รายงานวิจัยฉบับสมบูรณ์

โครงการ ระเบียบวิธีอันดับสูงสำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบนอยมันน์

โดย ผศ.ดร.ดำรงศักดิ์ แย้มบางหวาย

มิถุนายน 2558

สัญญาเลขที่ MRG5580014

รายงานวิจัยฉบับสมบูรณ์

โครงการ ระเบียบวิธีอันดับสูงสำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบนอยมันน์

ผศ.ดร.ดำรงศักดิ์ แย้มบางหวาย
มหาวิทยาลัยพะเยา

สนับสนุนโดยสำนักงานกองทุนสนับสนุนการวิจัย

(ความเห็นในรายงานนี้เป็นของผู้วิจัย สกว.ไม่จำเป็นต้องเห็นด้วยเสมอไป)

บทคัดย่อ

รหัสโครงการ : MRG5580014

ชื่อโครงการ : ระเบียบวิธีอันดับสูงสำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบนอยมันน์

ชื่อนักวิจัย : ผศ.ดร.ดำรงศักดิ์ แย้มบางหวาย มหาวิทยาลัยพะเยา

E-mail Address : damrong.sut@gmail.com

ระยะเวลาโครงการ: 2 กรกฎาคม 2555 ถึง 1 กรกฎาคม 2557

ในงานวิจัยชิ้นนี้ แนวคิดของระเบียบวิธียืดเวลาแก้ไข (deferred correction technique) และระเบียบวิธีของแครงค์-นิโคลสัน (Crank-Nikolson scheme) จะถูกนำมาใช้สำหรับสร้างระเบียบวิธีเชิงตัวเลข สำหรับหาผลเฉลยเชิงตัวเลขของปัญหาการถ่ายเทความร้อนในกรณีที่เงื่อนไขขอบเป็นแบบนอยมันน์ ที่มีความแม่นยำอันดับสองเทียบกับเวลา และมีความแม่นยำอันดับที่สี่เทียบกับปริภูมิ ซึ่งระเบียบวิธีที่ดังกล่าวจะถูกเรียกว่า ระเบียบวิธีอันดับสูงแบบยืดเวลาการแก้ไข หลังจากนั้นจะนำระเบียบวิธีอันดับสูงแบบยืดเวลาการแก้ไขที่ได้ ไปทดสอบกับปัญหาทดสอบ และนำไปเปรียบเทียบกับ ระเบียบวิธีอันดับสูงแบบกระชับ เพื่อแสดงให้เห็นประสิทธิภาพ และประสิทธิผลของระเบียบวิธีอันดับสูงแบบยืดเวลาการแก้ไขที่ได้ทำการสร้างขึ้น

คำหลัก : Heat Conducting Problem; High-order Compact Finite Difference Scheme; High-order Deferred Correction Scheme

Abstract

Project Code : MRG5580014

Project Title : High Order Schemes for Heat Conduction Problem with Neumann Boundary Conditions

Investigator : Dr.Damrongsak Yambangwai, University of Phayao

E-mail Address : damrong.sut@gmail.com

Project Period : 2 July 2012 to 1 July 2014

In this research, the idea of deferred correction is utilized to construct a set of high-order (fourth-order) deferred correction scheme for the solution of one-dimensional heat conducting problems with Neumann boundaries. Crank-Nikolson scheme for the temporal discretization and high-order deferred correction approach for the spatial discretization are used. Numerical examples are given to demonstrate the performance of the method proposed and to compare mostly with the spatial higher-order compact scheme.

Keywords : Heat Conducting Problem; High-order Compact Finite Difference Scheme; High-order Deferred Correction Scheme

Executive Summary

1. Introduction to the research problem and its significance

Finite difference methods are among the commonest approximation schemes used for numerical solution of ordinary and partial differential equations, mainly, because of their simplicity of use and the fact that they lend themselves quite easily to the Taylor series analysis of any incurred errors. While there are a number of problems which can be solved with low-order approximation methods (second or lower) with reasonable accuracies, there is also a large class of problems including those of acoustics and fluid dynamics, the solutions of which typically require higher order approximation solution schemes for higher levels of accuracy.

Low order approximations generally require compact stencils which utilize three nodal points in any direction. Any approximation method which involves grid nodes outside those of a compact stencil is said to be non-compact. Higher order (greater than 2) finite difference approximations are possible but these methods typically require non-compact stencils. Also the application of non-compact stencils at or near boundaries of the problem domain usually requires inclusion of fictitious nodes. Thus complicating the resulting numerical formulations and the usual consequences of those complications include increases in the overall number of grid points as well as increases in the bandwidths of the resulting system matrices. High-order compact schemes (HOCs) are frequently used now-day because they can provide accurate results on compact stencils. A compact method must have about the same accuracy at the boundary and near boundary points as that of the interior points. Most existing HCSs are constructed for problem with Dirichlet boundary condition. The Neumann (insulated or exchange) boundary condition is often encountered in engineering application. However, fewer HCSs have been constructed for problem with Neumann (insulated or exchange) boundary condition, which are much more difficult to handle than that of Dirichlet condition. Even for those less popular compact difference schemes involving Neumann boundary condition, very often, the schemes are fourth-order or sixth-order at the interior points, but less at the boundary. Another way of constructing compact stencil and obtaining accurate results for all derivative terms is using deferred correction approach which is easier development in the case of Neumann boundary conditions. Deferred correction takes a low order scheme and promotes it to a high order scheme by calculating the residual and solving for the error. The objective of research proposal

is to construct high-order (fourth) accurate spatial differencing scheme by using deferred correction technique, demonstrate the performance of the method proposed and to compare mostly with the spatial high order compact scheme in the case of Neumann boundary conditions. Additional to high-order of accuracy the stability and convergence of proposed scheme will be proved in this paper.

2. Objectives

To construct high order scheme for solving heat conducting problem and related problem when the Neumann boundary conditions are specified which are proved to stability and convergence.

3. Methodology

3.1 Study concept of high order schemes.

3.2 Study papers, books and documents in the topic of high order schemes when Neumann boundary conditions are specified.

3.3 Using previous knowledge from 10.1 and 10.2 to construct high order schemes which are proved stability and convergence.

- 3.3.1 Analyze and implement high order compact scheme and high order deferred correction scheme for heat conducting problem with Neumann boundary conditions.
- 3.3.2 Develop computer codes for high order compact scheme and high order deferred correction scheme with Neumann conditions.
- 3.3.3 Compare the efficiency of high order compact scheme and high order deferred correction scheme by using several tests problems.
- 3.3.4 Continue developing a new set of high order scheme which is proved stability, convergence provide much more accurate numerical solution.

3.4 Writing and submitting the researches for publication.

เนื้อหางานวิจัย

เนื้อหางานวิจัยจะแบ่งเป็น 3 ส่วนใหญ่ ได้แก่

1. ที่มาและความสำคัญของปัญหา
2. เซตของระเบียบวิธีอันดับสูงสำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ ดิริเคล และนอยมันน์
3. ความเสถียรของระเบียบวิธีแบบยืดเวลาการแก้ไขอันดับสูง สำหรับผลเฉลยของปัญหาการถ่ายเทความร้อน
4. ผลเชิงตัวเลข
5. สรุปผลที่ได้จากงานวิจัย

(ดูรายระเอียดทั้งหมดดังเอกสารแนบหน้าถัดไป)

Contents

Chapter 1

Introduction

The desired properties of finite difference schemes are stability, accuracy and efficiency. These requirements are in conflict with each other. In many applications a high-order accuracy is required in the spatial discretization. To reach better stability, implicit approximation is desired. For a high-order method of traditional type (not a High-order compact schemes (HCS)), the stencil becomes wider with increasing order of accuracy. For a standard centered discretization of order p, the stencil is $p + 1$ points wide. This inflicts problems at the fictional boundaries, and using an implicit method results in the solution of an algebraic system of equations with large bandwidth. In light of conflict requirements of stability, accuracy and computational efficiency, it is desired to develop schemes that have a wide range of stability, high-order of accuracy and lead to the solution of a systems of linear equations with a tridiagonal matrix.

High-order compact schemes (HCS) are frequently used because they can provide accurate results on compact stencils [2-20]. The high-order compact schemes obtain all the numerical derivative along grid lines using smaller stencils solving a linear system of equations. Since the size of computational molecule affects both the storage requirements and effort needed to solve the linear equation system, we would like to keep it as small as possible. Another way of constructing compact stencil and obtaining accurate results for all derivative terms is using deferred-correction approach [7] . The high-order deferred correction scheme (HDS) is constructed and compared with HCS for the heat conducting problem with the Dirichlet boundary condition.

Most existing HCSs are constructed for problem with Dirichlet boundary condition [2-20]. The Neumann (insulated or exchange) boundary condition is often encountered in engineering application, such as ultra-heat transfer and reaction-diffusion. The conventional finite difference scheme for the Neumann boundary condition are either first-order accurate or second-order accurate but needs a ghost points outside the domain

[1, 4, 7]. However, fewer HCSs have been constructed for problem with Neumann (insulated or exchange) boundary condition, which are much more difficult to handle than dirichlet condition. Even for those less popular compact difference schemes involving neumann boundary condition, very often, the schemes are fourth-order or sixth-order at the interior points, but less at the boundary [2,21,22].

In this report a new set of fourth-order deferred correction schemes (HDS4) for one-dimensional heat conducting problem with Dirichlet and Neumann boundary conditions are presented.

A set of scheme is constructed for the heat conducting problem with initial data and Dirichlet and Neumann boundary conditions

$$u_t = \beta u_{xx} + f(x,t), \quad 0 < x < l, \quad t > 0, \tag{1.1}$$

$$u(x,0) = u_0(x), \quad 0 < x < l, \tag{1.2}$$

$$\text{Dirichlet BC: } u(0,t) = \alpha_1(t), \quad u(l,t) = \alpha_2(t), \quad t > 0, \tag{1.3}$$

$$\text{Neumann BC: } u_x(0,t) = \gamma_1(t), \quad u_x(l,t) = \gamma_2(t), \quad t > 0, \tag{1.4}$$

Here $u(x,t)$ represents the temperature at point (x,t), β is the diffusion constant, and $f(x,t)$, $\alpha_1(t)$, $\alpha_2(t)$, $\gamma_1(t)$, $\gamma_2(t)$ are sufficiently smooth functions. Problems (1.1)– (1.3) and (1.1), (1.2),(1.4) are model of transient heat conduction in a slab of material with thickness l.

The organization of the paper is as follow. In chapter 2, for self completeness we present a list of second-order Crank-Nicolson schemes for one-dimensional heat conducting problems with Dirichlet boundary conditions (DCNS2) [19], a list of Crank-Nicolson schemes with first-order Neumann boundary conditions (NCNS1) [19], a list of fourth-order compact schemes with Dirichlet boundary boundary conditions (DHCS4) [11, 14, 15], and a list of a fourth-order compact schemes with Neumann boundary (NHCS4) [21, 22], a set of second-order Crank-Nicolson schemes for one-dimensional heat conducting problems with the second-order deferred correction Neumann boundary conditions (DHDS2). A new set of fourth-order deferred correction with Dirichlet (DHDS4), and a new set of fourth-order deferred correction scheme with Neumann boundary are constructed and presented in the following. In chapter 3, we present numerical examples to compare the efficiency of standard schemes and high-order deferred correction schemes (DHDS4, NHDS2, and NHDS4) for problems with Dirichlet and Neumann boundary conditions. Results developed in this paper compared with the previous study [11, 14, 21] demonstrate the superior performance of the high-order deferred correction schemes .

Chapter 2

The Sets of Schemes

Let Δt denotes the spatial mesh size. For simplicity, we consider a uniform $1-D$ mesh, consisting of N points: $x_1, x_2, \ldots, x_N$ where $x_i = (i-1) * \Delta x$, and the mesh size $\Delta x = l/(N-1)$. Below we use notations u_i^n, $(u_{xx})_i^{n+1}$ to represent the numerical approximations of $u(x_i, t^n)$ and $u_{xx}(x_i, t^n)$ where $t^n = n\Delta t$ and $i = 1, 2, \ldots, N$.

2.1 A Set of Deferred Correction Scheme

A new set of high-order deferred correction schemes is based on an iterative method. Using second upper index $''s''$ to denote the number of iteration, one writes

$$\frac{u_i^{n+1,s+1} - u_i^n}{\Delta t} = \beta \frac{(u_{xx})_i^{n+1,s+1} + (u_{xx})_i^n}{2} + f_i^{n+1/2}, \tag{2.1}$$

$$\tag{2.2}$$

where

$$f_i^{n+1/2} = \frac{f_i^{n+1} + f_i^n}{2},$$

$s = 0, \ldots, \hat{S}$ and $i = 1, \ldots, N$.

The high-order finite difference approximation is utilized to approximate the second derivatives $(u_{xx})_i^n$, $i = 1, \ldots, N$. The deferred correction technique [7] is utilized to approximate the second derivatives $(u_{xx})_i^{n+1,s+1}$, $i = 1, \ldots, N$ by iterative method

$$(u_{xx})_i^{n+1,s+1} = (u_{xx}^l)|_i^{n+1,s+1} + \left[(u_{xx}^h)|_i^{n+1,s} - (u_{xx}^l)|_i^{n+1,s}\right]. \tag{2.3}$$

where

$$(u_{xx}^h)_i^{n+1,s}, i = 1, \ldots, N, s = 0, \ldots, \widehat{S}$$

is high-order approximation and

$$(u_{xx}^l)_i^{n+1,k}, k = s, s+1, i = 1, \ldots, N$$

is the lower-order approximation. The expression in the square brackets of (2.3) are evaluated by using values known from the previous iteration. When $s = 0$ we use solution from level n (so $u^{n+1,0} = u^n$). Once the iterations converge, the lower-order approximation terms drop out and the obtained solution corresponds to the high-order approximation of u_{xx}, the same order of approximation as $(u^h_{xx})^{n+1,s}$ approximate u_{xx}. For example, to get second order we do not need any iterations. If we denote

$$u_i^{n+1,s+1} = u_i^{n+1}$$

and

$$(u^l_{xx})_i^{n+1,s+1} = (u^l_{xx})_i^{n+1,s} = (u^h_{xx})_i^{n+1,s} = \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{\Delta x^2}$$

as result we get standard Crank-Nickolson scheme for interior points.

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \beta \frac{(u_{xx})_i^{n+1} + (u_{xx})_i^n}{2} + f_i^{n+1/2}, \quad i = 2, 3, \ldots, N-1, \tag{2.4}$$

where

$$f_i^{n+1/2} = \frac{f_i^{n+1} + f_i^n}{2}.$$

This scheme has a truncation error of $O(\Delta t^2)$ in time. Order of spatial truncation error depends on order of approximation of $(u_{xx})_i^k$, $k = n, n+1$ and order of approximation of boundary conditions as well.

Let us consider a set of fourth-order deferred correction scheme on the interior points by using equation (2.3). For the lower-order approximation of $(u^l_{xx})|_i^{n+1,k}$, $k = s, s+1$, $i = 2, \ldots, N-1$ in equation (2.3), we use the central second-order finite difference approximation

$$(u^l_{xx})_i^{n+1,k} = \frac{1}{\Delta x^2}\left(u_{i-1}^{n+1,k} - 2u_i^{n+1,k} + u_{i+1}^{n+1,k}\right). \tag{2.5}$$

For the high-order approximation of $(u^h_{xx})|_i^{n+1,s}$, $i = 2, \ldots, N-1$ (case $s = 0$ corresponds to $(u^h_{xx})|_i^n$) we use the fourth-order finite difference approximation

$$(u^h_{xx})_2^{n+1,s} = \frac{1}{12\Delta x^2}\left(10u_1^{n+1,s} - 15u_2^{n+1,s} - 4u_3^{n+1,s} + 14u_4^{n+1,s} - 6u_5^{n+1,s} + u_6^{n+1,s}\right), \tag{2.6}$$

$$(u^h_{xx})_i^{n+1,s} = \frac{1}{12\Delta x^2}\left(-u_{i-2}^{n+1,s} + 16u_{i-1}^{n+1,s} - 30u_i^{n+1,s} + 16u_{i+1}^{n+1,s} - u_{i+2}^{n+1,s}\right), \quad i = 3, \ldots, N-2, \tag{2.7}$$

$$(u^h_{xx})_{N-1}^{n+1,s} = \frac{1}{12\Delta x^2}\Big(10u_N^{n+1,s} - 15u_{N-1}^{n+1,s} - 4u_{N-2}^{n+1,s} + 14u_{N-3}^{n+1,s} - 6u_{N-4}^{n+1,s} + u_{N-5}^{n+1,s}\Big). \tag{2.8}$$

Substitute (2.6)–(2.8) into equation (2.3) the following version of fourth-order deferred correction approximation of $(u_{xx})|_i^{n+1,s+1}$, $i = 2, \ldots, N-1$ are

$$\begin{aligned}(u_{xx})_2^{n+1,s+1} &= \frac{1}{\Delta x^2}\Big(u_1^{n+1,s+1} - 2u_2^{n+1,s+1} + u_3^{n+1,s+1}\Big) + \\ &\frac{1}{12\Delta x^2}\Big(-2u_1^{n+1,s} + 9u_2^{n+1,s} - 16u_3^{n+1,s} + 14u_4^{n+1,s} - 6u_5^{n+1,s} + u_6^{n+1,s}\Big),\end{aligned} \tag{2.9}$$

$$\begin{aligned}(u_{xx})_i^{n+1,s+1} &= \frac{1}{\Delta x^2}\Big(u_{i-1}^{n+1,s+1} - 2u_i^{n+1,s+1} + u_{i+1}^{n+1,s+1}\Big) + \\ &\frac{1}{12\Delta x^2}\Big(-u_{i-2}^{n+1,s} + 4u_{i-1}^{n+1,s} - 6u_i^{n+1,s} + 4u_{i+1}^{n+1,s} - u_{i+2}^{n+1,s}\Big), \quad i = 3, \ldots, N-2,\end{aligned} \tag{2.10}$$

$$\begin{aligned}(u_{xx})_{N-1}^{n+1,s+1} &= \frac{1}{\Delta x^2}\Big(u_{N-2}^{n+1,s+1} - 2u_{N-1}^{n+1,s+1} + u_N^{n+1,s+1}\Big) + \\ &\frac{1}{12\Delta x^2}\Big(-2u_N^{n+1,s} + 9u_{N-1}^{n+1,s} - 16u_{N-2}^{n+1,s} + 14u_{N-3}^{n+1,s} - 6u_{N-4}^{n+1,s} + u_{N-5}^{n+1,s}\Big).\end{aligned} \tag{2.11}$$

Substitution (2.9)-(2.11) into equation (2.1) where $i = 2, \ldots, N-1$, results a fourth-order deferred correction scheme on the interior domain.

2.1.1 A Set of Fourth-Order Deferred Correction Scheme with Dirichlet Boundary

Let us consider the heat conducting problem with initial data and Dirichlet boundary conditions (1.1)-(1.3)

$$u_1^{n+1} = \alpha_1(t^{n+1}), \quad u_N^{n+1} = \alpha_2(t^{n+1}). \tag{2.12}$$

The set of schemes, consisting of equations (2.1), (2.9)-(2.11) and (2.12) can be written in the following form

$$a_i u_{i-1}^{n+1,s+1} + b_i u_i^{n+1,s+1} + c_i u_{i+1}^{n+1,s+1} = d_i; \quad i = 1, \ldots, N, \tag{2.13}$$

where the coefficients of tri-diagonal matrix are

$$\begin{aligned}&a_1 = 0, \quad a_i = -r\,;\ i = 2, ..., N-1, \quad a_N = -1,\\ &b_1 = 1, \quad b_i = 1 + 2r\,;\ i = 2, ..., N-1, \quad b_N = 1,\\ &c_1 = -1, \quad c_i = -r\,;\ i = 2, ..., N-1, \quad c_N = 0,\end{aligned} \tag{2.14}$$

$$
\begin{aligned}
&d_1 = \alpha_1(t^{n+1}),\\
&d_2 = \frac{r}{12}\Big(-2u_1^{n+1,s} + 9u_2^{n+1,s} - 16u_3^{n+1,s} + 14u_4^{n+1,s} - 6u_5^{n+1,s} + u_6^{n+1,s}\Big)\\
&+ \frac{r}{12}\Big(10u_1^{n} - 15u_2^{n} - 4u_3^{n} + 14u_4^{n} - 6u_5^{n} + u_6^{n}\Big) + \Delta t f_2^{n+\frac{1}{2}},\\
&d_i = \frac{r}{12}\Big(-u_{i-2}^{n+1,s} + 4u_{i-1}^{n+1,s} - 6u_i^{n+1,s} + 4u_{i+1}^{n+1,s} - u_{i+2}^{n+1,s}\Big)\\
&+ \frac{r}{12}\Big(-u_{i-2}^{n} + 16u_{i-1}^{n} - 30u_i^{n} + 16u_{i+1}^{n} - u_{i+2}^{n}\Big) + \Delta t f_i^{n+\frac{1}{2}},\ i = 3,\dots,N-2,\\
&d_{N-1} = \frac{r}{12}\Big(-2u_N^{n+1,s} + 9u_{N-1}^{n+1,s} - 16u_{N-2}^{n+1,s} + 14u_{N-3}^{n+1,s} - 6u_{N-4}^{n+1,s} + u_{N-5}^{n+1,s}\Big)\\
&+ \frac{r}{12}\Big(10u_N^{n} - 15u_{N-1}^{n} - 4u_{N-2}^{n} + 14u_{N-3}^{n} - 6u_{N-4}^{n} + u_{N-5}^{n}\Big) + \Delta t f_{N-1}^{n+\frac{1}{2}}.\\
&d_N = \alpha_2(t^{n+1}).
\end{aligned}
\tag{2.15}
$$

where

$$r = \frac{\beta\Delta t}{2\Delta x^2}, \quad f_i^{n+1/2} = \frac{f_i^{n+1} + f_i^n}{2}, \quad i = 2,\dots,N-1.$$

It can be seen that the truncation error of the heat conducting problem with initial data and Dirichlet boundary condition (1.3) depends on order of approximation in interior points, because Dirichlet BCs are approximated exactly. Crank-Nikolson scheme (2.4), (2.5 and (2.12) has an order $O(\Delta t^2, \Delta x^2)$ over all grid points. Scheme (2.1), (2.9)-(2.11) and (2.12) is the fourth-order deferred correction scheme with Dirichlet boundary (DHDS4) and has the order of approximation $O(\Delta t^2, \Delta x^4)$ in uniform norm.

2.1.2 A Set of Fourth-Order Deferred Correction Scheme with Neumann Boundary

We now develop the first, second and fourth order approximations of Neuman boundary conditions (1.4) (based on principle of deferred corrections) for u_{xx} at the boundary points $x = 0, l$. The main idea is to use given Neumann BC (value of first derivative) to approximate second derivative at boundary similar approach have been used in [20] to construct high-order compact scheme.

