [0, 0, 0, 0, 0, 0]^T$, $\psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5, x_6]^T, M = 10, D = 5, I = 12, L_e = 5$.

5. [37] (CSTCR) $g = x_1^2 + x_2^2 + 0.1u^2, t_0 = 0, t_f = 0.78, f = [-(2 + u)(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)), 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.09, 0.09]^T$.

6. [37] (MSNIC) $\phi = x_3, t_0 = 0, t_f = 1, f = [x_2, -x_2 + u, x_1^2 + x_2^2 + 0.005u^2]^T, d = [-(20 - u)(20 + u), x_2 + 0.5 - 8(t - 0.5)^2]^T, x_0 = [0, -1, 0]^T$.

7. [21] $g = 2x_1, t_0 = 0, t_f = 3, f = [x_2, u]^T, d = [-(2 - u)(2 + u), -6 - x_1]^T, x_0 = [2, 0]^T$.

8. [15] $g = u^2, t_0 = 0, t_f = 1, f = \frac{1}{2}x^2 \sin x + u,\ x_0 = 0, \psi = x - 0.5$.

9. [41, 19] $g = \frac{1}{2}(x^2 + u^2),\ t_0 = 0, t_f = 1,\ f = -x + u,\ x_0 = 1$.

10. [18] $g = x^2 \cos^2 u,\ t_0 = 0, t_f = \pi,\ f = \sin \frac{u}{2},\ x_0 = \frac{\pi}{2}$.

11. [18] $g = \frac{1}{2}(x_1^2 + x_2^2 + u^2), t_0 = 0, t_f = 5, f = [x_2, -x_1 + (1 - x_1^2)x_2 + u]^T, d = -(x_2 + 0.25), x_0 = [1, 0]^T$.

12. [18] $g = x_1^2 + x_2^2 + 0.005u^2, t_0 = 0, t_f = 1, f = [x_2, -x_2 + u]^T, d = [-(20 - u)(20 + u), 0.5 + x_2 - 8(t - 0.5)^2]^T, x_0 = [0, -1]^T$.

13. [18] $g = \frac{1}{2}u^2, t_0 = 0, t_f = 2, f = [x_2, u]^T, x_0 = [1, 1]^T, \psi = [x_1, x_2]^T$.

14. [18] $g = -x_2, t_0 = 0, t_f = 1, f = [x_2, u]^T, d = -(1 - u)(1 + u), x_0 = [0, 0]^T, \psi = x_2$.

15. [18] $g = \frac{1}{2}(x_1^2 + x_2^2 + 0.1u^2), t_0 = 0, t_f = 0.78, f = [-2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)) - (x_1 + 0.25)u, 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T$.

16. [18] $g = \frac{1}{2}u^2, t_0 = 0, t_f = 10, f = [\cos u - x_2, \sin u]^T, d = -(\pi - u)(\pi + u), x_0 = [3.66, -1.86]^T, \psi = [x_1, x_2]^T$.

17. [18] $g = \frac{1}{2}(x_1^2 + x_2^2), t_0 = 0, t_f = 0.78, f = [-2(x_1 + 0.25) + (x_2 + 0.5)exp(25x_1/(x_1 + 2)) - (x_1 + 0.25)u, 0.5 - x_2 - (x_2 + 0.5)exp(25x_1/(x_1 + 2))]^T, d = -(1 - u)(1 + u), x_0 = [0.05, 0]^T, \psi = [x_1, x_2]^T$.

18. [18] $\phi = x_3, t_0 = 0, t_f = 1, f = [x_2, u, \frac{1}{2}u^2]^T, d = x_1 - 1.9, x_0 = [0, 0, 0]^T, \psi = [x_1, x_2 + 1]^T$.

19. [18] $\phi = -x_3, t_0 = 0, t_f = 5, f = [x_2, -2 + \frac{u}{x_3}, -0.01u]^T, d = -(30 - u)(30 + u), x_0 = [10, -2, 10]^T, \psi = [x_1, x_2]^T$.

20. [18] $\phi = (x_1 - 1)^2 + x_2^2 + x_3^2,\ g = \frac{1}{2}u^2,\ t_0 = 0, t_f = 5, f = [x_3 \cos u, x_3 \sin u, \sin u]^T, x_0 = [0, 0, 0]^T$.

21. [18] $g = \frac{1}{2}(u_1^2 + u_2^2 + u_3^2 + u_4^2), t_0 = 0, t_f = 5, f = [x_2, ((u_1 + u_3) \cos x_5 - (u_2 + u_4) \sin x_5)/M, x_4, ((u_1 + u_3) \sin x_5 + (u_2 + u_4) \cos x_5)/M, x_6, (D(u_1 + u_3) - L_e(u_2 + u_4))/I]^T, x_0 = [0, 0, 0, 0, 0, 0]^T, \psi = [x_1 - 4, x_2, x_3 - 4, x_4, x_5 - \frac{\pi}{4}, x_6]^T, M = 10, D = 5, I = 12, L_e = 5$.

22. [18] $g = 4.5(x_3^2 + x_6^2) + 0.5(u_1^2 + u_2^2), t_0 = 0, t_f = 1, f = [9x_4, 9x_5, 9x_6, 9(u_1 + 17.25x_3), 9u_2, -9(u_1 - 27.0756x_3 + 2x_5x_6)/x_2]^T, x_0 = [0, 22, 0, 0, -1, 0]^T, \psi = [x_1 - 10, x_2 - 14, x_3, x_4 - 2.5, x_5, x_6]^T$.

Iranian Journal of Numerical Analysis and Optimization
Vol 5, No. 1, (2015), pp 37-48

Numerical solution of multi-order fractional differential equations via the sinc collocation method

E. Hesameddini* and E. Asadollahifard

Abstract

In this paper, the sinc collocation method is proposed for solving linear and nonlinear multi-order fractional differential equations based on the new definition of fractional derivative which is recently presented by Khalil, R., Al Horani, M., Yousef, A. and Sababeh, M. in A new definition of fractional derivative, *J. Comput. Appl. Math.* **264** (2014), 65–70. The properties of sinc functions are used to reduce the fractional differential equation to a system of algebraic equations. Several numerical examples are provided to illustrate the accuracy and effectiveness of the presented method.

Keywords: Sinc function; Fractional differential equations; Multi-order FDEs; Collocation method.

1 Introduction

One of the old fields of mathematics is fractional calculus which dates back to the time of Leibniz [1] and from then many studies were done in this field [14]–[12]. Fractional differential equations (FDEs) have attracted the interest of researchers in many areas such as Physics, Chemistry, Engineering and Social Sciences [22, 15]. The analytic results on the existence and uniqueness of solutions to the FDEs have been investigated by many authors [11, 22, 16]. Generally, most of the FDEs do not have analytic solutions, so one has to resort to approximation and numerical methods.

One class of FDEs is multi-order fractional differential equations. They have been used to model various types of visco-elastic damping [22] and are expressed as follows

*Corresponding author

Received 12 August 2014; revised 14 January 2015; accepted 3 February 2015

E. Hesameddini
Department of Mathematics, Shiraz University of Technology, Shiraz , Iran. e-mail: hesameddini@sutech.ac.ir

E. Asadollahifard
Department of Mathematics, Shiraz University of Technology, Shiraz , Iran. email: e.asadolahifard@sutech.ac.ir

$$D^{(\alpha)}y(x) = F(x, y(x), D^{(\beta_1)}y(x), ..., D^{(\beta_k)}y(x)), \quad x \in I = [0, l], \tag{1}$$

with initial conditions

$$D^{(i)}(0) = d_i, \quad i = 0, 1, ..., m-1, \quad m \in N \tag{2}$$

where $m-1 < \alpha \leq m$, $0 < \beta_1 < \beta_2 < ... < \beta_k < \alpha$ and the values of d_i $(i = 0, 1, ..., m-1)$ describe the initial state of $y(x)$. $D^{(\alpha)}y$ indicates the fractional derivative of order α of y. Up to now, whenever this equation was under study, in most cases the fractional derivative was in the sense of Caputo definition. In this paper, we imply the new definition of conformable fractional derivative [18] which will be defined later. Depending on F, this equation classifies into linear and nonlinear.

In [14], it has been proved that equation (1) subject to the initial conditions (2) and under natural Lipschitz conditions imposed on F has a unique continuous solution.

Since the last decade, extensive research has been conducted on the development of numerical methods for equation (1). Doha et al.[25] proposed an efficient spectral tau and collocation method based on the Chebyshev polynomials for solving this equation. Extension of the tau method based on the shifted Legendre Gauss-Lobbato quadrature is used for solving equation (1) in [9]. In [12], this equation is converted into a system of FDEs and the shifted Chebyshev operational matrix method is used to solve the resultant system. Some other works on this problem are: piecewise polynomial collocation [17], Haar wavelet method [20], Lagrange wavelet method [23] and second kind Chebyshev wavelet method [30].

In this work, we apply the sinc collocation method for solving equation (1). The sinc method is an efficient method developed by Stenger [24]. It was widely used for the numerical solution of initial and boundary value problems [13, 19, 8], not only because of its exponential convergence rate but also due to its ability in handling problems with singularities. To the best of our knowledge, the sinc collocation method has not been used for solving FDEs directly. In this work, based on the new definition of fractional derivative [18], we compute the fractional derivative of the sinc function and apply it for solving equation (1).

The remainder of this paper is organized as follows: in Section 2, some definitions and theorems are presented that will be used in later sections. The proposed method is discussed in Section 3. Section 4 is devoted to numerical experiments. Finally some remarks are concluded.

2 Preliminaries

In this section, we recall some necessary definitions and mathematical preliminaries of the fractional theory and sinc method which will be used further in this paper.

2.1. The fractional derivative

The fractional calculus involves different definitions of fractional derivative operators such as Caputo and Riemmann-Lioville fractional derivative[22, 1]. One of the most recent works on the theory of derivatives of fractional order is done by Khalil et al. [18] which is the simplest definition. Up to now, some works were done based on this new definition [1, 2, 22]. In what follows, at first the conformable fractional derivative is defined and then some fantastic properties of this definition are presented.

Definition 1. [18] Let $\alpha \in (n, n+1]$, and f be an $n-$differentiable function at t, where $t > 0$. Then the conformable fractional derivative of f of order α is defined as

$$T_\alpha(f)(t) = \lim_{\varepsilon \to 0} \frac{f^{(\lceil\alpha\rceil-1)}(t + \varepsilon t^{(\lceil\alpha\rceil-\alpha)}) - f^{(\lceil\alpha\rceil-1)}(t)}{\varepsilon}, \tag{3}$$

where $\lceil\alpha\rceil$ is the smallest integer greater than or equal to α.

When the conformable fractional derivative of f of order α exists, we say f is $\alpha-$differentiable and we write $f^{(\alpha)}(t)$ for $T_\alpha(f)(t)$.

Remark 1. [18] As a consequence of Definition 1, one can easily show that

$$T_\alpha(f)(t) = t^{1+(\lceil\alpha\rceil-\alpha)} f^{(\lceil\alpha\rceil)}(t), \tag{4}$$

where $\alpha \in (n, n+1]$, and f is $(n+1)-$differentiable at $t > 0$.

Theorem 1. [18] Let $\alpha \in (0, 1]$, and f, g be $\alpha-$differentiable at a point $t > 0$. Then
1. $T_\alpha(af + bg) = aT_\alpha(f) + bT_\alpha(g), \quad \textit{for all } a, b \in R,$
2. $T_\alpha(fg) = fT_\alpha(g) + gT_\alpha(f).$

In [1], Abdeljawad was defined the left and right conformable fractional derivative. Since the left fractional derivative on $[0, \infty)$ is the conformable fractional derivative, we can have the following theorems according to [1].

Theorem 2. (Chain Rule) Assume $f, g : (0, \infty) \longrightarrow R$ be $\alpha-$differentiable functions, where $0 < \alpha \leq 1$. Let $h(t) = f(g(t))$. Then $h(t)$ is $\alpha-$differentiable and for all t with $t \neq 0$ and $g(t) \neq 0$ we have

$$(T_\alpha h)(t) = (T_\alpha f)(g(t))(T_\alpha g)(t)g(t)^{\alpha-1}.$$

If $t = 0$, then

$$(T_\alpha h)(0) = \lim_{t\to 0^+}(T_\alpha f)(g(t))(T_\alpha g)(t)g(t)^{\alpha-1}.$$

Theorem 3. Let $f : (0,\infty) \longrightarrow R$ be twice differentiable on $(0,\infty)$ and $0 < \alpha, \beta \leq 1$ such that $1 < \alpha + \beta \leq 2$. Then

$$(T_\alpha T_\beta f)(t) = T_{\alpha+\beta} f(t) + (1-\beta)tT_\alpha f(t).$$

2.2. Sinc function

The sinc function is defined on the whole real line, $-\infty < x < \infty$, by

$$sinc(x) = \begin{cases} \frac{sin\pi x}{\pi x} & x \neq 0, \\ 1 & x = 0. \end{cases}$$

For each integer k and the mesh size h, the translated sinc basis function is defined as

$$s(k,h)(x) = sinc(\frac{x-kh}{h}).$$

If a function $f(x)$ is defined on the real axis, then for any $h > 0$, the Whittaker cardinal expansion of $f(x)$ is as follows

$$c(f,h)(x) = \sum_{k=-\infty}^{\infty} f(kh)sinc(\frac{x-kh}{h}),$$

whenever this series converges. The properties of Whittaker cardinal expansion are derived in the infinite strip D_s of the complex w-planes where for $d > 0$

$$D_s = \{w = t + is : |s| < d \leq \frac{\pi}{2}\}.$$

These properties have been studied thoroughly in [24]. In order to approximate on the finite interval (a,b), which is used in this paper, we consider the one-to-one conformal map $w = \phi(z) = ln(\frac{z-a}{b-z})$, which maps the eye-shaped domain

$$D_E = \{z = x + iy : |arg\frac{z-a}{b-z}| < d \leq \frac{\pi}{2}\},$$

onto the infinite strip D_s. The basis functions on (a,b) are taken to be the composite translated sinc functions

$$s_k(x) = s(k,h)o\phi(x) = sinc(\frac{\phi(x) - kh}{h}), \quad k \in Z \tag{5}$$

where $s(k,h)o\phi(x)$is defined by $s(k,h)(\phi(x))$.
Let $\psi = \phi^{-1}$. We define the range of ψ on the real line as

$$\Gamma = \{\psi(w) \in D_E : -\infty < w < \infty\}.$$

For the uniform grid $\{kh\}_{k=-\infty}^{\infty}$ on the real line, the image which corresponds to these nodes is denoted by

$$x_k = \psi(kh) = \frac{a + be^{kh}}{1 + e^{kh}}, k = 0, \pm 1, \pm 2, ... \tag{6}$$

For discretizing the problem we need the following definition and theorems.

