View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Scholarship at UWindsor

University of Windsor

Scholarship at UWindsor

Electronic Theses and Dissertations

Theses, Dissertations, and Major Papers

2014

Development of a Generalized Finite Difference Scheme for Convection-Diffusion Equation

Shixin He
University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation

He, Shixin, "Development of a Generalized Finite Difference Scheme for Convection-Diffusion Equation" (2014). *Electronic Theses and Dissertations*. 5132.
https://scholar.uwindsor.ca/etd/5132

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.

Development of a Generalized Finite Difference Scheme for Convection-Diffusion Equation

By

Shixin He

A Thesis
Submitted to the Faculty of Graduate Studies
through the Department ofMechanical, Automotive and Materials Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science
at theUniversity of Windsor

Windsor, Ontario, Canada

2014

© 2014 Shixin He

Development of a Generalized Finite Difference Scheme for Convection-Diffusion Equation

by

Shixin He

APPROVED BY:

R. Balachandar,
Department of Civil and Environmental Engineering

G.W. Rankin,
Departmentof Mechanical, Automotive and Materials Engineering

R.M. Barron, Advisor
Department of Mechanical, Automotive and Materials Engineering
/ Department of Mathematics and Statistics

May 20, 2014

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

The traditional finite difference method has an important limitation in practical applications, which is the requirement of a structured grid. The purpose of this thesis is to improve the finite difference scheme for application on complex domains. The analysis of the Finite Difference method is carried out for 1D model problems governed by the convection-diffusion equation. The Stencil Mapping method is developed for complex domains. One of the features of this new scheme is that the value at a node can be calculated by using only the neighbouring values on the 3-point stencil. This allows finite differencing for arbitrary nodal distribution in the mesh, and is developed for 2nd-order and 4th-order differencing schemes. The numerical solutions for typical boundary and initial value problems are compared with exact solutions. Local truncation error is introduced as an effective parameter to assess accuracy of the scheme. An adaptive meshing procedure is also presented.

DEDICATION

To my parents

For their endless love, support and encouragement

And to my fiancé

For his moral support and putting up with me when I got stressed

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor, Dr. Barron, for the continuous support of my master's study and research, for his patience, motivation, enthusiasm and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my master study.

Besides my supervisor, I would like to thank the rest of my thesis committee: Dr. Balachandar and Dr. Rankin, for their encouragement and insightful comments.

TABLE OF CONTENTS

DECLARATION OF ORIGINALITY iii

ABSTRACT iv

DEDICATION v

ACKNOWLEDGEMENTS vi

LIST OF TABLES x

LIST OF FIGURES xi

LIST OF ABBREVIATIONS/SYMBOLS xiv

NOMENCLATURE xv

CHAPTER 1 INTRODUCTION 1

- *1.1 Background* 1
- *1.2 One-dimensional Models* 2
- *1.3 Classification of Model Equations* 5
- *1.4 Clustering Functions* 6
- *1.5 Research Objectives* 7
- *1.6 Thesis Layout* 9

CHAPTER 2 TRADITIONAL FINITE DIFFERENCE METHOD 10

- *2.1 Uniform Mesh* 10
- *2.2 Multi-block Mesh* 12
  - *2.2.1 Method A* 13
  - *2.2.2 Method B* 15
- *2.3 Clustered Mesh* 16
- *2.4 Examples* 18
  - *2.4.1 Uniform Mesh* 18
    - *2.4.1.1 Laplace Equation* 18
    - *2.4.1.2 Convection-Diffusion Equation* 19
  - *2.4.2 Multi-block Mesh* 19
    - *2.4.2.1 Laplace Equation* 20

2.4.2.2 Convection-Diffusion Equation 20
2.4.3 Clustered Mesh 21
2.4.3.1 Laplace Equation 21
2.4.3.2 Convection-Diffusion Equation 22
2.5 Summary 26

CHAPTER 3 CELL-CENTRED FINITE DIFFERENCE METHOD 27
3.1 General 1D CCFD Formulation 28
3.2 Examples for Steady Equations 29
3.2.1 Uniform Mesh 29
3.2.1.1 Laplace Equation 29
3.2.1.2 Steady Convection-Diffusion Equation 30
3.2.2 Clustered Mesh 31
3.2.2.1 Laplace Equation 31
3.2.2.2 Steady Convection-Diffusion Equation 32
3.3 Analysis of CCFD Results 33
3.4 Other Methods to Update Nodal Values 35
3.4.1 Shifting Method 35
3.4.2 Differencing Method 37
3.4.2.1 Uniform Mesh 37
3.4.2.2 Clustered Mesh 38
3.5 Unsteady Problems 39
3.5.1 Parabolic Equation 40
3.5.1.1 Stability Analysis 40
3.5.1.2 Example 41
3.5.1.3 Analysis of Results 42
3.5.2 Hyperbolic Equation 42
3.5.2.1 Stability Analysis 43
3.5.2.2 Example 43
3.5.2.3 Analysis of Results 44
3.6 Summary 45

CHAPTER 4 A NEW STENCIL MAPPING METHOD 46
4.1 The Stencil Mapping 47

4.2 Transformed Equation and Boundary Conditions 48
4.3 Discretisation and Implementation Issues 49
  4.3.1 FDE – Second Order Scheme 49
    4.3.1.1 Local Truncation Error 49
    4.3.1.2 Calculation of Local Truncation Error 51
  4.3.2 FDE – Fourth Order Scheme 51
    4.3.2.1 Local Truncation Error 53
    4.3.2.2Calculation of the Local Truncation Error 54
4.4 Results 55
  4.4.1 Poisson Equation 55
    4.4.1.1 Uniform Mesh 55
    4.4.1.2 Clustered Mesh 59
    4.4.1.3 Largest Absolute Error 64
    4.4.1.4 2nd-order scheme vs. 4th-order scheme 65
  4.4.2 Convection-Diffusion Equation 66
    4.4.2.1 Uniform Mesh 66
    4.4.2.2 Clustered mesh 67
    4.4.2.3 Largest Absolute Error 69
4.5 Adaptive Meshing 69
4.6 Summary 71

CHAPTER 5 CONCLUSIONS AND RECOMMENDATIONS 73
5.1 Conclusions 73
5.2 Recommendations 74

REFERENCES 75

VITA AUCTORIS 77

LIST OF TABLES

Table 4.1 Largest absolute error for each case for Poisson equation 65
Table 4.2 Maximum local truncation error for Poisson equation 65
Table 4.3 Largest absolute error for each case for convection-diffusion equation 69

LIST OF FIGURES

Figure 1.1 Boundary conditions and domain in 1D 4

Figure 2.1 Single block 1D uniform mesh with five cells 11

Figure 2.2 1D 4-block mesh 13

Figure 2.3 1D 4-block mesh with position of the “ghost” nodes 13

Figure 2.4 Mapping a 1D non-uniform mesh to a uniform mesh 17

Figure 2.5 Comparison of exact and TFDM solutions of Laplace equation (uniform mesh; N = 40) 18

Figure 2.6 Comparison of exact and TFDM solutions of steady convection-diffusion equation (uniform mesh; N = 40; *R* = 10) 19

Figure 2.7 Comparison of exact and TFDM solutions of Laplace equation (4-block mesh; N = 24) 20

Figure 2.8 Comparison of exact and TFDM solutions of steady convection-diffusion equation (4-block mesh; N = 24; R = 10) 21

Figure 2.9 Comparison of exact and TFDM solutions of Laplace equation (clustered mesh; N = 50) 22

Figure 2.10 Comparison of exact and TFDM solutions of steady convection-diffusion equation (clustered mesh; N = 50; *R* = 10) 23

Figure 2.11 Comparison of exact and TFDM solutions of steady convection-diffusion equation on meshes with different number of cells (clustered mesh; *R* = 10) 24

Figure 2.12 Comparison of exact and TFDM solutions of steady convection-diffusion equation (clustered mesh with *B* = 1.2 and *B* = 1.75; N = 50; *R* = 10) 25

Figure 2.13 Comparison of exact and TFDM solutions of steady convection-diffusion equation (clustered mesh 02 with *B* = 1.75; N = 50; *R* = 10, 30, 50) 25

Figure 3.1 Cell-Centred stencils in an arbitrary 2D mesh 28

Figure 3.2 Two adjacent cells in a 1D arbitrary mesh 28

Figure 3.3 Comparison of exact and CCFD solutions of the Laplace equation (uniform mesh; N = 50; averaging method) 30

Figure 3.4 Comparison of exact and CCFD solutions of convection-diffusion equation (uniform mesh; N = 50; averaging method) 31

Figure 3.5 Comparison of exact and CCFD solutions of Laplace equation (clustered mesh; N = 50; averaging method) 32

Figure 3.6 Comparison of exact and CCFD solutions of steady convection-diffusion equation (clustered mesh; N = 50; averaging method) 33

Figure 3.7 Mesh with nodes at domain boundaries 36

Figure 3.8 Mesh with cell-centres at domain boundaries 36

Figure 3.9 Comparison of exact and CCFD solutions of steady convection-diffusion equation (uniform mesh; N = 50; shifting method) 37

Figure 3.10 Comparison of exact and CCFD solutions of steady convection-diffusion equation (uniform mesh; N = 50; differencing method) 38

Figure 3.11 Comparison of exact and CCFD solutions of steady convection-diffusion equation (clustered mesh; N = 50; differencing method) 39

Figure 3.12 Solution of the model parabolic equation (uniform mesh; N = 40; averaging method; FTCS) 41

Figure 3.13 Solution of model hyperbolic equation (uniform mesh; N = 80; averaging method; FTCS) 44

Figure 4.1 Arbitrary mesh along *x*-axis 47

Figure 4.2 Stencil map 47

Figure 4.3 West boundary for a 1D mesh 48

Figure 4.4 Stencil with five discretisation points for the 4th-order scheme 52

Figure 4.5 Numerical and exact results for Poisson equation (uniform mesh; N = 50; Dirichlet boundaries) 56

Figure 4.6 Local truncation error for Poisson equation (uniform mesh; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme) 56

Figure 4.7 Numerical and exact results for Poisson equation (uniform mesh; N = 50; west Neumann boundary) 57

Figure 4.8 Local truncation error for Poisson equation (uniform mesh; N = 50; west Neumann boundary; 2nd-order vs. 4th-order scheme) 57

Figure 4.9 Numerical and exact results for Poisson equation (uniform mesh; N = 50; east Neumann boundary) 58

Figure 4.10 Local truncation error for Poisson equation (uniform mesh; N = 50; east Neumann boundary; 2nd-order vs. 4th-order scheme) 58

Figure 4.11 Numerical and exact results for Poisson equation (clustered mesh 01; N = 50; Dirichlet boundaries) 59

Figure 4.12 Local truncation error for Poisson equation (clustered mesh 01; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme) 59

Figure 4.13 Numerical and exact results for Poisson equation (clustered mesh 02; N = 50; Dirichlet boundaries) 60

Figure 4.14 Local truncation error for Poisson equation (clustered mesh 02; N = 50; Dirichlet boundary; 2nd-order vs. 4th-order scheme) 60

Figure 4.15 Numerical and exact results for Poisson equation (clustered mesh 01; N = 50; west Neumann boundary)........61

Figure 4.16 Local truncation error for Poisson equation (clustered mesh 01; N = 50; west Neumann boundary; 2nd-order vs. 4th-order scheme)........61

Figure 4.17 Numerical and exact results for Poisson equation (clustered mesh 02; N = 50; west Neumann boundary)........62

Figure 4.18 Local truncation error for Poisson equation (clustered mesh 02; N = 50; west Neumann boundary; 2nd-order vs. 4th-order scheme)........62

Figure 4.19 Numerical and exact results for Poisson equation (clustered mesh 01; N = 50; east Neumann boundary)........63

Figure 4.20 Local truncation error for Poisson equation (clustered mesh 01; N =50; east Neumann boundary; 2nd-order vs. 4th-order scheme)........63