For the lower order approximation of the second derivative terms

$$(u_{xx}^l)|_i^{n+1,k}, i = 1, N, k = s, s+1$$

we construct the first-order finite difference formula

$$(u_{xx}^l)_1^{n+1,k} = a_1 u_1^{n+1,k} + a_2 u_2^{n+1,k} + a_3 (u_x^l)_1^{n+1,k}, \tag{2.16}$$

where the coefficients can be found by matching the Taylor series expansion of left-hand

side up to the term $O(\Delta x^2)u_{xx}$ which gives us the following linear system

$$\begin{aligned} a_1 + a_2 &= 0, \\ a_2\Delta x + a_3 &= 0, \\ a_2 &= \frac{2}{\Delta x^2}. \end{aligned}$$

The solution to the above system is

$$a_1 = -\frac{2}{\Delta x^2}, \quad a_2 = \frac{2}{\Delta x^2}, \quad a_3 = -\frac{2}{\Delta x}. \tag{2.17}$$

Similarly at boundary point N, the first-order formula for the second derivative terms $(u^l_{xx})|^{m,k}_N$ is

$$(u^l_{xx})^{n+1,k}_N = b_1 u^{n+1,k}_N + b_2 u^{n+1,k}_{N-1} + b_3 (u^l_x)^{n+1,k}_N \tag{2.18}$$

where

$$b_1 = -\frac{2}{\Delta x^2}, \quad b_2 = \frac{2}{\Delta x^2}, \quad b_3 = \frac{2}{\Delta x}. \tag{2.19}$$

Substitute equation (2.17) into (2.16) and (2.19) into (2.18), the second derivatives

$$(u^l_{xx})|^{n+1,k}_i, i = 1, N, k = s, s+1$$

are approximated with the first-order approximation by the following formula

$$\begin{aligned} (u^l_{xx})^{n+1,k}_1 &= \frac{2}{\Delta x^2}\left(-u^{n+1,k}_1 + u^{n+1,k}_2\right) - \frac{2}{\Delta x}\gamma_1(t^{n+1}), \\ (u^l_{xx})^{n+1,k}_N &= \frac{2}{\Delta x^2}\left(-u^{n+1,k}_N + u^{n+1,k}_{N-1}\right) + \frac{2}{\Delta x}\gamma_2(t^{n+1}). \end{aligned} \tag{2.20}$$

This approximation is used for Crank-Nicholson scheme in case $u = 0$, $u^{n+1,k} = u^{n+1}$. To approximate $(u^h_{xx})^{n+1,s}_i$, $i = 1, N$, with second-order we use finite–difference approximation in the form

$$(u^h_{xx})^{n+1,s}_1 = a_1 u^{n+1,s}_1 + a_2 u^{n+1,s}_2 + a_3 u^{n+1,s}_3 + a_4\gamma_1(t^{n+1}), \tag{2.21}$$

where the coefficients can be found by matching the Taylor series expansion up to the term $O(\Delta x^3)u_{xxx}$ which gives us the following linear system

$$\begin{aligned} a_1 + a_2 + a_3 &= 0 \\ a_2 + 2a_3 + \frac{a_4}{\Delta x^2} &= 0, \\ a_2 + 2^2 a_3 &= \frac{2}{\Delta x^2} \\ a_2 + 2^3 a_3 &= 0. \end{aligned}$$

The solution to the above system is

$$a_1 = -\frac{7}{2\Delta x^2}, \quad a_2 = \frac{4}{\Delta x^2}, \quad a_3 = -\frac{1}{2\Delta x^2}, \quad a_4 = -\frac{3}{\Delta x^2}. \tag{2.22}$$

Similarly at boundary point x_N, the second-order finite difference formula for the second derivative terms $(u^h_{xx})|^{n+1,s}_N$ is

$$(u^h_{xx})^{n+1,s}_N = b_1 u^{n+1,s}_N + b_2 u^{n+1,s}_{N-1} + b_3 u^{n+1,s}_{N-2} + b_4 \gamma_2(t^{n+1}), \tag{2.23}$$

where

$$b_1 = -\frac{7}{2\Delta x^2}, \quad b_2 = \frac{4}{\Delta x^2}, \quad b_3 = -\frac{1}{2\Delta x^2}, \quad b_4 = \frac{3}{\Delta x^2}. \tag{2.24}$$

Substitute equation (2.22) into (2.21) and (2.24) into (2.23), the second derivatives $(u^h_{xx})|^{m,k}_i$, $i = 1, N$, are approximated with the second-order finite difference approximation by the following formula

$$\begin{aligned}
(u^h_{xx})^{n+1,s}_1 &= \frac{1}{2\Delta x^2}\left(-7u^{n+1,s}_1 + 8u^{n+1,s}_2 - u^{n+1,s}_3\right) - \frac{3}{\Delta x}\gamma_1(t^{n+1}), \\
(u^h_{xx})^{n+1,s}_N &= \frac{1}{2\Delta x^2}\left(-7u^{n+1,s}_N + 8u^{n+1,s}_{N-1} - u^{n+1,s}_{N-2}\right) + \frac{3}{\Delta x}\gamma_2(t^{n+1}).
\end{aligned} \tag{2.25}$$

Substitute equations (2.20) and (2.25) into equation (2.3), the following second-order deferred correction approximation of $(u_{xx})|^{n+1,s+1}_i$ where $i = 1, N$ are

$$\begin{aligned}
(u_{xx})^{n+1,s+1}_1 &= \frac{2}{\Delta x^2}\left(-u^{n+1,s+1}_1 + u^{n+1,s+1}_2\right) - \frac{3}{\Delta x}\gamma_1(t^{n+1}) \\
&\quad + \frac{1}{2\Delta x^2}\left(-3u^{n+1,s}_1 + 4u^{n+1,s}_2 - u^{n+1,s}_3\right), \\
(u_{xx})^{n+1,s+1}_N &= \frac{2}{\Delta x^2}\left(-u^{n+1,s+1}_N + u^{n+1,s+1}_{N-1}\right) + \frac{3}{\Delta x}\gamma_2(t^{n+1}) \\
&\quad + \frac{1}{2\Delta x^2}\left(-3u^{n+1,s}_N + 4u^{n+1,s}_{N-1} - u^{n+1,s}_{N-2}\right),
\end{aligned} \tag{2.26}$$

To approximate $(u^h_{xx})|^{n+1,k}_i$, $i = 1, N$, with fourth order we use the following representation

$$(u^h_{xx})^{n+1,s}_1 = a_1 u^{n+1,s}_1 + a_2 u^{n+1,s}_2 + a_3 u^{n+1,s}_3 + a_4 u^{n+1,s}_4 + a_5 u^{n+1,s}_5 + a_6 \gamma_1(t^{n+1}), \tag{2.27}$$

where the coefficients are found by matching the Taylor series expansion up to the term$O(\Delta x^5)u_{xxxxx}$ which gives us the following linear system

$$\begin{aligned}
a_1 + a_2 + a_3 + a_4 + a_5 &= 0 \\
a_2 + 2a_3 + 3a_4 + 4a_5 + \frac{a_6}{\Delta x} &= 0, \\
a_2 + 2^2 a_3 + 3^2 a_4 + 4^2 a_5 &= \frac{2}{\Delta x^2} \\
a_2 + 2^3 a_3 + 3^3 a_4 + 4^3 a_5 &= 0, \\
a_2 + 2^4 a_3 + 3^4 a_4 + 4^4 a_5 &= 0, \\
a_2 + 2^5 a_3 + 3^5 a_4 + 4^5 a_5 &= 0.
\end{aligned}$$

The solution to the above system is

$$\begin{aligned} a_1 &= -\frac{415}{72\Delta x^2}, \quad a_2 = \frac{8}{\Delta x^2}, \quad a_3 = -\frac{3}{\Delta x^2}, \\ a_4 &= \frac{8}{9\Delta x^2}, \quad a_5 = -\frac{1}{8\Delta x^2}, \quad a_6 = -\frac{25}{6\Delta x} \end{aligned} \tag{2.28}$$

Similarly at boundary point N, the fourth-order finite difference formula for the second derivative terms $(u^h_{xx})|^{n+1,s}_N$ is

$$(u^h_{xx})^{n+1,s}_N = b_1 u^{n+1,s}_N + b_2 u^{n+1,s}_{N-1} + b_3 u^{n+1,s}_{N-2} + b_4 u^{n+1,s}_{N-3} + b_{N-4} u^{n+1,s}_5 + b_6 \gamma_2(t^{n+1}), \tag{2.29}$$

where

$$\begin{aligned} b_1 &= -\frac{415}{72\Delta x^2}, \quad b_2 = \frac{8}{\Delta x^2}, \quad b_3 = -\frac{3}{\Delta x^2}, \\ b_4 &= \frac{8}{9\Delta x^2}, \quad b_5 = -\frac{1}{8\Delta x^2}, \quad b_6 = \frac{25}{6\Delta x}. \end{aligned} \tag{2.30}$$

Substitute equation (2.28) to (2.27) and (2.30) to (2.29), the second derivatives $(u^h_{xx})|^{n+1,s}_i$ $i = 1, N$ are approximated with the fourth-order approximation by the following formula

$$\begin{aligned} (u^h_{xx})^{n+1,s}_1 &= \frac{1}{72\Delta x^2}\Big(-415u^{n+1,s}_1 + 576u^{n+1,s}_2 - 216u^{n+1,s}_3 \\ &\qquad + 64u^{n+1,s}_4 - 9u^{n+1,s}_5 \Big) - \frac{25}{6\Delta x}\gamma_1(t^{n+1}), \\ (u^h_{xx})^{n+1,s}_N &= \frac{1}{72\Delta x^2}\Big(-415u^{n+1,s}_N + 576u^{n+1,s}_{N-1} - 216u^{n+1,s}_{N-2} \\ &\qquad + 64u^{n+1,s}_{N-3} - 9u^{n+1,s}_{N-4}\Big) + \frac{25}{6\Delta x}\gamma_2(t^{n+1}). \end{aligned} \tag{2.31}$$

Substitute equations (2.20) and (2.31) into equation (2.3), the following version of fourth-order deferred correction approximation of $(u_{xx})|^{n+1,s+1}_i$ where $i = 1, N$ are

$$\begin{aligned} (u_{xx})^{n+1,s+1}_1 &= \frac{2}{\Delta x^2}\left(-u^{n+1,s+1}_1 + u^{n+1,s+1}_2\right) - \frac{2}{\Delta x}\gamma_1(t^{n+1}) \\ &\quad + \frac{1}{72\Delta x^2}\Big(-271u^{n+1,s}_1 + 432u^{n+1,s}_2 - 216u^{n+1,s}_3 \\ &\qquad + 64u^{n+1,s}_4 - 9u^{n+1,s}_5 \Big), \\ (u_{xx})^{n+1,s+1}_N &= \frac{2}{\Delta x^2}\left(-u^{n+1,s+1}_N + u^{n+1,s+1}_{N-1}\right) + \frac{2}{\Delta x}\gamma_2(t^{n+1}) \\ &\quad + \frac{1}{72\Delta x^2}\Big(-271u^{n+1,s}_N + 432u^{n+1,s}_{N-1} - 216u^{n+1,s}_{N-2} \\ &\qquad + 64u^{n+1,s}_{N-3} - 9u^{n+1,s}_{N-4} \Big) \end{aligned} \tag{2.32}$$

2.2 A Set of Fourth-order Compact Scheme

Most existing high-order compact schemes are constructed for problem with Dirichlet boundary conditions. In paper by Zhao et. al [21] a set of fourth order compact finite-difference schemes is developed to solve a heat conduction problem with Neumann boundary conditions. Let us shortly represent main idea and final formulae of this set. Spatial derivatives are evaluated by the fourth-order compact finite differences implicit scheme [12, 14, 20, 21].

2.2.1 A Set of Fourth-Order Compact Scheme with Dirichlet Boundary

In [8, 15], the Dirichlet boundary conditions $u(0, m\Delta t) = \alpha_1(t^m) = u_1^m$, and $u(l, m\Delta t) = \alpha_2(t^m) = u_N^m$ are used to derive the following four-order schemes at the boundary points (x_1, t^m), (x_2, t^m), (x_{N-1}, t^m) and (x_N, t^m)

$$\begin{aligned}(u_{xx})_1^m + \alpha(u_{xx})_2^m &= \frac{1}{12\Delta x^2}\left(a_1u_1^m + a_2u_2^m + a_3u_3^m + a_4u_4^m + a_5u_5^m\right), \\ &= \frac{1}{12\Delta x^2}\left(a_1\alpha_1(t^m) + a_2u_2^m + a_3u_3^m + a_4u_4^m + a_5u_5^m\right),\end{aligned} \tag{2.33}$$

where the coefficients can be found by matching the Taylor series expansion up to order $O(\Delta x^5)u_{xxxxx}$ which gives us the following linear system [15]

$$\begin{aligned} a_1 + a_2 + a_3 + a_4 + a_5 &= 0, \\ a_2 + 2a_3 + 3a_4 + 4a_5 &= 0, \\ a_2 + 2^2a_3 + 3^2a_4 + 4^2a_5 &= 2!(1+\alpha) \\ a_2 + 2^3a_3 + 3^3a_4 + 4^3a_5 &= 3!\alpha, \\ a_2 + 2^4a_3 + 3^4a_4 + 4^4a_5 &= \frac{4!}{2!}\alpha, \\ a_2 + 2^3a_3 + 3^3a_4 + 4^3a_5 &= \frac{5!}{3!}\alpha. \end{aligned}$$

The solution to the above system is

$$\alpha = 10, \quad a_1 = \frac{145}{12}, \quad a_2 = -\frac{76}{3}, \quad a_3 = \frac{29}{2}, \quad a_4 = -\frac{4}{3}, \quad a_5 = \frac{1}{12}. \tag{2.34}$$

Similarly at boundary point N, the fourth-order formula for the second derivative terms $(u_{xx}^l)|_N^{m,k}$ is

$$\begin{aligned}\alpha(u_{xx})_{N-1}^m + (u_{xx})_N^m &= \frac{1}{12\Delta x^2}\left(b_1u_N^m + b_2u_{N-1}^m + b_3u_{N-2}^m + b_4u_{N-3}^m + b_5u_{N-4}^m\right), \\ &= \frac{1}{12\Delta x^2}\left(b_1\alpha_2(t^m) + b_2u_{N-1}^m + b_3u_{N-2}^m + b_4u_{N-3}^m + b_5u_{N-4}^m\right),\end{aligned} \tag{2.35}$$

where

$$\alpha = 10, \quad b_1 = \frac{145}{12}, \quad b_2 = -\frac{76}{3}, \quad b_3 = \frac{29}{2}, \quad b_4 = -\frac{4}{3}, \quad b_5 = \frac{1}{12}. \tag{2.36}$$

The following fourth-order implicit relation at all interior points and any time level, is derived in [12,14,21] to approximate $(u_{xx})_i^m$

$$\frac{1}{10}(u_{xx})_{i-1}^m + (u_{xx})_i^m + \frac{1}{10}(u_{xx})_{i+1}^m = \frac{6}{5\Delta x^2}(u_{i-1}^m - 2u_i^m + u_{i+1}^m), \quad i = 2, \ldots, N-1. \tag{2.37}$$

Let us introduce the following matrix-vector notations

$$\vec{u}^m = \left(\alpha_1(t^m), u_2^m, \ldots, u_{N-1}^m, \alpha_2(t^m)\right), \quad (\vec{u}_{xx})^m = \left((u_{xx})_1^m, \ldots, (u_{xx})_N^m\right),$$

$$A^{\mathcal{D}} = \begin{bmatrix} 1 & 10 & & & \\ 1 & 10 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 10 & 1 \\ & & & 10 & 1 \end{bmatrix}_{N\times N} \qquad B^{\mathcal{D}} = \begin{bmatrix} -\frac{145}{24} & \frac{38}{3} & -\frac{29}{4} & \frac{2}{3} & -\frac{1}{24} & \\ -6 & 12 & -6 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & -6 & 12 & -6 \\ & -\frac{1}{24} & \frac{2}{3} & -\frac{29}{4} & \frac{38}{3} & -\frac{145}{24} \end{bmatrix}_{N\times N}.$$

With this notation, equation (2.33) and (2.35) can be combined and express in the following matrix form

$$A^{\mathcal{D}}\,(\vec{u}_{xx})^m = -\frac{2}{\Delta x^2} B^{\mathcal{D}} \vec{u}^m. \tag{2.38}$$

By the Gerschgorin Theorem [6], it can prove that A is invertible. So, equation (2.38) can be simplified to

$$(\vec{u}_{xx})^m = -\frac{2}{\Delta x^2}\left(A^{\mathcal{D}}\right)^{-1} B^{\mathcal{D}} \vec{u}^m, \quad m = n, n+1. \tag{2.39}$$

Substituting equation (2.39) into equations (2.4), a fourth-order accurate compact finite difference scheme on the interior points is

$$(A^{\mathcal{D}} + rB^{\mathcal{D}})\vec{u}^{n+1} = (A^{\mathcal{D}} - rB^{\mathcal{D}})\vec{u}^n + \Delta t A^{\mathcal{D}} \vec{f}^{n+\frac{1}{2}}, \tag{2.40}$$

where $r = \dfrac{\beta \Delta t}{\Delta x^2}$ and $\vec{f}^{n+\frac{1}{2}} = \left(f_1^{n+\frac{1}{2}}, \ldots, f_N^{n+\frac{1}{2}}\right)$.

The truncation error of the heat conducting problem with initial data and Dirichlet boundary condition (1.1)–(1.3) has an order of approximation $O(\Delta t^2, \Delta x^4)$ over all grid points which indicates a set of fourth order compact scheme with Dirichlet boundary (DHCS4) and consistent with the differential equation.

2.2.2 A Set of Fourth-Order Compact Scheme with Neumann Boundary

In [21], the Neumann boundary conditions $u_x(0, m\Delta t) = \gamma_1(t^m)$, and $u_x(l, m\Delta t) = \gamma_2(t^m)$ are used to derive the following four-order schemes at the boundary points (x_2, t^m), (x_3, t^m), (x_{N-1}, t^m) and (x_N, t^m)

$$\frac{11}{6}(u_{xx})_2^m - \frac{1}{3}(u_{xx})_3^m = -\frac{\gamma_1(t^m)}{\Delta x} + \frac{1}{\Delta x^2}(u_3^m - u_2^m), \tag{2.41}$$

$$\frac{11}{6}(u_{xx})_{N-1}^m - \frac{1}{3}(u_{xx})_{N-2}^m = -\frac{\gamma_2(t^m)}{\Delta x} + \frac{1}{\Delta x^2}(u_{N-2}^m - u_{N-1}^m). \tag{2.42}$$

Let us introduce the following matrix-vector notations

$$\vec{u}^m = \left(u_2^m, \ldots, u_{N-1}^m\right), (\vec{u}_{xx})^m = \left((u_{xx})_2^m, \ldots, (u_{xx})_{N-1}^m\right),$$

$$\vec{\gamma}(t^m) = \left(-\frac{\gamma_1(t^m)}{\Delta x}, 0, \ldots, 0, -\frac{\gamma_2(t^m)}{\Delta x}\right),$$

$$A^{\mathcal{N}} = \begin{bmatrix} 22 & -4 & & & \\ 1 & 10 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 10 & 1 \\ & & & -4 & 22 \end{bmatrix}_{(N-2)\times(N-2)} \quad B^{\mathcal{N}} = \begin{bmatrix} 6 & -6 & & & \\ -6 & 12 & -6 & & \\ & \ddots & \ddots & \ddots & \\ & & -6 & 12 & -6 \\ & & & -6 & 6 \end{bmatrix}_{(N-2)\times(N-2)}.$$

With these notations, the scheme (2.37), (2.41), and (2.42) can be expressed in the following matrix-vector form

$$\frac{1}{10}A^{\mathcal{N}}(\vec{u}_{xx})^m = -\frac{1}{5\Delta x^2}B^{\mathcal{N}}\vec{u}^m + \frac{6}{5}\vec{\gamma}(t^m). \tag{2.43}$$

Matrix $A^{\mathcal{N}}$ is invertible. Equation (2.43) can be solved with respect $(\vec{u}_{xx})^m$

$$(\vec{u}_{xx})^m = -\frac{2}{\Delta x^2}\left(A^{\mathcal{N}}\right)^{-1}B^{\mathcal{N}}\vec{u}^m + 12\left(A^{\mathcal{N}}\right)^{-1}\vec{\gamma}(t^m). \tag{2.44}$$

After substituting (2.44) into (2.4) we have the following vector equation

$$(A^{\mathcal{N}} + rB^{\mathcal{N}})\vec{u}^{n+1} = (A^{\mathcal{N}} - rB^{\mathcal{N}})\vec{u}^n + 12\beta\Delta t\vec{\gamma}(t^{n+\frac{1}{2}}) + \Delta t A^{\mathcal{N}}\vec{f}^{n+\frac{1}{2}}, \tag{2.45}$$

where $\vec{f}^{n+\frac{1}{2}} = \left(f_2^{n+\frac{1}{2}}, \ldots, f_{N-1}^{n+\frac{1}{2}}\right)$.

It can be seen that the truncation error of the heat conducting problem with initial data and Neumann boundary condition (1.4) has an order $O(\Delta t^2, \Delta x^4)$ at interior grid points $i = 3, \ldots, N-2$ [21], and an order of $O(\Delta t^2, \Delta x^3)$ at grid points x_2 and x_{N-1} [23] which indicates a set of fourth order compact scheme with Neumann boundary (NHCS4) and consistent with the differential equation.

2.2.3 A Set of Update Fourth-Order Compact Scheme with Neumann Boundary

Let us represent shortly main idea and final formulae of this set. Spatial derivatives are evaluated by the fourth-order compact finite differences implicit scheme [12, 14, 20, 21] on interior points and fourth-order deferred correction approach on the Neumann boundary.

The Neumann boundary conditions $u_x(0, m\Delta t) = \gamma_1(t^m)$, and $u_x(l, m\Delta t) = \gamma_2(t^m)$ are used to derive the following fourth-order deferred correction schemes at the boundary points (x_1, t^m) and (x_N, t^m) by iterative methods

$$\begin{aligned}
(u_{xx})_1^{n+1,s+1} &= \frac{2}{\Delta x^2}\left(-u_1^{n+1,s+1} + u_2^{n+1,s+1}\right) - \frac{2}{\Delta x}\gamma_1(t^{n+1}) \\
&+\frac{1}{72\Delta x^2}\Big(-271u_1^{n+1,s} + 432u_2^{n+1,s} - 216u_3^{n+1,s} \\
&\qquad + 64u_4^{n+1,s} - 9u_5^{n+1,s} \Big), \\
(u_{xx})_N^{n+1,s+1} &= \frac{2}{\Delta x^2}\left(-u_N^{n+1,s+1} + u_{N-1}^{n+1,s+1}\right) + \frac{2}{\Delta x}\gamma_2(t^{n+1}) \\
&+\frac{1}{72\Delta x^2}\Big(-271u_N^{n+1,s} + 432u_{N-1}^{n+1,s} - 216u_{N-2}^{n+1,s} \\
&\qquad + 64u_{N-3}^{n+1,s} - 9u_{N-4}^{n+1,s} \Big).
\end{aligned} \tag{2.46}$$

and

$$\begin{aligned}
(u_{xx})_1^n &= \frac{1}{72\Delta x^2}\Big(-415u_1^n + 576u_2^n - 216u_3^n \\
&\qquad + 64u_4^n - 9u_5^n \Big) - \frac{2}{\Delta x}\gamma_1(t^n), \\
(u_{xx})_N^n &= \frac{1}{72\Delta x^2}\Big(-415u_N^n + 576u_{N-1}^n - 216u_{N-2}^n \\
&\qquad + 64u_{N-3}^n - 9u_{N-4}^n\Big) + \frac{2}{\Delta x}\gamma_2(t^n).
\end{aligned} \tag{2.47}$$

The following fourth-order implicit relation at all interior points and any time level, is derived in [12,14,21] to approximate $(u_{xx})_i^{n+1,s+1}$ by iterative methods

$$\begin{aligned}
&\frac{1}{10}(u_{xx})_{i-1}^{n+1,s+1} + (u_{xx})_i^{n+1,s+1} + \frac{1}{10}(u_{xx})_{i+1}^{n+1,s+1} \\
&\quad = \frac{6}{5\Delta x^2}(u_{i-1}^{n+1,s+1} - 2u_i^{n+1,s+1} + u_{i+1}^{n+1,s+1}),
\end{aligned} \tag{2.48}$$

and

$$\frac{1}{10}(u_{xx})_{i-1}^{n} + (u_{xx})_i^{n} + \frac{1}{10}(u_{xx})_{i+1}^{n} = \frac{6}{5\Delta x^2}(u_{i-1}^{n} - 2u_i^{n} + u_{i+1}^{n}), \tag{2.49}$$

where $i = 2, \ldots, N-1$. When $s = 0$, we use solution from level n so

$$u_{xx}^{n+1,0} = u_{xx}^n.$$

Let us introduce the following matrix-vector notations

$$\begin{aligned}
\vec{u}^{n+1,s+1} &= \left(u_1^{n+1,s+1}, \ldots, u_N^{n+1,s+1}\right), \\
(\vec{u}_{xx})^{n+1,s+1} &= \left((u_{xx})_1^{n+1,s+1}, \ldots, (u_{xx})_N^{n+1,s+1}\right),
\end{aligned} \tag{2.50}$$

$$\begin{aligned} \vec{u}^n &= (u_1^n, \ldots, u_N^n)\,, \\ (\vec{u}_{xx})^n &= ((u_{xx})_1^n, \ldots, (u_{xx})_N^n)\,, \end{aligned} \tag{2.51}$$

$$\begin{aligned} \vec{\varphi}^{n+1,s} &= (\varphi_1, 0, \ldots, 0, \varphi_2)\,, \\ \vec{\psi}^n &= (\psi_1, 0, \ldots, 0, \psi_2)\,, \end{aligned} \tag{2.52}$$

where

$$\psi_1 = -\frac{2}{\Delta x}\gamma_1(t^n), \quad \psi_2 = -\frac{2}{\Delta x}\gamma_2(t^n),$$

$$\begin{aligned} \varphi_1 &= \frac{1}{72\Delta x^2}\Big(-415u_1^{n+1,s} + 576u_2^{n+1,s} - 216u_3^{n+1,s} \\ &\qquad + 64u_4^{n+1,s} - 9u_5^{n+1,s} \Big) - \frac{2}{\Delta x}\gamma_1(t^{n+1}), \\ \varphi_2 &= \frac{1}{72\Delta x^2}\Big(-415u_N^{n+1,s} + 576u_{N-1}^{n+1,s} - 216u_{N-2}^{n+1,s} \\ &\qquad + 64u_{N-3}^{n+1,s} - 9u_{N-4}^{n+1,s} \Big) - \frac{2}{\Delta x}\gamma_2(t^{n+1}), \end{aligned}$$

$$A^{\mathcal{U}} = \begin{bmatrix} 1 & 0 & & & \\ 1 & 10 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 10 & 1 \\ & & & 0 & 1 \end{bmatrix}_{(N)\times(N)} \qquad B^{\mathcal{U}} = \begin{bmatrix} -1 & 1 & & & \\ -6 & 12 & -6 & & \\ & \ddots & \ddots & \ddots & \\ & & -6 & 12 & -6 \\ & & & 1 & -1 \end{bmatrix}_{(N)\times(N)}$$

$$C^{\mathcal{U}} = \begin{bmatrix} \frac{271}{144} & -3 & \frac{3}{2} & -\frac{4}{9} & \frac{1}{16} & & \\ -6 & 12 & -6 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & -6 & 12 & -6 \\ & & \frac{1}{16} & -\frac{4}{9} & \frac{3}{2} & -3 & \frac{271}{144} \end{bmatrix}_{N\times N}.$$

The scheme (2.46)-(2.48) can be expressed in the following matrix-vector form

$$\begin{aligned} A^{\mathcal{U}}\,(\vec{u}_{xx})^{n+1,s+1} &= -\frac{2}{\Delta x^2}B^{\mathcal{U}}\vec{u}^{n+1,s+1} + \vec{\varphi}^{n+1,s}, \\ A^{\mathcal{U}}\,(\vec{u}_{xx})^{n} &= -\frac{2}{\Delta x^2}C^{\mathcal{U}}\vec{u}^{n} + \vec{\psi}^{n}, \end{aligned} \tag{2.53}$$

After substituting (2.45) into (2.4) we have the following vector equation

$$(A^{\mathcal{U}} + rB^{\mathcal{U}})\vec{u}^{n+1} = (A^{\mathcal{U}} - rC^{\mathcal{U}})\vec{u}^{n} + \beta\Delta t\vec{\gamma}^{s,n,n+1} + \Delta t A^{\mathcal{U}}\vec{f}^{n+\frac{1}{2}}, \tag{2.54}$$

where

$$\vec{\gamma}^{s,n,n+1} = \left(\frac{\varphi_1 + \psi_1}{2}, 0, \ldots, 0, \frac{\varphi_2 + \psi_2}{2}\right)$$

and

$$\vec{f}^{n+\frac{1}{2}} = \left(f_1^{n+\frac{1}{2}}, \ldots, f_N^{n+\frac{1}{2}} \right).$$

The truncation error of the heat conducting problem with initial data (1.1)–(1.2) and Neumann boundary condition (1.4) has an order of approximation $O(\Delta t^2, \Delta x^4)$ over all grid points which indicates a set of fourth order compact scheme with Neumann boundary (UHCS4) and consistent with the differential equation.