Definition 2. [24] Let $L_\beta(D_E)$ be the set of all analytic functions, for which there exist a constant, C, such that

$$|y(z)| \leq C\frac{|\rho(z)|^\beta}{[1 + |\rho(z)|]^{2\beta}}, \quad z \in D_E, \quad 0 < \beta \leq 1,$$

where $\rho(z) = e^{\phi(z)}$.
Theorem 4. [21] Let $y \in L_\beta(D_E)$, N be a positive integer and h be selected by the formula

$$h = (\frac{\pi d}{\beta N})^{\frac{1}{2}}, \tag{7}$$

then there exists a positive constant c_1, independent of N, such that

$$sup_{z\in\Gamma}|y(z) - \sum_{j=-N}^{N} y(z_j)s(j,h)o\phi(z)| \leq c_1 e^{-(\pi d\beta N)^{\frac{1}{2}}}.$$

Theorem 5. [21] Let ϕ be a conformal one-to-one map of the simply connected domain D_E onto D_S.Then

$$\delta_{kj}^{(0)} = s_k(x)|_{x=x_j} = \begin{cases} 1 & k = j, \\ 0 & k \neq j. \end{cases}$$

$$\delta_{kj}^{(1)} = \frac{d}{d\phi}[s_k(x)]|_{x=x_j} = \frac{1}{h}\begin{cases} 0 & k = j, \\ \frac{(-1)^{(j-k)}}{j-k} & k \neq j. \end{cases}$$

$$\delta_{kj}^{(2)} = \frac{d^2}{d\phi^2}[s_k(x)]|_{x=x_j} = \frac{1}{h^2}\begin{cases} \frac{-\pi^2}{3} & k=j, \\ \frac{-2(-1)^{(j-k)}}{(j-k)^2} & k \neq j. \end{cases}$$

3 Method of Solution

Consider equation (1) in $I=[0,1]$ where $D^{\alpha}y$ denotes the fractional derivative which is defined in (3) i.e. $D^{(\alpha)}y = y^{(\alpha)}$.
The approximate solution of equation (1) based on the sinc basis functions (5), should satisfy the initial conditions (2). But this basis functions do not have a derivative when x tends to 0 or 1 so we modify them as

$$w(x)s_k(x), \tag{8}$$

where $w(x) = (x(1-x))^{(m-1)}$ [6].
In order to approximate the solution, we construct a polynomial $p(x)$ that satisfies initial conditions [6]. So the approximate solution is represented by

$$y_N(x) = u_N(x) + p(x), \tag{9}$$

where

$$u_N(x) = \sum_{k=-N}^{N} c_k w(x) s_k(x), \tag{10}$$

and

$$p(x) = a_0 + a_1 x + ... + a_m x^m, \quad m-1 < v \leq m. \tag{11}$$

The unknown coefficients $a_0, a_1, ..., a_m$ and $\{c_k\}_{k=-N}^{N}$ are determined by substituting $y_N(x)$ into equation (1) and evaluating the result at the sinc points

$$x_j = \frac{e^{jh}}{1+e^{jh}}, j = -N-1, ..., N. \tag{12}$$

Notice that according to Theorem 1 and Remark 1, we have

$$(w(x)s_k(x))^{(\alpha)} = x^{1+[\alpha]-\alpha}(w(x)s_k(x))^{(1+[\alpha])}, \quad n < \alpha \leq n+1, \tag{13}$$

so

$$u_N^{(\alpha)}(x) = \Sigma_{k=-N}^{N} c_k (w(x)s_k(x))^{(\alpha)}. \tag{14}$$

Also it should be noted that when x tends to 1 or 0, we have

$$u_N(x) = u'_N(x) = ... = u_N^{(m-1)}(x) = 0.$$

Using equations (13) and (14), one can obtain

$$y_N^{(\alpha)}(x_j) = u_N^{(\alpha)}(x_j) + p^{(\alpha)}(x_j), \quad j = -N-1, ..., N. \tag{15}$$

Now by substituting this equation into equation (1), we obtain the following system of algebraic equations which can be solved for unknowns

$$y_N^{(v)}(x_j) = F(x_j, y_N(x_j), y_N^{(\beta_1)}(x_j), ..., y_N^{(\beta_k)}(x_j)), \quad -N-1 \leq j \leq N,$$
$$y_N^{(i)}(0) = d_i, \quad i = 0, 1, ..., m-1.$$

4 Applications and results

In this section, we solve some examples by the presented method and compare the numerical results with the exact solutions and some earlier works.

Example 1. As the first example, we consider the following nonlinear fractional initial value problem [5] on $[0, 1]$

$$y'''(x) + y^{(2.5)}(x) + y^2(x) = x^4, \quad y(0) = y'(0) = 0, \quad y''(0) = 2, \tag{16}$$

whose exact solution is $y(x) = x^2$. Following the procedure of the presented method, we consider the following approximate solution

$$y_N(x) = \sum_{k=-N}^{N} c_k w(x) s_k(x) + a_0 + a_1 x + a_2 x^2 + a_3 x^3,$$

where $w(x) = x^2(1-x)^2$. By substituting this approximate solution into equation (16) and evaluating at sinc points (12), we arrive at the following nonlinear system of algebraic equations which can be solved for unknown coefficients

$$6a_3 + \Sigma_{k=-N}^{N} c_k \{w_j'' \delta_{kj}^{(0)} + \delta_{kj}^{(1)}(3w_j''\phi_j' + 3w_j'\phi_j'' + w_j\phi_j''') + \delta_{kj}^{(2)}(3w_j'(\phi_j')^2 +$$
$$3w_j\phi_j''\phi_j') + \delta_{kj}^{(3)} w_j (\phi_j')^3\} + 6x_j^{0.5} a_3 + \Sigma_{k=-N}^{N} c_k \{x_j^{0.5} w''' j \delta_{kj}^{(0)} + w_j x_j^{0.5}(\phi_j''' \delta_{kj}^{(1)} +$$
$$3\phi_j''\phi_j'\delta_{kj}^{(2)} + (\phi_j')^3 \delta_{kj}^{(3)})\} + (a_0 + a_1 x_j + a_2 x_j^2 + a_3 x_j^3 + \Sigma_{k=-N}^{N} c_k w_j \delta_{kj}^{(0)})^2 = x_j^4$$
$$j = -N-1, ..., N,$$
$$y(0) = 0 \Rightarrow a_0 = 0, \quad y'(0) = 0 \Rightarrow a_1 = 0, \quad y''(0) = 2 \Rightarrow a_2 = 1.$$

According to relation (7), by taking $d = \frac{\pi}{2}$ and $\beta = 2$, we have $h = \frac{\pi}{2\sqrt{N}}$. Then by applying the well known Newton method with starting points $c_k = 0, k = -N, ..., N,\ a_0 = a_1 = a_3 = 0, a_2 = 1$, we obtain $c_k = 0, k = -N, ..., N$ and $a_0 = a_1 = a_3 = 0, a_2 = 1$. So the approximate solution is $y_N(x) = x^2$, which is the exact solution.

Example 2. Consider the fractional Ricatti equation on $[0, 1]$

$$y^{(\alpha)}(x) = 2y(x) - y^2(x) + 1, \quad 0 < \alpha \leq 1, \quad y(0) = 0.$$

For $\alpha = 1$, the exact solution of this equation is $y(t) = 1 + \sqrt{2}tanh(\sqrt{2}t + \frac{1}{2}ln(\frac{\sqrt{2}-1}{\sqrt{2}+1}))$. Consider the following approximate solution based on the sinc collocation method

$$y_N(x) = \Sigma_{k=-N}^{N} c_k s_k(x) + a_0 + a_1 x.$$

Odibat and Momani [20], solve this equation by using the modified homotopy perturbation method. Also in [4], this equation is solved by the Chebyshev wavelet operational matrices of fractional integration. For comparison, the results of this method are presented in Tables 1 and 2 with 192–set of Block Pulse Functions (Chebyshev wavelets was expanded into an 192-term block pulse functions).

In Table 1, the results of the presented method with $N = 1$ for $\alpha = 0.5$ and $\alpha = 0.75$ are compared with earlier works [4, 20]. We see that our results are in a good agreement with them. For $\alpha = 1$, the results are presented in Table 2. It is clear that by increasing N, the approximate solution becomes more and more accurate and for $N = 85$ the exact solution is obtained whereas Refs [4, 20] can not reach the exact solution. In Figure 1. the approximate solution for different values of α is shown. Numerical results show that as

Table 1: Numerical results with comparison to [4, 20] for $\alpha = 0.5$ and $\alpha = 0.75$ in Example 2

	$\alpha = 0.5$			$\alpha = 0.75$		
x	Ours[N=1]	[4]	[20]	Ours[N=1]	[4]	[20]
0.1	0.3956920	0.592756	0.321730	0.2321153	0.310732	0.216866
0.2	0.9184524	0.9331796	0.629666	0.4961556	0.584307	0.428892
0.3	1.2973611	1.1739836	0.940941	0.7523005	0.822173	0.654614
0.4	1.5802323	1.3466546	1.250737	0.9998683	1.024974	0.891404
0.5	1.7987123	1.4738876	1.549439	1.2372036	1.198621	1.132764
0.6	1.9690794	1.5705716	1.825456	1.4604023	1.349150	1.370240
0.7	2.0982657	1.646199	2.066523	1.6619744	1.481449	1.594278
0.8	2.1867519	1.706880	2.260633	1.8278045	1.599235	1.794879
0.9	2.2352250	1.756644	2.396839	1.9347648	1.705303	1.962239
1.0	2.3926026	1.798220	2.466004	2.0825668	1.801763	2.087384

Table 2: Numerical results with comparison to [4, 20] for $\alpha = 1$ in Example 2

x	Ours[N=10]	Ours[N=50]	Ours[N=80]	[4]	[20]	*Exact*
0.1	0.1134865	0.1103047	0.110295	0.1103111	0.110294	0.110295
0.2	0.2458331	0.2419881	0.241977	0.241995	0.241965	0.241977
0.3	0.3993884	0.3951178	0.395105	0.395123	0.395106	0.395105
0.4	0.5726929	0.5678265	0.567812	0.567829	0.568115	0.567812
0.5	0.7610790	0.7560297	0.756014	0.756029	0.757564	0.756014
0.6	0.9589295	0.9535820	0.953566	0.953576	0.958259	0.953566
0.7	1.1581332	1.1529646	1.152949	1.152955	1.163459	1.152949
0.8	1.3514117	1.3463785	1.366364	1.346365	1.365240	1.366364
0.9	1.5314497	1.5269249	1.526911	1.526909	1.554960	1.526911
1	1.6949935	1.6895135	1.689498	1.689494	1.723810	1.689498

α approaches to its integer value, the solution of fractional order differential equation approaches to the solution of integer order differential equation.

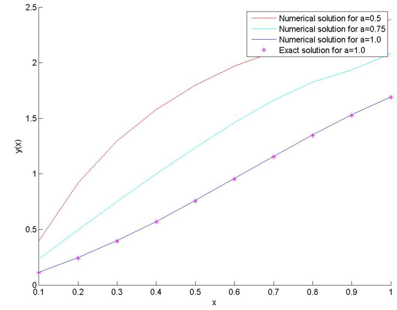


Figure 1: Approximate solution of Example 2 for different values of α

Example 3. [3] As the last example, consider the following inhomogeneous Bagley-Torvik equation

$$y''(x) + y^{(1.5)}(x) + y(x) = 1 + x,$$

subject to initial conditions

$$y(0) = y'(0) = 1.$$

The exact solution of this equation is $y(x) = 1 + x$.
In a same manner of last examples, by considering the approximate solution as

$$y_N(x) = \Sigma_{k=-N}^{N} c_k w(x) s_k(x) + a_0 + a_1 x + a_2 x^2,$$

where $w(x) = x(1 - x)$, one can obtain $y_N(x) = 1 + x$ which is the exact solution.

5 Conclusion

In this work the sinc-collocation method is used to approximate the solution of multi-order fractional differential equations with initial conditions. This method converts the FDEs into a system of algebraic equations which can be solved more easier. In this work, the fractional derivatives are described in the sense of new definition which makes us able to solve fractional differential equation directly by the sinc method for the first time. Also this method can be applied to other types of FDEs easily. Several examples are included to demonstrate the reliability and efficiency of our method.

6 Acknowledgments

The authors are indebted to the unknown referees for their constructive suggestions toward improving this paper.

References

1. Abdeljawad, T. *On conformable fractional calculus*, J. Comput. Appl. Math. (2014), http://dx.doi.org/10.1016/j.cam.2014.10.016.

2. Abu Hammad, M. and Khalil, R. *Conformable fractional Heat differential equation*, ijpam. 94 (2014) 215–221.

3. Amairi, M., Aoun, M., Najar, S. and Abdelkrim, M.N. *A constant enclosure method for validating existence and uniqueness of the solution of an initial value problem for a fractional differential equation*, Appl. Math. Comput. 217 (2010) 2162–2168.

4. Atabakzadeh, M.H., Akrami, M.H. and Erjaee, G.H. *Chebyshev operational matrix method for solving multi-order fractional ordinary differential equations*, Appl. Math. Model. 37 (2013) 8903–8911.

5. Bhrawy, A.H., Alofi, A.S. and Ezz–Eldien, S. S. *A quadrature tau method for fractional differentials equations with variable coefficients*, Appl. Math. Lett. 24 (2011) 2146–2152.

6. Deng, J. and Ma, L. *Existence and uniqueness of solutions of initial value problems for nonlinear fractional differential equations*, Appl. Math. Lett. 23 (2010) 676–680.

7. Diethelm, K. and Ford, N.J. *Multi-order fractional differential equations and their numerical solutions*, Appl. Math. Comput. 154 (2004) 621–640.

8. Diethelm, K. and Ford, N.J. *Numerical solution of the Bagley-Torvik equation*, BIT. 42 (2002) 490–507.

9. Doha, E.H., Bhrawy, A.H. and Ezz-Eldien, S.S. *A Chebyshev spectral method based on operational matrix for initial and boundary value problems of fractional order*, Comput. Math. Appl. 62 (2011) 2364–2373.

10. Doha, E.H., Bhrawy, A.H. and Ezz-Eldien, S.S. *Efficient Chebyshev spectral methods for solving multi-term fractional orders differential equations*, Appl.Math. Modell. 35 (2011) 5662–5672.

11. El–Gamel, M., Cannon, J.R. and Zayed, A.I. *Sinc–Galerkin method for solving linear Sixth order boundary-value problems*, Math. Comp. 73 (2004) 1325–1343.

12. Garrappa, R. and Popolizio, M. *On the use of matrix functions for fractional partial differential equations*, Math. Comput. Simulation. 81 (2011) 1045–1056.

13. Jafari, H., Das, S. and Tajadodi, H. *Solving a multi-order fractional differential equation using homotopy analysis method*, J. King Saud Univ. Sci. 23 (2011) 151–155.

14. Khalil, R., Al Horani, M., Yousef, A. and Sababeh, M. *A new definition of fractional derivative*, J. Comput. Appl. Math. 264 (2014) 65–70.

15. Kilbas, A.A., Srivastava, H.M. and Trujillo, J.J. *Theory and Applications of Fractional Differential Equations*, Elsevier, San Diego, 2006.

16. LI, Y. *Solving a nonlinear fractional differential equation using Chebyshev wavelets*, Commun. Nonlinear Sci. Numer. Simul. 15 (9) (2010), 2284–2292.