Figure 4.21 Numerical and exact results for Poisson equation (clustered mesh 02; N = 50; east Neumann boundary)........64

Figure 4.22 Local truncation error for Poisson equation (clustered mesh 02; N = 50; east Neumann boundary; 2nd-order vs. 4th-order scheme)........64

Figure 4.23 Numerical and exact results for steady convection-diffusion equation (uniform mesh; N = 50; Dirichlet boundaries)........66

Figure 4.24 Local truncation error for steady convection-diffusion equation (uniform mesh; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme)........67

Figure 4.25 Numerical and exact results for steady convection-diffusion equation (clustered mesh 01; N = 50; Dirichlet boundaries)........67

Figure 4.26 Local truncation error for steady convection-diffusion equation (clustered mesh 01; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme)........68

Figure 4.27 Numerical and exact results for steady convection-diffusion equation (clustered mesh 02; N = 50; Dirichlet boundaries)........68

Figure 4.28 Local truncation error for steady convection-diffusion equation (clustered mesh 01; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme)........69

Figure 4.29 Absolute error comparison for adaptive meshing (initial uniform mesh with N = 20; convection-diffusion equation)........71

Figure 4.30 Numerical and exact results for adaptive meshes (initial uniform mesh with N = 20; convection-diffusion equation)........71

LIST OF ABBREVIATIONS/SYMBOLS

CFD	Computational Fluid Dynamics
DIFCA	Distance-Function-Based Cartesian
1D	One-Dimensional
2D	Two-Dimensional
3D	Three-Dimensional
PDE	Partial Differential Equation
CCFD	Cell-Centred Finite Difference
FD	Finite Difference
BLK	Block
TFDM	Traditional Finite Difference Method
ODE	Ordinary Differential Equation
MODE	Modified Ordinary Differential Equation
FDE	Finite Difference Equation
L.T.E.	Local Truncation Error
BVP	Boundary Value Problem
Abs_error	Absolute Error

NOMENCLATURE

φ , u	general variables
S_{φ},S_u	source terms
v	viscosity
U	reference velocity
L	reference length
$\bar{u}$, $\bar{x}$, $\bar{t}$, $\bar{p}$	dimensionless variables
Re	Reynolds number
u_{LB}	left boundary value
u_{RB}	right boundary value
κ	thermal conductivity
R	diffusion coefficient
a	convective speed
x	physical coordinate
ξ	computational coordinate
B	clustering parameter
β	coefficient to determine central or backward differencing
u_i	nodal value at node i
u_0	west (left) boundary value
u_L	east (right) boundary value
N	number of cells

NJ	number of cells in blocks J
uJ_i	nodal value at node *i* in block J
x', x''	transformation metrics
BLK J	block J
i	node or cell number
c_i	cell centre in cell *i*
Δx_i	length of cell *i*
d	diffusion number
C	Courant number
U_{n+1}/U_n	amplification factor
Δt	time step
t_{max}	maximum time
x_W	west node coordinate in the stencil
x_P	centre node coordinate in the stencil
x_E	east node coordinate in the stencil
x_L	left central coordinate in the stencil
x_R	right central coordinate in the stencil
u_L	left central value in the stencil
u_R	right central value in the stencil
u_W	west nodal value in the stencil
u_E	east nodal value in the stencil
u_P	solution value at node *P*

CHAPTER 1

INTRODUCTION

1.1 Background

Generally, for Computational Fluid Dynamics (CFD), there are three types of mesh-based discretisation methods: Finite Difference, Finite Volume and Finite Element. Each method has their strengths and weaknesses. Among these methods, Finite Difference is the most efficient and, since it is a relatively straightforward method, it is often used in developing numerical formulations to deal with initial and boundary value problems. Another strength of the finite difference method is that Taylor series expansion can be easily applied to analyze local truncation errors. This property can be exploited to analyze the accuracy of the solution. But there is an important limitation in applying the finite difference method, which is the requirement of a structured grid. Therefore, the method cannot easily be applied on complex domains, making the finite difference method unpopular in commercial CFD software. However, since the finite volume and finite element methods are also not without restrictions, some researchers have preferred to improve the traditional finite difference method to make it useful for a wider range of applications. This has led to the development of structured grid generation techniques, algorithms for solution of the governing equations in curvilinear coordinates and multi-block methods. The fundamental concepts of the finite difference and finite volume methods are explained in details in many CFD books, eg., Hoffmann and Chiang[1], Ramshaw[2], Lomax et al.[3], Anderson et al.[4], Roache[5], Anderson[6], Versteeg and Malalasekera[7], Ferziger and Peric[8], Chung[9] and Patankar[10].

The general convection-diffusion equation is often used to study the utility of new algorithms for the Navier-Stokes equations (eg. see [7, 8, 10]). The convection-diffusion equation is also popular for the study of some specific issues in numerical schemes, such as Kalita's[11] study on the effects of clustering on simulations, the significance of "wiggles" in the numerical results by Gresho and Lee[12], Thiart's[13] research on solving fluid flow and heat transfer problems on non-staggered grids, and the work of Date[14] on collocated variables for unstructured meshes. A variety of new methods have been developed by researchers attempting to find an approach that makes the application to complex meshes easier. For example, Chai and Yap[15] developed a distance-function-based Cartesian (DIFCA) grid finite volume method for irregular geometries and applied the algorithm on the convection-diffusion equation. A mesh-free finite difference method based on the Poisson equation has been developed by Seibold[16].

Due to the significant commonalities among the governing equations of the flow of a Newtonian fluid, such as the continuity equation, momentum equations and energy equation, it is convenient to introduce a general variable φ to express the conservative form of these equations. The general conservative form of the convection-diffusion equation can be written as

$$\frac{\partial\varphi}{\partial t} + \vec{\nabla} \cdot (\varphi\vec{u}) - \vec{\nabla} \cdot \left(\Gamma\vec{\nabla}\varphi\right) = S_{\varphi} \, . \tag{1.1}$$

Equation (1.1) is the general transport equation for incompressible flow. Property φ can stand for velocity components, temperature or some other variable in physical problems. To be precise, if φ equals to 1, $\Gamma = 0$ and $S_{\varphi} = 0$, equation (1.1) is the mass conservation equation; if φ equals to u, $\Gamma = v$ (viscosity) and $S_{\varphi} = S_u - \frac{1}{\rho}\frac{\partial p}{\partial x}$, equation (1.1) is the x-momentum equation; if φ equal to v, $\Gamma = v$ and $S_{\varphi} = S_v - \frac{1}{\rho}\frac{\partial p}{\partial y}$, equation (1.1) is y-momentum equation; if φ equal to w, $\Gamma = v$ and $S_{\varphi} = S_w - \frac{1}{\rho}\frac{\partial p}{\partial z}$, equation (1.1) is z-momentum equation; if φ equal to e_t, $\Gamma = \kappa\rho$ and $S_{\varphi} = S_{e_t} - \frac{\emptyset}{\rho} - \frac{(\vec{\nabla}\cdot\vec{u})P}{\rho}$, equation (1.1) is the energy equation.

Equation (1.1) also clearly emphasizes the physical background in fluid mechanics for transport processes: (rate of increase of φ in the fluid element) + (net rate of flow of φ out of the fluid element) + (rate of increase of φ due to diffusion) = (rate of increase of φ due to sources).

1.2 One-dimensional Models

The focus in this thesis is the development of a new generalized finite difference methodology. This new approach is explained and subsequently validated using one-dimensional mathematical models. The following examples illustrate how these model equations are obtained from the transport equation (1.1) and are representative of the equations typically encountered in engineering and physical problems.

Example 1.Nondimensional 1D Transport Equation

If the general transport equation is used to express the *x*-momentum equation, equation (1.1) becomes

$$\frac{\partial u}{\partial t} + \vec{\nabla} \cdot (u\vec{u}) - \nu \nabla^2 u = -\frac{1}{\rho}\frac{\partial p}{\partial x} + S_u \,. \qquad (1.2)$$

Re-arranging equation (1.2) using differentiation rules,

$$\frac{\partial u}{\partial t} + \vec{u} \cdot \left(\vec{\nabla} u\right) + u\vec{\nabla} \cdot \vec{u} - \nu \nabla^2 u = -\frac{1}{\rho}\frac{\partial p}{\partial x} + S_u. \qquad (1.3)$$

Due to the conservation of mass, $\vec{\nabla} \cdot \vec{u} = 0$. Therefore, equation (1.3) is

$$\frac{\partial u}{\partial t} + \vec{u} \cdot \left(\vec{\nabla} u\right) - \nu \nabla^2 u = -\frac{1}{\rho}\frac{\partial p}{\partial x} + S_u \,. \qquad (1.4)$$

The one-dimensional (1D) version of this equation is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + S_u \,. \qquad (1.5)$$

The numerical modeling of equation (1.5) is simplified if it is expressed in nondimensional form. Generally, there are three major advantages in nondimensionalization: reduce the total number of parameters; nondimensionalized parameters have a more transparent meaning; parameters and variables can be rescaled to make the computed quantities have relatively similar magnitudes[17]. Furthermore, the nondimensional transport equation can be applied to problems with the same boundary conditions for different cases and the results from these cases can be easily compared. For nondimensionalization, a set of dimensionless variables needs to be defined.

Define reference velocity magnitude U, length L, and dimensionless variables $\bar{u}$, $\bar{x}$, $\bar{t}$, $\bar{p}$ by $\bar{u} = \frac{u}{U}$, $\bar{x} = \frac{x}{L}$, $\bar{t} = \frac{t}{L/U}$, $\bar{p} = \frac{p}{\rho U^2}$. Substituting these expressions into equation (1.5), the nondimensional *x*-momentum equation in 1D is the convection-diffusion equation

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \bar{u}\frac{\partial \bar{u}}{\partial \bar{x}} - \frac{1}{Re}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} = -\frac{\partial \bar{p}}{\partial \bar{x}} + \bar{S}_u \qquad (1.6)$$

where $\bar{u}$ is the velocity,$Re = \frac{LU}{\nu}$ is the Reynolds number, $-\frac{\partial \bar{p}}{\partial \bar{x}}$ is the pressure gradient and $\bar{S}_u$ is a source term.

This transport equation is classified as a non-linear partial differential equation (PDE) since it includes a product of the dependent variable and its derivative.

Example 2. Linear 1D Initial Value Problem/ Boundary Value Problem

A model linear PDE is obtained by assuming that the convective speed, which is the coefficient of the $\frac{\partial u}{\partial x}$ term, is constant. Combining the pressure gradient with the source term in (1.5), the 1D linear transport equation becomes

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = S \tag{1.7}$$

where a is the constant convective velocity. Let us consider equation (1.7) on $x \in [0, L]$ with initial and boundary conditions $u(x, 0) = u^0$, $u(0, t) = u_{LB}$ and $u(L, t) = u_{RB}$ respectively. Here u_{LB} and u_{RB} are constant values. Figure 1.1 shows the physical domain and the imposed boundary conditions.

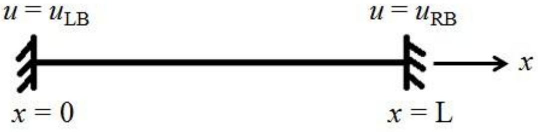


Figure 1.1 Boundary conditions and domain in 1D

Define dimensionless variables $\bar{u} = \frac{u-u_{LB}}{\Delta u}$, $\bar{x} = \frac{x}{L}$, $\bar{t} = \frac{\Delta u}{L}t$, $\bar{a} = \frac{a}{\Delta u}$, $\bar{S} = \frac{L}{(\Delta u)^2}S$, where $\Delta u = u_{RB} - u_{LB}$. Note that $x = 0$, is mapped to $\bar{x} = 0$ and $x = L$ is mapped to $\bar{x} = 1$.