Chapter 3

Stability analysis

A set of high order deferred correction schemes is based on the well-known Crank-Nikolson type of scheme in the following form,

$$\begin{aligned}\frac{u_i^{n+1,s+1}-u_i^n}{\Delta t} &= \frac{\beta}{2}\left[(u_{xx})_i^{n+1,s+1}+(u_{xx})_i^n\right] \\ &\quad + f_i^{n+1/2}, \quad f_i^{n+1/2}=\frac{f_i^{n+1}+f_i^n}{2}\end{aligned} \tag{3.1}$$

the second superscript "s" denotes the number of iterations $s = 0, \ldots, \hat{S}$ and $i = 2, \ldots, N-1$.

The deferred correction technique is utilized to approximate the second-order derivatives at higher time levels $(u_{xx})_i^{n+1,s+1}$, $i = 2, \ldots, N-1$ by the iterative method

$$(u_{xx})_i^{n+1,s+1} = (u_{xx}^l)_i^{n+1,s+1} + \left[(u_{xx}^h)_i^{n+1,s} - (u_{xx}^l)_i^{n+1,s}\right], \tag{3.2}$$

where

$$(u_{xx}^h)_i^{n+1,s}, i = 2, \ldots, N-1, s = 0, \ldots, \widehat{S}$$

is high-order approximation on wide stencil, and

$$(u_{xx}^l)_i^{n+1,k}, k = s, s+1, i = 2, \ldots, N-1$$

is the lower order approximation on compact stencil (usually three point stencil). The expression in the square brackets of (3.2) is evaluated explicitly using the values known from the previous iteration. When $s = 0$ we use the solution from the time level n (so $u^{n+1,0} = u^n$ and $(u_{xx})_i^{n+1,0} = (u_{xx})_i^n$). Once the iterations converge, the lower order approximation terms drop out and the approximation of $(u_{xx})_i^{n+1,s+1}$ obtained has the same order of approximation as $(u_{xx}^h)_i^{n+1,\widehat{S}}$. There are no difficulties to construct high-order approximation for interior points.

To preserve a compact using three wide stencil in the finite difference scheme at higher time level $(n+1, s+1)$, we use the central second-order finite difference approximation to approximate the lower order term in (3.2)

$$(u_{xx}^{l})_i^{n+1,k} = \frac{1}{\Delta x^2}\Lambda_l u_i^{n+1,k}, k = s, s+1, \tag{3.3}$$

$$\Lambda_l u_i^{n+1,k} = u_{i-1}^{n+1,k} - 2u_i^{n+1,k} + u_{i+1}^{n+1,k}, i = 3, \ldots, N-2.$$

For the high-order approximation term in (3.2), we use a symmetric five point wide stencil for the inner points to reach the fourth-order of approximation

$$(u_{xx}^{h})_i^{n+1,s} = \frac{1}{\Delta x^2}\Lambda_h u_i^{n+1,s}, \quad i = 3, \ldots, N-2, \tag{3.4}$$

$$\Lambda_h u_i^{n+1,s} = \frac{1}{12}\big(- u_{i-2}^{n+1,s} + 16u_{i-1}^{n+1,s} - 30u_i^{n+1,s} + 16u_{i+1}^{n+1,s} - u_{i+2}^{n+1,s}\big).$$

To study the stability of scheme (3.1)-(3.4), we use the Von-Neumann stability analysis. For simplicity, we assume that $f_i^{n+1/2} \equiv 0$ in (3.1), and u is periodic in x. Let us recast scheme (3.1)-(3.4) in the following form,

$$\left(E + \alpha\Lambda_l\right) u_i^{n+1,s+1} = \alpha\left(\Lambda_l - \Lambda_h\right) u_i^{n+1,s} + \left(E - \alpha\Lambda_l\right) u_i^{n}, \tag{3.5}$$

where $\alpha = \beta\Delta t/(2\Delta x^2)$. If we define the following operators:

$$A = E + \alpha\Lambda_l, B = E - \alpha\Lambda_h, C = E + \alpha\Lambda_h,$$

where E is the identity operator, then (3.5) can be rewritten as follows

$$Au_i^{n+1,s+1} = (A - C)u_i^{n+1,s} + Bu_i^{n}. \tag{3.6}$$

Assuming that the operators commute, $(A-C)A = A(A-C)$ (for example in the case of uniform grid), it is easy to demonstrate that if $u_i^{n+1,\widehat{S}+1} = u_i^{n+1}$ and $u_i^{n+1,0} = u_i^{n}$ we get

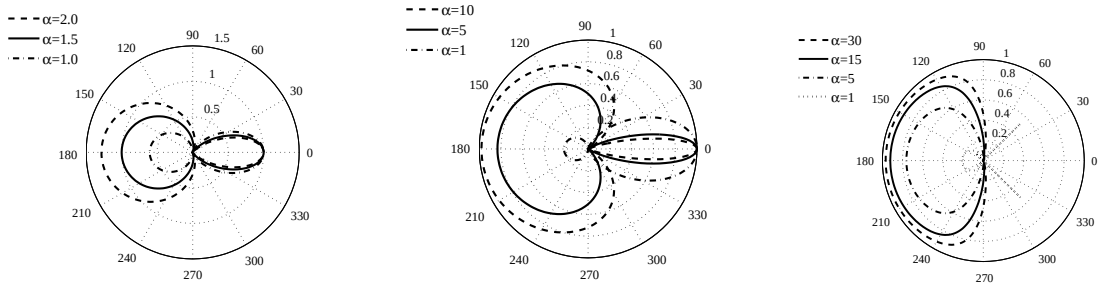
$$A^{\widehat{S}+1}u_i^{n+1} = \left(\sum_{k=0}^{\widehat{S}} A^{\widehat{S}-k}(A-C)^k\right) Bu_i^{n} + (A-C)^{\widehat{S}+1}u_i^{n}. \tag{3.7}$$

Let $u_i^n = \xi^n e^{I\Theta i}$, $I = \sqrt{-1}$, be the solution of (3.1)-(3.4), where $\Theta = 2\pi\triangle x/l$ is the phase angle with wavelength l. From (3.7), we can derive an equation for the amplification factor in the form

$$|\xi| = |\varphi(\Theta, \widehat{S}, \alpha)|, \tag{3.8}$$

where $\widehat{S}$ is the number of iterations, and

$$|\varphi(\Theta, \widehat{S}, \alpha)| = \frac{\left|\left[\left(\sum_{k=0}^{\widehat{S}} A^{\widehat{S}-k}(A-C)^k\right) B + (A-C)^{\widehat{S}+1}\right] e^{I\Theta i}\right|}{|A^{\widehat{S}+1}e^{I\Theta i}|}.$$



(a) (b) (c)

Figure 3.1: Variation of amplification factor with Θ. (a)–$\widehat{S} = 1$, dashed line $\alpha = 2.0$, solid line $\alpha = 1.5$, dash-doted line $\alpha = 1.0$, (b)–$\widehat{S} = 3$, dashed line $\alpha = 10.0$, solid line $\alpha = 5.0$, dash-doted line $\alpha = 1.0$, (c)–$\widehat{S} = 5$, dashed line $\alpha = 30.0$, solid line $\alpha = 15$, dash-doted line $\alpha = 5.0$, doted line $\alpha = 1.0$

For stability of the method it is necessary that the absolute values of the amplification factor is less than one, i.e.

$$|\xi| < 1. \tag{3.9}$$

Calculations are tedious and almost impossible to do by hand without mistake. We have therefore automate all calculations in a computer algebra environment based on REDUCE to obtain an explicit form of $|\varphi(\Theta, \widehat{S}, \alpha)|$. Figure 3.1 shows the values of $|\xi|^2$ in the polar coordinate system $(|\xi|^2, \Theta)$ for $\widehat{S} = 1, 3 \,\text{and}\, 5$. If only one iteration executes in (3.1), $\widehat{S} = 1$, inequality (3.9) holds if $\alpha < 1.5$, as can be seen from Figure 3.1 a). If 3 iterations are done in (**??**) (Figure 3.1 b) , $\widehat{S} = 3$, the amplification factor remains bounded by one at least for $\alpha \leq 10$. In case of $\widehat{S} = 5$, the stability criteria hold up to $\alpha = 30$ as can be seen from Figure 3.1 c). It can be seen that increasing the number of internal iterations results in increasing the range of α needed for stability. This tendency allows to assume that as $\widehat{S} \to \infty$, our method becomes the unconditionally stable Crank-Nikolson method for the heat equation.

Chapter 4

Numerical Examples

In this section, several numerical examples are carried to verify and compare the accuracy for the schemes DCNS2, DHCS4, DHDS4, NCNS1, NHDS2, NHCS4, NHDS4, and UNHDS4. For convenient information about abbreviations used for different approaches as well as reference on original papers are summarized in Table 4.1

Abbreviation	Method	Equations	refrence
DCNS2	Dirichlet BC, Crank-Nikolson, Second order	2.4, 2.5, 2.12	
DHCS4	Dirichlet BC, High-order compact, fourth order	2.40	
DHDS4	Dirichlet BC,differed correction, fourth order	2.13-2.15	
NCNS1	Neuman BC, Crank-Nikolson, first order	2.4, 2.5, [21]	
NHDS2	Neuman BC, differed correction, second order	2.1, 2.9-2.11, 2.26	
NHDS4	Neuman BC, differed correction, fourth order	2.1, 2.9-2.11, 2.32	
NHCS4	Neuman BC, High-order compact,fourth order	2.45	
UNHCS	Neuman BC, High-order compact, fourth order, Differed corrected BC	2.54	

Table 4.1: Abreviation used for different approaches

Below, we denote $\Delta t = \Delta x^2$ the time step size, and the following stopping criterion is applied:

$$\max_{1\leq i\leq N} \mid u_i^{n+1,s+1} - u_i^{n+1,s} \mid < \epsilon, \quad s = 0, \ldots, \hat{S}$$

where $''s''$ denotes the number of iteration. It should be pointed out, although a small Δt is chosen here, our set of scheme is unconditionally stable with no restriction on either space mesh or time increment. For testing purpose only, all computations are performed for $0 \leq t \leq 1$.

In the first part of this section, the numerical examples are provided to verify the accuracy for DCNS2, DHCS4, and DHDS4 with zero-Dirichlet and nonzero-Dirichlet boundary conditions. The computation are performed using uniform grids of $11, 21, 41, 81$, and 161 nodes.

In the second part of this section, the numerical example are provided to verify and compare the accuracy for NCNS1, NHDS2, NHCS4, and NHDS4 with nonzero-Neumann boundary. We apply a set of fourth-order deferred correction Neumann boundary for the HCS4 and call it as UNHCS4 to update the accuracy of NHCS4. The computation are performed using the uniform grids of $11, 21, 41, 81$, and 161 nodes.

Example 1 (Zero-Dirichlet boundary conditions and properly selected initial condition)

$$u_t = u_{xx} + (\pi^2 - 1)e^{-t}\sin(\pi x), \quad 0 \le x \le 1, \quad t > 0,$$
$$u(x,0) = sin(\pi x), \quad u(0,t) = 0, \quad u(1,t) = 0. \tag{4.1}$$

The exact solution is $u(x,t) = e^{-t}\sin(\pi x)$. The performance over the time domain $t \in [0,1]$ for DCNS2 , DHCS4, and DHDS4 are compared. The maximum error, and order of accuracy at $t = 1$ and $\epsilon = 10^{-8}$ are shown in Table 2.

Types of scheme	Number of grids	Maximum error	Order of convergence
DCNS2	11	3.39×10^{-3}	–
	21	8.43×10^{-4}	2.00
	41	2.10×10^{-4}	2.00
	81	5.26×10^{-5}	2.00
	161	1.32×10^{-5}	2.00
DHCS4	11	1.17×10^{-5}	–
	21	7.30×10^{-7}	4.01
	41	4.56×10^{-8}	4.00
	81	2.85×10^{-9}	4.00
	161	1.78×10^{-10}	4.00
DHDS4	11	1.30×10^{-5}	–
	21	2.24×10^{-6}	2.55
	41	1.52×10^{-7}	3.88
	81	9.59×10^{-9}	3.98
	161	6.00×10^{-10}	4.00

Table 4.2: Comparison of maximum error for equation (4.1) at $t = 1$

The fourth-order differed correction scheme demonstrates a little bit larger error compare with fourth-order compact scheme.

Example 2 (The non-homogeneous Dirichlet boundary conditions and properly selected initial condition

$$u_t = u_{xx} + (\pi^2 - 1)e^{-t}\cos(\pi x) + 4x - 2, \quad 0 \le x \le 1, \quad t > 0,$$
$$u(x,0) = \cos(\pi x) + x^2, \quad u(0,t) = e^{-t}, \quad u(1,t) = -e^{-t} + 4t + 1. \tag{4.2}$$

The exact solution is $u(x,t) = e^{-t}\cos(\pi x) + x^2 + 4xt$. The performance over the time domain $t \in [0,1]$ for DCNS2, DHCS4, and DHDS4 are compared. The maximum error, and order of accuracy at $t = 1$ and $\epsilon = 10^{-8}$ are compared in Table 3.

Types of scheme	Number of grids	Maximum error	Order of convergence
DCNS2	11	6.51×10^{-4}	–
	21	1.62×10^{-4}	2.00
	41	4.08×10^{-5}	1.99
	81	1.02×10^{-5}	2.00
	161	2.55×10^{-6}	2.00
DHCS4	11	3.20×10^{-6}	–
	21	2.00×10^{-7}	4.00
	41	1.26×10^{-8}	3.99
	81	7.85×10^{-10}	4.00
	161	5.11×10^{-11}	4.00
DHDS4	11	1.78×10^{-5}	–
	21	2.92×10^{-7}	5.92
	41	2.44×10^{-8}	3.58
	81	1.78×10^{-9}	3.78
	161	1.15×10^{-10}	3.95

Table 4.3: Comparison of maximum error for equation (4.2) at $t = 1$

Example 3(The non-homogeneous Neumann boundary indicate that the boundary are insulated and properly selected initial condition)

$$u_t = u_{xx} + (4\pi^2 - 1)e^{-t}\cos(2\pi x) + x - 2, \quad 0 \le x \le 1, \quad t > 0,$$
$$u(x,0) = \cos(2\pi x) + x^2, \quad u_x(0,t) = t, \quad u_x(1,t) = 2 + t. \tag{4.3}$$

The exact solution is $u(x,t) = e^{-t}\cos(2\pi x) + x^2 + xt$. The performance over the time domain $t \in [0,1]$ for NCNS1, NHDS2, NHCS4, UNHCS4, and NHDS4 are compared. The maximum error, and order of accuracy at $t = 1$ and $\epsilon = 10^{-8}$ are compared in Table 4.

Types of scheme	Number of grids	Maximum error	Order of convergence
NCNS1	21	1.26×10^{0}	–
	41	6.15×10^{-1}	1.03
	81	3.04×10^{-1}	1.02
	161	1.51×10^{-1}	1.01
NHDS2	21	5.71×10^{-2}	–
	41	7.74×10^{-3}	2.88
	81	1.07×10^{-3}	2.85
	161	1.58×10^{-4}	2.76
NHCS4	21	8.97×10^{-2}	–
	41	1.18×10^{-2}	2.92
	81	1.50×10^{-3}	2.98
	161	1.89×10^{-4}	2.99
UNHCS4	21	4.25×10^{-3}	–
	41	1.61×10^{-4}	4.72
	81	6.10×10^{-6}	4.72
	161	2.30×10^{-7}	4.73
NHDS4	21	3.11×10^{-3}	–
	41	1.19×10^{-4}	4.71
	81	4.77×10^{-6}	4.64
	161	1.92×10^{-7}	4.63

Table 4.4: Comparison of maximum error for equation (4.3) at $t = 1$

The last table shows the maximum internal iteration(MI) of deferred correction scheme (DHDS4, NHDS2, UNHCS4, NHDS4) depending on the stopping criterion (ϵ) and the number of grid points.

Types of scheme	Stopping criterion	MI of 11 grids	MI of 21 grids	MI of 41 grids	MI of 81 grids	MI of 161 grids
DHDS4	10^{-4}	2	2	2	2	1
	10^{-6}	3	2	2	2	2
	10^{-8}	5	3	2	2	2
	10^{-10}	7	5	3	2	2
NHDS2	10^{-4}	6	4	2	2	1
	10^{-6}	10	7	5	3	2
	10^{-8}	13	11	9	7	4
	10^{-10}	17	15	13	11	9
UNHCS4	10^{-4}	11	4	2	2	1
	10^{-6}	24	17	11	5	2
	10^{-8}	37	30	24	18	12
	10^{-10}	50	43	37	31	25
NHDS4	10^{-4}	10	4	2	2	1
	10^{-6}	22	15	10	4	2
	10^{-8}	33	27	21	16	11
	10^{-10}	44	38	32	27	22

Table 4.5: Maximum of internal iteration of deferred correction scheme.

Chapter 5

Conclusion

In this Report, a new set of high-order deferred correction schemes is constructed for a heat conduction problem with Dirichlet and Neumann boundary conditions. The greatest significance of this set of schemes, comparing to other similar ones, is that it is exactly fourth order accurate in space at all grid points, including both interior and boundary point. HDS schemes have better accuracy in the case of Neumann boundary conditions. Numerical examples are provided to confirm the accuracy. The construction of HDS requires only a regular three-point stencil similar to that in standard second-order Crank-Nicolson methods. Numerical examples are provided to confirm the accuracy. HDS approach can be easily extend to multidimesional case.

Bibliography

[1] John, C.T., Dale A.A., and Richard, H.P., Computational Fluid Mechanics and Heat Transfer, Hemisphere, New York. 1798

[2] Adam Y., Highly accurate compact implicit methods and boundary conditions, Journal of Computational Physics. 1977; 24: 10-22.

[3] Navon I.M. and Riphagen H.A., An Implicit compact Fourth-order Algorithm for Solving shallow water equation in conservative law form, Mon Weather Rev. 1979; 107: 1107-1127.

[4] Patankar, S.V., Numerical Heat Transfer and Fluid Flow, Hemisphere, New York. 1980

[5] Christie I., Upwind compact finite difference schemes, Journal of Computational Physics. 1985; 53: 353-368.

[6] Atkinson K.E., An Introduction to Numerical Analysis, John Wiley & Son, New York. (1988)

[7] Fletcher, C.A.J., Computaional Techniues for Fluid Dynmics 1, Springer-Verlag, New York. (1990)

[8] Lele, S.K., Compact finite difference scheme with spectral-like solution, J. Comp. Phys. 1992; 103: 16-42.

[9] Carpenter M. H., Gottlieb D., and Abarbanel S., Stable and accurate boundary treatments forcompact, high-order finite difference schemes, Appl Numer Math. 1993; 12: 55-87.

[10] Carey G. F. and Spotz W. F., High-order compact mixed methods, Commun Numer Methods Eng. 1997; 13: 553-564.

[11] Deng X. and Maekawa H., Compact high-order accurate nonlinear scheme, Journal of Computational Physics. 1997: 130: 77-91.

[12] Chu P. C. and Fan C., A Three-point Combined Compact Difference Scheme, Journal ofComputational Physics. 1998: 140: 370-399.

[13] Chu P. C. and Fan C., A three-point six-order non-uniform combined compact difference scheme, Journal of Computational Physics. 1999; 148: 663-674.

[14] Dai W. and Nassar R., A compact finite difference scheme for solving a three-dimensional heat transport equation in a thin film, Numerical Methods Partial Differential Equations. 2000: 16: 441-458.

[15] Dai W. and Nassar R., Compact ADI method for solving parabolic differential equations, Numerical Methods Partial Differential Equations. 2002: 18: 129-142.

[16] Kalita J.C., Dalal D.C., and Dass A.K., A class of Higher order compact schemes for the Unsteady Two-dimensional Convection-diffusion equation with variables convection coefficients, International Journal for Numerical Methods in Fluids. 2002: 18: 1111-1131.

[17] Nihei T. and Ishii K., A Fast Solver of the Shallow Water Equations on a Sphere using aCombined Compact Difference Scheme, Journal of Computational Physics. 2003: 187: 639-659.

[18] Karaa S., Dala D.C., and Dass A.K., High-order ADI Method for solving unsteady convection-diffusion Problem, Journal of Computational Physics. 2004: 198: 1-9.

[19] Gerald W.R., Finite-Difference Approximations to the Heat Equation, Mechanical Engineering Department Portland State University. 2004: 1-27.

[20] Li J., Chen Y., and Liu G., High-Order Compact ADI Methods for Parabolic Equations, An International Journal of Computers and Mathematics with Application. 2006; 52: 1343-1356.

[21] Zhao J., Dai W., and Niu T., Fourth order compact schemes of a heat conduction problem withNeumann boundary conditions, Numerical Methods Partial Differential Equations. 2007; 23: 949-959.

[22] Zhao J., Dai W., and Niu T., Fourth-Order Compact Schemes for Solving Multidimensional Heat Problems with Neumann Boundary Conditions, Numer Methods Partial Differential Equations.2008; 24: 165-178.

[23] Dai W., A New Accurate Finite Difference Scheme for Neumann (insulated) Boundary Condition of Heat Conduction, International Jounal of Themal Sciences.2010; 49: 571-579.

Output ที่ได้จากโครงการ

ผลงานตีพิมพ์ในวารสารวิชาการนานาชาติ

D.Yambangwai and N.P.Moshkin. Fourth-Order Deferred Correction Scheme for Solving Heat Conduction Problem. Mathematical Problems in Engineering, 2013, Article ID 5746205, 9 pages, http://dx.doi.org/10.1155/2013/574620.

ภาคผนวก

ประกอบด้วย
ผลงานตีพิมพ์ระดับนานาชาติ 1 เรื่อง ได้แก่

D.Yambangwai and N.P.Moshkin. Fourth-Order Deferred Correction Scheme for Solving Heat Conduction Problem. Mathematical Problems in Engineering, 2013, Article ID 5746205, 9 pages, http://dx.doi.org/10.1155/2013/574620.

โปรแกรมสำหรับทดสอบผลเชิงตัวเลข สำหรับระเบียบวิธีอันดับสูงสำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ ดิริเคล และนอยมันน์ ประกอบไปด้วย

1. ระเบียบวิธีผลต่างอันดับสอง (Second-Order Finite Difference Scheme) สำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ ดิริเคล (DCNS2)
2. ระเบียบวิธีผลต่างแบบกระชับอันดับสี่ (Fourth-Order Compact Finite Difference Scheme) สำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ ดิริเคล (DHCS4)
3. ระเบียบวิธีผลต่างแบบยืดเวลาแก้ไขอันดับสี่ (Fourth-Order Deferred ฉ orrection Finite Difference Scheme) สำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ ดิริเคล (DHDS4)
4. ระเบียบวิธีผลต่างอันดับ (Second-Order Finite Difference Scheme) สำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ นอยมันน์ (NCNS2)
5. ระเบียบวิธีผลต่างแบบกระชับอันดับสี่ (Fourth-Order Compact Finite Difference Scheme) สำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ นอยมันน์ (NHCS4)
6. ระเบียบวิธีผลต่างแบบยืดเวลาแก้ไขอันดับสี่ (Fourth-Order Deferred ฉ orrection Finite Difference Scheme) สำหรับผลเฉลยของปัญหาการถ่ายเทความร้อนในกรณีเงื่อนไขขอบเป็นแบบ นอยมันน์ (NHDS4)

Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 574620, 9 pages
http://dx.doi.org/10.1155/2013/574620

Research Article

Fourth-Order Deferred Correction Scheme for Solving Heat Conduction Problem

D. Yambangwai[1] and N. P. Moshkin[2]

[1] *Department of Mathematics, School of Science, University of Phayao, Phayao 56000, Thailand*
[2] *School of Mathematics, Institute of Science, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand*

Correspondence should be addressed to D. Yambangwai; damrong.sut@gmail.com

Received 3 December 2012; Accepted 27 February 2013

Academic Editor: Safa Bozkurt Coskun

A deferred correction method is utilized to increase the order of spatial accuracy of the Crank-Nicolson scheme for the numerical solution of the one-dimensional heat equation. The fourth-order methods proposed are the easier development and can be solved by using Thomas algorithms. The stability analysis and numerical experiments have been limited to one-dimensional heat-conducting problems with Dirichlet boundary conditions and initial data.