17. Lund, J. and Bowers, K.L. *Sinc methods for quadrature and differential equations*, PA, Philadelphia, SIAM, 1992.

18. Narasimhan, S., Majdalani, J. and Stenger, F. *A first step in applying the Sinc collocation method to the nonlinear Navier Stokes Equations*, Numer. Heat Transfer Part B. 41 (2002) 447–462.

19. Nurmuhammad, A., Muhammad, M., Mori, M. and Sugihara, M. *Double exponential transformation in the Sinc-collocation method for a boundary value problem with fourth-order ordinary differential equation*, J. Comput. Appl. Math. 182 (2005) 32–50.

20. Odibat, Z. and Momani, S. *Modified homotopy perturbation method: application to quadratic Riccati differential equation of fractional order*, Chaos Solitons Fract. 36 (2008) 167–74.

21. Oldham, K.B. and Spanier, J. *The Fractional Calculus*, Academic Press, New York, 1974.

22. Ortigueiraa, M.D. and Machado, J.A.T. *What is a fractional derivative?*, J. Comput. Phys. (2014), http://dx.doi.org/10.1016/j.jcp.2014.07.019.

23. Pedas, A. and Tamme, E. *Piecewise polynomial collocation for linear boundary value problems of fractional differential equations*, J. Comput. Appl. Math. 236 (2012) 3349–3359.

24. Podlubny, I. *Fractional Differential Equations*, in: Mathematics in Science and Engineering, vol. 198, Academic Press Inc., San Diego, CA, 1999.

25. Rehman, M.U. and Khan, R.A. *A numerical method for solving boundary value problems for fractional differential equations*, Appl. Math. Model. 36 (2012) 894–907.

26. Rehman, M.U. and Khan, R.A. *The Legendre wavelet method for solving fractional differential equations*, Commun. Nonlinear Sci. Numer. Simulat. 16 (11) (2011) 4163–4173.

27. Saadatmandi, A. and Dehghan, M. *The use of Sinc-Collocation Method for Solving multi-point boundary value problems*, Commun. Nonlinear Sci. Numer. Simul. 17 (2) (2012) 593–601.

28. Stenger, F. *Numerical Methods Based on Sinc and Analytic Functions*, New York: Springer, 1993.

29. Sugihara, M. *Double exponential transformation in the Sinc-collocation method for two-point boundary value problems*, J. Comput. Appl. Math. 149 (2002) 239–250.

30. Wang, Y. and Fan, Q. *The second kind Chebyshev wavelet method for solving fractional differential equations*, Appl. Math. Comput. 218 (2012) 8592–8601.

31. Yuanlu, L. *Solving a nonlinear fractional differential equation using Chebyshev wavelets*, Commun. Nonlinear Sci. Numer. Simul. 15 (9) (2010) 2284–2292.

Iranian Journal of Numerical Analysis and Optimization
Vol 5, No. 1, (2015), pp 49-61

RBFs meshless method of lines based on adaptive nodes for Burgers' equations

A.R. Soheili*, M. Arab Ameri and M. Barfeie

Abstract

We introduce a RBFs mesheless method of lines that decomposes the interior and boundary centers to obtain the numerical solution of the time dependent PDEs. Then, the method is applied with an adaptive algorithm to obtain the numerical solution of one dimensional problems. We show that in the problems in which the solutions contain region with rapid variation, the adaptive RBFs methods are successful so that the PDE solution can be approximated well with a small number of basis functions. The method is described in detail, and computational experiments are performed for one-dimensional Burgers' equations.

Keywords: Method of Lines; Radial basis functions; Adaptive Method; Burgers' Equations.

1 Introduction

The radial basis functions (RBFs) methods are one of the most attractive meshless methods. These methods are easy to implement, very suitable for problems in irregular geometries and the formulation for different dimensional problems are similar. Also, this method can be spectrally accurate [11]. A set of points called centers are needed to define the RBFs. Therefore, a RBF can be defined anywhere in a given domain, independently to the other RBFs.

*Corresponding author

Received 2 March 2014; revised 6 January 2015; accepted 17 February 2015

A.R. Soheili
The Center of Excellence on Modeling and Control Systems, Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran. e-mail: soheili@um.ac.ir

M. Arab Ameri
Faculty of Mathematics, University of Sistan and Baluchestan, Zahedan, Iran. email: arabameri@math.usb.ac.ir

M. Barfeie
Faculty of Mathematics, University of Sistan and Baluchestan, Zahedan, Iran. email: m.barfeie@gmail.com

Both the approximation quality and the stability of the RBFs interpolation depend on the positions of the centers set [9].

The condition number of RBFs collocation methods becomes large when the number of centers increases, while reducing the number of centers improves the conditioning [9, 13]. In order to obtain numerical solution with the minimal numbers of centers, we can use a set of adaptive nodes rather than uniform ones. Especially in problems whose solutions contain regions of rapid variation, adaptive methods are preferred over fixed grid methods, [17]. The goal of an adaptive method is to obtain a numerical solution such that the error is less than a prescribed accuracy but with the minimal number of grid points. By using adaptive methods, the computational grid should reflect the profile of the solution. Clearly, grids with finer spacing should be concentrated in regions, where high variations occur, and much coarser grids can be used in other regions.

Some methods have been constructed to select centers of RBFs. In [6, 26], the power function is used to iteratively obtain an optimal set of nodes. In [25], an adaptive algorithm so-called residual sub-sampling is introduced such that nodes can be added or removed based on residuals evaluated at a finer set of nodes. Our goal is to move a fixed numbers of nodes in such a way that nodes move with time and concentrate in region of domain that the solution has rapid variations. To this goal in this paper, we use a simple adaptive nodes generation method that is used for finite difference computations [24] and RBFs method [23]. Also we introduce a RBFs meshless method of lines to solve time dependent PDE with adaptive centers. In this method, we divide centers to interior and boundary data centers and obtain the expansion coefficients of boundary centers as a function of interior ones. This gives an ODEs system that is only related to the expansion coefficients of the interior data centers instead of all data centers. Actually after approximation spatial derivatives of equation and boundary condition with RBFs, we have a system of differential algebraic equations (DAEs) [5]. By decomposing centers and replacing boundary coefficients as a function of interior ones we obtain a smaller system of ODEs. The resultant system of ODEs can be solved with a proper ODE solver. We use the function *ode15s* in Matlab for solving the resulting system of ordinary differential equations.

In this paper, in order to combine the adaptive method and the RBFs method of lines, we start with a set of uniform centers, then the adaptive method is used to obtain new centers for initial condition. After obtaining the adaptive centers, the PDE is advanced for a small time step. The *ode15s* in Matlab is used for solving the resultant ODEs system. Then, the numerical solution of the PDE is used to obtain adaptive centers for next time. The procedure is repeated until the final time. We perform computational experiment for unsteady Burgers' equations and demonstrate the benefits of adaptation in the numerical experiments.

The rest of the paper is organized as follows. In Section 2 at first the RBFs method of lines is introduced, then adaptive method is extended for time dependent PDEs. Numerical experiment are given in Section 3. Finally, the conclusion is given in Section 4.

2 Meshless method of lines with adaptive RBFs

In this section, we introduce a RBFs mesheless method of lines that decomposes the interior and boundary centers to obtain the numerical solution of the time dependent PDEs. Then, apply the method with an adaptive algorithm to obtain the numerical solution of one dimensional Burgers' equations.

2.1 RBFs meshless method of lines

There are two classes of RBFs, known as globally supported and locally supported [22, 16]. Globally supported RBFs are infinitely smoothed and contain a free parameter ϵ, called shape parameter. This parameter affects both accuracy of the solutions and conditioning of the collocation matrix. As ϵ decreases, numerical solution of PDEs gets more accurate and the condition number of the resulting matrix gets larger. If the shape parameter becomes too small, the ill-conditioned matrix leads to numerical instabilities and loss of precision. Thus it is important to select a good values for ϵ. There are some paper related to select an optimal value for RBFs shape parameter [21, 1, 14].

Generally a radial basis function is a function $\phi_i(x, \epsilon) = \phi\left(\epsilon \|x - x_i\|_2\right)$, which depends solely on the distance between $x \in \mathbb{R}$ and a fixed center $x_i \in \Omega$. $\phi_i : \mathbb{R}^+ \to \mathbb{R}$ is a continuous function and $\|\cdot\|_2$ represents the Euclidean norm. The multiquadrics (MQ) RBF proposed by Hardy [3], is one of the most used globally supported RBFs because of its spectral convergence property. In [4], Franke showed that the MQ RBF is one of the best methods among 29 scattered data interpolation schemes. We here use MQ RBF defined as $\phi(r, \epsilon) = \sqrt{1 + (\epsilon r)^2}$.

Let a set of N distinct centers $\{x_i\}_{i=1}^{N}$ is given in $\Omega \bigcup \partial\Omega$, where Ω is a bounded domain in $\mathbb{R}$. We assume that the arrangement of the centers is in such a way that the first N_I centers and the last N_B centers lie in Ω and $\partial\Omega$, respectively, $N = N_I + N_B$. Consider the following time dependent PDE of the general form

$$\frac{\partial u(x,t)}{\partial t} - \mathcal{L}u(x,t) = f(x,t), \quad x \in \Omega, \qquad \mathcal{B}u(x,t) = g(x,t), \qquad x \in \partial\Omega, \tag{1}$$

with the initial condition

$$u(x,0) = u_0(x). \tag{2}$$

$\mathcal{L}$ and $\mathcal{B}$ are differential and boundary operators respectively. We approximate the solution of equation (1) by

$$u^N(x) = \sum_{i=1}^{N_I} c_i(t)\phi\left(\|x - x_i\|\right) + \sum_{i=N_{I+1}}^{N} c_i(t)\phi\left(\|x - x_i\|\right). \tag{3}$$

Using collocation method to ensure that the approximation $u^N(x)$ satisfies in equations (1), one obtains the following system of equations for the expansion coefficients

$$A_{1,1}\dot{C}_1 + A_{1,2}\dot{C}_2 = F + \mathcal{L}_\phi(C_1, C_2), \tag{4}$$

$$0\,\dot{C}_1 + 0\,\dot{C}_2 = G(t) - \left(A_{2,1}C_1 + A_{2,2}C_2\right), \tag{5}$$

where

$$\begin{aligned}
&A_{1,1}(i,j) = \phi(\|x_i - x_j\|), \quad i = 1,\dots,N_I, j = 1,\dots,N_I,\\
&A_{1,2}(i,j) = \phi(\|x_i - x_j\|), \quad i = 1,\dots,N_I, j = N_{I+1},\dots,N,\\
&A_{2,1}(i,j) = \mathcal{B}\phi(\|x_i - x_j\|), \quad i = N_{I+1},\dots,N, j = 1,\dots,N_I,\\
&A_{2,2}(i,j) = \mathcal{B}\phi(\|x_i - x_j\|), \quad i = N_{I+1},\dots,N, j = N_{I+1},\dots,N,\\
&\mathcal{L}_\phi(C_1,C_2)^T = [\mathcal{L}_{1\phi}(C_1,C_2),\dots,\mathcal{L}_{N_I\phi}(C_1,C_2)],\\
&\mathcal{L}_{i\phi}(C_1,C_2) = \sum_{j=1}^{N_I} c_j(t)\mathcal{L}\phi\left(\|x_i - x_j\|\right) + \sum_{j=N_{I+1}}^{N} c_j(t)\mathcal{L}\phi\left(\|x_i - x_j\|\right),\\
&F^T = [f(x_1,t),\dots,f(x_{N_I},t)],
\end{aligned}$$

and

$$G(t)^T = [g(x_{N_{I+1}},t),\dots,g(x_N,t)].$$

Equations (4) and (5) are distinct from ODEs because the coefficient matrix of the $\dot{C}^T = [\dot{C}_1, \dot{C}_2]$ is singular and are referred to as differential-algebraic equations (DAEs). DAEs differ in many ways from ordinary differential equations and there are some problems to be expected in solving these systems. More information about differential-algebraic equations can be found in [10, 12]. In order to reach a system of ODEs, we obtain C_2 and $\dot{C}_2$ from equation (5) as follows:

$$C_2 = A_{2,2}^{-1}\left(G(t) - A_{2,1}C_1\right), \tag{6}$$

$$\dot{C}_2 = A_{2,2}^{-1}\left(\dot{G}(t) - A_{2,1}\dot{C}_1\right). \tag{7}$$

Note that unlike the interpolation problem the invertibility of $A_{2,2}$ may failed for some special centers arrangements. However, numerical experiments show that the cases of singularity for Kansa method is rare [19]. We substitute C_2

and $\dot{C}_2$ into equation (4) to obtain a $N_I \times N_I$ nonlinear system of ordinary differential equation for C_I as follows:

$$\left(A_{1,1} - A_{1,2}A_{2,2}^{-1}A_{2,1}\right)\dot{C}_1 = F + \mathcal{L}_\phi(C_1) - A_{1,2}A_{2,2}^{-1}\dot{G}(t), \tag{8}$$

where

$$\mathcal{L}_\phi(C_1)^T = [\mathcal{L}_{1\phi}(C_1), \dots, \mathcal{L}_{N_I\phi}(C_1)],$$
$$\mathcal{L}_{i\phi}(C_1) = \sum_{j=1}^{N_I} c_j(t)\mathcal{L}\phi\left(\|x_i - x_j\|\right) + \sum_{j=N_{I+1}}^{N} d_j(t)\mathcal{L}\phi\left(\|x_i - x_j\|\right),$$

and $d_j(t)$ is jth component of the vector $C_2 = A_{2,2}^{-1}\left(G(t) - A_{2,1}C_1\right)$.

After solving the reduced system using a proper ODE solver, its solution vector C_1 is applied to obtain C_2 and C, using the relations $C_2 = A_{2,2}^{-1}\left(G(t) - A_{2,1}C_1\right)$ and $C = [C_1, C_2]$. This method can be used for high dimensional problems. In case of one dimensional problem, we have only two boundary nodes x_1 and x_N.

2.2 Adaptive method

In this section, the proposed mesheless method of lines that decomposes the interior and boundary centers to reach a smaller system of equations is combined with an adaptive algorithm that is used for finite difference and RBFs computations [24, 23]. In this method, at first the arclength of the numerical solution is computed. Then, the total length is divided into $(N-1)$ equal part and the projection of each part onto x-axis determines the position of adaptive centers. The selected nodes on x-axis are such that the variation of the solution is equi-distributed on each section.

Suppose that the approximate solution and the centers are given at the time step t_n. The adaptive method is generalized for RBFs and introduced in the following algorithm:

1) $S_1 = 0, S_j = S_{j-1} + \sqrt{(h_j^n)^2 + (u_j^n - u_{j-1}^n)^2}, j = 2...N,$
$\left(u_j^n = u(x_j^n, t^n), h_j^n = x_j^n - x_{j-1}^n\right).$
This step compute the arclength of solution u at time step t^n.

2) $\delta = \frac{S_N}{N-1}, k = 2, \bar{x}_1^n = x_1^n, \bar{x}_N^n = x_N^n.$
In this step the total length is divided into $(N-1)$ equal part.
3) For $j = 2, \cdots, N-1, \Delta = (j-1)\delta.$

- while $\Delta > S_k$ put $k = k+1$,
- $\bar{x}_j^n = x_{k-1}^n + \frac{(\Delta - S_{k-1})h_k^n}{S_k - S_{k-1}}$, Next j.