Using $u = u_{LB} + (\Delta u)\bar{u}$, substitute u into equation (1.7) to obtain the nondimensional PDE

$$\frac{\partial \bar{u}}{\partial \bar{t}} + \bar{a}\frac{\partial \bar{u}}{\partial \bar{x}} - \frac{1}{D}\frac{\partial^2 \bar{u}}{\partial \bar{x}^2} = \bar{S} \tag{1.8}$$

where $\bar{a}$ is the constant wave speed, $D^{-1} = \frac{\nu}{L(\Delta u)}$ is called the “diffusion” coefficient and the nondimensionalized domain becomes a unit domain, $\bar{x} \in [0,1]$.

The initial condition becomes $u^0 = u_{LB} + (\Delta u)\bar{u}(\bar{x}, 0)$, which can be re-arranged to give

$$\bar{u}(\bar{x}, 0) = \overline{u^0} \, . \tag{1.9}$$

Similarly, the nondimensional boundary conditions become

$$\bar{u}(0,\bar{t}) = 0 \text{ and } \bar{u}(1,\bar{t}) = 1. \tag{1.10}$$

Example 3. 1D Heat Conduction in a Solid Material

Using the nondimensional partial differential equation (1.8), take $\bar{a} = 0, \frac{1}{D} = \kappa, \bar{u} = \bar{T}$.

Then the governing heat conduction equation is written as

$$\frac{\partial \bar{T}}{\partial \bar{t}} - \kappa \frac{\partial^2 \bar{T}}{\partial \bar{x}^2} = \bar{S} \tag{1.11}$$

where κ is the thermal conductivity.

1.3 Classification of Model Equations

The above three examples show how the general transport equation can be nondimensionalized, and how the nondimensional transport equation corresponds to specific mathematical models. These second-order PDEs can be further classified. The classification is crucial when deciding how to discretize these second-order PDEs so that the physics of the phenomenon is properly modeled. Clarifying the different types of PDEs is also beneficial since different forms of boundary and initial conditions are required to formulate the problems with different types of PDEs.

Dropping the bars from the notation for convenience, the 1D nondimensional transport equation (1.8) becomes

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} - \frac{1}{D}\frac{\partial^2 u}{\partial x^2} = S. \tag{1.12}$$

a. If the value for D goes to infinity, the diffusion term drops out and the equation becomes the hyperbolic equation

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = S\ . \tag{1.13}$$

From a geometric interpretation, there are two real characteristic curves for a hyperbolic PDE. Two initial conditions and two boundary conditions restrict the solution domain, which is a conic section. In fluid mechanics problems, when the Reynolds number becomes quite high, the solution is dominated by the convection terms. For hyperbolic equations, the information propagates in certain directions at a certain speed[1].

b. If the value for a, which is the coefficient of the convection term, equals to 0, the equation is the parabolic equation

$$\frac{\partial u}{\partial t} - \frac{1}{D}\frac{\partial^2 u}{\partial x^2} = S. \tag{1.14}$$

From a geometrical aspect, only one characteristic curve exists for a parabolic equation. The solution domain will be an open region. The solution forms from the initial plane of data to downstream within the domain, propagating forward in time, meanwhile restricted by the specified boundary conditions [1].

c. The third classification is elliptic equation. For an elliptic PDE, the characteristic curves are imaginary. Any disturbance propagates to every direction in the region at infinite speed. The solution domain is a closed region [1]. Elliptic equations, which describe equilibrium phenomena, can be divided into two groups: Poisson equation and convection-diffusion equation. Setting $a = 0$, and taking steady conditions, equation (1.8) becomes the Poisson equation

$$-u'' = DS\,. \tag{1.15}$$

Otherwise, the elliptic equation is the steady convection-diffusion equation

$$au' - \frac{1}{D}u'' = S\,. \tag{1.16}$$

1.4 Clustering Functions

Mesh quality is an important consideration for all mesh-based numerical simulation methods. If the solution is smooth with small gradients, a uniform mesh will usually suffice. However, if the solution undergoes large gradients, such as in boundary layer flows on no-slip walls, the mesh must be appropriately designed. This usually entails creating a mesh with variable spacing, perhaps small spacing close to the physical boundary and larger spacing further away from the boundary. These grids are referred to as clustered or stretched grids.

The new generalized finite difference method developed in this thesis is designed to easily handle arbitrary grid spacing without any knowledge of the clustering functions used to generate the mesh or, in fact, for nodes that are randomly placed without using any clustering function. However, for the purpose of comparison between the traditional finite difference method and the proposed method, the following two clustering functions have been used in this thesis:

$$x(\xi) = 0.5 + 0.5\,\frac{arcsinh\,[\alpha(2\xi-1)]}{arcsinh\,[\alpha]} \tag{1.17}$$

and

$$\xi(x) = 0.5 - 0.5\,\frac{ln\left|\frac{B+2x-1}{B-2x+1}\right|}{ln\left|\frac{B-1}{B+1}\right|} \tag{1.18}$$

Both of these functions map $x \in [0,1]$ into $\xi \in [0,1]$, with clustering at $x = 0$ and $x = 1$. The mesh obtained from equation (1.17) is referred to as mesh 01 in Chapter 2; the mesh generated from equation (1.18) is called mesh 02. The parameter α in (1.17) is related to the number of nodes in the mesh, $\alpha = 0.5(M + 1)$ where M is the number of nodes in the mesh including endpoints. The parameter B in equation (1.18) must satisfy the condition $1 < B < 2$. B controls the degree of clustering, with more clustering of the nodes near $x = 0$ and $x = 1$ as $B \rightarrow 1$.

1.5 Research Objectives

The objective of this research is to develop a new generalized finite difference method which can be used to solve any PDE in an arbitrarily discretized domain. Although this research is focused only on one-dimensional (1D) problems, it is essential to guarantee that the new method can be applied effectively in 2D and 3D without any potential problems. The development of the algorithm starts by considering the general convection-diffusion equation since it is a model for many physical phenomena encountered in engineering and science. The 1D formulation is easy to derive and the code programming is not as complicated as the 2D or 3D cases. Finding the potential problems in a 1D model and resolving them in a manner that can be extended to higher dimensions can ensure applicability in 2D or 3D.

Two types of finite difference schemes are formulated and analyzed, the Cell-Centred Finite Difference (CCFD) method and the Stencil Mapping method. The research begins with the CCFD method, which has been developed for 2D Poisson equations by Salih[18] and Situ[19]. Their investigations have shown that the method is very successful for Laplace equations, but loses accuracy for the convection-diffusion equation. After testing several cases, our research has identified the interpolation scheme as the source of this inaccuracy. Lakner and Plazl[20] proposed a symbolic-numerical solution procedure based on the finite difference method to solve PDEs on irregular domains. Their concept uses the theory of splines to develop appropriate interpolation schemes, which we have also taken into consideration for the CCFD method. However, although the interpolation problem can be solved by applying the theory of splines [21], it is time-consuming for computer calculation and cannot be easily applied in 2D and 3D. Thus, the research is re-directed to the development of a new generalized finite difference method which we refer to as the Stencil Mapping method. Spotz[22] showed that knowing the mapping derivatives is especially meaningful to the accuracy of the finite difference method. In the proposed Stencil Mapping method, each difference stencil is mapped individually to a generic computational stencil, rather than the entire physical domain being mapped to a computational domain. This is accomplished using a quadratic mapping function, from which the mapping derivatives can be analytically determined.

In this thesis, the proposed Stencil Mapping method is also applied to the development of 4th-order accurate finite difference schemes. High order finite difference schemes usually require non-compact stencils, which use grid points that are not adjacent to the node at which the differencing equations are applied. For example, Castillo et al.[23][24] developed mimetic and support-operator differencing methods for 4th-order schemes on a non-compact stencil. However, non-compact differencing schemes for higher-order accuracy always need special treatment for nodes near boundaries. Therefore, it is desirable to develop a compact stencil algorithm for higher-order schemes. The compact stencil algorithm only uses the two nodes adjacent to the node at which the differencing equation is formulated. In this way, no special equation is needed when applying the discretisation equation on the nodes near boundaries. Some researchers have worked on developing such differencing schemes. For example, Hemker[25] developed a defect correction technique to implement a higher-order scheme compactly, Noye[26] proposed a three-point third-order accurate scheme and Spotz[22] introduced a higher-order compact scheme. The Stencil Mapping method introduced in this thesis is applied to 4th-order differencing schemes, while retaining a compact 3-point stencil. The technique can be applied on schemes that are more accurate than 4th-order, such as 8th-order, and the same 3-point compact stencil can be preserved.

1.6 Thesis Layout

In this research, a computer code has been written for the finite difference method in C programming language, in order to compare the convenience of the proposed schemes from either the formulation or implementation aspect, and to assess the accuracy of the results.

Chapter 2 covers details of the traditional finite difference method, focusing mainly on formulations, how it is implemented in a code, where the complexity is when dealing with a multi-block and clustered meshes and the accuracy of numerical results compared with the exact results.

Chapter 3 discusses a recently proposed finite difference method called Cell-Centred Finite Difference (CCFD). The algorithm is developed for the general convection-diffusion equation. Three types of interpolation – averaging, shifting and differencing, are formulated. Consistency is investigated for the steady convection-diffusion equation and stability is analyzed for the unsteady equation.

Chapter 4 provides the formulation for the new Stencil Mapping method in 1D for the general convection-diffusion equation, applying both 2nd-order and 4th-order differencing schemes. The algorithm is implemented with either Dirichlet or Neumann boundary conditions on either a uniform mesh or clustered mesh to determine if the new scheme is applicable and if the results are accurate. The local truncation error is used to test the accuracy of the scheme.

CHAPTER 2

TRADITIONAL FINITE DIFFERENCE METHOD

As indicated in the introductory chapter, the convection-diffusion equation is often used to analyze new numerical methods for partial differential equations. In this chapter the 1D form of the steady convection-diffusion equation,

$$-\frac{d^2u}{dx^2} + R\frac{du}{dx} = S \tag{2.1}$$

is used to discuss the traditional implementation of the Finite Difference (FD) method. This is equation (1.16) with $R = aD$ and S redefined. Since the FD method is well-known, the emphasis in this chapter is to highlight the key issues that make the method more complicated when applied to complex geometries.

In the Finite Difference methodology, the derivatives are approximated by finite differences, and the differential equation (2.1) is applied at each node in the discretised domain. Several types of 1D mesh, with different size of cells, are tested in this thesis. These mesh types, chosen because they are commonly used for finite difference CFD simulations in complex domains, are known as uniform, clustered and multi-block. After discussing the developments related to these mesh features, results from the various schemes are presented at the end of the chapter.

2.1 Uniform Mesh

To illustrate the basic FD method, consider the solution of the convection-diffusion equation (2.1) on a 1D domain with five cells, as shown in Fig. 2.1. The domain length is 1 unit, with five cells in the domain, with four internal nodes and two boundary nodes. Since the grid spacing is equal, the length of each cell is $\Delta x = 1/5$. For Dirichlet boundary conditions, the west (left) boundary is set to be 0, and the east (right) boundary is set to be 1. As shown in Chapter 1, any Dirichlet boundary value problem associated with the convection-diffusion equation (2.1) can be set up in this generic way.

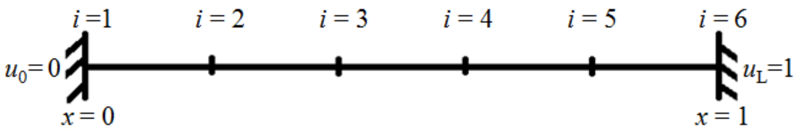


Figure 2.1 Single block 1D uniform mesh with five cells

The second-order accurate central differencing formula is used to approximate the diffusion term in equation (2.1), and either the second-order central differencing or first-order backward differencing scheme is applied for the convection term (assuming $R > 0$). The discretisation equation is

$$\frac{u_{i-1}-2u_i+u_{i+1}}{\Delta x^2} - R\left\{\frac{-(1+\beta)u_{i-1}+2\beta u_i+(1-\beta)u_{i+1}}{2\Delta x}\right\} = -S \qquad (2.2)$$

where index i refers to any node in the domain, i-1 refers to the left-side node of node i and i+1 refers to the right-side node of node i. If $\beta = 0$, central differencing is applied for the convection term, while $\beta = 1$ corresponds to backward differencing.