1. Introduction

The desired properties of finite difference schemes are stability, accuracy, and efficiency. These requirements are in conflict with each other. In many applications a high-order accuracy is required in the spatial discretization. To reach better stability, implicit approximation is desired. For a high-order method of traditional type (not a high-order compact (HOC)), the stencil becomes wider with increasing order of accuracy. For a standard centered discretization of order p, the stencil is $p + 1$ points wide. This inflicts problems at the fictional boundaries, and using an implicit method results in the solution of an algebraic system of equations with large bandwidth. In light of conflict requirements of stability, accuracy, and computational efficiency, it is desired to develop schemes that have a wide range of stability and highorder of accuracy and lead to the solution of a system of linear equations with a tri-diagonal matrix, that is, the system of linear equations arising from a standard second-order discretization of heat equation.

The development of high-order compact (HOC) schemes [1–18] is one approach to overcome the antagonism between stability, accuracy, and computational cost. However, the HOC becomes complicated when applie to multidimensional problems or to non-Cartesian coordinate cases.

Another way of preserving a compact stencil at higher time level and reaching high-order spatial accuracy is the deferred correction approach [11]. A classical deferred correction procedure is developed in [19, 20].

In this paper we use the deferred correction technique to obtain fourth-order accurate schemes in space for the one-dimensional heat-conducting problem with Dirichlet boundary conditions. The linear system that needs to be solved at each time step is similar to the standard Crank-Nikolson method of second order which is solved by using Thomas algorithms. The fourth-order deferred (FOD) correction schemes are compared with the fourth-order semi-implicit (FOS) schemes and fourth-order compact (FOC) schemes for the Dirichlet boundary value problems.

A set of schemes are constructed for the one-dimensional heat-conducting problem with Dirichlet boundary conditions and initial data:

$$u_t = \beta u_{xx} + f(x,t), \quad 0 < x < l, \ t > 0, \tag{1}$$

$$u(x,0) = u_0(x), \quad 0 < x < l, \tag{2}$$

$$\text{Dirichlet BC}: \ u(0,t) = \gamma_1(t), \quad u(l,t) = \gamma_2(t), \ t > 0, \tag{3}$$

where the diffusion coefficient β is positive, $u(x,t)$ represents the temperature at point (x,t), and $f(x,t)$, $\gamma_1(t)$, $\gamma_2(t)$ are sufficiently smooth functions.

The rest of this paper is organized as follows. Section 2 presents an FOD scheme which we use to compare performance of proposed scheme with FOS and FOC schemes. Section 3 provides examples of comparisons. Although FOD schemes have a higher computational cost than FOS and FOC schemes, it is evident from these examples that the FOD schemes have the advantage of accuracy in the uniform norm, robustness, and the ability to be extended easily to the multidimensional case. We conclude the paper in Section 4.

2. The Fourth-Order Schemes

Let Δt denote the temporal mesh size. For simplicity, we consider a uniform mesh consisting of N points: $x_1, x_2, \ldots, x_N$ where $x_i = (i-1)\Delta x$ and the mesh size is $\Delta x = l/(N-1)$. Below we use the notations u_i^m and $(u_{xx})_i^m$ to represent the numerical approximations of $u(x_i, t^m)$ and $u_{xx}(x_i, t^m)$ where $t^m = m\Delta t$ and $u^{(p)}$ is the value of the pth derivative of the given function u.

2.1. Fourth-Order Semi-Implicit Scheme. The application to the well-known Crank-Nikolson scheme to (1) results in the following expression:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\beta}{2}\left[(u_{xx})_i^{n+1} + (u_{xx})_i^n\right] + f_i^{n+1/2}, \tag{4}$$

where $f_i^{n+1/2} = (f_i^{n+1} + f_i^n)/2$, $i = 2, \ldots, N-1$. The Dirichlet boundary conditions

$$\begin{aligned} u(0, m\Delta t) &= \gamma_1(t^m) = u_1^m, \\ u(l, m\Delta t) &= \gamma_2(t^m) = u_N^m \end{aligned} \tag{5}$$

are used to derive the following fourth-order approximation of second derivative terms:

$$\begin{aligned} (u_{xx})_2^m &= \frac{1}{12\Delta x^2}\left(a_1 u_1^m + a_2 u_2^m + a_3 u_3^m + a_4 u_4^m \right. \\ &\qquad \left. + a_5 u_5^m + a_6 u_6^m\right) \\ &= \frac{a_1}{12\Delta x^2}\gamma_1(t^m) + \frac{1}{12\Delta x^2} \\ &\quad \times \left(a_2 u_2^m + a_3 u_3^m + a_4 u_4^m + a_5 u_5^m + a_6 u_6^m\right), \\ (u_{xx})_i^m &= \frac{1}{12\Delta x^2} \\ &\quad \times \left(-u_{i-2}^m + 16u_{i-1}^m - 30u_i^m + 16u_{i+1}^m - u_{i+2}^m\right), \\ &\qquad i = 3, \ldots, N-2, \\ (u_{xx})_{N-1}^m &= \frac{1}{12\Delta x^2}\left(a_1 u_N^m + a_2 u_{N-1}^m \right. \\ &\qquad + a_3 u_{N-2}^m + a_4 u_{N-3}^m \\ &\qquad \left. + a_5 u_{N-4}^m + a_6 u_{N-5}^m\right) \\ &= \frac{a_1}{12\Delta x^2}\gamma_2(t^m) + \frac{1}{12\Delta x^2} \\ &\quad \times \left(a_2 u_{N-1}^m + a_3 u_{N-2}^m \right. \\ &\qquad \left. + a_4 u_{N-3}^m + a_5 u_{N-4}^m + a_6 u_{N-5}^m\right), \end{aligned} \tag{6}$$

where the coefficients can be found by matching the Taylor series expansion of left-hand-side terms up to order $O(\Delta x^4)u^{(6)}$ which gives the following values of coefficients:

$$\begin{aligned} a_1 &= 10, \quad a_2 = -15, \quad a_3 = -4, \\ a_4 &= 14, \quad a_5 = -6, \quad a_6 = 1. \end{aligned} \tag{7}$$

Schemes (6) can be combined and expressed in the following matrix form:

$$\mathbf{u}_{xx}^m = \frac{1}{\Delta x^2}\Lambda_h \mathbf{u}^m + \gamma(t^m), \tag{8}$$

where Λ_h is the corresponding triangular and sparse $(N-2)\times(N-2)$ matrix,

$$\begin{aligned} \mathbf{u}_{xx}^m &= \left((u_{xx})_2^m, (u_{xx})_3^m, \ldots, (u_{xx})_{N-1}^m\right)^T, \\ \mathbf{u}^m &= \left(u_2^m, u_3^m, \ldots, u_{N-1}^m\right)^T, \\ \gamma(t^m) &= \left(\gamma_1(t^m), 0, \ldots, 0, \gamma_2(t^m)\right)^T. \end{aligned} \tag{9}$$

Substituting (6) into (4) gives us the following matrix form:

$$\begin{aligned} \left(E - \alpha\Lambda_h\right)\mathbf{u}^{n+1} &= \left(E + \alpha\Lambda_h\right)\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right) + \gamma\left(t^n\right)\right] \\ &\quad + \Delta t\mathbf{f}^{n+1/2}, \end{aligned} \tag{10}$$

where $\alpha = \beta\Delta t/(2\Delta x^2)$, $\mathbf{f}^{n+1/2} = (f_2^{n+1/2}, f_3^{n+1/2}, \ldots, f_{N-1}^{n+1/2})^T$, and E denote the $(N-2)\times(N-2)$ identity matrix. The scheme (10) is FOSs for the heat-conducting problem with Dirichlet boundary condition. The order of approximation is $O(\Delta t^2, \Delta x^4)$ in the uniform norm. The triangular and sparse $(N-2)\times(N-2)$ coefficient matrix in FOSs are time independent; hence, we have to store the inverse of the coefficient matrix $E - \alpha\Lambda_h$ before the time marching in the implementation for computational efficiency.

2.2. Fourth-Order Deferred Correction Schemes. A set of fourth-order deferred correction schemes is based on the well-known Crank-Nikolson type of scheme in the following form:

$$\frac{u_i^{n+1,s+1}-u_i^n}{\Delta t}=\frac{\beta}{2}\left[\left(u_{xx}\right)_i^{n+1,s+1}+\left(u_{xx}\right)_i^n\right]+f_i^{n+1/2}, \tag{11}$$

where $f_i^{n+1/2}=(f_i^{n+1}+f_i^n)/2$ and the second superscript "s" denotes the number of iterations $s=0,\ldots,\widehat{S}$ and $i=2,\ldots,N-1$.

The deferred correction technique [11] is utilized to approximate the second-order derivatives at higher time levels $(u_{xx})_i^{n+1,s+1}$, $i=2,\ldots,N-1$ by the iterative method

$$\left(u_{xx}\right)_i^{n+1,s+1}=\left(u_{xx}^l\right)_i^{n+1,s+1}+\left[\left(u_{xx}^h\right)_i^{n+1,s}-\left(u_{xx}^l\right)_i^{n+1,s}\right], \tag{12}$$

where $(u_{xx}^h)_i^{n+1,s}$, $i=2,\ldots,N-1$, $s=0,\ldots,\widehat{S}$, is high-order approximation on wide stencil and $(u_{xx}^l)_i^{n+1,k}$, $k=s$, $s+1$, $i=2,\ldots,N-1$, is the lower-order approximation on compact stencil (usually three-point stencil). The expression in the square brackets of (12) is evaluated explicitly using the values known from the previous iteration. When $s=0$ we use the solution from the time level n (so $u^{n+1,0}=u^n$ and $(u_{xx})_i^{n+1,0}=(u_{xx})_i^n$). Once the iterations converge, the lower-order approximation terms drop out and the approximation of $(u_{xx})_i^{n+1,s+1}$ obtained has the same order of approximation as $(u_{xx}^h)_i^{n+1,\widehat{S}}$. There are no difficulties to construct high-order approximation for interior points.

To preserve a compact three using wide stencil in the finite difference scheme at higher time level $(n+1,s+1)$, we use the central second-order finite difference approximation to approximate the lower-order term in (12):

$$\left(u_{xx}^l\right)_i^{n+1,k}=\frac{1}{\Delta x^2}\Lambda_l u_i^{n+1,k},\quad k=s,s+1,\quad i=3,\ldots,N-2,$$
$$\Lambda_l u_i^{n+1,k}=u_{i-1}^{n+1,k}-2u_i^{n+1,k}+u_{i+1}^{n+1,k}. \tag{13}$$

For the high-order approximation term in (12), we use a symmetric five-point wide stencil for the inner points to reach the fourth order of approximation:

$$\left(u_{xx}^h\right)_i^{n+1,s}=\frac{1}{\Delta x^2}\Lambda_h u_i^{n+1,s},\quad i=3,\ldots,N-2,$$
$$\Lambda_h u_i^{n+1,s}=\frac{1}{12}\left(-u_{i-2}^{n+1,s}+16u_{i-1}^{n+1,s}-30u_i^{n+1,s}+16u_{i+1}^{n+1,s}-u_{i+2}^{n+1,s}\right). \tag{14}$$

Case $s=0$ in (13) gives the fourth order of approximation to approximate the second-order derivatives at the time level n.

2.2.1. Stability Analysis. To study the stability of scheme (11)–(14), we use the Von-Neumann stability analysis. For simplicity, we assume that $f_i^{n+1/2}\equiv 0$ in (11) and u is periodic in x.

Let us recast scheme (11) in the following form:

$$\left(E+\alpha\Lambda_l\right)u_i^{n+1,s+1}=\alpha\left(\Lambda_l-\Lambda_h\right)u_i^{n+1,s}+\left(E-\alpha\Lambda_l\right)u_i^n, \tag{15}$$

where $\alpha=\beta\Delta t/(2\Delta x^2)$. If we define the following operators $A=E+\alpha\Lambda_l$, $B=E-\alpha\Lambda_h$, and $C=E+\alpha\Lambda_h$, where E is the identity operator, then (15) can be rewritten as follows:

$$Au_i^{n+1,s+1}=(A-C)u_i^{n+1,s}+Bu_i^n. \tag{16}$$

Assuming that the operators commute, $(A-C)A=A(A-C)$ (e.g., in the case of uniform grid), it is easy to demonstrate that if $u_i^{n+1,\widehat{S}+1}=u_i^{n+1}$ and $u_i^{n+1,0}=u_i^n$ we get

$$A^{\widehat{S}+1}u_i^{n+1}=\left(\sum_{k=0}^{\widehat{S}}A^{\widehat{S}-k}(A-C)^k\right)Bu_i^n+(A-C)^{\widehat{S}+1}u_i^n. \tag{17}$$

Let $u_i^n=\xi^n e^{I\Theta i}$, $I=\sqrt{-1}$, be the solution of (11)–(14), where $\Theta=2\pi\Delta x/l$ is the phase angle with wavelength l. From (17), we can derive an equation for the amplification factor in the form

$$\left|\xi\right|=\left|\varphi\left(\Theta,\widehat{S},\alpha\right)\right|, \tag{18}$$

where $\widehat{S}$ is the number of iterations, and

$$\left|\varphi\left(\Theta,\widehat{S},\alpha\right)\right|=\frac{\left|\left[\left(\sum_{k=0}^{\widehat{S}}A^{\widehat{S}-k}(A-C)^k\right)B+(A-C)^{\widehat{S}+1}\right]e^{I\Theta i}\right|}{\left|A^{\widehat{S}+1}e^{I\Theta i}\right|}. \tag{19}$$

For stability of the method it is necessary that the absolute values of the amplification factor are less than one; that is,

$$\left|\xi\right|<1. \tag{20}$$

Calculations are tedious and almost impossible to do by hand without mistake. We have therefore automate all calculations in a computer algebra environment based on REDUCE to obtain an explicit form of $|\varphi(\Theta,\widehat{S},\alpha)|$. Figure 1 shows the values of $|\xi|^2$ in the polar coordinate system $(|\xi|^2,\Theta)$ for $\widehat{S}=1,3,$ and 5. If only one iteration is executed in (11), $\widehat{S}=1$, inequality (20) holds if $\alpha<1.5$, as can be seen from Figure 1(a)). If 3 iterations are done in (11) (Figure 1(b)), $\widehat{S}=3$, the amplification factor remains bounded by one at least for $\alpha\le 10$. In case of $\widehat{S}=5$, the stability criteria hold up to $\alpha=30$ as can be seen from Figure 1(c)). It can be seen that increasing the number of internal iterations results in increasing the range of α needed for stability. This tendency allows to assume that as $\widehat{S}\to\infty$, our method becomes the unconditionally stable Crank-Nikolson method for the heat equation.

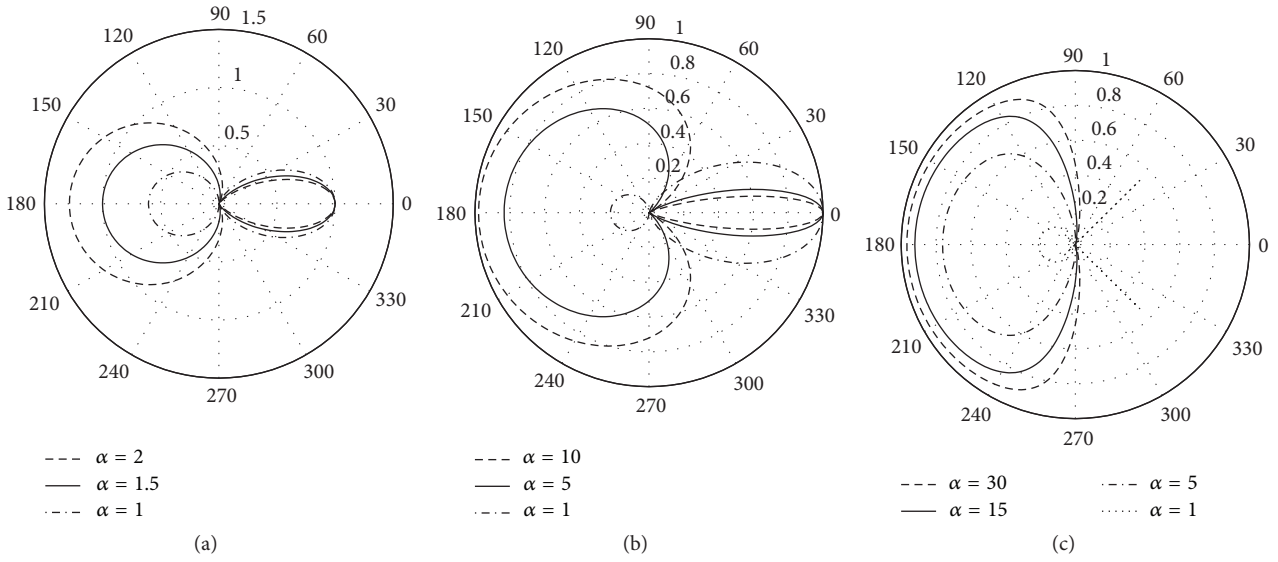


FIGURE 1: Variation of amplification factor with Θ. (a) $\widehat{S} = 1$, dashed line $\alpha = 2.0$, solid line $\alpha = 1.5$, dash-doted line $\alpha = 1.0$, (b) $\widehat{S} = 3$, dashed line $\alpha = 10.0$, solid line $\alpha = 5.0$, dash-dotted line $\alpha = 1.0$, and (c) $\widehat{S} = 5$, dashed line $\alpha = 30.0$, solid line $\alpha = 15$, dash-doted line $\alpha = 5.0$, doted line $\alpha = 1.0$.

2.2.2. Fourth-Order Deferred Correction Scheme. Let us first consider the one-dimensional heat conduction problem with initial data and Dirichlet boundary conditions (1)–(3):

$$u_1^{n+1,k} = \gamma_1\left(t^{n+1}\right), \qquad u_N^{n+1,k} = \gamma_2\left(t^{n+1}\right). \tag{21}$$

The finite difference approximations at x_2 and x_{N-1}, which are the points next to the left and right boundaries, are straightforward:

$$\begin{aligned}
\left(u_{xx}^{l}\right)_2^{n+1,k} &= \frac{1}{\Delta x^2}\left(\gamma_1\left(t^{n+1}\right) - 2u_2^{n+1,k} + u_3^{n+1,k}\right), \\
&\qquad k = s, s+1, \\
\left(u_{xx}^{h}\right)_2^{n+1,s} &= \frac{1}{12\Delta x^2}\left(10\gamma_1\left(t^{n+1}\right) - 15u_2^{n+1,s} - 4u_3^{n+1,s}\right. \\
&\qquad \left. +14u_4^{n+1,s} - 6u_5^{n+1,s} + u_6^{n+1,s}\right), \\
\left(u_{xx}^{h}\right)_{N-1}^{n+1,s} &= \frac{1}{12\Delta x^2}\left(10\gamma_2\left(t^{n+1}\right) - 15u_{N-1}^{n+1,s} - 4u_{N-2}^{n+1,s}\right. \\
&\qquad \left. +14u_{N-3}^{n+1,s} - 6u_{N-4}^{n+1,s} + u_{N-5}^{n+1,s}\right), \\
\left(u_{xx}^{l}\right)_{N-1}^{n+1,k} &= \frac{1}{\Delta x^2}\left(u_{N-2}^{n+1,k} - 2u_{N-1}^{n+1,k} + \gamma_2\left(t^{n+1}\right)\right), \\
&\qquad k = s, s+1.
\end{aligned} \tag{22}$$

Cases $s = 0$ or $k = 0$ give formulae to approximate $(u_{xx}^{l})_i^n$ and $(u_{xx}^{h})_i^n$. Substituting (13), (14), and (22) into (12) the following fourth-order deferred correction approximations of $(u_{xx})_i^{n+1,s+1}$, $i = 2, \ldots, N-1$, are

$$\begin{aligned}
\left(u_{xx}\right)_2^{n+1,s+1} &= \frac{5}{6\Delta x^2}\gamma_1\left(t^{n+1}\right) \\
&\quad + \frac{1}{\Delta x^2}\left(-2u_2^{n+1,s+1} + u_3^{n+1,s+1}\right) \\
&\quad + \frac{1}{12\Delta x^2}\left(9u_2^{n+1,s} - 16u_3^{n+1,s} + 14u_4^{n+1,s}\right. \\
&\qquad \left. - 6u_5^{n+1,s} + u_6^{n+1,s}\right), \\
\left(u_{xx}\right)_i^{n+1,s+1} &= \frac{1}{\Delta x^2}\left(u_{i-1}^{n+1,s+1} - 2u_i^{n+1,s+1} + u_{i+1}^{n+1,s+1}\right) \\
&\quad + \frac{1}{12\Delta x^2}\left(-u_{i-2}^{n+1,s} + 4u_{i-1}^{n+1,s} - 6u_i^{n+1,s}\right. \\
&\qquad \left. + 4u_{i+1}^{n+1,s} - u_{i+2}^{n+1,s}\right), \\
&\qquad i = 3, \ldots, N-2, \\
\left(u_{xx}\right)_{N-1}^{n+1,s+1} &= \frac{5}{6\Delta x^2}\gamma_2\left(t^{n+1}\right) \\
&\quad + \frac{1}{\Delta x^2}\left(-2u_{N-1}^{n+1,s+1} + u_N^{n+1,s+1}\right) \\
&\quad + \frac{1}{12\Delta x^2}\left(9u_{N-1}^{n+1,s} - 16u_{N-2}^{n+1,s} + 14u_{N-3}^{n+1,s}\right. \\
&\qquad \left. -6u_{N-4}^{n+1,s} + u_{N-5}^{n+1,s}\right).
\end{aligned} \tag{23}$$

Schemes (23) can be combined and expressed in the following matrix form:

$$\mathbf{u}_{xx}^{n+1,s+1} = \frac{1}{\Delta x^2}\Lambda_l \mathbf{u}^{n+1,s+1} + \frac{1}{\Delta x^2}\left(\Lambda_h - \Lambda_l\right)\mathbf{u}^{n+1,s} + \gamma\left(t^{n+1}\right), \tag{24}$$

where Λ_l is a tridiagonal $(N-2)\times(N-2)$ matrix and Λ_h is the corresponding triangular and sparse $(N-2)\times(N-2)$ matrix,

$$\mathbf{u}^{n+1,k} = \left(u_2^{n+1,k}, u_3^{n+1,k}, \ldots, u_{N-1}^{n+1,k}\right)^T, \quad k = s, s+1, \tag{25}$$

$$\mathbf{u}_{xx}^{n+1,s+1} = \left((u_{xx})_2^{n+1,s+1}, (u_{xx})_3^{n+1,s+1}, \ldots, (u_{xx})_{N-1}^{n+1,s+1}\right)^T. \tag{26}$$

Substituting (6), (23) into (11), the formulae can be written into matrix form

$$\begin{aligned}(E-\alpha\Lambda_l)\,\mathbf{u}^{n+1,s+1} &= \alpha\left(\Lambda_h-\Lambda_l\right)\mathbf{u}^{n+1,s} + \left(E+\alpha\Lambda_h\right)\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right)+\gamma\left(t^n\right)\right] + \Delta t\mathbf{f}^{n+1/2}.\end{aligned} \tag{27}$$

The above matrix form is called FODs for Dirichlet boundary value problem (1)–(3). Thomas algorithms can be used to compute the solutions of FODs. At each step of time t^n and the initial stage, the convergence of FODs requires more iterations to converge to the solution of the FOSs. The order of approximation of FODs is $O(\Delta t^2, \Delta x^4)$ which is the same as FOSs in the uniform norm.

2.3. Fourth-Order Compact Scheme. Let us briefly represent the main idea and final formulae of compact schemes. Spatial derivatives in the Crank-Nikolson scheme (4) are evaluated by the fourth-order compact finite differences implicit scheme [5, 7, 8, 13, 14, 17].