These steps project each part on solution onto x-axis.

The set $\bar{x}_j, j = 2, \cdots, (N-1)$ are adaptive interior nodes and $\bar{x}_1, \bar{x}_N$ are the boundary nodes which are fixed. In using adaptive centers in region with rapid variations, nodes are close to each other and hence a larger value of shape parameter is needed. In order to obtain results with a smaller shape parameter, the final set of centers are selected as $.9\bar{x}_j + .1x_j$.

In solving PDE problems, at first we apply the above adaptive algorithm for the initial condition to obtain the adaptive centers at $t = 0$. Then, adaptive centers are used for the RBFs method of lines to advance the PDE for a small time step. Next, the approximate solution at this time is used to obtain the adaptive centers again. Note that in each step we need to interpolate u at the adaptive centers to obtain initial condition for next time. The procedure is repeated until approximate solution is obtained at the final time.

3 Numerical experiments

In this section, the proposed method is applied to obtain numerical solution of Burgers' equation as follows:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{1}{Re}\frac{\partial^2 u}{\partial x^2}, \quad x \in (0,1), \tag{9}$$

where Re is the Reynolds number. Equation (9) has shock wave behavior when the coefficient of kinematic viscosity $\nu = 1/Re$ is small. Also, it is a useful model for many interesting physical problems such as modeling of fluid dynamics, turbulence, boundary layer behavior, shock wave formation, traffic flow and is an interesting test problem for establishing the efficiency of different methods [8, 20].

Example 1. We consider Equation (9) with the following exact solution [15]

$$u(x,t) = \frac{\alpha + \mu + (\mu - \alpha)exp(\eta)}{1 + exp(\eta)}, \tag{10}$$

where $\eta = \alpha.Re.(x - \mu t - \beta), \alpha, \mu$ and β are arbitrary constant.

In this example α, μ and β are .4, .6 and .125 respectively. The boundary conditions are

$$u(0,t) = 1, \quad u(1,t) = .2 \quad t > 0. \tag{11}$$

Initial condition is taken from the exact solution. In order to measure the error, root mean square error (rms) is computed at M evaluation nodes z_i as:

$$rms \quad error = \sqrt{\frac{\sum_{i=1}^{M} \left(u^N(z_i) - u(z_i)\right)^2}{M}}.$$

Table 1 shows the rms error at $t = .2, .4, .6, .8$ and $t = 1$ for $Re = 100$ and $Re = 500$. The results are computed for $N = 50$ adaptive centers. When Re increases, the gradient of solution become sharper and consequently a larger values of shape parameter is needed. In this example, the values of shape parameters for $Re = 100$ and $Re = 500$ are 50 and 150, respectively.

The numerical solution in Example 1 at $t = .1, t = .5$ and $t = 1$ for $Re = 100$ and $Re = 500$ are shown in Figures 1.a and 2.a respectively. Figures 1.b and 2.b show the corresponding nodes trajectories. Figures show that the nodes move with time and are concentrated in region with rapid variations. When Re increases, the gradient become sharper and the nodes are more concentrated in region with rapid variations.

The numerical and exact solutions of Example 1 at $t = 1$ are plotted in Figure 3. In order to obtain numerical solution with a set of uniform centers a larger number of nodes is needed [23]. Figure 4 shows the numerical solutions and absolute errors for $N = 50$ uniform and adaptive centers. In the case of using uniform centers, the numerical solution with some oscillations is obtained for $Re = 500$ and $\epsilon = 50$ at $t = .1$. As Figure 5 shows in order to obtain an acceptebale solution at this time, we need to use more uniform nodes or a set of adaptive centers.

Table 1: rms error values corresponding to Example 1

Re	rms error (t=.2)	rms error (t=.4)	rms error (t=.6)	rms error (t=.8)	rms error (t=1)
100	2.121018e-003	3.149610e-003	4.023452e-003	4.826538e-003	5.599954e-003
500	1.485532e-003	1.840686e-003	2.517397e-003	4.152420e-003	6.497818e-003

Example 2. We consider Burgers' equation (9) with the initial condition

$$u(x, 0) = sin(\pi x),$$

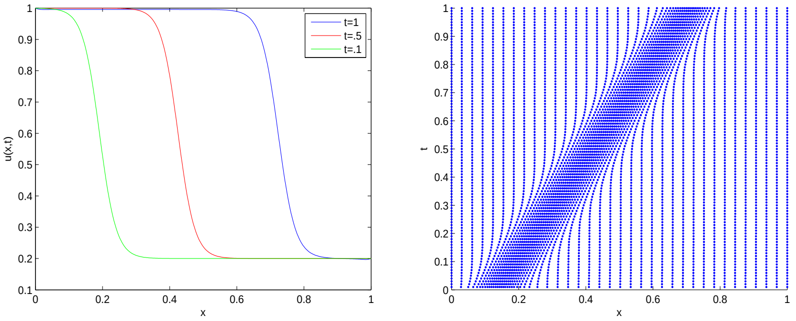
and the boundary conditions

$$u(0, t) = u(1, t) = 0, \quad t > 0.$$

The exact solution for this example is given by [15]

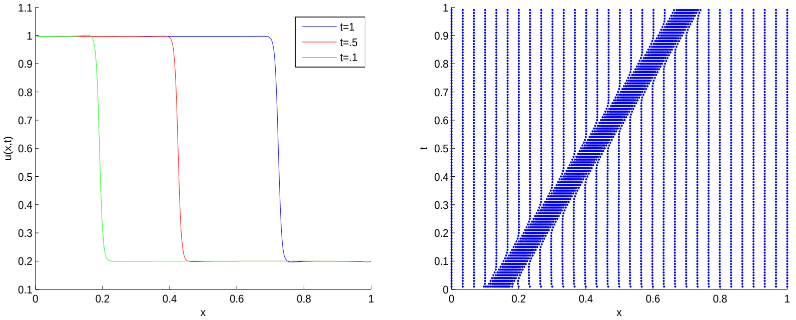
$$u(x, t) = \frac{2\pi\nu \sum_{i=1}^{\infty} iA_i \sin(i\pi x) exp(-i^2\pi^2\nu t)}{A_0 + \sum_{i=1}^{\infty} A_i \cos(i\pi x) exp(-i^2\pi^2\nu t)}, \tag{12}$$

with the Fourier coefficients



(a) (b)

Figure 1: The numerical solution and corresponding nodes trajectories for $N = 50$ and $Re = 100$



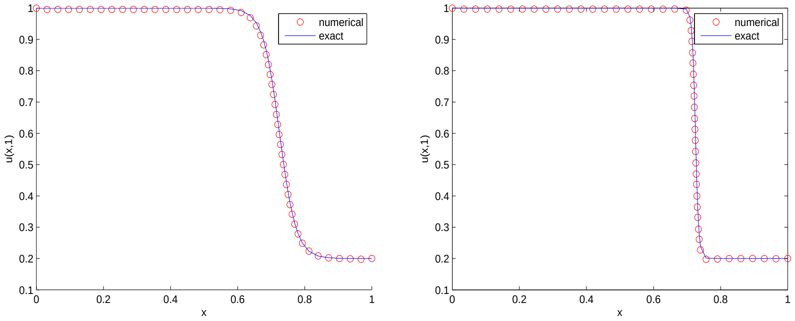
(a) (b)

Figure 2: The numerical solution and corresponding nodes trajectories for $N = 50$ and $Re = 500$

$$A_0 = \int_0^1 \exp\left\{-(2\pi\nu)^{-1}(1-\cos(\pi x)\right\} dx, \tag{13}$$

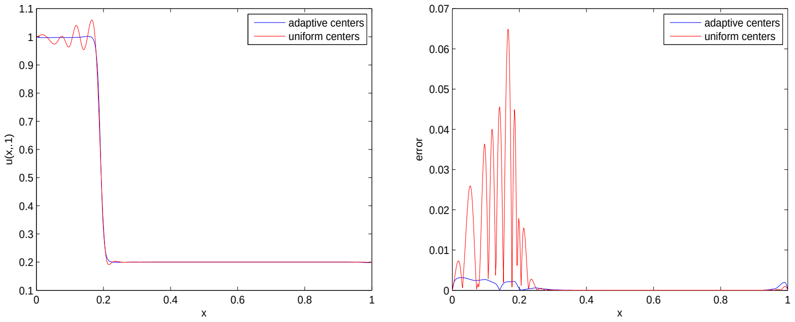
$$A_i = 2\int_0^1 \exp\left\{-(2\pi\nu)^{-1}(1-\cos(\pi x))\right\} \cos(i\pi x)dx, \quad i \geqslant 1. \tag{14}$$

In this example, $N = 50$ nodes are used. The computation are performed for a final time $t = 3$. The numerical solution at $t = .01, t = .1, t = 1, t = 2$ and $t = 3$ for $Re = 100$ and $Re = 500$ are shown in Figures 6.a and 7.a respectively. Initial condition in Example 2 does not have rapid variation, but the variation of the solution increases with time. The variation increases until a time t_0 less than $t = .75$. After this time, the variation of the solution decreases. Nodes trajectories also have such behavior. The nodes trajectories are shown in Figures 6.b and 7.b. Nodes are moved with time and concen-



(a) (b)

Figure 3: The numerical and exact solutions at $t = 1$ for (a) $Re = 100$, (b) $Re = 500$



(a) (b)

Figure 4: The numerical solutions (a) and absolute errors (b) in the case of using uniform and adaptive centers at $t = .1$ for $Re = 500$

trated in region with rapid variations. For $t < t_0$, the variation increases and nodes are concentrated in region with rapid variation. For $t \geqslant t_0$, the variation of the solution decreases with the time and hence the nodes trajectories diverge.

Figure 8 shows the numerical solution, exact solution and the absolute error at $t = 3$ when $Re = 100$, $\epsilon = 50$ and $N = 50$. We can see that, the error of the proposed method method is as small as 10^{-4}.

The numerical solution for $Re = 500$ are obtained for $\epsilon = 110$. In this case, obtaining numerical results with $N = 50$ and $Re = 500$ uniform centers is not possible as well.

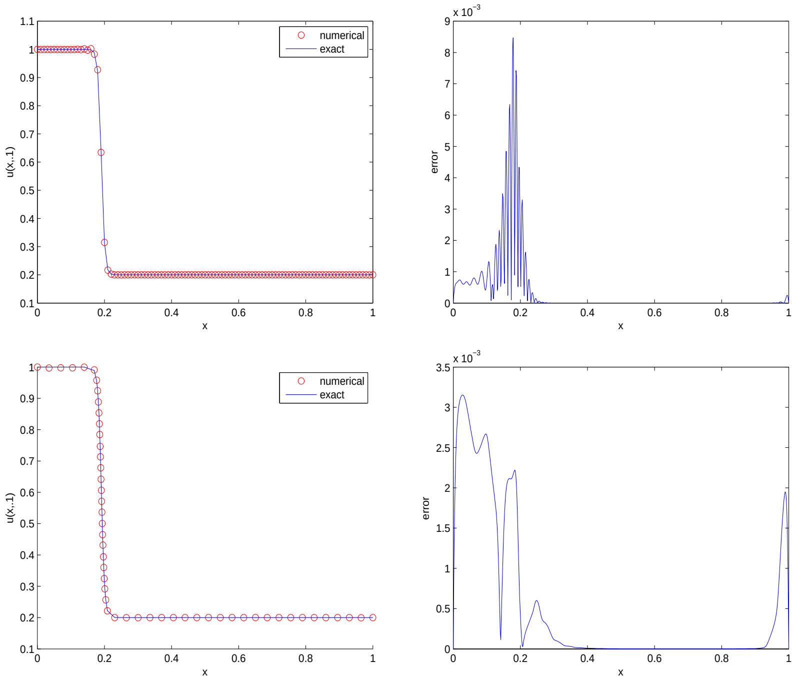
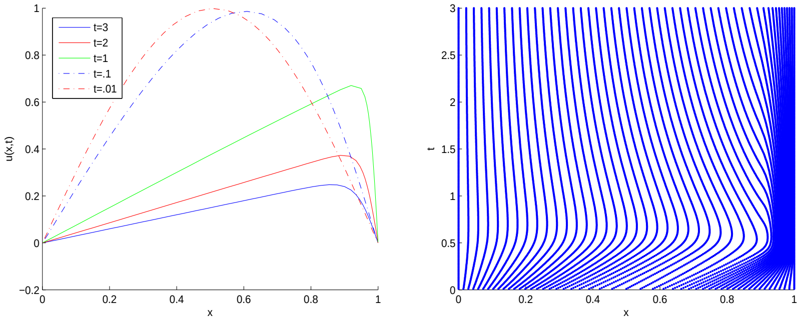


Figure 5: The numerical and exact solution and absolute error at $t = .1$ when $Re = 500$ for $N = 100$ uniform centers (top) and $N = 50$ adaptive centers (down)

4 Conclusion

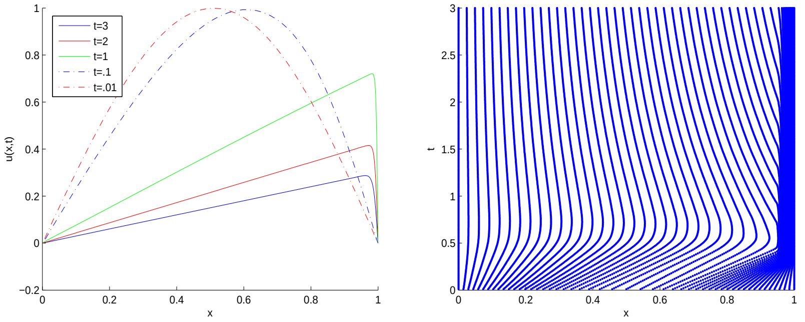
The adaptive MQ RBF method of lines has been proposed for obtaining the numerical solution of Burgers' equations. In the method of lines, centers in the domain were portioned into the interior and the boundary centers. By portioning centers and obtaining the expansion coefficients for boundary centers as a function of interior ones, the DAEs system was converted to a smaller ODEs system. The resulting ODE system was solved with *ode15s* in Matlab. Also, we have used a simple adaptive nodes generation method to enable the method for obtaining numerical solution of the problem with high gradient. In the adaptive method, the nodes moved with time and concentrated in region with rapid variation. When the gradient of solution increases the nodes become more closer in region with rapid variation. In this case numerical solution can be obtained with less number of centers in comparison with using uniform centers.

Numerical experiments have been performed for one-dimensional Burgers' equations. Numerical results show that the proposed adaptive method are preferred over fixed grid methods. For example, the adaptive method is able



(a) (b)

Figure 6: (a) The numerical solutions at different times for $Re = 100$ and (b) corresponding nodes trajectories in Example 2



(a) (b)

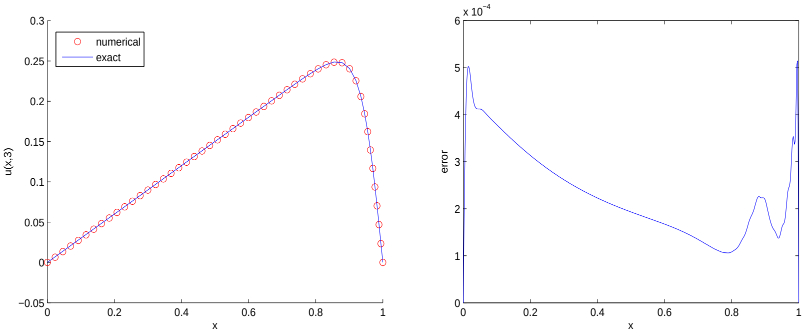
Figure 7: (a) The numerical solutions at different times for $Re = 500$ and (b) corresponding nodes trajectories in Example 2

to solve Burgers' equation for $Re = 500$ and $N = 50$ whereas the numerical solution could not be obtained for $N = 50$ uniform centers.