After applying the discretisation equation (2.2) at every internal node, four coupled linear algebraic equations with four unknowns can be formed. The difference equation (2.2) can be re-arranged as

$$[2+R\Delta x(1+\beta)]u_{i-1} + (-4-2R\beta\Delta x)u_i + [2-R\Delta x(1-\beta)]u_{i+1} = -S. \qquad (2.3)$$

Since the node numbers start from 1 and end at 6, u_1 and u_6 are the west and east boundary conditions, respectively, i.e., $u_1 = 0$ and $u_6 = 1$. Interior nodes run from node numbers 2 to 5.

Therefore, at $i = 2$:

$$[2+R\Delta x(1+\beta)]u_1 + (-4-2R\beta\Delta x)u_2 + [2-R\Delta x(1-\beta)]u_3 = -S \qquad (2.4)$$

At $i = 3$:

$$[2+R\Delta x(1+\beta)]u_2 + (-4-2R\beta\Delta x)u_3 + [2-R\Delta x(1-\beta)]u_4 = -S \qquad (2.5)$$

At $i = 4$:

$$[2+R\Delta x(1+\beta)]u_3 + (-4-2R\beta\Delta x)u_4 + [2-R\Delta x(1-\beta)]u_5 = -S \qquad (2.6)$$

At $i = 5$:

$$[2+R\Delta x(1+\beta)]u_4 + (-4-2R\beta\Delta x)u_5 + [2-R\Delta x(1-\beta)]u_6 = -S \qquad (2.7)$$

Since u_1 and u_6 are boundary nodes, their values are already known. Equations (2.4), (2.5), (2.6) and (2.7) can be written as a system of linear algebraic equations

$$\begin{bmatrix} -4-2R\beta\Delta x & 2-R\Delta x(1-\beta) & 0 & 0 \\ 2+R\Delta x(1+\beta) & -4-2R\beta\Delta x & 2-R\Delta x(1-\beta) & 0 \\ 0 & 2+R\Delta x(1+\beta) & -4-2R\beta\Delta x & 2-R\Delta x(1-\beta) \\ 0 & 0 & 2+R\Delta x(1+\beta) & -4-2R\beta\Delta x \end{bmatrix} \begin{bmatrix} u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix}$$

$$= \begin{bmatrix} -[2+R\Delta x(1+\beta)]u_1 - S \\ -S \\ -S \\ -[2-R\Delta x(1-\beta)]u_6 - S \end{bmatrix} \tag{2.8}$$

The value for β is set to 0 or 1, depending whether central differencing or backward differencing is chosen for discretisation of the convection term. Then, this tri-diagonal matrix equation (2.8) can be solved by applying Thomas' Algorithm [1] to obtain the results for all nodal values.

In general, suppose the uniform mesh has N cells, which means the mesh has (N-1) interior nodes. Applying difference equation (2.3) at each node leads to an (N-1)×(N-1) tri-diagonal matrix equation which can be expressed as

$$trid\big(2+R\Delta x(1+\beta), -4-2R\beta\Delta x, 2-R\Delta x(1-\beta)\big)\vec{u} = \vec{b}.$$

2.2 Multi-block Mesh

In order to use the Finite Difference method for complex domains, the multi-block technique is often implemented. For multi-block mesh problems, the domain is divided into several blocks. Each block (BLK) is then meshed with a structured grid. For purposes of the present discussion, and without loss of generality, the cells inside every block are assumed to be uniform, but the cells in one block can be different from the cells in other blocks. When solving multi-block problems, the key question is about how to deal with the block interfaces. Accurate inter-block communication is essential. The problem of transferring accurate information across block interfaces adds significantly to the complexity of the traditional FD method (TFDM) for practical applications. This is one of the motivations in the current research to develop a new finite difference method that can easily handle block interfaces. In this section, two methods to solve the interface problem are formulated within the context of the TFDM.

Figure 2.2 illustrates an example of a multi-block mesh, which has four blocks. This example will be used in the explanation of inter-block communication schemes.

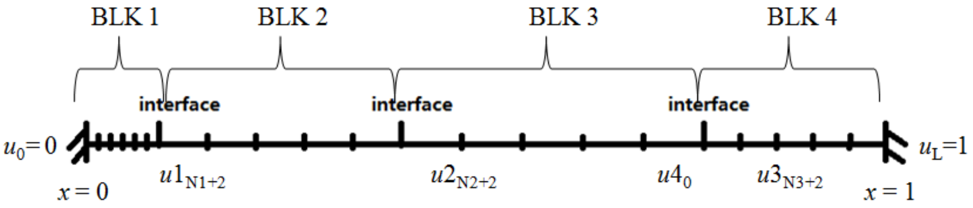


Figure 2.2 1D 4-block mesh

The convection-diffusion equation (2.1) is applied on this 4-block mesh. The domain and the boundary conditions are the same as in section 2.1. The length of the cells in each block is denoted as Δx_1, Δx_2, Δx_3 and Δx_4 for BLK 1, BLK 2, BLK 3 and BLK 4, respectively. The number of cells in BLK *i* is denoted as N*i*.

2.2.1 Method A

Method A deals with the evaluation of the interface nodes by introducing "ghost" or imaginary nodes and using an overlapping procedure. In this 4-block case, there are three interfaces, but four imaginary nodes should be taken into consideration to implement this overlapping method. To simplify the discussion, solution values in BLK *J* are denoted by *uJ*. The detailed iterative procedure is as follows:

i. Initialize imaginary nodal values. Since there are three interfaces, there should be three initialized imaginary nodal values, which are regarded as the east end nodes of BLK 1, BLK 2 and BLK 3. These three imaginary nodes are placed so as to preserve the uniformity in each block, and the value at these nodes is denoted as $u1_{N1+2}$, $u2_{N2+2}$ and $u3_{N3+2}$, respectively. These "ghost" nodes are shown in Fig. 2.3.

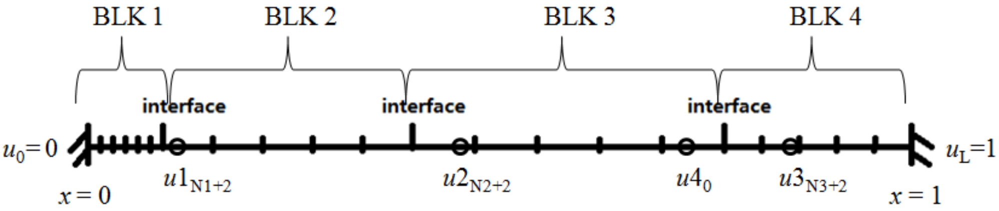


Figure 2.3 1D 4-block mesh with position of the "ghost" nodes

ii. After initializing the value for $u1_{N1+2}$, the east boundary condition (at N1+2) for BLK 1 is considered to be known. The west boundary condition for BLK 1 is the west boundary condition of the domain. Applying the discretisation equation at every interior node of BLK 1, nodal values on BLK 1 can be calculated by solving the resulting (N1-1)×(N1-1) tri-diagonal matrix.

iii. From step ii, the first interface node value $u1_{N1+1}$, which is also $u2_1$, is known. Since the nodal value $u2_{N2+2}$ has already been initialized, the west boundary condition is $u2_1$ and the east boundary condition is $u2_{N2+2}$ for BLK 2. Apply the difference equation (2.3) at every node in BLK 2, from which all nodal values in BLK 2 can be calculated.

iv. Apply the same procedure as in step iii to calculate all nodal values for BLK 3, including the interface node between BLK 3 and BLK 4.

v. The last block in the domain, which is BLK 4, contains the east boundary condition (at N4+1) of the domain. Thus, the east boundary condition for BLK 4 is known, which is $u4_{N4+1}$. The west boundary condition for BLK 4 is $u4_0$ at the fourth imaginary node, which is located on BLK 3 as shown in Fig. 2.3. Find the two neighbour nodes of the imaginary west boundary node located in BLK 3. The solution at these two neighbour nodes can be expressed as $u3_i$ and $u3_{i+1}$, and the locations are $x3_i$ and $x3_{i+1}$. The location for $u4_0$ is denoted as $x4_0$. Apply distance-weighted average to obtain the value for $u4_0$ as

$$u4_0 = \frac{\frac{u3_i}{x4_0 - x3_i} + \frac{u3_{i+1}}{x3_{i+1} - x4_0}}{\frac{1}{x4_0 - x3_i} + \frac{1}{x3_{i+1} - x4_0}} \ . \tag{2.9}$$

Therefore, the west boundary condition for BLK 4 is known. Apply the discretised equation (2.3) on every interior node in BLK 4. Solving the tri-diagonal matrix, the nodal values for BLK 4 can be obtained.

vi. After solving for all the nodal values in BLK 4, the value $u3_{N3+2}$ at imaginary node $x3_{N3+2}$ needs to be updated. Find the two neighbour nodes adjacent to node $x3_{N3+2}$ in BLK 4. The notations for the values at these two nodes are $u4_i$ and $u4_{i+1}$, and the locations of these two nodes are $x4_i$ and $x4_{i+1}$. Use distance-weighted average to obtain the updated value for $u3_{N3+2}$,

$$u3_{N3+2} = \frac{\frac{u4_i}{x3_{N3+2}-x4_i}+\frac{u4_{i+1}}{x4_{i+1}-x3_{N3+2}}}{\frac{1}{x3_{N3+2}-x4_i}+\frac{1}{x4_{i+1}-x3_{N3+2}}}\ . \tag{2.10}$$

vii. Repeat similar calculations as in step iv in BLK 3 to update the nodal value for $u2_{\mathrm{N2+2}}$. Then perform similar calculations as in step iii to update the nodal value for $u1_{\mathrm{N2+2}}$. To complete this iteration, carry out similar calculations as in step ii to update the nodal values on BLK 1.

viii. Repeat the calculations from step ii to vii to iteratively update the nodal values in the domain until the values for $u1_{\mathrm{N1+1}}$, $u2_{\mathrm{N2+1}}$ and $u3_{\mathrm{N3+1}}$ converge to within the user-specified tolerance.

2.2.2 Method B

The same setup as shown in Fig. 2.3 is used to illustrate this method. Method A connects the blocks through imaginary nodes overlapping at each interface during the calculations. For Method B, the connection is at the interface node itself. Assume that the derivative of the solution from the west side of the interface node is the same as the derivative from the east side of the interface node, i.e., assume that the solution is differentiable at the interface node. This condition can be expressed as

$$\frac{du^-}{dx} = \frac{du^+}{dx} \tag{2.11}$$

where – represents west and + represents east.

For example, taking BLK 1 and BLK 2, and using one-sided difference approximations for the derivatives in equation (2.11) gives

$$\frac{du^-}{dx} = \frac{u1_{N1+1}-u1_{N1}}{\Delta x_1} \tag{2.12}$$

$$\frac{du^+}{dx} = \frac{u2_2-u2_1}{\Delta x_2}\ . \tag{2.13}$$

The interface node is the last node in BLK 1, but also the first node in BLK 2. Thus,

$$u1_{N1+1} = u2_1\ . \tag{2.14}$$

Substituting equations (2.12), (2.13), (2.14) into equation (2.11) yields

$$u1_{N1} + \left(-2 + \frac{\Delta x_2}{\Delta x_1+\Delta x_2}\right) u1_{N1-1} = -\frac{\Delta x_1}{\Delta x_1+\Delta x_2} u2_1\ . \tag{2.15}$$

In this method, the nodal value of $u2_2$ should be initialized.