In [8, 14], the Dirichlet boundary conditions

$$u(0, m\Delta t) = \gamma_1\left(t^m\right) = u_1^m, \qquad u(l, m\Delta t) = \gamma_2\left(t^m\right) = u_N^m \tag{28}$$

are used to derive the following fourth-order schemes

$$\begin{aligned}&(u_{xx})_2^m + \sigma(u_{xx})_3^m \\ &\quad = \frac{1}{24\Delta x^2}\left(a_1u_1^m + a_2u_2^m + a_3u_3^m + a_4u_4^m + a_5u_5^m + a_6u_6^m\right) \\ &\quad = \frac{a_1}{24\Delta x^2}\gamma_1\left(t^m\right) + \frac{1}{24\Delta x^2}\left(a_2u_2^m + a_3u_3^m + a_4u_4^m + a_5u_5^m + a_6u_6^m\right), \\ &(u_{xx})_{i-1}^m + 10(u_{xx})_i^m + (u_{xx})_{i+1}^m \\ &\quad = \frac{2}{\Delta x^2}\left(6u_{i-1}^m - 12u_i^m + 6u_{i+1}^m\right), \\ &\qquad i = 2, \ldots, N-1, \\ &(u_{xx})_{N-1}^m + \sigma(u_{xx})_{N-2}^m \\ &\quad = \frac{1}{24\Delta x^2}\left(a_1u_N^m + a_2u_{N-1}^m + a_3u_{N-2}^m + a_4u_{N-3}^m + a_5u_{N-4}^m + a_6u_{N-5}^m\right) \\ &\quad = \frac{a_1}{24\Delta x^2}\gamma_2\left(t^m\right) + \frac{1}{24\Delta x^2}\left(a_2u_{N-1}^m + a_3u_{N-2}^m + a_4u_{N-3}^m + a_5u_{N-4}^m + a_6u_{N-5}^m\right),\end{aligned} \tag{29}$$

where the coefficients can be found by matching the Taylor series expansion of left-hand-side terms up to order $O(\Delta x^4)u^{(6)}$ which gives the following values of coefficients [8]:

$$\sigma = \frac{1}{2}, \qquad a_1 = 19, \qquad a_2 = -14, \qquad a_3 = -38, \qquad a_4 = 44, \qquad a_5 = -13, \qquad a_6 = 2. \tag{30}$$

Then all derivatives in (4) are approximated by the fourth-order compact formula; we can write

$$A\mathbf{u}_{xx}^m = \frac{1}{\Delta x^2}B\mathbf{u}^m + \gamma^m, \quad m = n, n+1, \tag{31}$$

where A and B are the corresponding triangular and sparse $(N-2)\times(N-2)$ matrices, $\mathbf{u}_{xx}^m = ((u_{xx})_2^m, (u_{xx})_3^m, \ldots, (u_{xx})_{N-1}^m)^T$, $\mathbf{u}^m = (u_2^m, u_3^m, \ldots, u_{N-1}^m)^T$ and $\gamma^m = (\gamma_1(t^m), 0, \ldots, 0, \gamma_2(t^m))^T$, $m = n, n+1$. Schemes (4) and (29) can be combined and expressed in the following matrix form:

$$\begin{aligned}(A-\alpha B)\,\mathbf{u}^{n+1} &= (A+\alpha B)\,\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right)+\gamma\left(t^n\right)\right] \\ &\quad + \Delta t\mathbf{f}^{n+1/2}.\end{aligned} \tag{32}$$

This scheme is called FOCs for Dirichlet boundary value problem (1)–(3). We like to mention that the above scheme has truncation error $O(\Delta t^2, \Delta x^4)$. Note that the triangular and sparse $(N-2)\times(N-2)$ coefficient matrices in FOCs are time independent; hence, we have to store the inverse of the coefficient matrix $A-\alpha B$ before the time marching in the implementation of computational efficiency.

3. Numerical Examples

In this section, three numerical examples are carried out. The first two are linear heat-conducting problem, with Dirichlet

boundary conditions, which are used to confirm our theoretical analysis. Then we apply the FODS to the Burgers equation. For simplicity, we fix our problem domain $\Omega = \{x \mid 0 \le x \le 1\}$. In all computations, we used $\Delta t = \Delta x^2/4$ and $\epsilon = 10^{-10}$. The following stopping criterion is used:

$$\max_{1\le i\le N} \left| u_i^{n+1,\widehat{S}+1} - u_i^{n+1,\widehat{S}} \right| < \epsilon, \quad s = 0, \ldots, \widehat{S}, \tag{33}$$

where "$\widehat{S}$" denotes the number of the last iteration.

The computations are performed using uniform grids of 11, 21, 41, 81, and 161 nodes. The initial and boundary conditions are obtained based on the exact solutions. For the testing purpose only, all computations are performed for $0 \le t \le 1$.

Example 1 (the homogeneous heat equation with the homogeneous Dirichlet boundary conditions). One has

$$\begin{aligned} u_t &= u_{xx}, \quad 0 \le x \le 1, \ t > 0, \\ u(x,0) &= \sin(\pi x), \quad u(0,t) = 0, \quad u(1,t) = 0. \end{aligned} \tag{34}$$

The exact solution is $u(x,t) = e^{-\pi^2 t}\sin(\pi x)$. The results of performance over the time interval $t \in [0,1]$ for the FOCs, FODs, and FOSs are represented in Table 1, where the maximum error and the rate of convergence at time instant $t = 1$ are shown.

Example 2 (the nonhomogeneous heat equation with non-homogeneous Dirichlet boundary conditions). One has

$$\begin{aligned} u_t &= u_{xx} + \left(\pi^2 - 1\right) e^{-t}\cos(\pi x) \\ &\quad + 4x - 2, \quad 0 \le x \le 1, \ t > 0, \\ u(x,0) &= \cos(\pi x) + x^2, \quad u(0,t) = e^{-t}, \\ u(1,t) &= -e^{-t} + 4t + 1. \end{aligned} \tag{35}$$

The exact solution is $u(x,t) = e^{-t}\cos(\pi x) + x^2 + 4xt$. The results of performance over the time domain $t \in [0,1]$ for the FOC, FOD, and FOS schemes are represented in Table 2, where the maximum error and the rate of convergence at time instant $t = 1$ are shown.

The last two columns of Tables 1 and 2 demonstrate the average number of iterations in FODs at one time step and the CPU time required to obtain the solution at time instant $t = 1$. The average number of iterations means the total number of iterations divided by the number of time steps. As a rule, at the initial stage the convergence of deferred correction requires more iterations. For larger instants of time, the convergence occurs after 2~7 iterations as can be seen from Tables 1 and 2. All of schemes are seen to be the fourth order of accuracy, as the error is reduced approximately by factor four when the mesh is refined by half. The maximum error of the FODs and FOCs is almost the same, since the iterative scheme FODs is constructed by applying the deferred correction technique on the FOSs. It can be stated that when the iterations converge, the solution of FODs, therefore, converges to the solution of FOSs in each step of time. As shown in Tables 1 and 2, there is hardly a difference in the computational efficiency between FODs and FOSs. Both schemes are more efficient than FODs. An explanation is due to the iteration needed for the convergence of solutions on each step of time.

Although the FODs use more computational time as compared with FOCs and FOSs, it is recommended that the construction of FODs can be easily implemented. Moreover, the scheme does not need to store the inverse of coefficient matrices as required in FOCs and FOSs. Therefore, the method is easily extended to multidimensional cases.

It is suggested that the differed correction technique can solve problems which need high accuracy of computational methods. Also this technique can be easily implemented and extended for solving problem with Neumann boundary conditions. In addition, such technique can be easily used to create standard code and applied in case of nonuniform grids.

Considering Burgers equation

$$u_t = \beta u_{xx} - u u_x, \quad 0 \le x \le 1, \ t > 0, \tag{36}$$

with the exact solution [21] is given by

$$u(x,t) = \frac{\xi + \eta + (\eta - \xi)\, e^{\rho}}{1 + e^{\rho}}, \tag{37}$$

where $\rho = \xi(x - \eta t - \upsilon)/\beta$. The initial and Dirichlet boundary conditions are considered to be in agreement with the exact solution proposed here. For Burgers equation (36), we solve it by the following fourth-order deferred correction scheme:

$$\frac{u_i^{n+1,s+1} - u_i^n}{\Delta t} = \frac{\beta}{2}\left[\left(u_{xx}\right)_i^{n+1,s+1} + \left(u_{xx}\right)_i^n \right] + f_i^n, \tag{38}$$

where $f_i^n = -[(u^2/2)_x]_i^n$. The nonlinear term f_i^n is approximated with the fourth-order approximation and all the second-derivative terms in (38) are approximated by the fourth-order formula (6) and the fourth-order deferred correction schemes (23). The scheme (38) can be combined and expressed in the following matrix form:

$$\begin{aligned} \left(E - \alpha\Lambda_l\right)\mathbf{u}^{n+1,s+1} &= \alpha\left(\Lambda_h - \Lambda_l\right)\mathbf{u}^{n+1,s} + \left(E + \alpha\Lambda_h\right)\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right) + \gamma\left(t^n\right)\right] + \Delta t\mathbf{f}^n, \end{aligned} \tag{39}$$

where E is identity matrix, Λ_l is tridiagonal $(N-2)\times(N-2)$ matrix, and Λ_h is the corresponding triangular and sparse $(N-2)\times(N-2)$ matrix and can be solved by using Thomas algorithm.

Example 3 (the Burgers equation (36) and the constant values $\nu = 0.125$, $\xi = 0.6$, $\eta = 0.4$, and $\beta = 1$ with appropriate initial and Dirichlet boundary condition in agreement with exact solution (37)). This problem was solved using different time step and mesh sizes over the time interval $0 < t \le 1$. The results of performance over the time interval $t \in [0,1]$ for the FODs are represented in Tables 3 and 4, where the maximum error and the rate of convergence at time instant $t = 1$ are shown.

Table 1: Maximum absolute error, order of convergence, and CPU time in seconds of the FOCs, FODs, and FOSs for test problem (34) at time instant $t = 1$.

Types of scheme	Grid points	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FOCs	11	3.8687×10^{-8}	—	1	0.006
	21	6.0426×10^{-10}	6.0005	1	0.042
	41	2.2454×10^{-11}	4.7501	1	0.326
	81	1.2821×10^{-12}	4.1304	1	2.564
	161	8.0164×10^{-14}	3.9994	1	20.437
FODs	11	9.9767×10^{-9}	—	4	0.015
	21	1.4996×10^{-9}	2.7361	3	0.085
	41	1.1193×10^{-10}	3.7438	2	0.438
	81	7.1438×10^{-12}	3.9698	2	3.450
	161	4.4797×10^{-13}	4.1875	2	27.495
FOSs	11	9.9763×10^{-9}	—	1	0.006
	21	1.4996×10^{-9}	2.7361	1	0.043
	41	1.1193×10^{-10}	3.7438	1	0.334
	81	7.1440×10^{-12}	3.9698	1	2.623
	161	4.4854×10^{-13}	4.1875	1	20.907

Table 2: Absolute error, the rate of convergence, and CPU time in seconds of the FOCs, FODs, and FOSs for the test problem (35) at time instant $t = 1$.

Types of scheme	Grid points	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FOCs	11	1.8470×10^{-5}	—	1	0.006
	21	3.6901×10^{-7}	5.6454	1	0.046
	41	7.5595×10^{-9}	5.6092	1	0.353
	81	6.6458×10^{-10}	3.5077	1	2.778
	161	4.8841×10^{-11}	3.7663	1	22.141
FODs	11	1.7132×10^{-5}	—	7	0.016
	21	2.6914×10^{-7}	5.9922	7	0.128
	41	2.7910×10^{-8}	3.2655	6	0.851
	81	2.0112×10^{-9}	3.7941	5	5.568
	161	1.3116×10^{-10}	3.9415	5	44.375
FOSs	11	1.2895×10^{-5}	—	1	0.006
	21	2.8544×10^{-7}	5.9922	1	0.046
	41	2.7306×10^{-8}	3.2655	1	0.359
	81	1.9590×10^{-9}	3.7941	1	2.821
	161	1.3130×10^{-10}	3.9415	1	22.484

Table 3: Maximum absolute error, order of convergence, and CPU time in seconds for Example 3 at time instant $t = 1$ with fixed mesh size $\Delta x = 0.05$.

Types of scheme	Time step sizes	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FODs	10^{-2}	8.0945×10^{-6}	—	10	0.015
	10^{-3}	8.0942×10^{-7}	1.0000	6	0.109
	10^{-4}	8.1144×10^{-8}	0.9989	3	0.656
	10^{-5}	8.2989×10^{-9}	0.9902	3	6.281

Table 4: Maximum absolute error, order of convergence, and CPU time in seconds for Example 3 at time instant $t = 1$ with time step size $\Delta t = \Delta x^4$.

Types of scheme	Grid points	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FODs	11	8.3375×10^{-8}	—	3	0.516
	21	5.2632×10^{-9}	3.9812	2	8.594
	41	3.3491×10^{-10}	3.9774	2	200.015

In order to analyze the results found in application to the Burgers equation (36), Table 3 demonstrates rate of convergence, average number of iteration at each time step, and CPU time required to obtain the solution of Example 3 by using FODs at time instant $t = 1$ when $\Delta x = 0.05$ with various time step sizes. Table 4 shows the rate of convergence, average number of iteration at each time step, and CPU time required to obtain the solution of Example 3 at time instant $t = 1$ and using uniform grids of 11, 21, and 41 with time step sizes $\Delta t = \Delta x^4$ and $\epsilon = 10^{-10}$.

It can be seen from Tables 3 and 4 that numerical results are in good agreement with the exact solution. We only observe $O(\Delta t)$ convergence rate and the error is dominated by time error. An explanation for this phenomenon is due to the nonlinear term, which is approximated at time level n, instead of at time level $n + 1/2$ for the FODs (38).

4. Conclusion

In this paper, a new set of fourth-order schemes for the one-dimensional heat conduction problem with Dirichlet boundary conditions is constructed using a deferred correction technique. The construction of high-order deferred correction schemes requires only a regular three-point stencil at higher time level which is similar to the standard second-order Crank-Nikolson method. The greatest significance of FODs, compared with FOCs and FOSs, is the easier development and that it can be solved by using Thomas algorithms. Numerical examples confirm the order of accuracy. We also implement our algorithms to nonlinear problems. However, theoretical analysis for nonlinear problems needs further investigation. Posterior idea for this project is to use another way to make uu_x term as follows [21, 22]:

$$u^{n+1}(u_x)_i^{n+1} \approx u^{n+1}(u_x)_i^n + u^n(u_x)_i^{n+1} - u^n(u_x)_i^n, \qquad (40)$$

where better results are expected to be found. The first two terms on the right-hand side of above equation make the coefficient matrices of FOCs, FODs, and FOSs vary with time. That is, the inverse coefficient matrices of FOCs and FOSs have to be stored on each step of time while FODs have no need. For this reason, the FODs is simple to implement although FODs need more iterations for the convergence of solution on each step of time.

Acknowledgments

This work is financially supported by the Commission on Higher Education (CHE), the Thailand Research Fund (TRF), the University of Phayao (UP), Project MRG5580014. The author would like to express their deep appreciation to Professor Sergey Meleshko for the kind assistance and valuable advice on the REDUCE calculations.

References

[1] Y. Adam, "Highly accurate compact implicit methods and boundary conditions," *Journal of Computational Physics*, vol. 24, no. 1, pp. 10–22, 1977.

[2] G. F. Carey and W. F. Spotz, "Higher-order compact mixed methods," *Communications in Numerical Methods in Engineering with Biomedical Applications*, vol. 13, no. 7, pp. 553–564, 1997.

[3] M. H. Carpenter, D. Gottlieb, and S. Abarbanel, "Stable and accurate boundary treatments for compact, high-order finite-difference schemes," *Applied Numerical Mathematics*, vol. 12, no. 1–3, pp. 55–87, 1993.

[4] I. Christie, "Upwind compact finite difference schemes," *Journal of Computational Physics*, vol. 59, no. 3, pp. 353–368, 1985.

[5] P. C. Chu and C. Fan, "A three-point combined compact difference scheme," *Journal of Computational Physics*, vol. 140, no. 2, pp. 370–399, 1998.

[6] P. C. Chu and C. Fan, "A three-point sixth-order nonuniform combined compact difference scheme," *Journal of Computational Physics*, vol. 148, no. 2, pp. 663–674, 1999.

[7] W. Dai and R. Nassar, "A compact finite difference scheme for solving a three-dimensional heat transport equation in a thin film," *Numerical Methods for Partial Differential Equations*, vol. 16, no. 5, pp. 441–458, 2000.

[8] W. Dai and R. Nassar, "Compact ADI method for solving parabolic differential equations," *Numerical Methods for Partial Differential Equations*, vol. 18, no. 2, pp. 129–142, 2002.

[9] W. Dai, "A new accurate finite difference scheme for Neumann (insulated) boundary condition of heat conduction," *International Journal of Thermal Sciences*, vol. 49, no. 3, pp. 571–579, 2010.

[10] X. Deng and H. Maekawa, "Compact high-order accurate nonlinear schemes," *Journal of Computational Physics*, vol. 130, no. 1, pp. 77–91, 1997.

[11] J. H. Ferziger and M. Peric, *Computational Methods in Fluid Dynamics*, Springer, Berlin, Germany, 2002.

[12] J. C. Kalita, D. C. Dalal, and A. K. Dass, "A class of higher order compact schemes for the unsteady two-dimensional convection-diffusion equation with variable convection coefficients," *International Journal for Numerical Methods in Fluids*, vol. 38, no. 12, pp. 1111–1131, 2002.

[13] S. Karaa and J. Zhang, "High order ADI method for solving unsteady convection-diffusion problems," *Journal of Computational Physics*, vol. 198, no. 1, pp. 1–9, 2004.

[14] S. K. Lele, "Compact finite difference schemes with spectral-like resolution," *Journal of Computational Physics*, vol. 103, no. 1, pp. 16–42, 1992.

[15] J. Li, Y. Chen, and G. Liu, "High-order compact ADI methods for parabolic equations," *Computers & Mathematics with Applications*, vol. 52, no. 8-9, pp. 1343–1356, 2006.

[16] I. M. Navon and H. A. Riphagen, "An implicit compact fourth-order algorithms for solving the shallow-water equations in conservation-law form," *Monthly Weather Review*, vol. 107, no. 9, pp. 1107–1127, 1979.

[17] J. Zhao, W. Dai, and T. Niu, "Fourth-order compact schemes of a heat conduction problem with Neumann boundary conditions," *Numerical Methods for Partial Differential Equations*, vol. 23, no. 5, pp. 949–959, 2007.

[18] J. Zhao, W. Dai, and S. Zhang, "Fourth-order compact schemes for solving multidimensional heat problems with Neumann boundary conditions," *Numerical Methods for Partial Differential Equations*, vol. 24, no. 1, pp. 165–178, 2008.

[19] V. Pereyra, "On improving an approximate solution of a functional equation by deferred corrections," *Numerische Mathematik*, vol. 8, pp. 376–391, 1966.

[20] V. Pereyra, "Iterated deferred corrections for nonlinear boundary value problems," *Numerische Mathematik*, vol. 11, pp. 111–125, 1968.

[21] M. C. Miriane, M. G. Paulo, and C. R. Estaner, "Error analysis in the solution of unsteady nonlinear convection-diffusion problems," *Research Journal of Physical and Applied Science*, vol. 1, no. 1, pp. 20–22, 2012.

[22] B. Jiang, *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics*, Scientific Computation, Springer, Berlin, Germany, 1998.

Program list for test Problems

Generated Grids and Test Problem Used

- Generated Grids

```
function[gx] = grid(nx,hx);
for i=1:nx
    gx(i,1) = hx + (i-1)*hx;
end
```

- Exact solution for test Problem

```
function[ex] = exact(ti,nx,v1);
k1=exp(-ti);
for i=1:nx
    ex(i,1)=(k1*cos(pi*v1(i,1))) + v1(i,1)^2 +(4*ti*v1(i,1));
end
```

```
function[ex] = exact(ti,nx,v1);
k1=exp(-pi^2*ti);
for i=1:nx
    ex(i,1)=k1*sin( pi*v1(i,1) );
end
```

```
function[ex] = exact(ti,nx,v1);
for i=1:nx
    ex(i,1)=2*v1(i,1)/( 1+2*ti );
end
```

```
function[ex] = exact(ti,nx,c1,c2,c3,Re,v1);
for i=1:nx
    c4 = c1*Re*(v1(i,1)-c2*ti-c3);
    c5 = exp(c4);
    ex(i,1)=( (c1+c2)+(c2-c1)*c5 )/(1+c5);
end
```

Main Program

DCNS2: Dirichlet Boundary Condition, Second-Order Finite Difference Scheme

DHCS4: Dirichlet Boundary Condition, Fourth-Order Compact Finite Difference Scheme

DHDS4: Dirichlet Boundary Condition, Fourth-Order Deferred correction
Finite Difference Scheme

NCNS2: Neuman Boundary Condition, Second-Order Finite Difference Scheme

NHDS4: Neuman Boundary Condition, Fourth-Order Deferred correction
Finite Difference Scheme

NHCS4: Neuman Boundary Condition, Fourth-Order Compact Finite Difference Scheme

DCNS2: Dirichlet Boundary Condition, Second-Order Finite Difference Scheme

- Main Program

```
clc,clear
format long
l  = 1;
n  = 11;
nx = n-2;
hx = 1./(n-1);
dt = hx.^2/10;
r = dt./(2.*hx.^2);
time = 1.;
eps  = 1.0e-10;
nit = round(time/dt);

A = zeros(nx,nx);   C  = zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
a1 = zeros(nx,1);   a2 = zeros(nx,1);   a3 = zeros(nx,1);
li = zeros(nx,1);

gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)= 2;         A(i,i+1)=-1;
    elseif i == nx
        A(i,i-1)=-1;     A(i,i)= 2;
    else
        A(i,i-1)=-1;     A(i,i)= 2;     A(i,i+1)=-1;
    end
end

C  = E+r*A;
nt  = 0;
ti  = 0.;
err = 0.;
k1  = (pi.^2)-1.;
u0  = exact(ti,nx,gx);
un  = u0;

while ti < time
    err  = 1.;
    nt   = nt+1;
    ti   = nt*dt;        ki   = k1*exp(-ti);
    th   = ti-(dt/2);    kh   = k1*exp(-th);
    tp   = ti-dt;        kp   = k1*exp(-tp);
    ex   = exact(ti,nx,gx);
    ap1 =  exp(-ti);
    ap2 = -exp(-ti)+(4*ti)+1;
    for i=1:nx;
        fi(i,1) = dt*( kh*cos(pi*gx(i,1)) + 4*gx(i,1) - 2 );
        if i == 1
            gi(i,1) = un(1,1)+fi(1,1) ...
                      +r*( ap1-2*un(1,1)+un(2,1) );
```

```
        elseif i == nx
            gi(nx,1)= un(nx,1)+fi(nx,1) ...
                      +r*( ap2-2*un(nx,1)+un(nx-1,1) );
        else
            gi(i,1) = un(i,1)+fi(i,1) ...
                      +r*( un(i-1,1)-2*un(i,1)+un(i+1,1) );
        end
    end

    ap1 =  exp(-tp);
    ap2 = -exp(-tp)+(4*tp)+1;
    for i=1:nx;
        if i == 1
            di(i,1) = r*ap1;
        elseif i == nx
            di(i,1) = r*ap2;
        else
            di(i,1) = 0.0;
        end
    end
    li   = gi + di;

    % Using TDMA
    for i=1:nx;
        if i == 1
            a1(i,1)= 0;         a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        elseif i == nx
            a1(i,1)=C(i,i-1);   a2(i,1)= C(i,i);    a3(i,1)= 0;
        else
            a1(i,1)=C(i,i-1);   a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        end
    end
    a3(1,1) = a3(1,1)/a2(1,1);
    li(1,1) = li(1,1)/a2(1,1);
    for i = 2:nx-1
        temp    = a2(i,1)-(a1(i,1)*a3(i-1,1));
        a3(i,1) = a3(i,1)/temp;
        li(i,1) = ( li(i,1)-(a1(i,1)*li(i-1,1)) )/temp;
    end
    li(nx,1) = ( li(nx,1)  - (a1(nx,1)*li(nx-1,1) ) )/ ...
               ( a2(nx,1) - (a1(nx,1)*a3(nx-1,1)) );
    % Now back substitute.
    unp1(nx,1) = li(nx,1);
    for i = nx-1:-1:1
        unp1(i,1) = li(i,1) - (a3(i,1)*unp1(i+1,1));
    end

% Check err
    err = max(abs(unp1-un));
    un  = unp1;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
end
toc
er(nit,1)
figure(1);
semilogy(t,er,'r-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-12 1.0E-02])
```

DHCS4: Dirichlet Boundary Condition, Fourth-Order Compact Finite Difference Scheme

- Main Program

```
clc,clear
format short
l  = 1;
n  = 161;
nx = n-2;
hx = 1/(n-1);
dt = hx^2/4;
r  = dt/hx^2;
time = 1;
nit = round(time/dt);

A  = zeros(nx,nx);  B = zeros(nx,nx);   C  = zeros(nx,nx);
D  = zeros(nx,nx);  IC= zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
li = zeros(nx,1);

gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)  =  2;    A(i,i+1)= 1;
        B(i,i)  = 7/12;  B(i,i+1)= 19/12;   B(i,i+2)=-11/6;
        B(i,i+3)= 13/24; B(i,i+4)=-1/12;
    elseif i == nx
        A(i,i)  =  2;    A(i,i-1)= 1;
        B(i,i)  = 7/12;  B(i,i-1)= 19/12;   B(i,i-2)=-11/6;
        B(i,i-3)= 13/24; B(i,i-4)=-1/12;
    else
        A(i,i-1)= 1;     A(i,i)=10;     A(i,i+1)= 1;
        B(i,i-1)=-6;     B(i,i)=12;     B(i,i+1)=-6;
    end
end
 C = A+r*B; D = A-r*B;  IC= inv(C);
nt  = 0;
ti  = 0.;
err = 0.;
k1  = (pi.^2)-1.;
u0  = exact(ti,nx,gx);
un  = u0;
while ti < time
    err  = 10.;          nt   = nt+1;
    ti   = nt*dt;        ki   = k1*exp(-ti);
    th   = ti-(dt/2);    kh   = k1*exp(-th);
    tp   = ti-dt;        kp   = k1*exp(-tp);
    ex   = exact(ti,nx,gx);
    ap1 =  exp(-th);
    ap2 = -exp(-th)+(4*th)+1;
    for i=1:nx;
        fi(i,1) = dt*( kh*cos(pi*gx(i,1)) + 4*gx(i,1) - 2 );
    end
```

```
    for i=1:nx;
        if i == 1
            gi(1,1) = D(1,1)*un(1,1) + D(1,2)*un(2,1) ...
                    + D(1,3)*un(3,1) + D(1,4)*un(4,1) ...
                    + D(1,5)*un(5,1) ...
                    + A(1,1)*fi(1,1) + A(1,2)*fi(2,1) ...
                    + A(1,3)*fi(3,1) + A(1,4)*fi(4,1) ...
                    + A(1,5)*fi(5,1);
            di(1,1) = 19*r*ap1/12;
        elseif i == nx
            gi(nx,1)= D(nx,nx)*un(nx,1)   + D(nx,nx-1)*un(nx-1,1) ...
                  + D(nx,nx-2)*un(nx-2,1) + D(nx,nx-3)*un(nx-3,1) ...
                  + D(nx,nx-4)*un(nx-4,1) ...
                  + A(nx,nx)*fi(nx,1)     + A(nx,nx-1)*fi(nx-1,1) ...
                  + A(nx,nx-2)*fi(nx-2,1) + A(nx,nx-3)*fi(nx-3,1) ...
                  + A(nx,nx-4)*fi(nx-4,1);
            di(nx,1)= 19*r*ap2/12;
        else
            gi(i,1) = D(i,i-1)*un(i-1,1) + D(i,i)*un(i,1) ...
                    + D(i,i+1)*un(i+1,1) ...
                    + A(i,i-1)*fi(i-1,1) + A(i,i)*fi(i,1) ...
                    + A(i,i+1)*fi(i+1,1);
            di(i,1) = 0.0;
        end
    end
    li   = gi+di;
    unp1 = IC*li;
    un   = unp1;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
end
toc
er(nit,1)
figure(1);
semilogy(t,er,'k-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-12 1.0E-04])
```