References

1. Barfeie, M., soheili, A.R. and Arab Ameri, M. *Application of variational mesh generation approach for selection of centers in radial basis functions collocation method*, Eng. Anal. Bound. Elem., 37 (2013) 1567-1575.

2. Boyd, J.P. and Gildersleeve, K.W. *Numerical experiments on the condition number of the interpolation matrices for radial basis functions*, Appl. Numer. Math., 61 (2011) 443-459.



(a) (b)

Figure 8: (a) The numerical and exact solutions at $t = 3$ for $Re = 100$ and (b) corresponding absolute error in Example 2

3. Brenan, K.E., Campbell, S.L. and Petzold, L. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, North-Holland, New York (1989).

4. Buhmann, M.D. *Radial Basis Functions: Theory and Implementations,* Cambridge University Press. (2003).

5. Driscoll, T.A. and Heryudono, A.R.H. *Adaptive residual subsampling methods for radial basis function interpolation and collocation problems*, Comput. Math. Appl., 53 (2007) 927-939.

6. Fasshauer, G.E. *Meshfree Approximation Methods with MATLAB*, World Scientific Publishers. (2007).

7. Fletcher, C.A.J. *Burgers' Equation: a Model for all Reasons*, Numerical Solutions of Partial Differential Equations (J. Noye, editor), New York, North-Holland Pub Co, 1982.

8. Franke, C. *Scattered data interpolation: tests of some methods*, Math. Comput., 38 (1982) 181-200.

9. Hardy, R. *Theory and applications of the multiquadric-biharmonic method. 20 years of discovery 1968-1988*, Comput. Math. Appl.,19 (1988), 163-208.

10. Hairer, E. and Wanner, G. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems,* Springer, Berlin Heidelberg New York Tokyo (1991).

11. Hon, Y.C. and Schaback, R. *On unsymmetric collocation by radial basis functions*, Appl. Math. Comput., 19 (2001) 177-186.

12. Marchi, S.D. *On optimal center locations for radial basis function interpolation: computational aspects*, Rend. Splines Radial Basis Functions and Applications, 61 (3) (2003) 343-358.

13. Marchi, S.D., Schaback, R. and Wendland, H. *Near-optimal data-independent point locations for radial basis function interpolation*, Adv. Comput. Math., 23 (2005) 317-330.

14. Min, X., Wang, R.H., Zhang, J.H. and Fang, Q. *A novel numerical scheme for solving Burgers' equation—*, Appl. Math. Comput., 217 (9) (2011), 4473-4482.

15. Rabier, P. and Rheinboldt, W. *Theoretical and Numerical Analysis of Differential-Algebraic Equations, Handbook of Numerical Analysis*, vol. VIII, edited by P.G. Ciarlet and J.L. Lions, Elsevier Science B.V. (2002).

16. Rippa, S. *An algorithm for selecting a good value for the parameter c in radial basis function interpolation*, Adv. Comput. Math., 11 (1999) 193-210.

17. Sanz-Serna, J. and Christie, I. *A simple adaptive technique for nonlinear wave problems*, J. Comput. Phys., 67 (1986) 348-360.

18. Sarra, S.A. *Adaptive radial basis function methods for time dependent partial differential equations*, Appl. Numer. Math., 54 (2005) 79-94.

19. Schaback, R. *Error estimates and condition numbers for radial basis function interpolation,* Adv. Comput. Math., 3 (1995) 251-264.

20. Soheili, A.R., Kerayechian, A. and Davoodi, N. *Adaptive numerical method for Burgers-type nonlinear equations*, Appl. Math. Comput., 219 (2012) 3486-3495.

21. Tsai, C.H., Kolibal, J. and Li, M. *The golden section search algorithm for finding a good shape parameter for meshless collocation methods*, Eng. Anal. Bound. Elem., 34 (2010) 738-740.

22. Wand, J.G. and Liu, G.R. *On the optimal shape parameters of radial basis functions used for 2-d meshless methods*, Comput. Meth. Appl. Mech. Eng., 191 (2002) 2611-2630.

23. Yoon, J. *Spectral approximation orders of radial basis function interpolation on the Sobolev space,* SIAM J. Math. Anal., 33 (2001) 946-958.

Iranian Journal of Numerical Analysis and Optimization
Vol 5, No. 1, (2015), pp 63-72

Numerical study of the nonlinear Cauchy diffusion problem and Newell-Whitehead equation via cubic B-spline quasi-interpolation

H. Aminikhah* and J. Alavi

Abstract

In this article, a numerical approximation to the solution of the Newell-Whitehead equation (NWE) and Cauchy problem of ill-posed non-linear diffusion equation have been studied. The presented scheme is obtained by using the derivative of the cubic B-spline quasi-interpolation (BSQI) to approximate the spatial derivative of the dependent variable and first order forward difference to approximate the time derivative of the dependent variable. Some numerical experiments are provided to illustrate the method. The results of numerical experiments are compared with analytical solutions. The main advantage of the scheme is that the algorithm is very simple and very easy to implement.

Keywords: B-spline quasi-interpolation; convection-diffusion equation; difference schemes.

1 Introduction

The use of spline function and its approximation plays an important role for the formation of stable numerical methods. Usually, a spline is a piecewise polynomial function defined in region, such that there exists a decomposition of D into subregions in each of which the function is a polynomial of some degree d. Also, the function, as a rule, is continuous in D, together with its derivatives of order up to $(d-1)$. As the piecewise polynomial, spline, especially B-spline, have become a fundamental tool for numerical methods to get the solution of the differential equations [9, 13, 15, 16, 26]. The numerical

*Corresponding author

Received 16 April 2014; revised 14 February 2015; accepted 17 February 2015

H. Aminikhah
Department of Applied Mathematics, School of Mathematical Sciences, University of Guilan, Rasht, Iran. e-mail: hossein.aminikhah@gmail.com

J. Alavi
Department of Applied Mathematics, School of Mathematical Sciences, University of Guilan, Rasht, Iran. email: javadealavi@gmail.com

solutions of partial differential equations by B-spline quasi-interpolation are introduced in [2, 5, 17, 20, 25].

Nonlinear equations play an important role in various filed of sciences. The world around us is nonlinear, so these kinds of equations arise naturally in a variety of models from theoretical physics, chemistry, and biology. The diffusion equation, one of these nonlinear equations, describes density dynamics in a material undergoing diffusion. It is also used to describe processes exhibiting diffusive-like behaviour, for instance the diffusion of alleles in a population in population genetics. It has also a great deal of application in different branches of sciences which have found a considerable amount of interest in recent years [1, 3, 4, 11, 14, 18, 23, 24].

Consider the nonlinear Cauchy diffusion equation as the following

$$Au = \phi(x,t), \quad x \in (a,b),\ t > 0 \tag{1}$$

with initial condition

$$u(x,0) = f(x), \quad x \in [a,b] \tag{2}$$

and boundary conditions of the form

$$u(a,t) = g_0(t), \quad u(b,t) = g_1(t), \quad t \geq 0 \tag{3}$$

$$A(u(x,t)) = \frac{\partial u}{\partial t} - \frac{\partial}{\partial x}\left((\kappa(t)u(x,t) + \omega(t))\,\frac{\partial u}{\partial x}\right) \tag{4}$$

such that $\kappa(t)u(x,t) + \omega(t)$ is positive [3, 11, 14, 23], a, b are constants, $g_0(t), g_1(t), \kappa(t), \omega(t), f(x)$ and $\phi(x,t)$ are known functions and $\phi(x,t)$ be a smooth function.

The Newell-Whitehead equation models the interaction of the effect of the diffusion term with the nonlinear effect of the reaction term. For instance an equation to describe nearly 1D traveling-wave patterns is put forward in the form of a dispersive generalization of the Newell-Whitehead equation. The Newell-Whitehead equation is written as:

$$\upsilon_t = \upsilon_{xx} + \alpha\upsilon + \beta\upsilon^n, \quad x \in [a,b], \quad t \geq 0 \tag{5}$$

where α, β are arbitrary constants, n is a positive integer and subscripts x and t denote differentiation.

Initial and boundary conditions are

$$\upsilon(x,0) = f_1(x), \quad x \in [a,b] \tag{6}$$

$$\upsilon(a,t) = g_2(t), \quad \upsilon(b,t) = g_3(t), \quad t \geq 0 \tag{7}$$

where $f_1(x), g_2(t), g_3(t)$ are known functions. The rest of this paper is organized as follows. In Section 2, we obtain the numerical schemes using cubic B-spline interpolation to solve the nonlinear Cauchy diffusion equation and

Newell-Whitehead equation. Some numerical examples are solved to assess the accuracy of the technique and the maximum absolute errors will be presented in Section 3.The conclusion appears in Section 4.

2 B-spline quasi-interpolant applied to the Cauchy problem and Newell-Whitehead equation

Assume that an interval $I = [a, b]$ is given, denoted by $S_d(X_n)$ the space of splines of degree d and class C^{d-1} on the uniform partition $X_n = \{x_i = a + ih, i = 0, 1, ..., n\}$ with meshlength $h = (b - a)/n$. Let a basis of $S_d(X_n)$ be $\{B_{j,d,r},\quad j = 1, 2, ..., n + d\}$ where $B_{j,d,r}$ is the jth B-spline of degree d for the knot sequence $r := (r_i)_{i=-d}^{n+d}$ where $r_{-d} = r_{-d+1} = ... = r_{-1} = a$, $r_n = r_{n+1} = ... = r_{n+d} = b$ and $r_i = x_i \quad 0 \le i \le n$. Since the cubic spline has become the most commonly used spline and we need the second order derivatives we use cubic B-spline quasi-interpolation in this paper.

From nonlinear differential equation (1) we have

$$u_t = \phi(x,t) + \kappa(t)\left(u_x^2 + uu_{xx}\right) + \omega(t)u_{xx} \tag{8}$$

and from discretizing this equation in time, we get

$$u_i^{k+1} = \tau\left(\phi(x_i,t_k) + \kappa(t_k)\left(\left((u_x)_i^k\right)^2 + u_i^k\,(u_{xx})_i^k\right) + \omega(t_k)\,(u_{xx})_i^k\right) + u_i^k \tag{9}$$

where $u_i^k, (u_x)_i^k, (u_{xx})_i^k$ are the approximation of the values $u(x,t), u_x(x,t)$, $u_{xx}(x,t)$ at $(x_i, t_k), t_k = k\tau$, and τ is the time step. For fixed k, we can get the cubic quasi-interpolation as follows [19]:

$$Q_3u^k = \sum_{j=1}^{n+3} \mu_j(u^k)B_{j,3,r}(x) \tag{10}$$

where $u^k = u(x, t_k)$ and the coefficient functionals are respectively:

$$\begin{aligned}
&\mu_1(u^k) = u_0^k,\\
&\mu_2(u^k) = \tfrac{1}{18}\left(7u_0^k + 18u_1^k - 9u_2^k + 2u_3^k\right)\\
&\mu_j(u^k) = \tfrac{1}{6}\left(-u_{j-3}^k + 8u_{j-2}^k - u_{j-1}^k\right), 3 \le \mathrm{j} \le \mathrm{n+1}\\
&\mu_{n+2}(u^k) = \tfrac{1}{18}\left(2u_{n-3}^k - 9u_{n-2}^k + 18u_{n-1}^k + 7u_n^k\right),\\
&\mu_{n+3}(u^k) = u_n^k.
\end{aligned} \tag{11}$$

Using the de Boor-Cox formula [12, 21], the cubic B-spline basis $B_{j,3,r}(x)$, and his derivatives can be computed.

For $u^k \in C^4(I)$ we have the error estimate [19] as

$$\left\| u^k - Q_3 u^k \right\|_\infty = O\left(h^4\right) \tag{12}$$

For approximate $(u_x)_i^k, (u_{xx})_i^k$ by derivatives of the cubic B-spline quasi-interpolant (10) up to the order h^3 we can evaluate the value of u^k at x_i by:

$$(Q_3 u_i^k)' = \sum_{j=1}^{n+3} \mu_j(u^k) B_j'(x_i), \;\; (Q_3 u_i^k)'' = \sum_{j=1}^{n+3} \mu_j(u^k) B_j''(x_i). \tag{13}$$

We set

$$\begin{aligned} &U^k = (u_0^k, u_1^k, \ldots, u_n^k)^T \;,\; U_x^k = ((u_0^k)', (u_1^k)', \ldots, (u_n^k)'), \\ &U_{xx}^k = ((u_0^k)'', (u_1^k)'', \ldots, (u_n^k)''), \end{aligned} \tag{14}$$

where

$$(u_i^k)' = (Q_3 u_i^k)' \;,\; (u_i^k)'' = (Q_3 u_i^k)'' \;,\quad i = 0, 1, \ldots, n. \tag{15}$$

By (15) we obtain

$$U_x^k = \frac{1}{h} D_1 U^k, \quad U_{xx}^k = \frac{1}{h^2} D_2 U^k \tag{16}$$

where $D_1,\ D_2 \in \mathbb{R}^{(n+1)\times(n+1)}$ are obtain as follows:

$$D_1 = \begin{bmatrix} -11/6 & 3 & -3/2 & 1/3 & 0 & 0 & \ldots & 0 & 0 \\ -1/3 & -1/2 & 1 & -1/6 & 0 & 0 & \ldots & 0 & 0 \\ 1/12 & -2/3 & 0 & 2/3 & -1/12 & 0 & \ldots & 0 & 0 \\ 0 & 1/12 & -2/3 & 0 & 2/3 & -1/12 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & 1/12 & -2/3 & 0 & 2/3 & -1/12 & 0 \\ 0 & 0 & \ldots & 0 & 1/12 & -2/3 & 0 & 2/3 & -1/12 \\ 0 & 0 & \ldots & 0 & 0 & 1/6 & -1 & 1/2 & 1/3 \\ 0 & 0 & \ldots & 0 & 0 & -1/3 & 3/2 & -3 & 11/6 \end{bmatrix}$$

$$D_2=\begin{bmatrix} 2 & -5 & 4 & -1 & 0 & 0 & \dots & 0 & 0\\ 1 & -2 & 1 & 0 & 0 & 0 & \dots & 0 & 0\\ -1/6 & 5/3 & -3 & 5/3 & -1/6 & 0 & \dots & 0 & 0\\ 0 & -1/6 & 5/3 & -3 & 5/3 & -1/6 & \dots & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \dots & -1/6 & 5/3 & -3 & 5/3 & -1/6 & 0\\ 0 & 0 & \dots & 0 & -1/6 & 5/3 & -3 & 5/3 & -1/6\\ 0 & 0 & \dots & 0 & 0 & 0 & 1 & -2 & 1\\ 0 & 0 & \dots & 0 & 0 & -1 & 4 & -5 & 2 \end{bmatrix}$$

From the initial conditions (2) and boundary conditions (3), we can compute the numerical solution of (1) step by step using the scheme (9) and formulas (16). For implementation of this method from (2) we have $U^0=(f(x_0),f(x_1),...,f(x_n))^T$ and from (16), (9) and (3) the following algorithm is obtained

```
$U^0 \leftarrow (f(x_0), f(x_1), ..., f(x_n))^T$;
for $k = 0, 1, ..., m$ do
    ${U_x}^k \leftarrow \frac{1}{h} D_1 U^k$;
    ${U_{xx}}^k \leftarrow \frac{1}{h^2} D_2 U^k$;
    $u_0^{k+1} \leftarrow g_0(t_{k+1})$;
    for $i = 1, 2, ..., n-1$ do
        $u_i^{k+1} \leftarrow \tau\left(f(x_i,t_k)+k(t_k)\left(\left(\left({U_x}^k\right)_i\right)^2+u_i^k\left({U_{xx}}^k\right)_i\right)\right)$
                $+\tau w(t_k)\left({U_{xx}}^k\right)_i+u_i^k$;
    end
    $u_n^{k+1} \leftarrow g_1(t_{k+1})$;
    $U^{k+1} \leftarrow \left(u_0^{k+1}, u_1^{k+1}, u_2^{k+1}, ..., u_{n-1}^{k+1}, u_n^{k+1}\right)$;
end.
```

Considering a maximum time like T that $0 \le t \le T$ we have $m = T/\tau$.