Considering the example illustrated in Fig. 2.3, the solution procedure is as follows:

i. Initialize the three nodal values $u2_2$, $u3_2$ and $u4_2$.

ii. Using the discretisation equation (2.3), set up the matrix system for BLK 1 similar to that in equation (2.8) and calculate the nodal values in BLK 1.

iii. Using similar equations as (2.12), (2.13), (2.14) and (2.15), calculate the nodal values for BLK 2 and BLK 3.

iv. After step iii, the nodal value for the third interface node $u3_{N3+1}$, which is also the nodal value for $u4_1$, is known. Since the east boundary condition for the domain is already known, which is $u4_{N4+1}$, the discretised equation (2.3) can be applied at every node in BLK 4. The nodal values for BLK 4 can be calculated by solving the tri-diagonal matrix.

v. From step iv, the updated nodal value for $u4_2$ is available. Repeat the calculation from step ii to iv, until the nodal values $u2_2$, $u3_2$ and $u4_2$ converge to within the user-specified tolerance.

2.3 Clustered Mesh

In a clustered mesh, the length of the cells is different from each other. In the TFDM, the approach is to map the non-uniform mesh to a uniform one. Figure 2.4 demonstrates this mapping approach. The domain $0 \le x \le 1$ is mapped to $0 \le \xi \le 1$, with the same number of cells. This mapping of the domain can be expressed by a mapping function,

$$x = x(\xi) \ . \tag{2.16}$$

The TFDM needs to be applied in the ξ system, where the nodes are uniformly distributed. As illustrated in Fig. 2.4, the length of every mesh cell in the ξ system is $\Delta\xi$ = 1/N, where N represents the number of cells. To apply the standard finite difference formulae in this mapping approach, the governing differential equation must be transformed into the ξ system. Using chain rules, the terms in the convection-diffusion equation (2.1) transform as

$$\frac{d^2u}{dx^2} = \frac{1}{x'}\frac{d}{d\xi}\left(\frac{1}{x'}\frac{du}{d\xi}\right) = \frac{1}{x'^2}\frac{d^2u}{d\xi^2} - \frac{x''}{x'^3}\frac{du}{d\xi}; \qquad R\frac{du}{dx} = R\frac{1}{x'}\frac{du}{d\xi} \tag{2.17}$$

where $x' = \frac{dx}{d\xi}$ and $x'' = \frac{d^2x}{d\xi^2}$ are the metrics of the transformation (2.16).

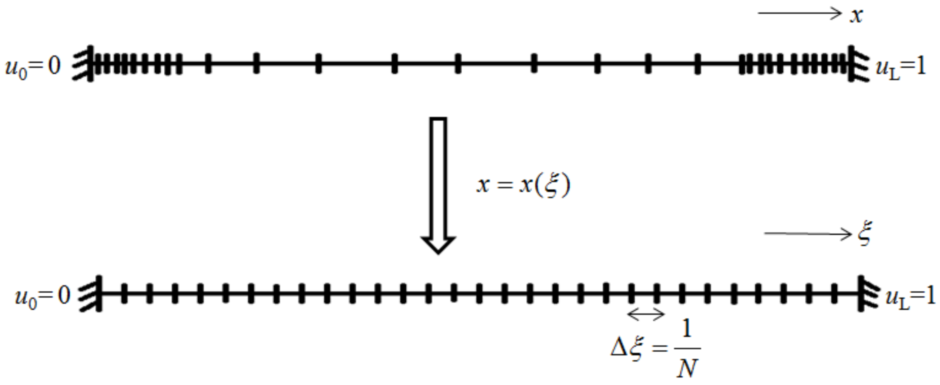


Figure 2.4 Mapping a 1D non-uniform mesh to a uniform mesh

Using equations (2.17), the governing differential equation (2.1) transforms to

$$\frac{1}{x'^2}\frac{d^2u}{d\xi^2}-\frac{x''}{x'^3}\frac{du}{d\xi}-R\frac{1}{x'}\frac{du}{d\xi}=-S \tag{2.18}$$

If the transformation function (2.16) is known analytically, such as the functions defined by equations (1.17) and (1.18), the metrics x' and x'' can be evaluated exactly at all nodes. If the clustered mesh is obtained from grid generation equations, the metrics are approximated at any node, which is denoted by *i* in the ξ system, as

$$x_i'=\frac{dx}{d\xi}|_i=\frac{x_{i+1}-x_{i-1}}{2\Delta\xi} \tag{2.19}$$

$$x_i''=\frac{d^2x}{d\xi^2}|_i=\frac{x_{i+1}-2x_i+x_{i-1}}{(\Delta\xi)^2}. \tag{2.20}$$

Applying three-point central differencing to approximate the diffusion term and backward differencing to approximate the convection term, equation (2.18) is discretized as

$$\frac{1}{x'^2}\left[\frac{u_{i+1}-2u_i+u_{i-1}}{(\Delta\xi)^2}\right]-\frac{x''}{x'^3}\left[\frac{u_{i+1}-u_{i-1}}{2\Delta\xi}\right]-\frac{R}{x'}\left[\frac{u_i-u_{i-1}}{\Delta\xi}\right]=-S \tag{2.21}$$

which can be re-arranged as

$$\left(\frac{1}{x'^2}+\frac{x''\Delta\xi}{2x'^3}+\frac{R\Delta\xi}{x'}\right)u_{i-1}+\left(-\frac{2}{x'^2}-\frac{R\Delta\xi}{x'}\right)u_i+\left(\frac{1}{x'^2}-\frac{x''\Delta\xi}{2x'^3}\right)u_{i+1}=-S(\Delta\xi)^2. \tag{2.22}$$

After substituting equation (2.19) and (2.20) into (2.22), since the location of each node in the *x*-coordinate system is known, a system of linear algebraic equations can be derived. Taking the mesh in Fig. 2.4 as an example, there are N+1 nodes in the domain. The west and east Dirichlet boundary conditions are known, which are 0 and 1 respectively. Thus, after applying discretised equation (2.22) at every node in the domain, an (N-1)×(N-1) matrix system can be set up. Every nodal value in the domain can be evaluated by solving this matrix system.

2.4 Examples

In the following examples, the solution domain is from 0 to 1, the west boundary condition is 0 and the east boundary condition is 1.

2.4.1 Uniform Mesh

The uniform mesh is the simplest of all meshes, and forms the basis for the TFDM. This is primarily due to the fact that the classical finite difference formulae used to approximate derivatives lose at least one order of accuracy if the spacing is non-uniform.

2.4.1.1 Laplace Equation

The Laplace equation, obtained by setting $R = 0$ and $S = 0$ in equation (2.1), i.e.

$$\frac{d^2u}{dx^2} = 0 \tag{2.23}$$

is solved on a uniform mesh by applying the traditional finite difference method. The number of cells is 40. Figure 2.5 shows the comparison between numerical results from the code and the exact solution, given by $u(x) = x$. The numerical results are identical to the exact results.

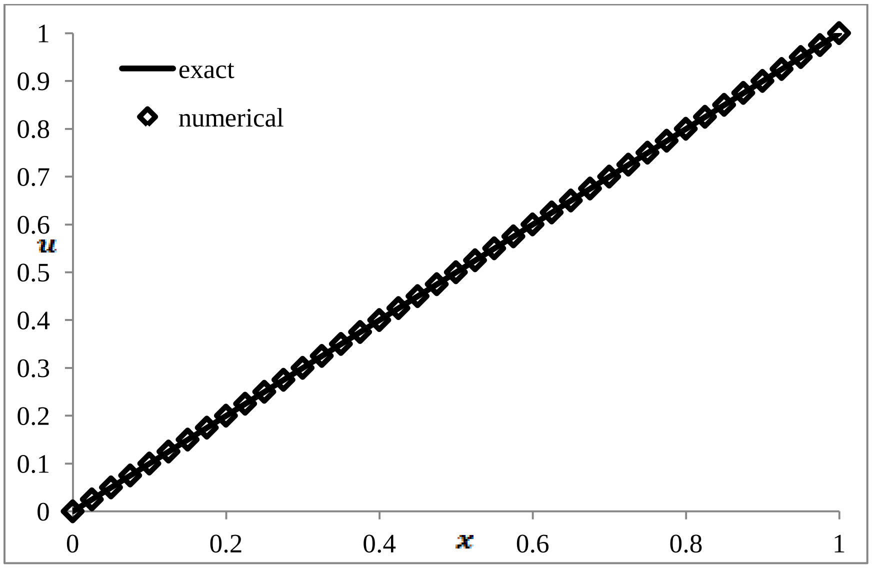


Figure 2.5 Comparison of exact and TFDM solutions of Laplace equation (uniform mesh; N = 40)

2.4.1.2 Convection-Diffusion Equation

In this section, the homogeneous convection-diffusion equation

$$\frac{d^2u}{dx^2} - R\frac{du}{dx} = 0 \tag{2.24}$$

is solved on a uniform mesh by applying the traditional finite difference method. The exact solution of the boundaryvalue problem is

$$u(x) = \frac{1-e^{Rx}}{1-e^{R}}\,. \tag{2.25}$$

Figure 2.6 shows the comparison between the numerical results and the exact solution. The number of cells used for the simulation is 40. Central differencing is applied to the diffusion term, backward differencing is applied on the convection term, and *R* is set to 10.

As illustrated in Fig. 2.6, there are some differences between the numerical results and the exact solution. The largest absolute error has the value 0.042 and occurs at node 37.

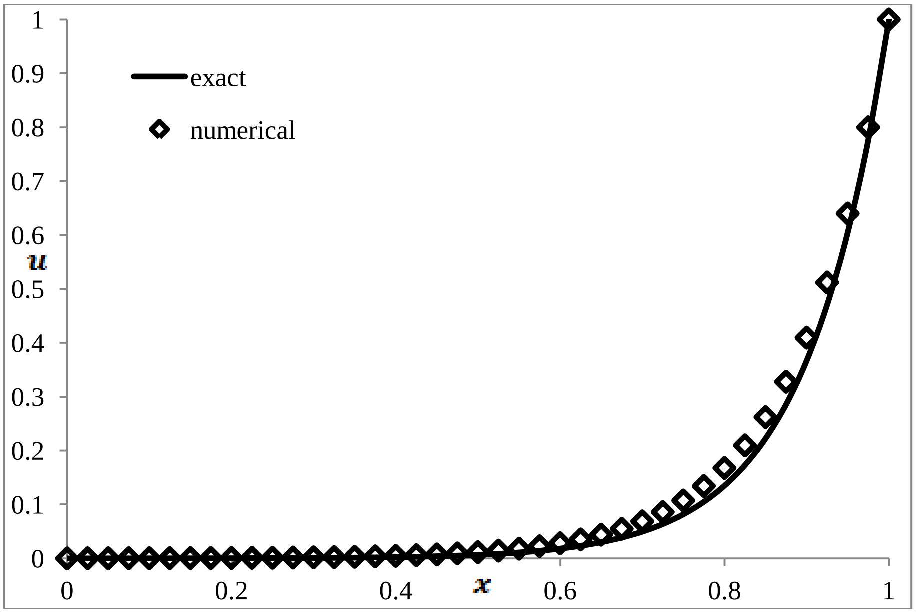


Figure 2.6 Comparison of exact and TFDM solutions of steady convection-diffusion equation (uniform mesh; N = 40; *R* = 10)

2.4.2 Multi-block Mesh

Consider the 4-block mesh on the interval [0,1] illustrated in Fig. 2.3. Boundary conditions are 0 for the west boundary and 1 for the east boundary. Block interfaces are at *x* equal to 0.1, 0.4 and 0.8. The numbers of cells in each block are 8, 5, 5 and 5 for BLK 1, BLK 2, BLK 3 and BLK 4, respectively. The convergence tolerance is set to be 10^{-6}. After running the code using both Method A and Method B to deal with the interface problem, it was determined that Method A provides a better solution for both the Laplace equation and convection-diffusion equation. Therefore, only the results from Method A are shown for the following cases.

2.4.2.1 Laplace Equation

The Laplace equation (2.23) is solved on the 4-block mesh by applying the traditional finite difference method. Figure 2.7 illustrates the results from the numerical calculation and the exact solution. The numerical results are not exactly the same as the exact solution, but the largest absolute error is only 8.75×10^{-6}, occurring at node 14.