DHDS4: Dirichlet Boundary Condition, Fourth-Order Deferred correction

Finite Difference Scheme

- Main Program 1

```
clc,clear
format short
l  = 1;
n  = 11;
nx = n-2;
hx = 1./(n-1);
dt = hx.^2/4;
r = dt./(2.*hx.^2);
time = 1;
eps  = 1.0e-10;
nit = round(time/dt);

A = zeros(nx,nx);   C  = zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
a1 = zeros(nx,1);   a2 = zeros(nx,1);   a3 = zeros(nx,1);
li = zeros(nx,1);

gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)=2;           A(i,i+1)=-1;
    elseif i == nx
        A(i,i-1)=-1;        A(i,i)=2;
    else
        A(i,i-1)=-1;        A(i,i)=2;           A(i,i+1)=-1;
    end
end

C  = E+r*A;

np  = 0;
nt  = 0;
ti  = 0.;
k1  = (pi.^2)-1.;
u0  = exact(ti,nx,gx);
un  = u0;
it  = 0;
itp = 0;

while ti < time
    err  = 1.;
    nt   = nt+1;
    ti   = nt*dt;       ki   = k1*exp(-ti);
    th   = ti-(dt/2);   kh   = k1*exp(-th);
    tp   = ti-dt;       kp   = k1*exp(-tp);
    ex   = exact(ti,nx,gx);
    ap1 =  exp(-tp);
    ap2 =  -exp(-tp)+(4*tp)+1;
```

```
    for i=1:nx;
        fi(i,1) = dt*( kh*cos(pi*gx(i,1)) + 4*gx(i,1) - 2 );
        if i == 1
            gi(i,1) = u0(1,1)+fi(1,1) ...
                        +r*( 10*ap1-15*u0(1,1)-4*u0(2,1)...
                        +14*u0(3,1)-6*u0(4,1)+u0(5,1) )/12;
        elseif i == 2
            gi(2,1) = u0(2,1)+fi(2,1) ...
                        +r*( -ap1+16*u0(1,1)-30*u0(2,1)...
                        +16*u0(3,1)-u0(4,1) )/12;
        elseif i == nx-1
            gi(nx-1,1) = u0(nx-1,1)+fi(nx-1,1) ...
                        +r*( -ap2+16*u0(nx,1)-30*u0(nx-1,1)...
                        +16*u0(nx-2,1)-u0(nx-3,1) )/12;
        elseif i == nx
            gi(nx,1) = u0(nx,1)+fi(nx,1) ...
                        +r*( 10*ap2-15*u0(nx,1)-4*u0(nx-1,1)...
                        +14*u0(nx-2,1)-6*u0(nx-3,1)+u0(nx-4,1) )/12;
        else
            gi(i,1) = u0(i,1)+fi(i,1) ...
                        +r*( -u0(i-2,1)+16*u0(i-1,1)-30*u0(i,1)...
                        +16*u0(i+1,1)-u0(i+2,1) )/12;
        end
    end

    ap1 =  exp(-ti);
    ap2 =  -exp(-ti)+(4*ti)+1;
    while err > eps
    itp  = itp+1;
        for i=1:nx;
            if i == 1
                di(i,1) = r*( 10*ap1+9*un(1,1)-16*un(2,1)...
                            +14*un(3,1)-6*un(4,1)+un(5,1) )/12;
            elseif i == 2
                di(i,1) = r*( -ap1+4*un(1,1)-6*un(2,1)...
                             +4*un(3,1)-un(4,1) )/12;
            elseif i == nx-1
                di(i,1) = r*( -ap2+4*un(nx,1)-6*un(nx-1,1)...
                             +4*un(nx-2,1)-un(nx-3,1) )/12;
            elseif i == nx
                di(i,1) = r*( 10*ap2+9*un(nx,1)-16*un(nx-1,1)...
                            +14*un(nx-2,1)-6*un(nx-3,1)+un(nx-4,1) )/12;
            else
                di(i,1) = r*( -un(i-2,1)+4*un(i-1,1)-6*un(i,1)...
                             +4*un(i+1,1)-un(i+2,1) )/12;
            end
        end
    li   = gi + di;

% Using TDMA
    for i=1:nx;
        if i == 1
            a1(i,1)= 0;         a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        elseif i == nx
            a1(i,1)=C(i,i-1);   a2(i,1)= C(i,i);    a3(i,1)= 0;
        else
            a1(i,1)=C(i,i-1);   a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        end
    end
    a3(1,1) = a3(1,1)/a2(1,1);
    li(1,1) = li(1,1)/a2(1,1);
```

```
    for i = 2:nx-1
        temp    = a2(i,1)-(a1(i,1)*a3(i-1,1));
        a3(i,1) = a3(i,1)/temp;
        li(i,1) = ( li(i,1)-(a1(i,1)*li(i-1,1)) )/temp;
    end
    li(nx,1) = ( li(nx,1)  - (a1(nx,1)*li(nx-1,1) ) )/ ...
               ( a2(nx,1) - (a1(nx,1)*a3(nx-1,1)) );
    % Now back substitute.
    unp1(nx,1) = li(nx,1);
    for i = nx-1:-1:1
        unp1(i,1) = li(i,1) - (a3(i,1)*unp1(i+1,1));
    end

% Check err
    err = max(abs(unp1-un));
    un  = unp1;
    end
    u0  = un;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
    it        = it+1;
end
toc
itp/it
er(nit,1)
figure(1);
semilogy(t,er,'b-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-12 1.0E-04])
```

- Main Program 2

```
clc,clear
format long
l  = 1;
n  = 11;
nx = n-2;
hx = 1./(n-1);
dt = hx.^2/10;
Re = 1.;
r = dt/(2*Re*hx^2);
rr = dt/(2*hx);
time = 1;
eps  = 1.0e-10;
nit = round(time/dt);

A = zeros(nx,nx);   C  = zeros(nx,nx);  D = zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
a1 = zeros(nx,1);   a2 = zeros(nx,1);   a3 = zeros(nx,1);

gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)=2/3;          A(i,i+1)=-2/3;
        a1(i,1)= 0;          a2(i,1)= A(i,i);    a3(i,1)= A(i,i+1);
    elseif i == nx
        A(i,i-1)=-2/3;       A(i,i)=2/3;
        a1(i,1)=A(i,i-1);    a2(i,1)= A(i,i);    a3(i,1)= 0;
    else
        A(i,i-1)=-1;         A(i,i)=2;         A(i,i+1)=-1;
        a1(i,1)=A(i,i-1);    a2(i,1)= A(i,i);    a3(i,1)= A(i,i+1);
    end
end

C  = E+r*A;


np  = 0;
nt  = 0;
ti  = 0.;
err = 0.;
u0  = exact(ti,nx,gx);
un  = u0;
it  = 0;
itp = 0;

while ti < time
    err = 10.;
    nt  = nt+1;
    ti  = nt*dt;
    th  = ti-(dt/2);
    tp  = ti-dt;
    ex  = exact(ti,nx,gx);
    ap1 =  0.0;
    ap2 =  2/(1+2*tp);
```

```
for i=1:nx;
    if i == 1
        fi(1,1) =-rr*(-3*ap1^2-10*u0(1,1)^2+18*u0(2,1)^2 ...
                      -6*u0(3,1)^2+u0(4,1)^2 )/12;
    elseif i == 2
        fi(2,1) =-rr*( ap1^2-8*u0(1,1)^2 ...
                      +8*u0(3,1)^2-u0(4,1)^2 )/12;
    elseif i == nx-1
        fi(nx-1,1) =-rr*(-ap2^2+8*u0(nx,1)^2 ...
                      -8*u0(nx-2,1)^2+u0(nx-3,1)^2 )/12;
    elseif i == nx
        fi(nx,1) =-rr*( 3*ap2^2+10*u0(nx,1)^2-18*u0(nx-1,1)^2 ...
                      +6*u0(nx-2,1)^2-u0(nx-3,1)^2 )/12;
    else
        fi(i,1) =-rr*( u0(i-2,1)^2-8*u0(i-1,1)^2 ...
                      +8*u0(i+1,1)^2-u0(i+2,1)^2 )/12;
    end
end
for i=1:nx;
    if i == 1
        gi(1,1) = u0(1,1)+fi(1,1) ...
                     +r*( 10*ap1-15*u0(1,1)-4*u0(2,1)...
                         +14*u0(3,1)-6*u0(4,1)+u0(5,1) )/12;
    elseif i == 2
        gi(2,1) = u0(2,1)+fi(2,1) ...
                     +r*( -ap1+16*u0(1,1)-30*u0(2,1)...
                          +16*u0(3,1)-u0(4,1) )/12;
    elseif i == nx-1
        gi(nx-1,1) = u0(nx-1,1)+fi(nx-1,1) ...
                     +r*( -ap2+16*u0(nx,1)-30*u0(nx-1,1)...
                          +16*u0(nx-2,1)-u0(nx-3,1) )/12;
    elseif i == nx
        gi(nx,1) = u0(nx,1)+fi(nx,1) ...
                     +r*( 10*ap2-15*u0(nx,1)-4*u0(nx-1,1)...
                     +14*u0(nx-2,1)-6*u0(nx-3,1)+u0(nx-4,1) )/12;
    else
        gi(i,1) = u0(i,1)+fi(i,1) ...
                     +r*( -u0(i-2,1)+16*u0(i-1,1)-30*u0(i,1)...
                          +16*u0(i+1,1)-u0(i+2,1) )/12;
    end
end

ap1 =  0.0;
ap2 =  2/(1+2*ti);
ga1 =  2/(1+2*ti);
ga2 =  2/(1+2*ti);
while err > eps
itp  = itp+1;
    for i=1:nx;
        if i == 1
            di(1,1) = r*(-(50*ga1*hx/137)+(2041*un(1,1)/1644) ...
                        -(387*un(2,1)/137)+(653*un(3,1)/274) ...
                        -(131*un(4,1)/137)+(257*un(5,1)/1644) );
        elseif i == 2
            di(2,1) = r*( -ap1+4*un(1,1)-6*un(2,1)...
                          +4*un(3,1)-un(4,1) )/12;
        elseif i == nx-1
            di(nx-1,1) = r*( -ap2+4*un(nx,1)-6*un(nx-1,1)...
                          +4*un(nx-2,1)-un(nx-3,1) )/12;
```

```
            elseif i == nx
                di(nx,1) = r*((50*ga2*hx/137)+(2041*un(nx,1)/1644)...
                       -(387*un(nx-1,1)/137)+(653*un(nx-2,1)/274) ...
                       -(131*un(nx-3,1)/137)+(257*un(nx-4,1)/1644) );
            else
                di(i,1) = r*( -un(i-2,1)+4*un(i-1,1)-6*un(i,1)...
                              +4*un(i+1,1)-un(i+2,1) )/12;
            end
        end
    D   = gi + di;

    % Using TDMA
    for i=1:nx;
        if i == 1
            a1(i,1)= 0;         a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        elseif i == nx
            a1(i,1)=C(i,i-1);  a2(i,1)= C(i,i);    a3(i,1)= 0;
        else
            a1(i,1)=C(i,i-1);  a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        end
    end
    a3(1,1) = a3(1,1)/a2(1,1);
    D(1,1)  = D(1,1)/a2(1,1);
    for i = 2:nx-1
        temp    = a2(i,1)-(a1(i,1)*a3(i-1,1));
        a3(i,1) = a3(i,1)/temp;
        D(i,1)  = ( D(i,1)-(a1(i,1)*D(i-1,1)) )/temp;
    end
    D(nx,1) = ( D(nx,1)  - (a1(nx,1)*D(nx-1,1) ) )/ ...
              ( a2(nx,1) - (a1(nx,1)*a3(nx-1,1)) );
    % Now back substitute.
    unp1(nx,1) = D(nx,1);
    for i = nx-1:-1:1
        unp1(i,1) = D(i,1) - (a3(i,1)*unp1(i+1,1));
    end

% Check err
    err = max(abs(unp1-un));
    un  = unp1;
    end
    u0  = un;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
    it        = it+1;
end
toc
itp/it
er(nit,1)
figure(1);
semilogy(t,er,'b-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-07 1.0E-03])
```

NCNS2: Neuman Boundary Condition, Second-Order Finite Difference Scheme

- Main Program

```
clc,clear
format short
l  = 1;
n  = 11;
nx = n-2;
hx = 1/(n-1);
dt = hx^2/10;
r  = dt/(2*hx^2);
rr = dt/hx;
time = 1;
nit = round(time/dt);

A  = zeros(nx,nx);  B = zeros(nx,nx);
C  = zeros(nx,nx);  D  = zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
a1 = zeros(nx,1);   a2 = zeros(nx,1);   a3 = zeros(nx,1);
li = zeros(nx,1);

gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)  = 22;    A(i,i+1)= -4;
        B(i,i)  = 12;    B(i,i+1)= -12;
    elseif i == nx
        A(i,i)  = 22;    A(i,i-1)= -4;
        B(i,i)  = 12;    B(i,i-1)= -12;
    else
        A(i,i)=1;
        B(i,i-1)=-1;     B(i,i)=2;      B(i,i+1)=-1;
    end
end

C = A+r*B; D = A-r*B;

nt  = 0;
ti  = 0.;
err = 0.;
k1  = pi^2/2;
u0  = exact(ti,nx,gx);
un  = u0;

while ti < time
    err = 10.;
    nt  = nt+1;
    ti   = nt*dt;       ki   = k1*exp(-k1*ti);
    th   = ti-(dt/2);   kh   = k1*exp(-k1*th);
    tp   = ti-dt;       kp   = k1*exp(-k1*tp);
    ex  = exact(ti,nx,gx);
    for i=1:nx;
        fi(i,1) = dt*( kh*cos(pi*gx(i,1)) + gx(i,1) - 2 );
    end
```

```
    ga1 = th;
    ga2 = th+2;
    for i=1:nx;
        if i == 1
            gi(1,1) = D(1,1)*un(1,1) + D(1,2)*un(2,1) ...
                    + A(1,1)*fi(1,1) + A(1,2)*fi(2,1);
            di(1,1) =-12*rr*ga1;
        elseif i == nx
            gi(nx,1) = D(nx,nx)*un(nx,1) + D(nx,nx-1)*un(nx-1,1) ...
                     + A(nx,nx)*fi(nx,1) + A(nx,nx-1)*fi(nx-1,1);
            di(nx,1)= 12*rr*ga2;
        else
            gi(i,1) = D(i,i-1)*un(i-1,1)+D(i,i)*un(i,1) ...
                    + D(i,i+1)*un(i+1,1) ...
                    + A(i,i-1)*fi(i-1,1)+A(i,i)*fi(i,1) ...
                    + A(i,i+1)*fi(i+1,1);
            di(i,1) = 0.0;
        end
    end
    li   = gi + di;

    % Using TDMA
    for i=1:nx;
        if i == 1
            a1(i,1)= 0;         a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        elseif i == nx
            a1(i,1)=C(i,i-1);   a2(i,1)= C(i,i);    a3(i,1)= 0;
        else
            a1(i,1)=C(i,i-1);   a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        end
    end
    a3(1,1) = a3(1,1)/a2(1,1);
    li(1,1) = li(1,1)/a2(1,1);
    for i = 2:nx-1
        temp    = a2(i,1)-(a1(i,1)*a3(i-1,1));
        a3(i,1) = a3(i,1)/temp;
        li(i,1) = ( li(i,1)-(a1(i,1)*li(i-1,1)) )/temp;
    end
    li(nx,1) = ( li(nx,1)  - (a1(nx,1)*li(nx-1,1) ) )/ ...
               ( a2(nx,1) - (a1(nx,1)*a3(nx-1,1)) );
    % Now back substitute.
    unp1(nx,1) = li(nx,1);
    for i = nx-1:-1:1
        unp1(i,1) = li(i,1) - (a3(i,1)*unp1(i+1,1));
    end

% Check err
    err = max(abs(unp1-un));
    un  = unp1;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
end
toc
er(nit,1)
figure(1);
semilogy(t,er,'k-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-06 1.0E-02])
```

NHDS4: Neuman Boundary Condition, Fourth-Order Deferred correction

Finite Difference Scheme

- Main Program 1

```
clc,clear
format long
l  = 1;
n  = 11;
nx = n-2;
hx = 1./(n-1);
dt = hx.^2/20;
r = dt./(2.*hx.^2);
time = 1;
eps  = 1.0e-12;
nit = round(time/dt);

A = zeros(nx,nx);   C  = zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
a1 = zeros(nx,1);   a2 = zeros(nx,1);   a3 = zeros(nx,1);
li = zeros(nx,1);

gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)=2/3;         A(i,i+1)=-2/3;
    elseif i == nx
        A(i,i-1)=-2/3;      A(i,i)=2/3;
    else
        A(i,i-1)=-1;        A(i,i)=2;        A(i,i+1)=-1;
    end
end

C  = E+r*A;

np  = 0;
nt  = 0;
ti  = 0.;
err = 0.;
k1  = pi^2/2;
u0  = exact(ti,nx,gx);
un  = u0;
it  = 0;
itp = 0;

while ti < time
    err  = 10.;
    nt   = nt+1;
    ti   = nt*dt;       ki   = k1*exp(-k1*ti);
    th   = ti-(dt/2);   kh   = k1*exp(-k1*th);
    tp   = ti-dt;       kp   = k1*exp(-k1*tp);
    ex   = exact(ti,nx,gx);
    ap1 =  exp(-k1*tp);
    ap2 = -exp(-k1*tp)+tp+1;
```

```
    for i=1:nx;
        fi(i,1) = dt*( kh*cos(pi*gx(i,1)) + gx(i,1) - 2 );
        if i == 1
            gi(1,1) = u0(1,1)+fi(1,1) ...
                        +r*( 10*ap1-15*u0(1,1)-4*u0(2,1)...
                            +14*u0(3,1)-6*u0(4,1)+u0(5,1) )/12;
        elseif i == 2
            gi(2,1) = u0(2,1)+fi(2,1) ...
                        +r*( -ap1+16*u0(1,1)-30*u0(2,1)...
                             +16*u0(3,1)-u0(4,1) )/12;
        elseif i == nx-1
            gi(nx-1,1) = u0(nx-1,1)+fi(nx-1,1) ...
                        +r*( -ap2+16*u0(nx,1)-30*u0(nx-1,1)...
                             +16*u0(nx-2,1)-u0(nx-3,1) )/12;
        elseif i == nx
            gi(nx,1) = u0(nx,1)+fi(nx,1) ...
                        +r*( 10*ap2-15*u0(nx,1)-4*u0(nx-1,1)...
                        +14*u0(nx-2,1)-6*u0(nx-3,1)+u0(nx-4,1) )/12;
        else
            gi(i,1) = u0(i,1)+fi(i,1) ...
                        +r*( -u0(i-2,1)+16*u0(i-1,1)-30*u0(i,1)...
                             +16*u0(i+1,1)-u0(i+2,1) )/12;
        end
    end

    ap1 =  exp(-k1*ti);
    ap2 = -exp(-k1*ti)+ti+1;
    ga1 =  ti;
    ga2 =  ti+2;
    while err > eps
    itp  = itp+1;
        for i=1:nx;
            if i == 1
                di(1,1) = r*(-(50*ga1*hx/137)+(2041*un(1,1)/1644) ...
                            -(387*un(2,1)/137)+(653*un(3,1)/274) ...
                            -(131*un(4,1)/137)+(257*un(5,1)/1644) );
            elseif i == 2
                di(2,1) = r*( -ap1+4*un(1,1)-6*un(2,1)...
                             +4*un(3,1)-un(4,1) )/12;
            elseif i == nx-1
                di(nx-1,1) = r*( -ap2+4*un(nx,1)-6*un(nx-1,1)...
                             +4*un(nx-2,1)-un(nx-3,1) )/12;
            elseif i == nx
                di(nx,1)=r*((50*ga2*hx/137)+(2041*un(nx,1)/1644) ...
                       -(387*un(nx-1,1)/137)+(653*un(nx-2,1)/274) ...
                       -(131*un(nx-3,1)/137)+(257*un(nx-4,1)/1644) );
            else
                di(i,1) = r*( -un(i-2,1)+4*un(i-1,1)-6*un(i,1)...
                             +4*un(i+1,1)-un(i+2,1) )/12;
            end
        end
    li   = gi + di;
% Using TDMA
    for i=1:nx;
        if i == 1
            a1(i,1)= 0;         a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        elseif i == nx
            a1(i,1)=C(i,i-1);  a2(i,1)= C(i,i);    a3(i,1)= 0;
        else
            a1(i,1)=C(i,i-1);  a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        end
    end
```

```
    a3(1,1) = a3(1,1)/a2(1,1);
    li(1,1) = li(1,1)/a2(1,1);
    for i = 2:nx-1
        temp    = a2(i,1)-(a1(i,1)*a3(i-1,1));
        a3(i,1) = a3(i,1)/temp;
        li(i,1) = ( li(i,1)-(a1(i,1)*li(i-1,1)) )/temp;
    end
    li(nx,1) = ( li(nx,1)  - (a1(nx,1)*li(nx-1,1) ) )/ ...
               ( a2(nx,1) - (a1(nx,1)*a3(nx-1,1)) );
    % Now back substitute.
    unp1(nx,1) = li(nx,1);
    for i = nx-1:-1:1
        unp1(i,1) = li(i,1) - (a3(i,1)*unp1(i+1,1));
    end

% Check err
    err = max(abs(unp1-un));
    un  = unp1;
    end
    u0  = un;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
    it        = it+1;
end
toc
itp/it
er(nit,1)
figure(1);
semilogy(t,er,'b-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-12 1.0E-03])
```