Similarly from discretizing the Newell-Whitehead equation (5), we get

$$\upsilon_i^{k+1}=\tau\left((\upsilon_{xx})_i^k+\alpha\upsilon_i^k+\beta\left(\upsilon_i^k\right)^n\right)+\upsilon_i^k \tag{17}$$

where $\upsilon_i^k,(\upsilon_{xx})_i^k$ are the approximation of the values $\upsilon(x,t),\upsilon_{xx}(x,t)$ at $(x_i,t_k),t_k=k\tau$, and τ is the time step. For approximation of $(\upsilon_{xx})_i^k$, in relations (10), (11) and (13)-(16) we set $\upsilon^k=\upsilon(x,t_k)$ and replacing $\upsilon_i^k,V,V_{xx}^k,i=0,1,...,n$ respectively. Then from the initial conditions (6) and boundary conditions (7), we can compute the numerical solution of (5) step by step.

3 Numerical examples

In this section, two examples of the nonlinear Cauchy diffusion equation and Newell-Whitehead equation are considered and will be solved by B-spline quasi-interpolation method. To show the accuracy of the present method for our examples in comparison with the exact solutions, the amounts of errors is given in some mesh points and we report error norm which is defined by

$$|e|_1 = \frac{1}{n}\sum_{i=1}^{n-1} \frac{\left|u_i^{\text{exact}} - u_i^{\text{numerical}}\right|}{\left|u_i^{\text{exact}}\right|} \tag{18}$$

For the computational work we select the following examples from [7, 8, 10, 22].

Example 1. Let us consider the following nonlinear differential equation

$$\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}\left(\left(\frac{1}{6}e^{-t}u + (t+5)\,e^{-t}\right)\frac{\partial u}{\partial x}\right) = -\frac{7}{3}t - 9,\ (x,t) \in [0,1]\times[0,1] \tag{19}$$

which has the exact solution $u(x,t) = x^2e^t + t$. In (19) $\phi(x,t) = -\frac{7}{3}t - 9, \kappa(t) = \frac{1}{6}e^{-t}, \omega(t) = (t+5)e^{-t}$. In Table1, relative errors at different time levels are compared with the relative errors obtained by Zakeri et al. in [10]. In Figures 1 and 2 exact and numerical solutions are depicted.

Example 2. Relative errors at different time levels are compared with the relative errors obtained by Nourazar et al. [8]. for Eq. (5) with $\alpha = 3, \beta = -4, n = 3, a = 0, b = 1$ and $t = 1$ in Table 2. The exact solution of this example is $\upsilon(x,t) = \sqrt{\frac{3}{4}}\frac{e^{\sqrt{6}x}}{e^{\sqrt{6}x} + e^{\left(\frac{\sqrt{6}}{2}x - \frac{9}{2}t\right)}}$. The graph of the exact and numerical solution, are shown in Figures 3 and 4.

Table 1: Comparison of relative errors obtained from proposed method and method in [10].

x	Relative errors of proposed method			Relative errors obtained in [10]		
	$t = 0.25$	$t = 0.50$	$t = 0.75$	$t = 0.25$	$t = 0.50$	$t = 0.75$
0.2	8.0999e-08	6.6631e-08	6.9815e-08	2.7500e-08	4.7700e-09	3.4500e-08
0.4	1.0121e-07	9.3251e-08	1.0112e-07	4.8100e-07	4.3300e-07	2.8500e-07
0.6	8.1254e-08	8.1761e-08	9.1375e-08	2.2800e-06	2.2700e-06	1.9100e-06
0.8	4.4684e-08	4.7547e-08	5.4249e-08	6.7100e-06	6.7700e-06	5.8700e-06
$\lvert e\rvert_1$	6.4079e-08	5.9977e-08	6.5621e-08	–	–	–

From the test examples, we can say that the BSQI scheme is feasible and the accuracy is better than the multi-quadric quasi-interpolation (MQQI) method [6]. Moreover, MQQI method has very close relation to the shape

Table 2: Comparison of errors of Example 2 with the errors obtained in [8].($h = 0.02, \tau = 0.0001$)

x	Relative errors of proposed method				Relative errors obtained in [8]		
	$t = 0.1$	$t = 0.15$	$t = 0.2$	$t = 1$	$t = 0.1$	$t = 0.15$	$t = 0.2$
0.2	4.7533e-06	6.0414e-06	6.6041e-06	5.2440e-07	4.9987e-06	5.6384e-05	3.1193e-04
0.4	6.8110e-06	8.4592e-06	9.1422e-06	7.1097e-07	6.3997e-06	6.8760e-05	3.6460e-04
0.8	4.7680e-06	5.4354e-06	5.6195e-06	4.0217e-07	3.6819e-06	3.7324e-05	1.8700e-04
$\|e\|_1$	4.8486e-06	5.8747e-06	6.2722e-06	4.7741e-07	–	–	–

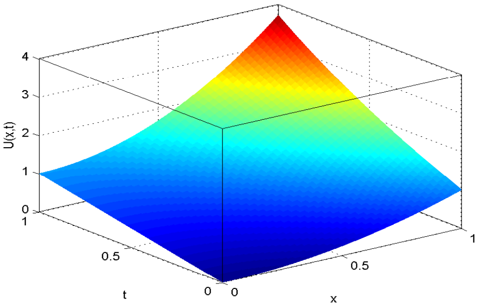


Figure 1: The exact solution of Example 1 for $h = 0.02, \tau = 0.00001$

parameter c in MQ. In fact, the choice of the shape parameter is still a pendent question. Furthermore, the MQQI is required to calculate derivatives of MQ quasi interpolant once for all, which is not easy to compute when h is small. Although the accuracy of BSQI is not better than that of other methods, we know that, at each time step, the complexity of BSQI is lower than theirs. The proposed method is an acceptable and valid scheme. Moreover, it can be implemented very easily.

4 Conclusions

In this article, we have applied the cubic B-spline quasi-interpolation method for solving the nonlinear Cauchy diffusion problem and Newell-Whitehead equation. The results have been compared with the exact solutions and demonstrated the good performance of the schemes. This method offers several advantages in reducing computational costs. On the other hand, this method is very simple to apply and to make an algorithm. Thus, this method may be reckoned as a simple and accurate solver for PDEs and it is worthy to note that this method can be utilized as an accurate algorithm to solve linear and nonlinear functional equations arising in physics and other

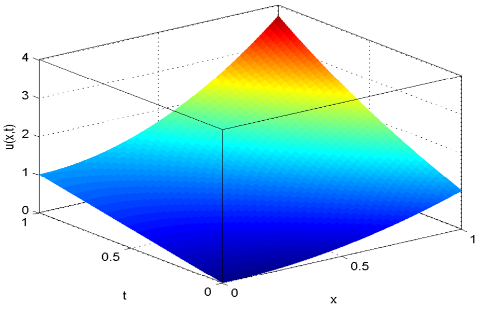


Figure 2: The numerical results of Example 1 for $h = 0.02, \tau = 0.00001$

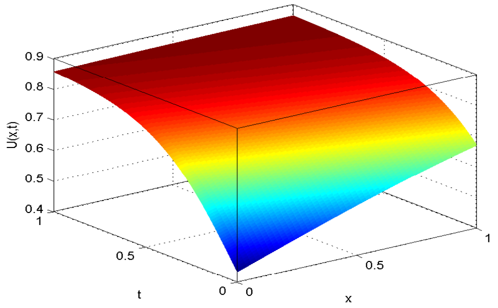


Figure 3: The exact solution of Example 2 for $h = 0.02, \tau = 0.0001$

fields of applied mathematics. The computations associated with the examples in this article were performed using MATLAB R2013a.

References

1. Burgers, J. M. *The Nonlinear Diffusion Equation*, Reidel Publishing Company, 1973.

2. Chen, R. and Wu, Z. *Solving partial differential equation by using multiquadric quasi-interpolation*, Applied Mathematics and Computation, 186 (2007) 1502-1510.

3. Chen, Z., Huan, G. and Ma, Y. *Computational Methods for Multiphase Flows in Porous Media*, Computational Science and Engineering, Society for Industrial and Applied Mathematics, Philadelphia, Pa, USA, 2006.

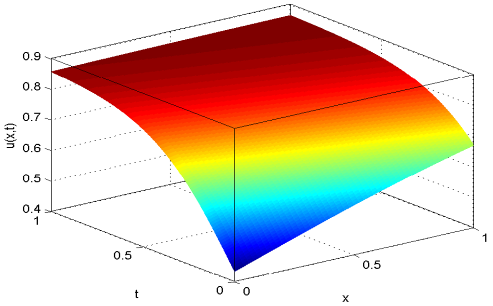


Figure 4: The numerical results of Example 2 for $h = 0.02, \tau = 0.0001$

4. Cunningham, R. E. and Williams, R.J.J. *Diffusion in Gasses and Porous-Media*, Plenum Press, New York, NY, USA, 1980.

5. Dosti, M. and Nazemi, A.R. *Solving one-dimensional hyperbolic telegraph equation using cubic B-spline quasi-interpolation*, International Journal of Mathematical and Computer Sciences, 7 (2011) 57-62.

6. El-Hawary, H.M. and Mahmoud, S.M. *Spline collocation methods for solving delay-differential equations*, Applied Mathematics and Computation, 146 (2003) 359372.

7. Ezzati, R. and Shakibi, K. *Using Adomian's decomposition and multiquadric quasi-interpolation methods for solving newell-whitehead equation*, Procedia Computer Science, 3 (2011) 1043-1048.

8. Farin, G. *Curves and Surfaces for CAGD*, fifth ed., Morgan Kaufmann, San Francisco, 2001.

9. Goh, J., Ahmad Abd. Majid, Ahmad Izani Md. Ismail, A quartic B-spline for second-order singular boundary value problems, Computers and Mathematics with Applications, 64 (2012), 115120.

10. Jäger, W., Rannacher, R. and Warnatz, J. *Reactive Flows, Diffusion and Transport*, Springer, Berlin, Germany, 2007.

11. Kadalbajoo, M.K., Tripathi, L.P. and Kumar, A. *A cubic B-spline collocation method for a numerical solution of the generalized Black-Scholes equation*, Mathematical and Computer Modelling, 55 (2012) 14831505.

12. Kärger, J. and Heitjans, P. *Diffusion Condesed Matter*, Springer, Berlin, Germany, 2005.

13. Khuri, S.A. and Sayfy, A. *A spline collocation approach for a generalized parabolic problem subject to non-classical conditions*, Applied Mathematics and Computation, 218 (2012) 91879196.

14. Kolokoltsov, V. *Semi Classical Analysis for Diffusion and Stochastic Processes*, Springer, Berlin, Germany, 2000.

15. Matinfar, M., Eslami, M. and Saeidy, M. *An efficient method for Cauchy problem of ill-posed nonlinear diffusion equation*, International Journal of Numerical Methods for Heat and Fluid Flow, 23 (2013) 427-435.

16. Mehrer, H. *Diffusion Solids Fundamentals Methods Materials Diffusion Controlled Processes*, Springer, Berlin, Germany, 2007.

17. Mittal, R.C. and Jain, R.K. *Numerical solutions of nonlinear Burgers' equation with modified cubic B-splines collocation method*, Applied Mathematics and Computation, 218 (2012) 7839-7855.

18. Nourazar, S.S., Soori, M. and Nazari-Golshan, A. *on the exact solution of Newell-Whitehead-Segel equation using the Homotopy Perturbation Method*, Australian Journal of Basic and Applied Sciences, 5 (2011) 1400-1411.

19. Sablonnire, P. *Univariate spline quasi-interpolants and applications to numerical analysis*, Rendiconti del Seminario Matematico, 63 (2005) 211-222.

20. Schumaker, L.L. *Spline Functions: Basic Theory*, third ed., Cambridge University Press, 2007.

21. V'azquez, J. L. *The Porous Medium Equation*, Oxford Mathematical Monographs, The Clarendon Press, Oxford, UK, 2007.

22. Yuab, R.G., Wanga, R.H. and Zhu, C.G. *A numerical method for solving KdV equation with multilevel B-spline quasi-interpolation*, Applicable Analysis: An International Journal, 92 (2012) 1682-1690.

23. 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, Volume 2010, Article ID 780207, 10 pages.

24. Zhu, C.G. and Kang, W.S. *Applying Cubic B-Spline Quasi-Interpolation To Solve Hyperbolic Conservation Laws*, U.P.B. Sci. Bull., Series D, 72 (2010) 49-58.

25. Zhu, C.G. and Kang, W.S. *Numerical solution of Burgers-Fisher equation by cubic B-spline quasi-interpolation*, Applied Mathematics and Computation, 216 (2010) 26792686.

26. Zhu, C.G. and Wang, R.H. *Numerical solution of Burgers' equation by cubic B-spline quasi-interpolation*, Applied Mathematics and Computation, 208 (2009) 260272.

Iranian Journal of Numerical Analysis and Optimization
Vol 5, No. 1, (2015), pp 73-83

Numerical solution of damped forced oscillator problem using Haar wavelets

I. Singh and S. Kumar*

Abstract

We present here the numerical solution of damped forced oscillator problem using Haar wavelet and compare the numerical results obtained with some well-known numerical methods such as Runge-Kutta fourth order classical and Taylor Series methods. Numerical results show that the present Haar wavelet method gives more accurate approximations than above said numerical methods.

Keywords: Haar wavelet method; Differential equation; Operational matrix; Damped forced oscillator.