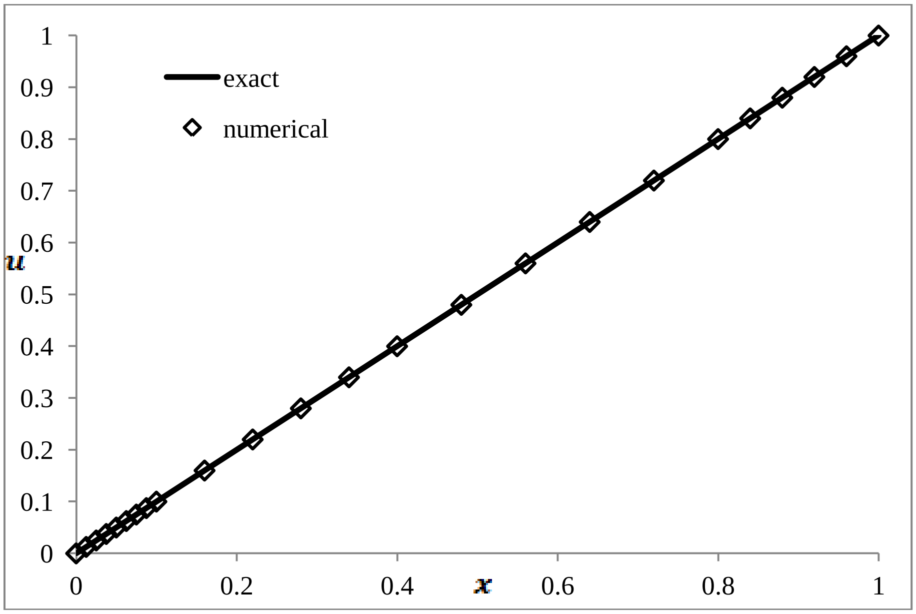


Figure 2.7 Comparison of exact and TFDM solutions of Laplace equation (4-block mesh; N = 24)

2.4.2.2 Convection-Diffusion Equation

The convection-diffusion equation (2.24) is solved on the same 4-block mesh by applying the TFDM with central differencing for the diffusion term and backward differencing for the convection term.

Figure 2.8 shows the comparison between the numerical calculation and the exact solution. The largest absolute error appears at node 21, with a value of 0.089. The solution of the convection-diffusion equation is not as accurate as the Laplace equation, but one should note that the discretization of the Laplace equation is fully second order, while the discretization of the convection-diffusion equation uses first order upwinding for the convective term.

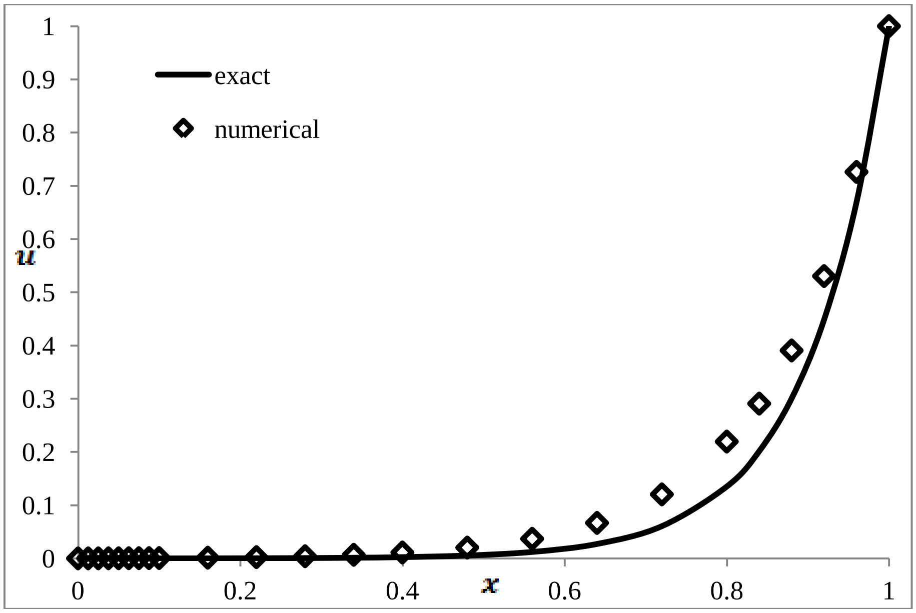


Figure 2.8 Comparison of exact and TFDM solutions of steady convection-diffusion equation (4-block mesh; N = 24; R = 10)

2.4.3 Clustered Mesh

The mesh is clustered at both ends, based on the clustering functions defined by equations (1.17) and (1.18). For clustered mesh 01, the function is *arcsinh*. For clustered mesh 02, the function is *Ln* with *B* = 1.2.

2.4.3.1 Laplace Equation

The solution of the Laplace equation (2.23) is illustrated in Fig. 2.9, showing the comparison between the numerical and exact solutions. The numerical results do not agree with the exact solution. For the clustered mesh 01, the largest absolute error is 0.349, which appears at node 31. The largest absolute error appears at node 30 for the clustered mesh 02, with a value of 0.099. The TFDM does not predict the correct solution on these clustered meshes.

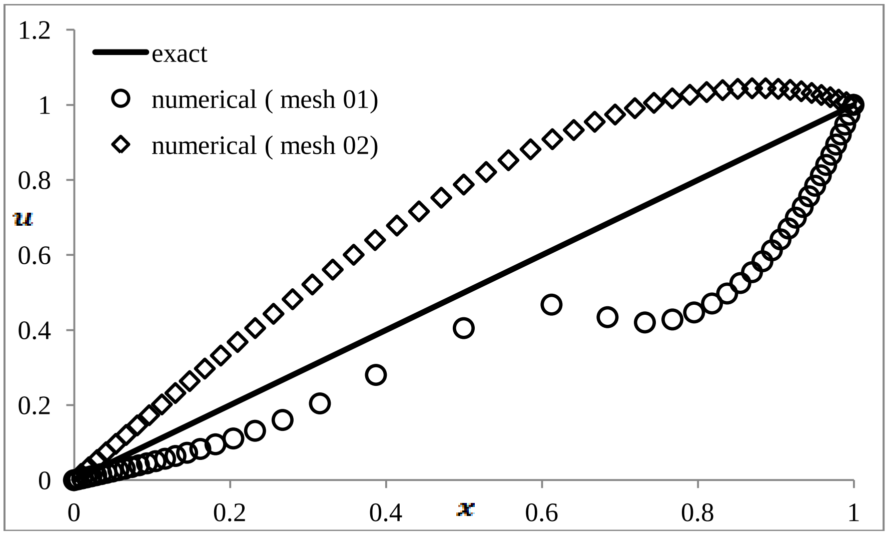


Figure 2.9 Comparison of exact and TFDM solutions of Laplace equation (clustered mesh; N = 50)

2.4.3.2 Convection-Diffusion Equation

The convection-diffusion equation (2.24) is solved on the same clustered meshes by applying the TFDM with central differencing scheme applied to the diffusion term and backward differencing for the convection term. *R* is set to be 10.

Figure 2.10 shows the comparison between the numerical calculation and exact solution. The numerical results are not accurate compared to the exact solution, especially for mesh 01. The largest absolute error is 2.08 at node 34 and 0.233 at node 43, for clustered mesh 01 and 02, respectively. Certainly, the number of nodes in the mesh will influence the solution accuracy, but refining the mesh does not always resolve the accuracy issue.

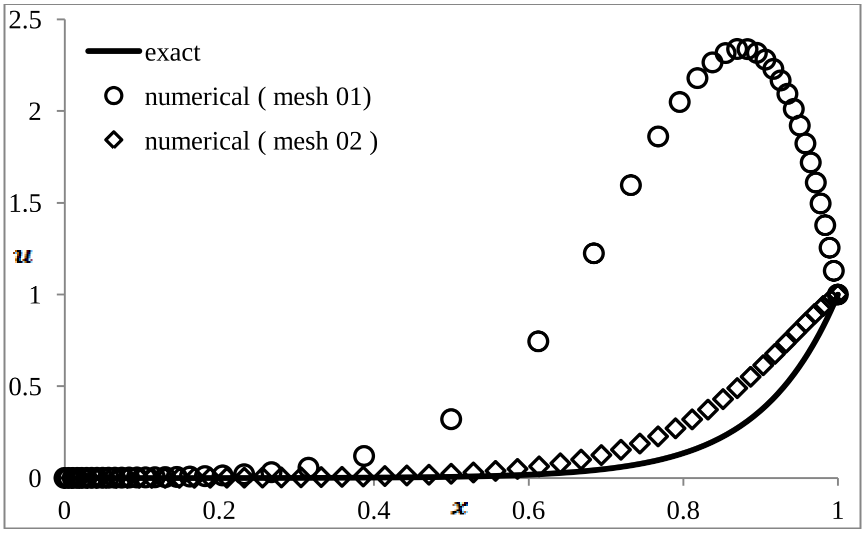


Figure 2.10 Comparison of exact and TFDM solutions of steady convection-diffusion equation (clustered mesh; N = 50; *R* = 10)

Figure 2.11 below shows the results from TFDM solutions on clustered meshes with different number of cells. These mesh files were generated based on the *arcsinh* function defined by equation (1.17). As shown in this figure, increasing the number of cells does not produce more accurate results in this particular example. In fact, the numerical results become worse as the number of grid points increases, even becoming negative for large N. This example confirms that clustered meshes can produce erroneous results if they are not properly designed, requiring considerable expertise on the part of the user.

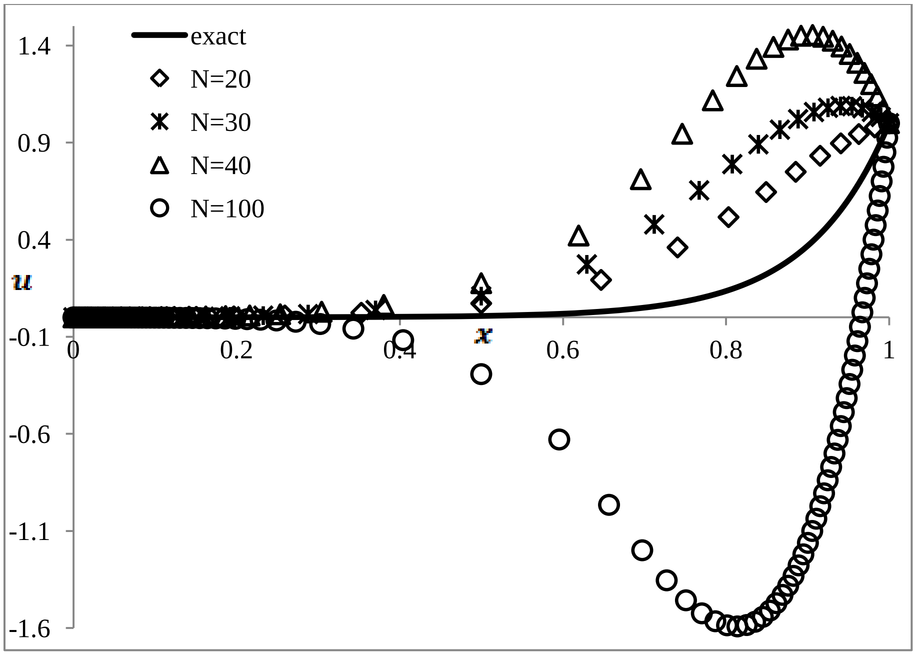


Figure 2.11 Comparison of exact and TFDM solutions of steady convection-diffusion equation on meshes with different number of cells (clustered mesh; *R* = 10)

The logarithmic function defined in equation (1.18) was used to create two clustered meshes with different values of B. Recall that $1 < B < 2$ and the mesh becomes more clustered for B closer to 1. Results from the solutions on these two meshes are compared in Fig. 2.12. It is obvious that the results are more accurate with the higher value of B, which means the mesh is more uniform. This comparison demonstrates that the degree of clustering has an effect on the solution accuracy, as has been pointed out by Kalita[11] and others.