- Main Program 2

```
clc,clear
format long
l  = 1;
n  = 11;
nx = n-2;
hx = 1./(n-1);
dt = hx.^2/10;
Re = 1.;
r = dt/(2*Re*hx^2);
rr = dt/(2*hx);
time = 1;
eps  = 1.0e-10;
nit = round(time/dt);

A = zeros(nx,nx);   C  = zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
a1 = zeros(nx,1);   a2 = zeros(nx,1);   a3 = zeros(nx,1);
li = zeros(nx,1);

gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)=2/3;          A(i,i+1)=-2/3;
    elseif i == nx
        A(i,i-1)=-2/3;       A(i,i)=2/3;
    else
        A(i,i-1)=-1;         A(i,i)=2;         A(i,i+1)=-1;
    end
end

C  = E+r*A;

np  = 0;
nt  = 0;
ti  = 0.;
err = 0.;
u0  = exact(ti,nx,gx);
un  = u0;
it  = 0;
itp = 0;

while ti < time
    err = 10.;
    nt  = nt+1;
    ti  = nt*dt;
    th  = ti-(dt/2);
    tp  = ti-dt;
    ex  = exact(ti,nx,gx);
    ap1 =  0.0;
    ap2 =  2/(1+2*tp);
```

```
for i=1:nx;
    if i == 1
        fi(1,1) =-rr*(-3*ap1^2-10*u0(1,1)^2+18*u0(2,1)^2 ...
                      -6*u0(3,1)^2+u0(4,1)^2 )/12;
    elseif i == 2
        fi(2,1) =-rr*( ap1^2-8*u0(1,1)^2 ...
                      +8*u0(3,1)^2-u0(4,1)^2 )/12;
    elseif i == nx-1
        fi(nx-1,1) =-rr*(-ap2^2+8*u0(nx,1)^2 ...
                      -8*u0(nx-2,1)^2+u0(nx-3,1)^2 )/12;
    elseif i == nx
        fi(nx,1) =-rr*( 3*ap2^2+10*u0(nx,1)^2-18*u0(nx-1,1)^2 ...
                      +6*u0(nx-2,1)^2-u0(nx-3,1)^2 )/12;
    else
        fi(i,1) =-rr*( u0(i-2,1)^2-8*u0(i-1,1)^2 ...
                      +8*u0(i+1,1)^2-u0(i+2,1)^2 )/12;
    end
end
for i=1:nx;
    if i == 1
        gi(1,1) = u0(1,1)+fi(1,1) ...
                     +r*( 10*ap1-15*u0(1,1)-4*u0(2,1)...
                         +14*u0(3,1)-6*u0(4,1)+u0(5,1) )/12;
    elseif i == 2
        gi(2,1) = u0(2,1)+fi(2,1) ...
                     +r*( -ap1+16*u0(1,1)-30*u0(2,1)...
                          +16*u0(3,1)-u0(4,1) )/12;
    elseif i == nx-1
        gi(nx-1,1) = u0(nx-1,1)+fi(nx-1,1) ...
                     +r*( -ap2+16*u0(nx,1)-30*u0(nx-1,1)...
                          +16*u0(nx-2,1)-u0(nx-3,1) )/12;
    elseif i == nx
        gi(nx,1) = u0(nx,1)+fi(nx,1) ...
                     +r*( 10*ap2-15*u0(nx,1)-4*u0(nx-1,1)...
                     +14*u0(nx-2,1)-6*u0(nx-3,1)+u0(nx-4,1) )/12;
    else
        gi(i,1) = u0(i,1)+fi(i,1) ...
                     +r*( -u0(i-2,1)+16*u0(i-1,1)-30*u0(i,1)...
                          +16*u0(i+1,1)-u0(i+2,1) )/12;
    end
end

ap1 =  0.0;
ap2 =  2/(1+2*ti);
ga1 =  2/(1+2*ti);
ga2 =  2/(1+2*ti);
while err > eps
itp  = itp+1;
    for i=1:nx;
        if i == 1
            di(1,1) = r*(-(50*ga1*hx/137)+(2041*un(1,1)/1644) ...
                        -(387*un(2,1)/137)+(653*un(3,1)/274) ...
                        -(131*un(4,1)/137)+(257*un(5,1)/1644) );
        elseif i == 2
            di(2,1) = r*( -ap1+4*un(1,1)-6*un(2,1)...
                          +4*un(3,1)-un(4,1) )/12;
        elseif i == nx-1
            di(nx-1,1) = r*( -ap2+4*un(nx,1)-6*un(nx-1,1)...
                          +4*un(nx-2,1)-un(nx-3,1) )/12;
```

```
            elseif i == nx
                di(nx,1)=r*((50*ga2*hx/137)+(2041*un(nx,1)/1644) ...
                       -(387*un(nx-1,1)/137)+(653*un(nx-2,1)/274) ...
                       -(131*un(nx-3,1)/137)+(257*un(nx-4,1)/1644) );
            else
                di(i,1) = r*( -un(i-2,1)+4*un(i-1,1)-6*un(i,1)...
                              +4*un(i+1,1)-un(i+2,1) )/12;
            end
        end
    li   = gi + di;

% Using TDMA
    for i=1:nx;
        if i == 1
            a1(i,1)= 0;         a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        elseif i == nx
            a1(i,1)=C(i,i-1);  a2(i,1)= C(i,i);    a3(i,1)= 0;
        else
            a1(i,1)=C(i,i-1);  a2(i,1)= C(i,i);    a3(i,1)= C(i,i+1);
        end
    end
    a3(1,1) = a3(1,1)/a2(1,1);
    li(1,1) = li(1,1)/a2(1,1);
    for i = 2:nx-1
        temp    = a2(i,1)-(a1(i,1)*a3(i-1,1));
        a3(i,1) = a3(i,1)/temp;
        li(i,1) = ( li(i,1)-(a1(i,1)*li(i-1,1)) )/temp;
    end
    li(nx,1) = ( li(nx,1)  - (a1(nx,1)*li(nx-1,1) ) )/ ...
               ( a2(nx,1) - (a1(nx,1)*a3(nx-1,1)) );
    % Now back substitute.
    unp1(nx,1) = li(nx,1);
    for i = nx-1:-1:1
        unp1(i,1) = li(i,1) - (a3(i,1)*unp1(i+1,1));
    end

% Check err
    err = max(abs(unp1-un));
    un  = unp1;
    end
    u0  = un;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
    it        = it+1;
end
toc
itp/it
er(nit,1)
figure(1);
semilogy(t,er,'b-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-07 1.0E-03])
```

NHCS4: Neuman Boundary Condition, Fourth-Order Compact Finite Difference Scheme

- Main Program

```
clc,clear
format short
l  = 1;
n  = 11;
nx = n-2;
hx = 1/(n-1);
dt = hx^2/10;
r  = dt/hx^2;
rr = dt/hx;
time = 1;
nit = round(time/dt);

A  = zeros(nx,nx);  B = zeros(nx,nx);   C  = zeros(nx,nx);
D  = zeros(nx,nx);  F = zeros(nx,nx);   G = zeros(nx,nx);
IC = zeros(nx,nx);

gx = zeros(nx,1);   u0 = zeros(nx,1);   un = zeros(nx,1);
unp1=zeros(nx,1);   fi = zeros(nx,1);   di = zeros(nx,1);
gi = zeros(nx,1);   er = zeros(nit,1);  t  = zeros(nit,1);
gx = grid(nx,hx);   E = eye(nx,nx);

tic

for i=1:nx;
    if i == 1
        A(i,i)  = 22;    A(i,i+1)= -4;
        B(i,i)  = 6;     B(i,i+1)= -6;
    elseif i == nx
        A(i,i)  = 22;    A(i,i-1)= -4;
        B(i,i)  = 6;     B(i,i-1)= -6;
    else
        A(i,i-1)= 1;     A(i,i)=10;     A(i,i+1)= 1;
        B(i,i-1)=-6;     B(i,i)=12;     B(i,i+1)=-6;
    end
end

C = A+r*B; D = A-r*B;  IC= inv(C); F = IC*D; G = IC*A;

nt  = 0;
ti  = 0.;
err = 0.;
k1  = pi^2/2;
u0  = exact(ti,nx,gx);
un  = u0;

while ti < time
    err = 10.;
    nt  = nt+1;
    ti   = nt*dt;       ki   = k1*exp(-k1*ti);
    th   = ti-(dt/2);   kh   = k1*exp(-k1*th);
    tp   = ti-dt;       kp   = k1*exp(-k1*tp);
    ex  = exact(ti,nx,gx);
    ga1 = th;
    ga2 = th+2;
```

```
    for i=1:nx;
        fi(i,1) = dt*( kh*cos(pi*gx(i,1)) + gx(i,1) - 2 );
        if i == 1
            di(1,1) =-12*rr*ga1;
        elseif i == nx
            di(nx,1)= 12*rr*ga2;
        else
            di(i,1) = 0.0;
        end
    end
    unp1 = F*un + IC*di + G*fi;
    un   = unp1;
    er(nt,1)  = max(abs(ex-unp1));
    t(nt,1)   = nt*dt;
end
end
toc
er(nit,1)
figure(1);
semilogy(t,er,'k-.');
hold on
xlabel('t')
ylabel('Maximum norm')
axis([0 1 1.0E-12 1.0E-02])
```

Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 574620, 9 pages
http://dx.doi.org/10.1155/2013/574620

Research Article

Fourth-Order Deferred Correction Scheme for Solving Heat Conduction Problem

D. Yambangwai[1] and N. P. Moshkin[2]

[1] *Department of Mathematics, School of Science, University of Phayao, Phayao 56000, Thailand*
[2] *School of Mathematics, Institute of Science, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand*

Correspondence should be addressed to D. Yambangwai; damrong.sut@gmail.com

Received 3 December 2012; Accepted 27 February 2013

Academic Editor: Safa Bozkurt Coskun

A deferred correction method is utilized to increase the order of spatial accuracy of the Crank-Nicolson scheme for the numerical solution of the one-dimensional heat equation. The fourth-order methods proposed are the easier development and can be solved by using Thomas algorithms. The stability analysis and numerical experiments have been limited to one-dimensional heat-conducting problems with Dirichlet boundary conditions and initial data.

1. Introduction

The desired properties of finite difference schemes are stability, accuracy, and efficiency. These requirements are in conflict with each other. In many applications a high-order accuracy is required in the spatial discretization. To reach better stability, implicit approximation is desired. For a high-order method of traditional type (not a high-order compact (HOC)), the stencil becomes wider with increasing order of accuracy. For a standard centered discretization of order p, the stencil is $p + 1$ points wide. This inflicts problems at the fictional boundaries, and using an implicit method results in the solution of an algebraic system of equations with large bandwidth. In light of conflict requirements of stability, accuracy, and computational efficiency, it is desired to develop schemes that have a wide range of stability and highorder of accuracy and lead to the solution of a system of linear equations with a tri-diagonal matrix, that is, the system of linear equations arising from a standard second-order discretization of heat equation.

The development of high-order compact (HOC) schemes [1–18] is one approach to overcome the antagonism between stability, accuracy, and computational cost. However, the HOC becomes complicated when applie to multidimensional problems or to non-Cartesian coordinate cases.

Another way of preserving a compact stencil at higher time level and reaching high-order spatial accuracy is the deferred correction approach [11]. A classical deferred correction procedure is developed in [19, 20].

In this paper we use the deferred correction technique to obtain fourth-order accurate schemes in space for the one-dimensional heat-conducting problem with Dirichlet boundary conditions. The linear system that needs to be solved at each time step is similar to the standard Crank-Nikolson method of second order which is solved by using Thomas algorithms. The fourth-order deferred (FOD) correction schemes are compared with the fourth-order semi-implicit (FOS) schemes and fourth-order compact (FOC) schemes for the Dirichlet boundary value problems.

A set of schemes are constructed for the one-dimensional heat-conducting problem with Dirichlet boundary conditions and initial data:

$$u_t = \beta u_{xx} + f(x,t), \quad 0 < x < l, \ t > 0, \tag{1}$$

$$u(x,0) = u_0(x), \quad 0 < x < l, \tag{2}$$

$$\text{Dirichlet BC}: \ u(0,t) = \gamma_1(t), \quad u(l,t) = \gamma_2(t), \ t > 0, \tag{3}$$

where the diffusion coefficient β is positive, $u(x,t)$ represents the temperature at point (x,t), and $f(x,t)$, $\gamma_1(t)$, $\gamma_2(t)$ are sufficiently smooth functions.

The rest of this paper is organized as follows. Section 2 presents an FOD scheme which we use to compare performance of proposed scheme with FOS and FOC schemes. Section 3 provides examples of comparisons. Although FOD schemes have a higher computational cost than FOS and FOC schemes, it is evident from these examples that the FOD schemes have the advantage of accuracy in the uniform norm, robustness, and the ability to be extended easily to the multidimensional case. We conclude the paper in Section 4.

2. The Fourth-Order Schemes

Let Δt denote the temporal mesh size. For simplicity, we consider a uniform mesh consisting of N points: $x_1, x_2, \ldots, x_N$ where $x_i = (i-1)\Delta x$ and the mesh size is $\Delta x = l/(N-1)$. Below we use the notations u_i^m and $(u_{xx})_i^m$ to represent the numerical approximations of $u(x_i, t^m)$ and $u_{xx}(x_i, t^m)$ where $t^m = m\Delta t$ and $u^{(p)}$ is the value of the pth derivative of the given function u.

2.1. Fourth-Order Semi-Implicit Scheme. The application to the well-known Crank-Nikolson scheme to (1) results in the following expression:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\beta}{2}\left[(u_{xx})_i^{n+1} + (u_{xx})_i^n\right] + f_i^{n+1/2}, \tag{4}$$

where $f_i^{n+1/2} = (f_i^{n+1} + f_i^n)/2$, $i = 2, \ldots, N-1$. The Dirichlet boundary conditions

$$\begin{aligned} u(0, m\Delta t) &= \gamma_1(t^m) = u_1^m, \\ u(l, m\Delta t) &= \gamma_2(t^m) = u_N^m \end{aligned} \tag{5}$$

are used to derive the following fourth-order approximation of second derivative terms:

$$\begin{aligned} (u_{xx})_2^m &= \frac{1}{12\Delta x^2}\left(a_1 u_1^m + a_2 u_2^m + a_3 u_3^m + a_4 u_4^m \right. \\ &\qquad \left. + a_5 u_5^m + a_6 u_6^m\right) \\ &= \frac{a_1}{12\Delta x^2}\gamma_1(t^m) + \frac{1}{12\Delta x^2} \\ &\quad \times \left(a_2 u_2^m + a_3 u_3^m + a_4 u_4^m + a_5 u_5^m + a_6 u_6^m\right), \\ (u_{xx})_i^m &= \frac{1}{12\Delta x^2} \\ &\quad \times \left(-u_{i-2}^m + 16u_{i-1}^m - 30u_i^m + 16u_{i+1}^m - u_{i+2}^m\right), \\ &\qquad i = 3, \ldots, N-2, \\ (u_{xx})_{N-1}^m &= \frac{1}{12\Delta x^2}\left(a_1 u_N^m + a_2 u_{N-1}^m \right. \\ &\qquad + a_3 u_{N-2}^m + a_4 u_{N-3}^m \\ &\qquad \left. + a_5 u_{N-4}^m + a_6 u_{N-5}^m\right) \\ &= \frac{a_1}{12\Delta x^2}\gamma_2(t^m) + \frac{1}{12\Delta x^2} \\ &\quad \times \left(a_2 u_{N-1}^m + a_3 u_{N-2}^m \right. \\ &\qquad \left. + a_4 u_{N-3}^m + a_5 u_{N-4}^m + a_6 u_{N-5}^m\right), \end{aligned} \tag{6}$$

where the coefficients can be found by matching the Taylor series expansion of left-hand-side terms up to order $O(\Delta x^4)u^{(6)}$ which gives the following values of coefficients:

$$\begin{aligned} a_1 &= 10, \quad a_2 = -15, \quad a_3 = -4, \\ a_4 &= 14, \quad a_5 = -6, \quad a_6 = 1. \end{aligned} \tag{7}$$

Schemes (6) can be combined and expressed in the following matrix form:

$$\mathbf{u}_{xx}^m = \frac{1}{\Delta x^2}\Lambda_h \mathbf{u}^m + \gamma(t^m), \tag{8}$$

where Λ_h is the corresponding triangular and sparse $(N-2)\times(N-2)$ matrix,

$$\begin{aligned} \mathbf{u}_{xx}^m &= \left((u_{xx})_2^m, (u_{xx})_3^m, \ldots, (u_{xx})_{N-1}^m\right)^T, \\ \mathbf{u}^m &= \left(u_2^m, u_3^m, \ldots, u_{N-1}^m\right)^T, \\ \gamma(t^m) &= \left(\gamma_1(t^m), 0, \ldots, 0, \gamma_2(t^m)\right)^T. \end{aligned} \tag{9}$$

Substituting (6) into (4) gives us the following matrix form:

$$\begin{aligned} \left(E - \alpha\Lambda_h\right)\mathbf{u}^{n+1} &= \left(E + \alpha\Lambda_h\right)\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right) + \gamma\left(t^n\right)\right] \\ &\quad + \Delta t\mathbf{f}^{n+1/2}, \end{aligned} \tag{10}$$

where $\alpha = \beta\Delta t/(2\Delta x^2)$, $\mathbf{f}^{n+1/2} = (f_2^{n+1/2}, f_3^{n+1/2}, \ldots, f_{N-1}^{n+1/2})^T$, and E denote the $(N-2)\times(N-2)$ identity matrix. The scheme (10) is FOSs for the heat-conducting problem with Dirichlet boundary condition. The order of approximation is $O(\Delta t^2, \Delta x^4)$ in the uniform norm. The triangular and sparse $(N-2)\times(N-2)$ coefficient matrix in FOSs are time independent; hence, we have to store the inverse of the coefficient matrix $E - \alpha\Lambda_h$ before the time marching in the implementation for computational efficiency.

2.2. Fourth-Order Deferred Correction Schemes. A set of fourth-order deferred correction schemes is based on the well-known Crank-Nikolson type of scheme in the following form:

$$\frac{u_i^{n+1,s+1}-u_i^n}{\Delta t}=\frac{\beta}{2}\left[\left(u_{xx}\right)_i^{n+1,s+1}+\left(u_{xx}\right)_i^n\right]+f_i^{n+1/2}, \tag{11}$$

where $f_i^{n+1/2}=(f_i^{n+1}+f_i^n)/2$ and the second superscript "s" denotes the number of iterations $s=0,\ldots,\widehat{S}$ and $i=2,\ldots,N-1$.

The deferred correction technique [11] is utilized to approximate the second-order derivatives at higher time levels $(u_{xx})_i^{n+1,s+1}$, $i=2,\ldots,N-1$ by the iterative method

$$\left(u_{xx}\right)_i^{n+1,s+1}=\left(u_{xx}^l\right)_i^{n+1,s+1}+\left[\left(u_{xx}^h\right)_i^{n+1,s}-\left(u_{xx}^l\right)_i^{n+1,s}\right], \tag{12}$$

where $(u_{xx}^h)_i^{n+1,s}$, $i=2,\ldots,N-1$, $s=0,\ldots,\widehat{S}$, is high-order approximation on wide stencil and $(u_{xx}^l)_i^{n+1,k}$, $k=s$, $s+1$, $i=2,\ldots,N-1$, is the lower-order approximation on compact stencil (usually three-point stencil). The expression in the square brackets of (12) is evaluated explicitly using the values known from the previous iteration. When $s=0$ we use the solution from the time level n (so $u^{n+1,0}=u^n$ and $(u_{xx})_i^{n+1,0}=(u_{xx})_i^n$). Once the iterations converge, the lower-order approximation terms drop out and the approximation of $(u_{xx})_i^{n+1,s+1}$ obtained has the same order of approximation as $(u_{xx}^h)_i^{n+1,\widehat{S}}$. There are no difficulties to construct high-order approximation for interior points.

To preserve a compact three using wide stencil in the finite difference scheme at higher time level $(n+1,s+1)$, we use the central second-order finite difference approximation to approximate the lower-order term in (12):

$$\left(u_{xx}^l\right)_i^{n+1,k}=\frac{1}{\Delta x^2}\Lambda_l u_i^{n+1,k},\quad k=s,s+1,\quad i=3,\ldots,N-2,$$
$$\Lambda_l u_i^{n+1,k}=u_{i-1}^{n+1,k}-2u_i^{n+1,k}+u_{i+1}^{n+1,k}. \tag{13}$$

For the high-order approximation term in (12), we use a symmetric five-point wide stencil for the inner points to reach the fourth order of approximation:

$$\left(u_{xx}^h\right)_i^{n+1,s}=\frac{1}{\Delta x^2}\Lambda_h u_i^{n+1,s},\quad i=3,\ldots,N-2,$$
$$\Lambda_h u_i^{n+1,s}=\frac{1}{12}\left(-u_{i-2}^{n+1,s}+16u_{i-1}^{n+1,s}-30u_i^{n+1,s}+16u_{i+1}^{n+1,s}-u_{i+2}^{n+1,s}\right). \tag{14}$$

Case $s=0$ in (13) gives the fourth order of approximation to approximate the second-order derivatives at the time level n.

2.2.1. Stability Analysis. To study the stability of scheme (11)–(14), we use the Von-Neumann stability analysis. For simplicity, we assume that $f_i^{n+1/2}\equiv 0$ in (11) and u is periodic in x.

Let us recast scheme (11) in the following form:

$$\left(E+\alpha\Lambda_l\right)u_i^{n+1,s+1}=\alpha\left(\Lambda_l-\Lambda_h\right)u_i^{n+1,s}+\left(E-\alpha\Lambda_l\right)u_i^n, \tag{15}$$

where $\alpha=\beta\Delta t/(2\Delta x^2)$. If we define the following operators $A=E+\alpha\Lambda_l$, $B=E-\alpha\Lambda_h$, and $C=E+\alpha\Lambda_h$, where E is the identity operator, then (15) can be rewritten as follows:

$$Au_i^{n+1,s+1}=(A-C)u_i^{n+1,s}+Bu_i^n. \tag{16}$$

Assuming that the operators commute, $(A-C)A=A(A-C)$ (e.g., in the case of uniform grid), it is easy to demonstrate that if $u_i^{n+1,\widehat{S}+1}=u_i^{n+1}$ and $u_i^{n+1,0}=u_i^n$ we get

$$A^{\widehat{S}+1}u_i^{n+1}=\left(\sum_{k=0}^{\widehat{S}}A^{\widehat{S}-k}(A-C)^k\right)Bu_i^n+(A-C)^{\widehat{S}+1}u_i^n. \tag{17}$$

Let $u_i^n=\xi^n e^{I\Theta i}$, $I=\sqrt{-1}$, be the solution of (11)–(14), where $\Theta=2\pi\Delta x/l$ is the phase angle with wavelength l. From (17), we can derive an equation for the amplification factor in the form

$$\left|\xi\right|=\left|\varphi\left(\Theta,\widehat{S},\alpha\right)\right|, \tag{18}$$

where $\widehat{S}$ is the number of iterations, and

$$\left|\varphi\left(\Theta,\widehat{S},\alpha\right)\right|=\frac{\left|\left[\left(\sum_{k=0}^{\widehat{S}}A^{\widehat{S}-k}(A-C)^k\right)B+(A-C)^{\widehat{S}+1}\right]e^{I\Theta i}\right|}{\left|A^{\widehat{S}+1}e^{I\Theta i}\right|}. \tag{19}$$

For stability of the method it is necessary that the absolute values of the amplification factor are less than one; that is,

$$\left|\xi\right|<1. \tag{20}$$

Calculations are tedious and almost impossible to do by hand without mistake. We have therefore automate all calculations in a computer algebra environment based on REDUCE to obtain an explicit form of $|\varphi(\Theta,\widehat{S},\alpha)|$. Figure 1 shows the values of $|\xi|^2$ in the polar coordinate system $(|\xi|^2,\Theta)$ for $\widehat{S}=1,3,$ and 5. If only one iteration is executed in (11), $\widehat{S}=1$, inequality (20) holds if $\alpha<1.5$, as can be seen from Figure 1(a)). If 3 iterations are done in (11) (Figure 1(b)), $\widehat{S}=3$, the amplification factor remains bounded by one at least for $\alpha\le 10$. In case of $\widehat{S}=5$, the stability criteria hold up to $\alpha=30$ as can be seen from Figure 1(c)). It can be seen that increasing the number of internal iterations results in increasing the range of α needed for stability. This tendency allows to assume that as $\widehat{S}\to\infty$, our method becomes the unconditionally stable Crank-Nikolson method for the heat equation.

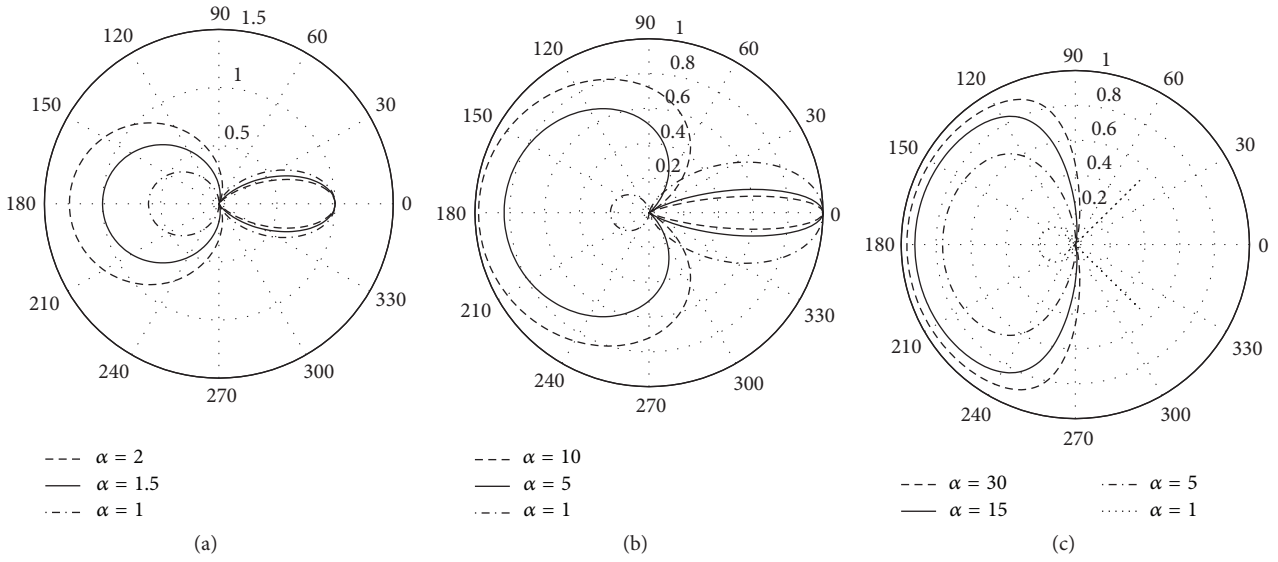


FIGURE 1: Variation of amplification factor with Θ. (a) $\widehat{S} = 1$, dashed line $\alpha = 2.0$, solid line $\alpha = 1.5$, dash-doted line $\alpha = 1.0$, (b) $\widehat{S} = 3$, dashed line $\alpha = 10.0$, solid line $\alpha = 5.0$, dash-dotted line $\alpha = 1.0$, and (c) $\widehat{S} = 5$, dashed line $\alpha = 30.0$, solid line $\alpha = 15$, dash-doted line $\alpha = 5.0$, doted line $\alpha = 1.0$.