1 Introduction

During the last few decades considerable efforts have been made using wavelet, towards the development of computational methods to solve numerically linear differential equations encountered in various fields of science and engineering. Wavelet analysis is a new branch of applied science. Wavelet methods are applied to find the numerical solution of problems related to science and engineering. In the last recent years, wavelet methods have been attracted the great interest of researchers of physical and mathematical sciences and many research papers were published in these fields. Recently, many researchers have used Haar and Daubechies wavelets to find the numerical solution of differential and integral equations. Before, the discovery of Haar wavelet, Daubechies wavelets were used in many published research papers for numerical solution of differential and integral equations.

In 1910, Alfred Haar [4] discovered a new wavelet known as Haar wavelet.

*Corresponding author

Received 13 January 2014; revised 27 January 2015; accepted 17 February 2015

I. Singh
Department of Mathematics, Dr. B. R. Ambedkar National Institute of Technology, Jalandhar, Punjab-144011, India. e-mail: indermaths12@yahoo.com

S. Kumar
Department of Mathematics, Dr. B. R. Ambedkar National Institute of Technology, Jalandhar, Punjab-144011, India. email: sheoks53@gmail.com

Among all wavelet families, Haar wavelet is most simple, accurate and efficient. It attracted, the interest of many researcher in the field of engineering and science. Haar wavelet has been used in wide variety of numerical methods developed for numerical solutions of differential and integral equations. Here, we present a survey of such methods for differential and integral equations. Chen and Hsiao [3] applied Haar wavelet method for solving lumped and distributed-parameter systems. Hsiao [6] used wavelet approach to time-varying functional differential equations. Razzaghi and Ordokhani [15] used Haar functions for variational problems. Ohkita and Kobayashi [13] applied rationalized Haar functions to solve linear differential equations. Cattani [2] suggested use of Haar wavelet splines for numerical solution of differential equations. Lepik [8, 9, 10, 11, 12] used Haar wavelets for solving differential and integral equations. Sunmonu [18] presented wavelet solution for second order differential equations with maple. Hariharan and Kannan [5] presented an overview of Haar wavelet method for solving differential and integral equations. Kouchi et al. [7] presented numerical solution of homogeneous and inhomogeneous harmonic differential equation with Haar wavelet. In [16], Quasilinearization technique and Haar wavelet operational matrix method both are applied to find the numerical solution of fractional order nonlinear oscillation equations. Also, Solutions of fractional order force-free and forced Duffing-Van der Pol oscillator and higher order fractional Duffing equation on large intervals are presented in [16].

In Section 2, we discussed damped forced oscillator. Haar wavelet method is presented in Section 3. Function approximation is presented in Section 4. In Section 5, we present convergence analysis of Haar wavelet method. In Section 6, the solution by Haar wavelet method is presented. In Section 7, Runge-Kutta method for second order differential equation is presented. Taylor-Series method is presented in Section 8. Comparison of numerical solutions is presented in Section 9 and in Section 10, conclusion is given.

2 Damped forced oscillation

Oscillation means repeated motion of a particle or a body, when displaced from its equilibrium position. The classifications of oscillating systems are presented in Thomsen [19] and in Bhat Rama and Dukkipati [14]. The mechanism that results in dissipation of the energy of an oscillator is called damping. In mechanical oscillator, the damping may be due to (1) Viscous drag (2) Friction and (3) Structure. An oscillator to which a continuous excitation is provided by some external agency is called forced oscillator.

Suppose a mass M attached to the end of a spring of stiffness constant S. One end of the spring is attached to a rigid support. Let the driven force acting on the system be $F(t)$. At any instant, the system will operate under the influence of the following forces:

(a) Restoring force, $F1 = -Sx$ where x is the displacement of the mass from the equilibrium position,
(b) Damping force, $F2 = -rdx/dt$, where r is damping constant,
(c) Driven force, $F3 = F(t)$.
The negative sign in the first two expression implies that both the restoring as well as damping forces opposes the displacement. By Newton second law of motion, we have

$$M\frac{d^2x}{dt^2} = -Sx - r\frac{dx}{dt} + F(t). \tag{1}$$

In this paper, we take special choice $F(t) = 2(1-sint)$, $M = 2kg$, $S = 1N/m$, $r = 0.3Ns/m$ and $x(0) = x'(0) = 0$ as initial conditions, see Simmons [17]. The exact solution of equation (1) by using classical method is:

$$x(t) = e^{-0.075t}(C_1cos(0.703118t) + C_2sin(0.703118t)) + 2 + \frac{200}{109}sin(t) + \frac{60}{109}cos(t). \tag{2}$$

applying initial conditions, we have $C_1 = -\frac{278}{109}$ and $C_2 = -\frac{110425000}{38319931}$.

3 Haar wavelet method

The Haar functions are an orthogonal family of switched rectangular waveforms where amplitudes can differ from one function to another. They are defined in the interval $[0, 1]$.

$$h_i(t) = \begin{cases} 1, & \alpha \le t < \beta, \\ -1, & \beta \le t < \gamma, \\ 0, & otherwise. \end{cases} \tag{3}$$

where $\alpha = \frac{k}{m}$, $\beta = \frac{k+0.5}{m}$ and $\gamma = \frac{k+1}{m}$.

Integer $m = 2^j$, $(j = 0, 1, 2, 3, 4, ...J)$ indicates the level of the wavelet.
$k = 0, 1, 2, 3, ..., m-1$ is the translation parameter.Maximal level of resolution is J. The index i is calculated according the formula $i = m+k+1$.In the case of minimal values, $m = 1$, $k = 0$ we have $i = 2$. The maximal value of i is $i = 2M$. where $M = 2^J$. It is assumed that the value $i = 1$, corresponding to the scaling function in $[0, 1]$.

$$h_1(t) = \begin{cases} 1, & 0 \le t \le 1, \\ 0, & \text{otherwise}. \end{cases} \tag{4}$$

Let us define the collocation points $t_l = \frac{(l-0.5)}{2M}$, where $l = 1, 2, 3, ..., 2M$ and discredits the Haar function $h_i(t)$.

In the collocation points, the fist four Haar functions can be expressed as follows:
$h_1(t) = [1, 1, 1, 1], h_2(t) = [1, 1, -1, -1], h_3(t) = [1, -1, 0, 0], h_4(t) = [0, 0, 1, -1]$.

We introduce the notation:

$$H_4(t) = [h_1(t), h_2(t), h_3(t), h_4(t)]^T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}. \tag{5}$$

Here $H_4(t)$ is called Haar coefficient matrix. It is a square matrix of order 4. The operational matrix of integration P, which is a $2M$ square matrix, is defined by the relations:

$$P_{i,1}(t) = \int_0^{t_l} h_i(t)dt. \tag{6}$$

$$P_{i,n+1}(t) = \int_0^{t_l} P_{i,n}(t)dt, \tag{7}$$

where $n = 1, 2, 3, 4....$

These integrals can be evaluated using equation (3) and first four of them are given below:-

$$P_{i,1}(t) = \begin{cases} t - \alpha, & t\epsilon[\alpha, \beta), \\ \gamma - t, & t\epsilon[\beta, \gamma), \\ 0, & elsewhere, \end{cases} \tag{8}$$

$$P_{i,2}(t) = \begin{cases} \frac{1}{2}(t-\alpha)^2, & t\epsilon[\alpha, \beta), \\ \frac{1}{4m^2} - \frac{1}{2}(\gamma - t)^2, & t\epsilon[\beta, \gamma), \\ \frac{1}{4m^2}, & t\epsilon[\gamma, 1), \\ 0, & elsewhere, \end{cases} \tag{9}$$

$$P_{i,3}(t) = \begin{cases} \frac{1}{6}(t-\alpha)^3, & t\epsilon[\alpha, \beta), \\ \frac{1}{4m^2}(t-\beta) - \frac{1}{6}(\gamma - t)^3, & t\epsilon[\beta, \gamma), \\ \frac{1}{4m^2}(t-\beta), & t\epsilon[\gamma, 1), \\ 0, & elsewhere, \end{cases} \tag{10}$$

$$P_{i,4}(t) = \begin{cases} \frac{1}{24}(t-\alpha)^4, & t\epsilon[\alpha, \beta), \\ \frac{1}{8m^2}(t-\beta)^2 - \frac{1}{24}(\gamma - t)^4 + \frac{1}{192m^4}, & t\epsilon[\beta, \gamma), \\ \frac{1}{8m^2}(t-\beta)^2 + \frac{1}{192m^4}, & t\epsilon[\gamma, 1), \\ 0, & elsewhere. \end{cases} \tag{11}$$

4 Function approximation

Any square integrable function $x(t)$ in the interval $[0,1]$ can be expanded by a Haar series of infinite terms:

$$x(t)=\sum_{i=1}^{\infty} a_i h_i(t), i\epsilon\{0\}\cup N \tag{12}$$

where the Haar coefficients a_i are determined as:

$$a_0=\int_0^1 x(t)h_0(t)dt \tag{13}$$

$$a_n=2^j\int_0^1 x(t)h_i(t)dt \tag{14}$$

where $i=2^j+k$, $j{\geq}0$ and $0{\leq}k<2^j$, $x\epsilon[0,1]$ such that the following integral square error ε is minimized:

$$\varepsilon=\int_0^1 [x(t)-\sum_{i=0}^{m-1} a_i h_i(t)]^2 dt \tag{15}$$

where $m=2^j$ and $j\epsilon\{0\}{\cup}N$.

Usually the series expansion of (12) contains infinite terms for smooth $x(t)$. if $x(t)$ is piecewise constant by itself or may be approximated as piecewise constant during each subinterval, then $x(t)$ will be terminated at finite m terms. This means

$$x(t)=\sum_{i=0}^{m-1} a_i h_i(t)={a_m}^T h_m(t) \tag{16}$$

where the coefficients ${a_m}^T$ and the Haar function vector $h_m(t)$ are defined as:

$${a_m}^T=[a_0,a_1,a_2,..........,a_{m-1}]$$

and

$$h_m(t)=[h_0(t),h_1(t),h_2(t),..........,h_{m-1}(t)]^T.$$

5 Convergence analysis of Haar wavelet method

Consider a differentiable function $x(t)$ with

$$|x(t)|\leqslant K_0, \tag{17}$$

such that

$$|x'(t)| \leqslant K_0, \tag{18}$$

for all $t\varepsilon(0,1)$. Where $K_0 > 0$ is a positive constant. Haar wavelet approximation for the function $x(t)$ is given by:

$$x_M(t) = \sum_{i=1}^{2M} a_i h_i(t) \tag{19}$$

The square of error norm for wavelet approximation in [1] is given by:

$$\|x(t) - x_M(t)\| \leq \frac{{K_0}^2}{3} . \frac{1}{(2M)^2} \tag{20}$$

Therefore,

$$\|x(t) - x_M(t)\| \leq O(\frac{1}{M}) \tag{21}$$

This means that error bound depends on level of resolution of Haar wavelets that is, error bound is inversely proportional to level of resolution of Haar wavelets. Therefore, when we increase the value of M, it yields the sure convergence of Haar wavelet approximation.

6 Method of solution

Consider the damped forced oscillatory equation (1). Assume that

$$x''(t) = \sum_{i=1}^{2M} a_i h_i(t). \tag{22}$$

Integrating twice with respect to t from 0 to t, we get

$$x'(t) = x'(0) + \sum_{i=1}^{2M} a_i P_{1,i}(t), \tag{23}$$

$$x(t) = x(0) + \sum_{i=1}^{2M} a_i P_{2,i}(t). \tag{24}$$

Apply initial conditions and substitute the values of $x''(t)$, $x'(t)$ and $x(t)$ in (1), we get,

$$\sum_{i=1}^{2M} a_i [M h_i(t) + r P_{1,i}(t) + S P_{2,i}(t)] = F(t) \tag{25}$$

where r, S, F and M are same as defined in Section 2. From here, wavelet coefficients a_i are calculated and solution $x(t)$ of equation (1) is obtained.

7 Runge-Kutta method of fourth order

Runge-Kutta method is famous numerical method for solving ordinary differential equations. Consider the second order ordinary differential equation

$$\frac{d^2y}{dx^2} = \phi(x, y, \frac{dy}{dx}) \tag{26}$$

By substituting $\frac{dy}{dx} = z$, it can reduced to two first order simultaneous differential equations

$$\frac{dy}{dx} = z = f(x, y, z) \tag{27}$$

and

$$\frac{dz}{dx} = \phi(x, y, z) \tag{28}$$

with initial conditions $y(x_0) = y_0$ and $z(x_0) = z_0$. Starting at (x_0, y_0, z_0) and taking the step-sizes for x, y, z to be h, k, l respectively, the Runge-Kutta method gives,

$$k_1 = hf(x_0, y_0, z_0), \tag{29}$$

$$l_1 = h\phi(x_0, y_0, z_0), \tag{30}$$

$$k_2 = hf(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1, z_0 + \frac{1}{2}l_1), \tag{31}$$

$$l_2 = h\phi(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1, z_0 + \frac{1}{2}l_1), \tag{32}$$

$$k_3 = hf(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2, z_0 + \frac{1}{2}l_2), \tag{33}$$

$$l_3 = h\phi(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2, z_0 + \frac{1}{2}l_2), \tag{34}$$

$$k_4 = hf(x_0 + h, y_0 + k_3, z_0 + l_3), \tag{35}$$

$$l_4 = h\phi(x_0 + h, y_0 + k_3, z_0 + l_3). \tag{36}$$

Using above relations, we have

$$y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), \tag{37}$$

and

$$z_1 = z_0 + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4), \tag{38}$$

To compute y_2 and z_2, we simply replace x_0, y_0, z_0 by x_1, y_1, z_1 in the above relations. Similarly by using above relations we compute x_2, y_2, z_2, x_3, y_3, z_3,..........so on.

8 Taylor-series method

Consider equations (26), (27) and (28). If h be the step-size, $y_1 = y(x_0 + h)$ and $z_1 = z(x_0 + h)$. Then, Taylor's algorithm for (26) and (27) gives

$$y_1 = y_0 + hy_0{}' + \frac{h^2}{2!}y_0{}'' + \frac{h^3}{3!}y_0{}''' + ... \tag{39}$$

$$z_1 = z_0 + hz_0{}' + \frac{h^2}{2!}z_0{}'' + \frac{h^3}{3!}z_0{}''' + ... \tag{40}$$

Differentiating (26) and (27) successively, we get y'', z'', etc. So the values $y_0{}', y_0{}'', y_0{}''', ...$ and $z_0{}', z_0{}'', z_0{}''', ...$ are known. Substituting these values in above equations, we get y_1, z_1. Similarly, we have the algorithms

$$y_2 = y_1 + hy_1{}' + \frac{h^2}{2!}y_1{}'' + \frac{h^3}{3!}y_1{}''' + ... \tag{41}$$

$$z_2 = z_1 + hz_1{}' + \frac{h^2}{2!}z_1{}'' + \frac{h^3}{3!}z_1{}''' + ... \tag{42}$$

Since y_1, z_1 are known. we can calculate $y_1{}', y_1{}'', y_1{}''', ...$ and $z_1{}', z_1{}'', z_1{}''', ...$ Substituting these values in above equations, we get y_2, z_2. Proceeding in this way, we can calculate the other values of y and z step by step.