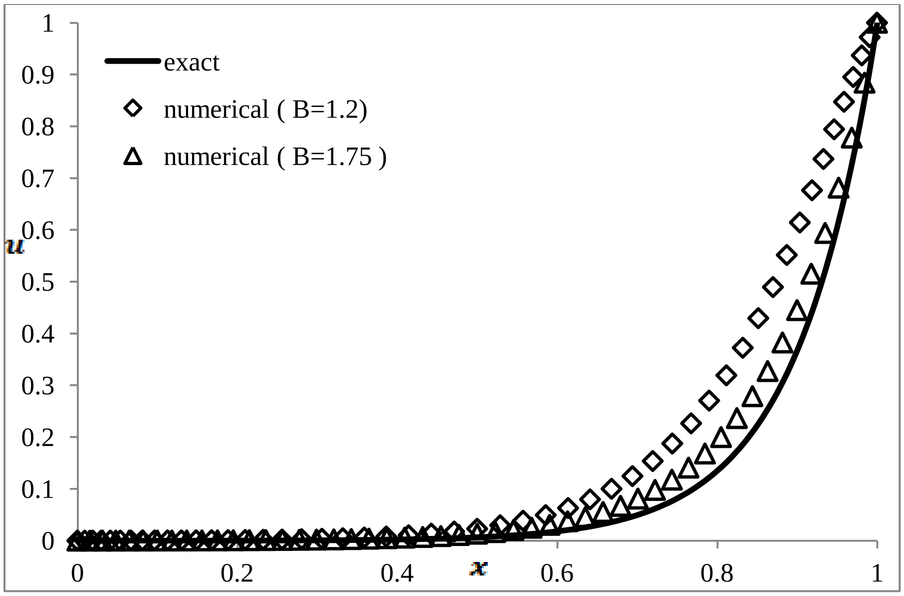


Figure 2.12 Comparison of exact and TFDM solutions of steady convection-diffusion equation (clustered mesh with *B* = 1.2 and *B* = 1.75; N = 50; *R* = 10)

Figure 2.13 illustrates the results from the solutions with different values for *R*. The clustered meshes are based on the function (1.18) with the same value for *B* = 1.75. The largest absolute error becomes higher when the value of *R* increases, with the value 0.077 at node 45, 0.09 at node 49 and 0.11 at node 50, respectively. These larger errors can be attributed to the higher gradients near the east boundary for larger *R*.

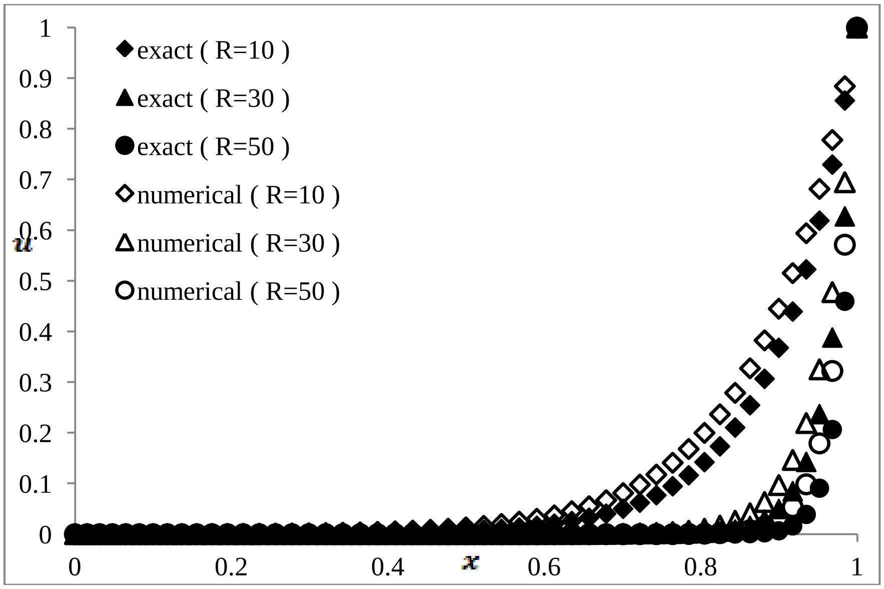


Figure 2.13 Comparison of exact and TFDM solutions of steady convection-diffusion equation (clustered mesh 02 with *B* = 1.75; N = 50; *R* = 10, 30, 50)

2.5 Summary

The traditional finite difference method can be used to accurately solve the Laplace equation on either a uniform mesh or a multi-block mesh. For steady convection-diffusion, although the results are not as accurate as the results for the Laplace equation, the error can still be kept small. Multi-block implementation is complicated due to the inter-block communication. The method to deal with the interface problem needs to be applied cautiously; otherwise, it may cause trouble in the programming and the accuracy. Applying the TFDM on a clustered mesh must also be done with care, since the results depend significantly on the functions used to generate the clustered mesh. One of the reasons for this is, in the TFDM, only one mapping function is applied to map the whole clustered mesh into a uniform mesh. The metrics are approximated based on the same mapping function, which is not always accurate for all cases, and this accuracy will affect the numerical results. Due to these reasons, a Cell-Centred Finite Difference Method, for which no mapping is needed when dealing with multi-block and clustered meshes, is investigated in the next chapter.

CHAPTER 3

CELL-CENTRED FINITE DIFFERENCE METHOD

Although the traditional finite difference method has been applied very successfully for many CFD applications, it has some weaknesses such as those illustrated in Chapter 2. The strengths of the method, in particular its simplicity in development and programming, are lost if one is interested in solving flows in very complex geometries. The main restriction that limits the applicability of the traditional finite difference method is that it cannot handle arbitrary mesh topologies. Since complicated flow domains are more easily meshed using arbitrary polygonal (2D) or polyhedral (3D) elements, the CFD community generally regards the finite difference method as non-applicable for industrial problems. For this reason, most commercial CFD codes are based on either the Finite Volume method or the Finite Element method, e.g., ANSYS Fluent[28], STAR-CCM+[29], CONVERGE[30], FLOW-3D[31] and COMSOL[32].

In an attempt to devise a finite difference scheme that can be implemented on an arbitrary mesh, Salih[18] and Situ[19] introduced the Cell-Centred Finite Difference (CCFD) method. In the CCFD method, the differential equation is approximated at the centre of each (arbitrary) cell in the domain by placing a local coordinate system at the cell centroid and aligning it with the global Cartesian coordinate system, with the local stencil arms cutting the edges of the cell, as shown in Fig. 3.1. The method derives its strength from the fact that the differencing stencil is confined to the cell. Instead of assembling a matrix system as in the traditional node-based finite difference method, a point-wise iterative approach is used in CCFD to determine the solution at all cell centres. Nodal values are then obtained by interpolation of the cell-centre values. Research by Salih[18] and Situ[19] focused mainly on implementation issues for 2D elliptic (Laplace) and parabolic (unsteady heat conduction) PDEs. Their simulations demonstrated that the CCFD method can handle triangulated domains as well as uniform and clustered Cartesian grids. However, there were some issues with accuracy for convection-diffusion equations, which may be related to the interpolation schemes used to determine the nodal values.

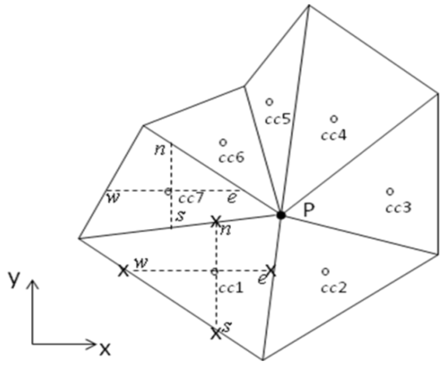


Figure 3.1 Cell-Centred stencils in an arbitrary 2D mesh

This chapter is devoted to a fundamental analysis of the CCFD method. In particular, the main source of inaccuracy due to numerical modeling errors is identified and some simple procedures to alleviate the problem are proposed. This analysis is carried out for the 1D model problems discussed in Chapter 1.

3.1 General 1D CCFD Formulation

In the Cell-Centred Finite Difference method, the differencing stencil is kept within each cell. Therefore, whether the mesh is uniform or not does not affect how the method works. The discretized equations are valid for every cell in the domain. Take an arbitrary 1D mesh and consider the general unsteady convection-diffusion equation (1.12). Figure 3.2 shows two cells in the arbitrary mesh with cell-centres and nodes denoted as c_{i-1} and c_i, and as *i*-1, *i* and *i*+1 respectively.

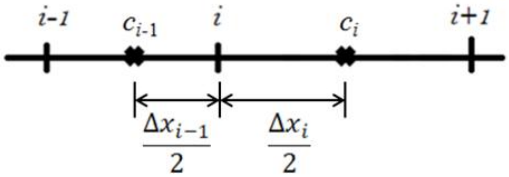


Figure 3.2 Two adjacent cells in a 1D arbitrary mesh

Applying three-point central differencing on cell-centres for the diffusion term and backward differencing on the convection term, the discretized form of equation (1.12) at any cell-centre c_i is:

$$\left.\frac{du}{dt}\right|_{c_i} - \frac{1}{D}\left[\frac{u_i - 2u_{c_i} + u_{i+1}}{\left(\frac{\Delta x_i}{2}\right)^2}\right] + a\left[\frac{u_{c_i} - u_i}{\frac{\Delta x_i}{2}}\right] = S_{c_i}\,. \tag{3.1}$$

Implementation of the CCFD method proceeds as follows:

Step 1: Make an initial guess for the solution at all internal nodes.

Step 2: Use discretisation equation (3.1) to determine the values at the cell-centres.

Step 3: Use distance-weighted averaging to update the nodal values, i.e., for each *i*,

$$u_i = \frac{\frac{u_{c_{i-1}}}{\Delta x_{i-1}/2} + \frac{u_{c_i}}{\Delta x_i/2}}{\frac{1}{\Delta x_{i-1}/2} + \frac{1}{\Delta x_i/2}} \ . \tag{3.2}$$

Then repeat the second and third steps until the differences for the nodal values between two successive iterations converge to within the specified tolerance.

Both the multi-block mesh and the clustered mesh problems can be solved based on the same procedures as for the uniform mesh. For the multi-block mesh, the interface nodes do not cause any complexity. For the clustered mesh, a mapping is no longer required to evaluate the nodal values. The procedures for CCFD calculations are much simpler, and programming the code is much more straightforward to accomplish. In brief, this CCFD method is much easier to implement compared with the traditional finite difference method.

3.2 Examples for Steady Equations

3.2.1 Uniform Mesh

For the following examples, the domain is of unit length, with west and east boundary conditions as 0 and 1, respectively. The number of cells in the domain is 50.

3.2.1.1 Laplace Equation

The Laplace equation (2.23) is solved on a uniform mesh by applying the above steps for the Cell-Centred Finite Difference method. Figure 3.3 shows the comparison between the numerical results from the code and the exact results. The numerical results perfectly match the exact solution $u(x) = x$.