2.2.2. Fourth-Order Deferred Correction Scheme. Let us first consider the one-dimensional heat conduction problem with initial data and Dirichlet boundary conditions (1)–(3):

$$u_1^{n+1,k} = \gamma_1\left(t^{n+1}\right), \qquad u_N^{n+1,k} = \gamma_2\left(t^{n+1}\right). \tag{21}$$

The finite difference approximations at x_2 and x_{N-1}, which are the points next to the left and right boundaries, are straightforward:

$$\begin{aligned}
\left(u_{xx}^{l}\right)_2^{n+1,k} &= \frac{1}{\Delta x^2}\left(\gamma_1\left(t^{n+1}\right) - 2u_2^{n+1,k} + u_3^{n+1,k}\right), \\
&\qquad k = s, s+1, \\
\left(u_{xx}^{h}\right)_2^{n+1,s} &= \frac{1}{12\Delta x^2}\left(10\gamma_1\left(t^{n+1}\right) - 15u_2^{n+1,s} - 4u_3^{n+1,s}\right. \\
&\qquad \left. +14u_4^{n+1,s} - 6u_5^{n+1,s} + u_6^{n+1,s}\right), \\
\left(u_{xx}^{h}\right)_{N-1}^{n+1,s} &= \frac{1}{12\Delta x^2}\left(10\gamma_2\left(t^{n+1}\right) - 15u_{N-1}^{n+1,s} - 4u_{N-2}^{n+1,s}\right. \\
&\qquad \left. +14u_{N-3}^{n+1,s} - 6u_{N-4}^{n+1,s} + u_{N-5}^{n+1,s}\right), \\
\left(u_{xx}^{l}\right)_{N-1}^{n+1,k} &= \frac{1}{\Delta x^2}\left(u_{N-2}^{n+1,k} - 2u_{N-1}^{n+1,k} + \gamma_2\left(t^{n+1}\right)\right), \\
&\qquad k = s, s+1.
\end{aligned} \tag{22}$$

Cases $s = 0$ or $k = 0$ give formulae to approximate $(u_{xx}^{l})_i^n$ and $(u_{xx}^{h})_i^n$. Substituting (13), (14), and (22) into (12) the following fourth-order deferred correction approximations of $(u_{xx})_i^{n+1,s+1}$, $i = 2, \ldots, N-1$, are

$$\begin{aligned}
\left(u_{xx}\right)_2^{n+1,s+1} &= \frac{5}{6\Delta x^2}\gamma_1\left(t^{n+1}\right) \\
&\quad + \frac{1}{\Delta x^2}\left(-2u_2^{n+1,s+1} + u_3^{n+1,s+1}\right) \\
&\quad + \frac{1}{12\Delta x^2}\left(9u_2^{n+1,s} - 16u_3^{n+1,s} + 14u_4^{n+1,s}\right. \\
&\qquad \left. - 6u_5^{n+1,s} + u_6^{n+1,s}\right), \\
\left(u_{xx}\right)_i^{n+1,s+1} &= \frac{1}{\Delta x^2}\left(u_{i-1}^{n+1,s+1} - 2u_i^{n+1,s+1} + u_{i+1}^{n+1,s+1}\right) \\
&\quad + \frac{1}{12\Delta x^2}\left(-u_{i-2}^{n+1,s} + 4u_{i-1}^{n+1,s} - 6u_i^{n+1,s}\right. \\
&\qquad \left. + 4u_{i+1}^{n+1,s} - u_{i+2}^{n+1,s}\right), \\
&\qquad i = 3, \ldots, N-2, \\
\left(u_{xx}\right)_{N-1}^{n+1,s+1} &= \frac{5}{6\Delta x^2}\gamma_2\left(t^{n+1}\right) \\
&\quad + \frac{1}{\Delta x^2}\left(-2u_{N-1}^{n+1,s+1} + u_N^{n+1,s+1}\right) \\
&\quad + \frac{1}{12\Delta x^2}\left(9u_{N-1}^{n+1,s} - 16u_{N-2}^{n+1,s} + 14u_{N-3}^{n+1,s}\right. \\
&\qquad \left. -6u_{N-4}^{n+1,s} + u_{N-5}^{n+1,s}\right).
\end{aligned} \tag{23}$$

Schemes (23) can be combined and expressed in the following matrix form:

$$\mathbf{u}_{xx}^{n+1,s+1} = \frac{1}{\Delta x^2}\Lambda_l \mathbf{u}^{n+1,s+1} + \frac{1}{\Delta x^2}\left(\Lambda_h - \Lambda_l\right)\mathbf{u}^{n+1,s} + \gamma\left(t^{n+1}\right), \tag{24}$$

where Λ_l is a tridiagonal $(N-2)\times(N-2)$ matrix and Λ_h is the corresponding triangular and sparse $(N-2)\times(N-2)$ matrix,

$$\mathbf{u}^{n+1,k} = \left(u_2^{n+1,k}, u_3^{n+1,k}, \ldots, u_{N-1}^{n+1,k}\right)^T, \quad k = s, s+1, \tag{25}$$

$$\mathbf{u}_{xx}^{n+1,s+1} = \left((u_{xx})_2^{n+1,s+1}, (u_{xx})_3^{n+1,s+1}, \ldots, (u_{xx})_{N-1}^{n+1,s+1}\right)^T. \tag{26}$$

Substituting (6), (23) into (11), the formulae can be written into matrix form

$$\begin{aligned}(E-\alpha\Lambda_l)\,\mathbf{u}^{n+1,s+1} &= \alpha\left(\Lambda_h-\Lambda_l\right)\mathbf{u}^{n+1,s} + \left(E+\alpha\Lambda_h\right)\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right)+\gamma\left(t^n\right)\right] + \Delta t\mathbf{f}^{n+1/2}.\end{aligned} \tag{27}$$

The above matrix form is called FODs for Dirichlet boundary value problem (1)–(3). Thomas algorithms can be used to compute the solutions of FODs. At each step of time t^n and the initial stage, the convergence of FODs requires more iterations to converge to the solution of the FOSs. The order of approximation of FODs is $O(\Delta t^2, \Delta x^4)$ which is the same as FOSs in the uniform norm.

2.3. Fourth-Order Compact Scheme. Let us briefly represent the main idea and final formulae of compact schemes. Spatial derivatives in the Crank-Nikolson scheme (4) are evaluated by the fourth-order compact finite differences implicit scheme [5, 7, 8, 13, 14, 17].

In [8, 14], the Dirichlet boundary conditions

$$u(0, m\Delta t) = \gamma_1\left(t^m\right) = u_1^m, \qquad u(l, m\Delta t) = \gamma_2\left(t^m\right) = u_N^m \tag{28}$$

are used to derive the following fourth-order schemes

$$\begin{aligned}&(u_{xx})_2^m + \sigma(u_{xx})_3^m \\ &\quad = \frac{1}{24\Delta x^2}\left(a_1u_1^m + a_2u_2^m + a_3u_3^m + a_4u_4^m + a_5u_5^m + a_6u_6^m\right) \\ &\quad = \frac{a_1}{24\Delta x^2}\gamma_1\left(t^m\right) + \frac{1}{24\Delta x^2}\left(a_2u_2^m + a_3u_3^m + a_4u_4^m + a_5u_5^m + a_6u_6^m\right), \\ &(u_{xx})_{i-1}^m + 10(u_{xx})_i^m + (u_{xx})_{i+1}^m \\ &\quad = \frac{2}{\Delta x^2}\left(6u_{i-1}^m - 12u_i^m + 6u_{i+1}^m\right), \\ &\qquad i = 2, \ldots, N-1, \\ &(u_{xx})_{N-1}^m + \sigma(u_{xx})_{N-2}^m \\ &\quad = \frac{1}{24\Delta x^2}\left(a_1u_N^m + a_2u_{N-1}^m + a_3u_{N-2}^m + a_4u_{N-3}^m + a_5u_{N-4}^m + a_6u_{N-5}^m\right) \\ &\quad = \frac{a_1}{24\Delta x^2}\gamma_2\left(t^m\right) + \frac{1}{24\Delta x^2}\left(a_2u_{N-1}^m + a_3u_{N-2}^m + a_4u_{N-3}^m + a_5u_{N-4}^m + a_6u_{N-5}^m\right),\end{aligned} \tag{29}$$

where the coefficients can be found by matching the Taylor series expansion of left-hand-side terms up to order $O(\Delta x^4)u^{(6)}$ which gives the following values of coefficients [8]:

$$\sigma = \frac{1}{2}, \qquad a_1 = 19, \qquad a_2 = -14, \qquad a_3 = -38, \qquad a_4 = 44, \qquad a_5 = -13, \qquad a_6 = 2. \tag{30}$$

Then all derivatives in (4) are approximated by the fourth-order compact formula; we can write

$$A\mathbf{u}_{xx}^m = \frac{1}{\Delta x^2}B\mathbf{u}^m + \gamma^m, \quad m = n, n+1, \tag{31}$$

where A and B are the corresponding triangular and sparse $(N-2)\times(N-2)$ matrices, $\mathbf{u}_{xx}^m = ((u_{xx})_2^m, (u_{xx})_3^m, \ldots, (u_{xx})_{N-1}^m)^T$, $\mathbf{u}^m = (u_2^m, u_3^m, \ldots, u_{N-1}^m)^T$ and $\gamma^m = (\gamma_1(t^m), 0, \ldots, 0, \gamma_2(t^m))^T$, $m = n, n+1$. Schemes (4) and (29) can be combined and expressed in the following matrix form:

$$\begin{aligned}(A-\alpha B)\,\mathbf{u}^{n+1} &= (A+\alpha B)\,\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right)+\gamma\left(t^n\right)\right] \\ &\quad + \Delta t\mathbf{f}^{n+1/2}.\end{aligned} \tag{32}$$

This scheme is called FOCs for Dirichlet boundary value problem (1)–(3). We like to mention that the above scheme has truncation error $O(\Delta t^2, \Delta x^4)$. Note that the triangular and sparse $(N-2)\times(N-2)$ coefficient matrices in FOCs are time independent; hence, we have to store the inverse of the coefficient matrix $A-\alpha B$ before the time marching in the implementation of computational efficiency.

3. Numerical Examples

In this section, three numerical examples are carried out. The first two are linear heat-conducting problem, with Dirichlet

boundary conditions, which are used to confirm our theoretical analysis. Then we apply the FODS to the Burgers equation. For simplicity, we fix our problem domain $\Omega = \{x \mid 0 \le x \le 1\}$. In all computations, we used $\Delta t = \Delta x^2/4$ and $\epsilon = 10^{-10}$. The following stopping criterion is used:

$$\max_{1\le i\le N} \left| u_i^{n+1,\widehat{S}+1} - u_i^{n+1,\widehat{S}} \right| < \epsilon, \quad s = 0, \ldots, \widehat{S}, \tag{33}$$

where "$\widehat{S}$" denotes the number of the last iteration.

The computations are performed using uniform grids of 11, 21, 41, 81, and 161 nodes. The initial and boundary conditions are obtained based on the exact solutions. For the testing purpose only, all computations are performed for $0 \le t \le 1$.

Example 1 (the homogeneous heat equation with the homogeneous Dirichlet boundary conditions). One has

$$\begin{aligned} u_t &= u_{xx}, \quad 0 \le x \le 1, \ t > 0, \\ u(x,0) &= \sin(\pi x), \quad u(0,t) = 0, \quad u(1,t) = 0. \end{aligned} \tag{34}$$

The exact solution is $u(x,t) = e^{-\pi^2 t}\sin(\pi x)$. The results of performance over the time interval $t \in [0,1]$ for the FOCs, FODs, and FOSs are represented in Table 1, where the maximum error and the rate of convergence at time instant $t = 1$ are shown.

Example 2 (the nonhomogeneous heat equation with non-homogeneous Dirichlet boundary conditions). One has

$$\begin{aligned} u_t &= u_{xx} + \left(\pi^2 - 1\right) e^{-t}\cos(\pi x) \\ &\quad + 4x - 2, \quad 0 \le x \le 1, \ t > 0, \\ u(x,0) &= \cos(\pi x) + x^2, \quad u(0,t) = e^{-t}, \\ u(1,t) &= -e^{-t} + 4t + 1. \end{aligned} \tag{35}$$

The exact solution is $u(x,t) = e^{-t}\cos(\pi x) + x^2 + 4xt$. The results of performance over the time domain $t \in [0,1]$ for the FOC, FOD, and FOS schemes are represented in Table 2, where the maximum error and the rate of convergence at time instant $t = 1$ are shown.

The last two columns of Tables 1 and 2 demonstrate the average number of iterations in FODs at one time step and the CPU time required to obtain the solution at time instant $t = 1$. The average number of iterations means the total number of iterations divided by the number of time steps. As a rule, at the initial stage the convergence of deferred correction requires more iterations. For larger instants of time, the convergence occurs after 2~7 iterations as can be seen from Tables 1 and 2. All of schemes are seen to be the fourth order of accuracy, as the error is reduced approximately by factor four when the mesh is refined by half. The maximum error of the FODs and FOCs is almost the same, since the iterative scheme FODs is constructed by applying the deferred correction technique on the FOSs. It can be stated that when the iterations converge, the solution of FODs, therefore, converges to the solution of FOSs in each step of time. As shown in Tables 1 and 2, there is hardly a difference in the computational efficiency between FODs and FOSs. Both schemes are more efficient than FODs. An explanation is due to the iteration needed for the convergence of solutions on each step of time.

Although the FODs use more computational time as compared with FOCs and FOSs, it is recommended that the construction of FODs can be easily implemented. Moreover, the scheme does not need to store the inverse of coefficient matrices as required in FOCs and FOSs. Therefore, the method is easily extended to multidimensional cases.

It is suggested that the differed correction technique can solve problems which need high accuracy of computational methods. Also this technique can be easily implemented and extended for solving problem with Neumann boundary conditions. In addition, such technique can be easily used to create standard code and applied in case of nonuniform grids.

Considering Burgers equation

$$u_t = \beta u_{xx} - u u_x, \quad 0 \le x \le 1, \ t > 0, \tag{36}$$

with the exact solution [21] is given by

$$u(x,t) = \frac{\xi + \eta + (\eta - \xi)\, e^{\rho}}{1 + e^{\rho}}, \tag{37}$$

where $\rho = \xi(x - \eta t - \upsilon)/\beta$. The initial and Dirichlet boundary conditions are considered to be in agreement with the exact solution proposed here. For Burgers equation (36), we solve it by the following fourth-order deferred correction scheme:

$$\frac{u_i^{n+1,s+1} - u_i^n}{\Delta t} = \frac{\beta}{2}\left[\left(u_{xx}\right)_i^{n+1,s+1} + \left(u_{xx}\right)_i^n \right] + f_i^n, \tag{38}$$

where $f_i^n = -[(u^2/2)_x]_i^n$. The nonlinear term f_i^n is approximated with the fourth-order approximation and all the second-derivative terms in (38) are approximated by the fourth-order formula (6) and the fourth-order deferred correction schemes (23). The scheme (38) can be combined and expressed in the following matrix form:

$$\begin{aligned} \left(E - \alpha\Lambda_l\right)\mathbf{u}^{n+1,s+1} &= \alpha\left(\Lambda_h - \Lambda_l\right)\mathbf{u}^{n+1,s} + \left(E + \alpha\Lambda_h\right)\mathbf{u}^n \\ &\quad + \Delta t\left[\gamma\left(t^{n+1}\right) + \gamma\left(t^n\right)\right] + \Delta t\mathbf{f}^n, \end{aligned} \tag{39}$$

where E is identity matrix, Λ_l is tridiagonal $(N-2)\times(N-2)$ matrix, and Λ_h is the corresponding triangular and sparse $(N-2)\times(N-2)$ matrix and can be solved by using Thomas algorithm.

Example 3 (the Burgers equation (36) and the constant values $\nu = 0.125$, $\xi = 0.6$, $\eta = 0.4$, and $\beta = 1$ with appropriate initial and Dirichlet boundary condition in agreement with exact solution (37)). This problem was solved using different time step and mesh sizes over the time interval $0 < t \le 1$. The results of performance over the time interval $t \in [0,1]$ for the FODs are represented in Tables 3 and 4, where the maximum error and the rate of convergence at time instant $t = 1$ are shown.

Table 1: Maximum absolute error, order of convergence, and CPU time in seconds of the FOCs, FODs, and FOSs for test problem (34) at time instant $t = 1$.

Types of scheme	Grid points	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FOCs	11	3.8687×10^{-8}	—	1	0.006
	21	6.0426×10^{-10}	6.0005	1	0.042
	41	2.2454×10^{-11}	4.7501	1	0.326
	81	1.2821×10^{-12}	4.1304	1	2.564
	161	8.0164×10^{-14}	3.9994	1	20.437
FODs	11	9.9767×10^{-9}	—	4	0.015
	21	1.4996×10^{-9}	2.7361	3	0.085
	41	1.1193×10^{-10}	3.7438	2	0.438
	81	7.1438×10^{-12}	3.9698	2	3.450
	161	4.4797×10^{-13}	4.1875	2	27.495
FOSs	11	9.9763×10^{-9}	—	1	0.006
	21	1.4996×10^{-9}	2.7361	1	0.043
	41	1.1193×10^{-10}	3.7438	1	0.334
	81	7.1440×10^{-12}	3.9698	1	2.623
	161	4.4854×10^{-13}	4.1875	1	20.907

Table 2: Absolute error, the rate of convergence, and CPU time in seconds of the FOCs, FODs, and FOSs for the test problem (35) at time instant $t = 1$.

Types of scheme	Grid points	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FOCs	11	1.8470×10^{-5}	—	1	0.006
	21	3.6901×10^{-7}	5.6454	1	0.046
	41	7.5595×10^{-9}	5.6092	1	0.353
	81	6.6458×10^{-10}	3.5077	1	2.778
	161	4.8841×10^{-11}	3.7663	1	22.141
FODs	11	1.7132×10^{-5}	—	7	0.016
	21	2.6914×10^{-7}	5.9922	7	0.128
	41	2.7910×10^{-8}	3.2655	6	0.851
	81	2.0112×10^{-9}	3.7941	5	5.568
	161	1.3116×10^{-10}	3.9415	5	44.375
FOSs	11	1.2895×10^{-5}	—	1	0.006
	21	2.8544×10^{-7}	5.9922	1	0.046
	41	2.7306×10^{-8}	3.2655	1	0.359
	81	1.9590×10^{-9}	3.7941	1	2.821
	161	1.3130×10^{-10}	3.9415	1	22.484

Table 3: Maximum absolute error, order of convergence, and CPU time in seconds for Example 3 at time instant $t = 1$ with fixed mesh size $\Delta x = 0.05$.

Types of scheme	Time step sizes	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FODs	10^{-2}	8.0945×10^{-6}	—	10	0.015
	10^{-3}	8.0942×10^{-7}	1.0000	6	0.109
	10^{-4}	8.1144×10^{-8}	0.9989	3	0.656
	10^{-5}	8.2989×10^{-9}	0.9902	3	6.281

Table 4: Maximum absolute error, order of convergence, and CPU time in seconds for Example 3 at time instant $t = 1$ with time step size $\Delta t = \Delta x^4$.

Types of scheme	Grid points	Maximum error	Rate of convergence	Aver. number of iteration	CPU time in sec.
FODs	11	8.3375×10^{-8}	—	3	0.516
	21	5.2632×10^{-9}	3.9812	2	8.594
	41	3.3491×10^{-10}	3.9774	2	200.015

In order to analyze the results found in application to the Burgers equation (36), Table 3 demonstrates rate of convergence, average number of iteration at each time step, and CPU time required to obtain the solution of Example 3 by using FODs at time instant $t = 1$ when $\Delta x = 0.05$ with various time step sizes. Table 4 shows the rate of convergence, average number of iteration at each time step, and CPU time required to obtain the solution of Example 3 at time instant $t = 1$ and using uniform grids of 11, 21, and 41 with time step sizes $\Delta t = \Delta x^4$ and $\epsilon = 10^{-10}$.

It can be seen from Tables 3 and 4 that numerical results are in good agreement with the exact solution. We only observe $O(\Delta t)$ convergence rate and the error is dominated by time error. An explanation for this phenomenon is due to the nonlinear term, which is approximated at time level n, instead of at time level $n + 1/2$ for the FODs (38).

4. Conclusion

In this paper, a new set of fourth-order schemes for the one-dimensional heat conduction problem with Dirichlet boundary conditions is constructed using a deferred correction technique. The construction of high-order deferred correction schemes requires only a regular three-point stencil at higher time level which is similar to the standard second-order Crank-Nikolson method. The greatest significance of FODs, compared with FOCs and FOSs, is the easier development and that it can be solved by using Thomas algorithms. Numerical examples confirm the order of accuracy. We also implement our algorithms to nonlinear problems. However, theoretical analysis for nonlinear problems needs further investigation. Posterior idea for this project is to use another way to make uu_x term as follows [21, 22]:

$$u^{n+1}(u_x)_i^{n+1} \approx u^{n+1}(u_x)_i^n + u^n(u_x)_i^{n+1} - u^n(u_x)_i^n, \qquad (40)$$

where better results are expected to be found. The first two terms on the right-hand side of above equation make the coefficient matrices of FOCs, FODs, and FOSs vary with time. That is, the inverse coefficient matrices of FOCs and FOSs have to be stored on each step of time while FODs have no need. For this reason, the FODs is simple to implement although FODs need more iterations for the convergence of solution on each step of time.

Acknowledgments

This work is financially supported by the Commission on Higher Education (CHE), the Thailand Research Fund (TRF), the University of Phayao (UP), Project MRG5580014. The author would like to express their deep appreciation to Professor Sergey Meleshko for the kind assistance and valuable advice on the REDUCE calculations.

References

[1] Y. Adam, "Highly accurate compact implicit methods and boundary conditions," *Journal of Computational Physics*, vol. 24, no. 1, pp. 10–22, 1977.

[2] G. F. Carey and W. F. Spotz, "Higher-order compact mixed methods," *Communications in Numerical Methods in Engineering with Biomedical Applications*, vol. 13, no. 7, pp. 553–564, 1997.

[3] M. H. Carpenter, D. Gottlieb, and S. Abarbanel, "Stable and accurate boundary treatments for compact, high-order finite-difference schemes," *Applied Numerical Mathematics*, vol. 12, no. 1–3, pp. 55–87, 1993.

[4] I. Christie, "Upwind compact finite difference schemes," *Journal of Computational Physics*, vol. 59, no. 3, pp. 353–368, 1985.

[5] P. C. Chu and C. Fan, "A three-point combined compact difference scheme," *Journal of Computational Physics*, vol. 140, no. 2, pp. 370–399, 1998.

[6] P. C. Chu and C. Fan, "A three-point sixth-order nonuniform combined compact difference scheme," *Journal of Computational Physics*, vol. 148, no. 2, pp. 663–674, 1999.

[7] W. Dai and R. Nassar, "A compact finite difference scheme for solving a three-dimensional heat transport equation in a thin film," *Numerical Methods for Partial Differential Equations*, vol. 16, no. 5, pp. 441–458, 2000.

[8] W. Dai and R. Nassar, "Compact ADI method for solving parabolic differential equations," *Numerical Methods for Partial Differential Equations*, vol. 18, no. 2, pp. 129–142, 2002.

[9] W. Dai, "A new accurate finite difference scheme for Neumann (insulated) boundary condition of heat conduction," *International Journal of Thermal Sciences*, vol. 49, no. 3, pp. 571–579, 2010.

[10] X. Deng and H. Maekawa, "Compact high-order accurate nonlinear schemes," *Journal of Computational Physics*, vol. 130, no. 1, pp. 77–91, 1997.

[11] J. H. Ferziger and M. Peric, *Computational Methods in Fluid Dynamics*, Springer, Berlin, Germany, 2002.

[12] J. C. Kalita, D. C. Dalal, and A. K. Dass, "A class of higher order compact schemes for the unsteady two-dimensional convection-diffusion equation with variable convection coefficients," *International Journal for Numerical Methods in Fluids*, vol. 38, no. 12, pp. 1111–1131, 2002.

[13] S. Karaa and J. Zhang, "High order ADI method for solving unsteady convection-diffusion problems," *Journal of Computational Physics*, vol. 198, no. 1, pp. 1–9, 2004.

[14] S. K. Lele, "Compact finite difference schemes with spectral-like resolution," *Journal of Computational Physics*, vol. 103, no. 1, pp. 16–42, 1992.

[15] J. Li, Y. Chen, and G. Liu, "High-order compact ADI methods for parabolic equations," *Computers & Mathematics with Applications*, vol. 52, no. 8-9, pp. 1343–1356, 2006.

[16] I. M. Navon and H. A. Riphagen, "An implicit compact fourth-order algorithms for solving the shallow-water equations in conservation-law form," *Monthly Weather Review*, vol. 107, no. 9, pp. 1107–1127, 1979.

[17] J. Zhao, W. Dai, and T. Niu, "Fourth-order compact schemes of a heat conduction problem with Neumann boundary conditions," *Numerical Methods for Partial Differential Equations*, vol. 23, no. 5, pp. 949–959, 2007.

[18] J. Zhao, W. Dai, and S. Zhang, "Fourth-order compact schemes for solving multidimensional heat problems with Neumann boundary conditions," *Numerical Methods for Partial Differential Equations*, vol. 24, no. 1, pp. 165–178, 2008.

[19] V. Pereyra, "On improving an approximate solution of a functional equation by deferred corrections," *Numerische Mathematik*, vol. 8, pp. 376–391, 1966.

[20] V. Pereyra, "Iterated deferred corrections for nonlinear boundary value problems," *Numerische Mathematik*, vol. 11, pp. 111–125, 1968.

[21] M. C. Miriane, M. G. Paulo, and C. R. Estaner, "Error analysis in the solution of unsteady nonlinear convection-diffusion problems," *Research Journal of Physical and Applied Science*, vol. 1, no. 1, pp. 20–22, 2012.

[22] B. Jiang, *The Least-Squares Finite Element Method: Theory and Applications in Computational Fluid Dynamics and Electromagnetics*, Scientific Computation, Springer, Berlin, Germany, 1998.