9 Comparison of numerical solutions

In this section, we compare the results of the present Haar wavelet method with two other numerical methods for the damped forced oscillatory problem. In order to verify the efficiency of Haar wavelet method in comparison to exact solution, Runge-kutta fourth order classical method and Taylor series method have been selected. For the Runge-kutta method, the step-size is 1/32. For Taylor's series method, step size is 1/32 and 7 terms are involved. Table-1 shows the numerical results from different numerical methods. Table-2 shows the errors arising from different numerical methods mentioned above. Further, graph in Figure 1 shows the comparison of graphical solution with the exact solution, obtained for $J = 3$ by (i) Haar wavelet method (ii) Runge-Kutta fourth order classical method and (iii) Taylor series method.

Table 1: Results from different numerical methods

$x(l)/32$	Exact solution	Haar wavelet	Runge-Kutta	Taylor series
1	0.0004824193	0.0004707036	0.0004630874	0.0004671912
3	0.0042356457	0.0042007228	0.0040510208	0.0040991982
5	0.0114690029	0.0114111581	0.0109524412	0.0110920238
7	0.0218950125	0.0218146369	0.0208783000	0.0211608344
9	0.0352250351	0.0351226225	0.0335339697	0.0340203049
11	0.0511707741	0.0510469174	0.0486211919	0.0493861034
13	0.0694457517	0.0693011385	0.0658400073	0.0669763459
15	0.0897667509	0.0896021596	0.0848906621	0.0865130140
17	0.1118552180	0.1116715126	0.1054754806	0.1077233303
19	0.1354386196	0.1352367450	0.1273006949	0.1303410858
21	0.1602517482	0.1600327242	0.1500782254	0.1541079134
23	0.1860379714	0.1858028875	0.1732274013	0.1787745029
25	0.2125504200	0.2123004292	0.1973766160	0.2041017526
27	0.2395531092	0.2392894218	0.2213649076	0.2298618522
29	0.2668219898	0.2665458670	0.2452434581	0.2558392936
31	0.2941459241	0.2938586718	0.2687770046	0.2818318053

10 Conclusion

Here, we used three numerical methods to approximate the solutions of damped forced oscillatory differential equation, and compared the results with exact solution. From above results, it is concluded that Haar wavelet method is simple, accurate and more efficient than other well known numerical methods for damped forced oscillatory differential equation.

References

1. Babolian, E. and Shahsawaran, A. *Numerical solution of non-linear fredholm integral equations of the second kind using haar wavelets*, J. Comput. Appl. Math. 225 (2009) 87-95.

2. Cattani, C. *Haar wavelet splines*, J. Interdisciplinary Math. 4 (2001) 35-47.

Table 2: Errors from different numerical methods

$x(l)/32$	Haar wavelet	Runge-Kutta	Taylor series
1	0.0000117157	0.0000193319	0.0000152281
3	0.0000349228	0.0001846249	0.0001364475
5	0.0000578447	0.0005165617	0.0003769791
7	0.0000803755	0.0010167125	0.0007341781
9	0.0001024125	0.0016910654	0.0012047302
11	0.0001238567	0.0025495822	0.0017846707
13	0.0001446131	0.0036057444	0.0024694058
15	0.0001645913	0.0048760888	0.0032537369
17	0.0001837053	0.0063797374	0.0041318877
19	0.0002018746	0.0081379247	0.0050975338
21	0.0002190239	0.0101735228	0.0061438348
23	0.0002350839	0.0128105701	0.0072634685
25	0.0002499908	0.0151738040	0.0084486674
27	0.0002636874	0.0181882016	0.0096912570
29	0.0002761227	0.0215785317	0.0109826962
31	0.0002872523	0.0253689195	0.0123141188

3. Chen, C.F. and Hsiao, C.H. *Haar wavelet method for solving lumped and distributed-parameter systems*, IEEE Proc.: Part D, 144(1) (1997) 87-94.

4. Haar, A. *Zur theorie der orthogonalen Funktionsysteme*, Math. Annal. 69 (1910) 331-371.

5. Hariharan, G. and Kannan, K. *An overview of Haar wavelet method for solving differential and integral equation*, World Applied Sciences Journal 23(12) (2013) 1-14.

6. Hsiao, C.H. *Wavelet approach to time-varying functional differential equations*, Int. J. Computer Math. 87(3) (2008) 528-540.

7. Kouchi, M.R., Khosravi, M. and Bahmani, J. *A numerical solution of Homogengous and Inhomogeneous Harmonic Differential equation with Haar wavelet*, Int. J. Contemp. Math. Sciences 6(41) (2011) 2009-2018.

8. Lepik, Ü. *Numerical solution of differential equations using Haar wavelets*, Math. Comput. Simulat. 68 (2005) 127-143.

9. Lepik, Ü. *Application of Haar wavelet transform to solving integral and differential equation*, Proc. Estonian Acad. Sci. Phys. Math. 56(1) (2007) 28-46.

10. Lepik, Ü. *Haar wavelet method for higher order differential equations*, Int. J. MAth. Comput. 1 (2008) 84-94.

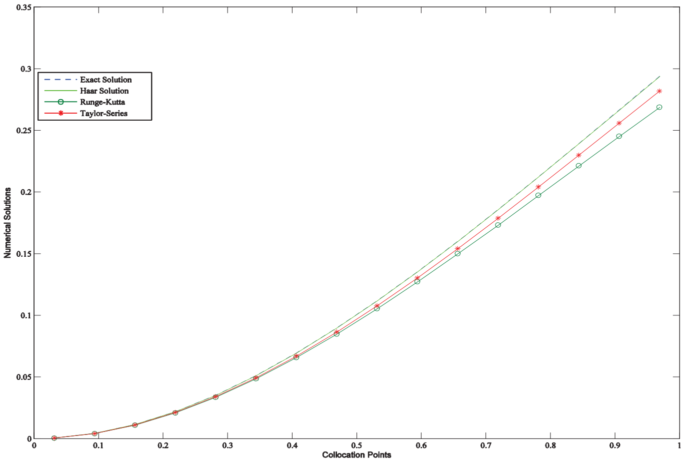


Figure 1: Comparison of graphical solution of different numerical methods.

11. Lepik, Ü. *Haar wavelet method for solving stiff differential equations*, Math. Modeling and Analysis 4 (2009) 467-489.

12. Lepik, Ü. *Solution of optimal control problems via Haar wavelets*, Int. J. Pure. Appl. Math. 55 (2009) 81-94.

13. Ohkita, M. and Kobayashi, Y. *An application of rationalized Haar functions to solution of linear differential equations*, IEEE Trans. Circuit System 9 (2003) 853-862.

14. Rama, B.B. and Dukkipati, V. *Advanced dynamics*, Alpha Science, Pangbourne, 2001.

15. Razzaghi, M. and Ordokhani,Y. *An application of rationalized Haar functions for variational problems*, Appl. MAth. Comput. 122 (2001) 353-364.

16. Saeed, U. and Rehman, M.U. *Haar wavelet operational matrix method for fractional oscillation equations*, International Journal of Mathematics and Mathematical Sciences, (2014) 1-8.

17. Simmons, G.F. *Differential equations with applications and historical notes*, Mcgraw-Hill, London. 1972.

18. Sunmonu, A. *Implementation of wavelet solution to second order differential equations with maple*, Applied Mathematical Sciences 6(127) (2012) 6311-6326.

19. Thomsen, J.J. *Vibrations and stability order and chaos*, Mc-graw-Hill, London, 1997.

Persian Translation of Abstracts

تقریب قوی برای معادلات دیفرانسیل تصادفی ایتو

مهران نامجو

دانشگاه ولی عصر رفسنجان، دانشکده علوم ریاضی، گروه ریاضی

چکیده : در این مقاله خانواده ای از روشهای رانگ–کوتا تصادفی دو مرحله ا ی نیمه ضمنی از مرتبه قوی یک با مینیمم ثابتهای خطا که برای حل معادلات دیفرانسیل تصادفی ایتو با یک فرایند وینر بکار می رود، معرفی شده است. کارایی این روش نسبت به روشهای رانگ–کوتا ایتو دو مرحله ای صریح، روش ایتو، روش مایلشتن و روشهای رانگ–کوتا استراتنویچ دو مرحله ای نیمه ضمنی و ضمنی با استفاده از نتایج عددی نشان داده شده است.

کلمات کلیدی : معادلات دیفرانسیل تصادفی؛ تقریب قوی؛ روشهای رانگ–کوتا.

یک الگوریتم دوفازی جستجوی همسایگی متغیر برای حل مسائل کنترل بهینه غیرخطی

رضا قنبری[۱] ، *عقیله حیدری*[۲] *و سعید نژاد حسین*[۳]

[۱] دانشگاه فردوسی مشهد، دانشکده علوم ریاضی، گروه ریاضی کاربردی
[۲] دانشگاه پیام نور، مشهد، گروه ریاضی کاربردی
[۳] دانشگاه پیام نور، تهران، گروه ریاضی کاربردی

چکیده : در این مقاله یک الگوریتم دوفازی به نام IVNS برای حل مسائل کنترل بهینه غیرخطی پیشنهاد شده است. در هر فاز الگوریتم پیشنهادی از روش جستجوی همسایگی متغیر (VNS) استفاده می‌کنیم که در آن از توزیع یکنواخت درگام لغزشو از روش برنامه‌ریزی درجه دو دنباله‌ای درگام جستجوی محلی استفاده شده است. در فاز اول، الگوریتم VNS با یک جواب اولیه کاملا تصادفی از متغیرهای ورودی کنترل اجرا می‌شود. به منظور افزایش دقت جواب بدست آمده از فاز اول، نقاطگره‌ای زمانی جدیدی اضافه می‌شوند و مقادیر ورودی کنترل در آنها با درون‌یابی اسپلاین تقریب زده می‌شوند. سپس در فاز دوم VNS با جواب جدید ساخته شده از فاز اول مجددا راه اندازی می‌شود. الگوریتم پیشنهادی روی ۲۰ مساله کنترل بهینه واقعی، به عنوان مسائل آزمون، پیاده سازی شده است. نتایج عددی با برخی از الگوریتم‌های پیشنهادی اخیر مقایسه شده است. نتایج نشان می‌دهد روش پیشنهادی جواب‌های عددی بهتری نسبت به سایر روش‌ها در زمان محاسباتی کمتر ارائه می‌دهد. همچنین برای مقایسهIVNS با VNS تک فاز (که تعداد نقاط گره‌ای دردو فاز مشابه است) یک آزمایش عددی انجام شده است. این مطالعه نشان داد IVNS زمان محاسباتی کمتری نسبت به VNS دارد، در حالی که کیفیت جواب‌های بدست آمده به صورت معنی داری تغییر نمی‌کند.

کلمات کلیدی : مساله کنترل بهینه غیرخطی؛ جستجوی همسایگی متغیر؛ برنامه ریزی درجه دو دنباله ای.

روش هم محلی سینکبرای حل عددی معادلات دیفرانسیل کسری با مرتبه چندگانه

اسماعیل حسام الدینی و الهام اسدالهی فرد

دانشگاه صنعتی شیراز، گروه ریاضی

چکیده : در این مقاله روش هم محلی سینک را برای حل عددی معادلات دیفرانسیل کسری با مرتبه چندگانه بر پایه ی تعریف جدید مشتق کسری که اخیراً توسط خلیل و همکارانش ارائه شده است، به کار می بریم. از خواص تابع سینک استفاده کرده و معادله دیفرانسیل کسری را به یک سیستم از معادلات جبری تبدیل می کنیم. چند مثال عددی که مؤید دقت و کارایی این روش می باشد نیز ارائه گردیده است.

کلمات کلیدی : تابع سینک؛ معادلات دیفرانسیل کسری؛ معادلات دیفرانسیل کسری با مرتبه چندگانه؛ روش هم محلی.

روش خطوط بدون شبکه توابع پایه شعاعي برمبناي نقاط تعدیل شده براي حل معادلات برگرز

علیرضا سهیلي[۱] ، *مریم عرب عامري*[۲] *و مهدیار برفه‌اي*[۲]

[۱] دانشگاه فردوسی مشهد، دانشکده علوم ریاضی، گروه ریاضی کاربردی
[۱] دانشگاه فردوسی مشهد، قطب علمی مدلسازی و محاسبات سیستم های خطی و غیر خطی
[۲] دانشگاه سیستان و بلوچستان، دانشکده ریاضی

چکیده : یک روش خطوط بر مبناي توابع پایه شعاعي براي حل معادلات با مشتقات جزیي معرفي مي‌شود که براي بدست آوردن جواب، نقاط مرزي و دروني را جدا مي‌کند. این روش همراه با یک الگوریتم تعدیل نقاط بکار مي‌رود. نشان مي‌دهیم براي مسایلي که با تغییرات شدید در جواب مواجه هستند، نقاط تعدیل شده نسبت به نقاط یکنواخت عملکرد بهتري دارند به گونه‌اي که با تعداد نقاط کمتر مي‌توان جواب را با دقت مورد نظر بدست آورد. روش براي حل معادله برگرز بکار رفته است.

کلمات کلیدی : روش خطوط؛ توابع پایه شعاعي؛ روش‌هاي تعدیل؛ معادلات برگرز.

مطالعه عددي مسأله غير خطي كوشي و معادله نيويل-وايتهد با روش شبه درونياب بي-اسپلاين مكعبي

حسين اميني خواه و جواد علوي

دانشگاه گيلان، دانشكده علوم رياضي، گروه رياضي كاربردي

چكيده : اين مقاله به مطالعه يک تقريب عددي از معادله نيويل-وايتهد و معادله بدوضع انتشار كوشي مي‌پردازد. در طرح ارائه شده از مشتق بي-اسپلاين شبه درونياب براي تقريب مشتق متغيرهاي وابسته و از تفاضل پيشرو مرتبه اول براي تقريب مشتق زمان استفاده مي‌شود. مثال‌هايي براي تشريح روش بيان شده و نتايج عددي مثال‌ها با جواب‌هاي دقيق مقاسيه شده‌اند. مزيت اصلي اين روش در الگوريتم و پياده سازي ساده آن است.

كلمات كليدی : شبه درونياب بي- اسپلاين مكعبي؛ معادله انتشار- انتقال؛ طرح تفاضلي.

حل عددی مساله نیروی میرای نوسانی با استفاده از موجکهای هار

ایندریپ سینک و شیوکومار

موسسه ملی فناوری دکتر آمبدکار، پنجاب، هندوستان

چکیده : در این مقاله، حل عددی مساله نیروی میرای نوسانی را با استفاده از موجکهای هار، ارائه می کنیم. نتایج عددی حاصل را با نتایج حاصل از به کار بردن بعضی از روشهای مشهور، مانند رانگ–کوتای مرتبه چهارم کلاسیک و روشهای سری تیلور مقایسه می کنیم.

نتایج عددی نشان می دهند که روش استفاده از موجک های هار، که در این مقاله ارائه گردیده است، جوابهای تقریبی دقیق تری را در مقایسه با روش های ذکر شده در بالا، بدست می دهند.

کلمات کلیدی : روش موجک های هار؛ معادلات دیفرانسیل؛ ماتریس عملیاتی؛ نیروی میرای نوسانی.

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://jm.um.ac.ir/index.php/math

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.1, pp 1-83, 2015

web site : http://jm.um.ac.ir/index.php/math

Email : mjms@um.ac.ir

ISSN : 1735-7144