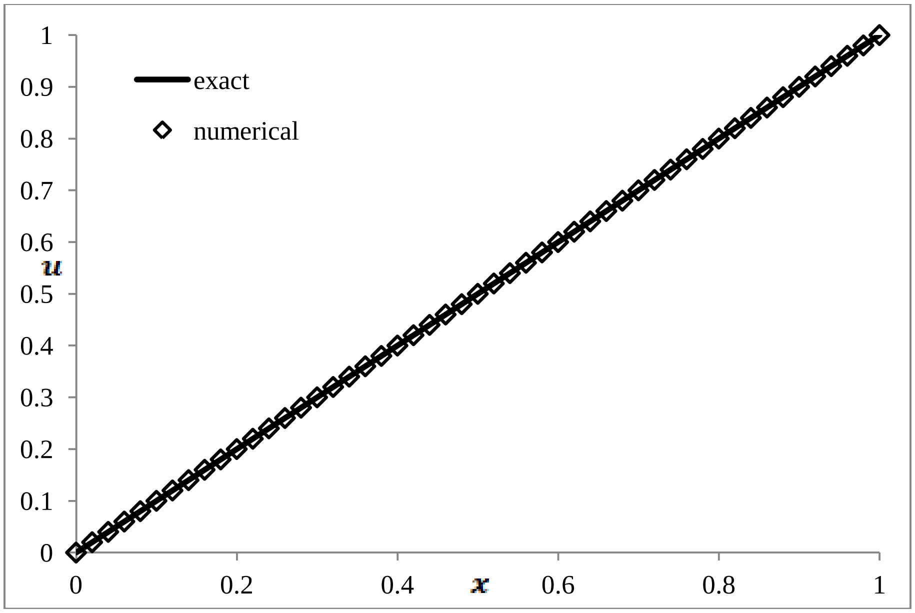


Figure 3.3 Comparison of exact and CCFD solutions of the Laplace equation (uniform mesh; N = 50; averaging method)

3.2.1.2 Steady Convection-Diffusion Equation

Consider the steady convection-diffusion equation (2.1) with R = 10 and S = 0. Second order central differencing is applied for the diffusion term and backward differencing is applied for convection term. Figure 3.4 shows the comparison between numerical results from the code and the exact solution given by equation (2.25).

As illustrated in Fig. 3.4, there are some significant differences between the numerical results and the exact results. The largest absolute error appears at node 44, with a value of 0.246. This corresponds to a large relative error of 1.0. This result is analyzed in section 3.3 below.

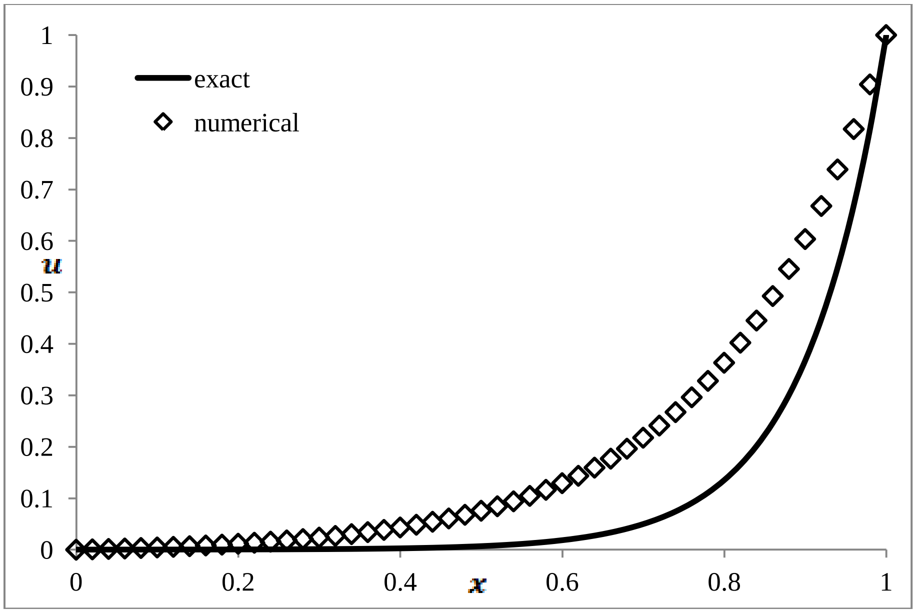


Figure 3.4 Comparison of exact and CCFD solutions of convection-diffusion equation (uniform mesh; N = 50; averaging method)

3.2.2 Clustered Mesh

The clustered meshes are generated from a separate code, using the functions defined in equations (1.17) and (1.18) to build the mesh files 01 and 02 as described in Chapter 1. The number of cells in each mesh is 50 and the domain has unit length. Boundary conditions are 0 and 1 at west and east, respectively, and both meshes are clustered towards the boundaries.

3.2.2.1 Laplace Equation

The Laplace equation (2.23) is solved on the clustered meshes by following the steps outlined above for the CCFD method. Figure 3.5 shows the comparison between the numerical results and the exact results. The numerical results perfectly match the exact solution $u(x) = x$.

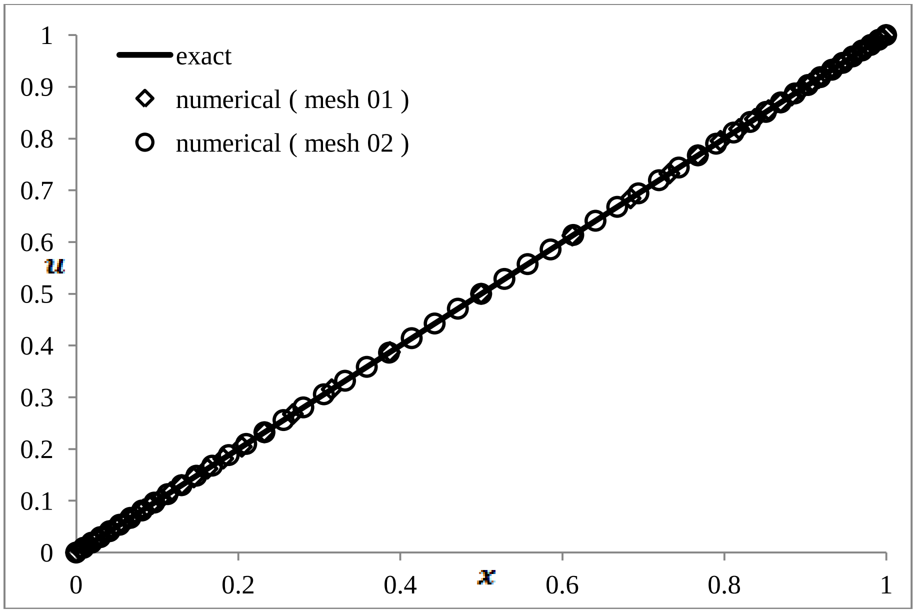


Figure 3.5 Comparison of exact and CCFD solutions of Laplace equation (clustered mesh; N = 50; averaging method)

3.2.2.2 Steady Convection-Diffusion Equation

Set R = 10 and S = 0 in the steady convection-diffusion equation (2.1) and approximate second derivative with three-point second order central differencing, and first derivative with backward differencing. Figure 3.6 shows the comparison between the numerical results from the code and the exact solution. As seen from Fig. 3.6, the numerical results are obviously higher than the exact results at many nodes in the mesh. For mesh 01, the largest absolute error occurs at node 35, and the value is 0.2464. For mesh 02, the largest absolute error occurs at node 41, and the value is 0.2462.

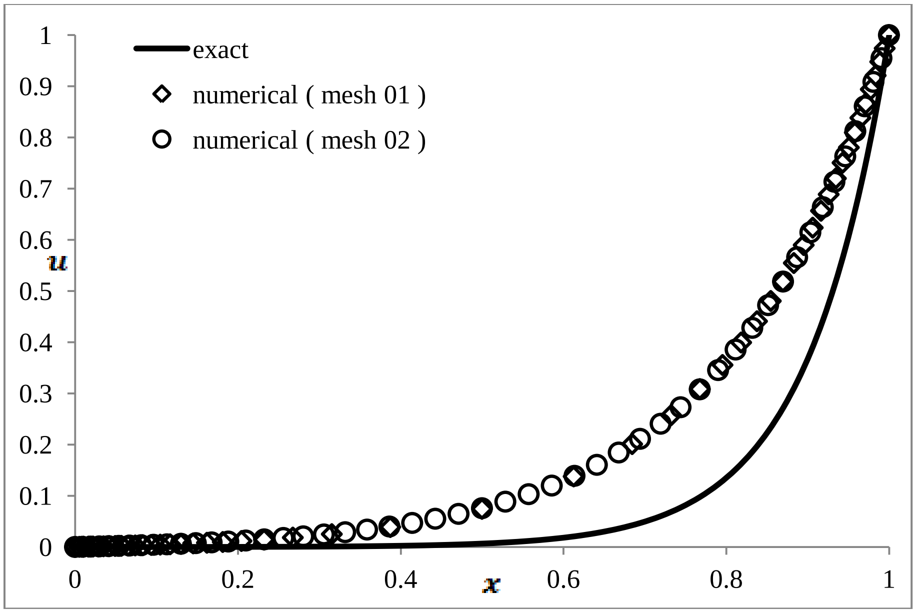


Figure 3.6 Comparison of exact and CCFD solutions of steady convection-diffusion equation (clustered mesh; N = 50; averaging method)

3.3 Analysis of CCFD Results

From the above figures, it is clear that the CCFD method can accurately solve the Laplace equation. However, if CCFD is applied to the convection-diffusion equation, the results are not accurate enough. In order to find the reasons for this, consistency needs to be checked. If the finite difference equation (FDE) reverts back to the differential equation as the grid spacing tends to zero, it shows that the numerical scheme is consistent. Therefore, if consistency is checked, it can be found whether the FDE, which is applied in the code, can reflect the ODE (or PDE in higher dimensions) which is required to be solved. For this purpose, the modified differential equation needs to be derived. Since the results for Laplace equation are correct, only the convection-diffusion equation needs to be checked.

The general procedure to derive the modified ODE (MODE) and check for consistency is as follows:

Step 1: Derive the FDE by applying finite difference formulae to approximate the terms in the ODE at node *i*.

Step 2: Expand each term in the FDE in a Taylor series about x_i.

Step 3: Collect coefficients of u_i and its derivatives.

Step 4: Re-arrange the infinite series so that the terms in the original ODE appear on one side of the equation, and all other terms appear on the other side. This is the MODE.

Step 5: Take the limit of the MODE as $\Delta x \rightarrow 0$. If

$$lim_{\Delta x\rightarrow 0}(MODE) = ODE,$$

then the numerical scheme is consistent. Otherwise it is inconsistent.

Essentially, if a numerical scheme is not consistent, the FDE does not correctly represent the ODE and the solution of the FDE (if it exists) will be the solution of some other ODE.

For the steady convection-diffusion equation (2.1), after applying second order central differencing on the diffusion term and allowing for central (β = 0), backward (β = 1) or forward (β = -1) differencing for the convection term, the discretisation equation becomes

$$-\frac{u_i-2u_{c_i}+u_{i+1}}{(\Delta x_i/2)^2}+R\left\{\frac{-(1+\beta)u_i+2\beta u_{c_i}+(1-\beta)u_{i+1}}{2(\Delta x_i/2)}\right\}=S_{c_i} \tag{3.3}$$

Re-arranging this equation gives

$$\{-4-R\Delta x_i(1+\beta)\}u_i+\{8+2\beta R\Delta x_i\}u_{c_i}+\{-4+R\Delta x_i(1-\beta)\}u_{i+1}=\Delta x_i{}^2 S_{c_i}. \tag{3.4}$$

Before introducing interpolations for the nodal values u_i and u_{i+1}, it is advisable to check equation (3.4) for consistency. Expanding u_i and u_{i+1} in Taylor series about x_{c_i}, equation (3.4) becomes (with $\Delta x \equiv \Delta x_i$)

$$\begin{aligned}&\{-4-R\Delta x(1+\beta)\}\left\{u_{c_i}-\frac{\Delta x}{2}u_{c_i}^{'}+\frac{(\Delta x)^2}{(4)2!}u_{c_i}^{''}-\frac{(\Delta x)^3}{(8)3!}u_{c_i}^{'''}+O(\Delta x)^4\right\}\\&\qquad-\{8+2\beta R\Delta x\}u_{c_i}\\&+\{-4+R\Delta x(1-\beta)\}\left\{u_{c_i}+\frac{\Delta x}{2}u_{c_i}^{'}+\frac{(\Delta x)^2}{(4)2!}u_{c_i}^{''}+\frac{(\Delta x)^3}{(8)3!}u_{c_i}^{'''}+O(\Delta x)^4\right\}=\Delta x^2 S_{c_i}\end{aligned} \tag{3.5}$$

which simplifies to

$$-u^{''}+Ru^{'}=S+\frac{\Delta x}{4}\beta Ru^{''}+O(\Delta x)^2\ . \tag{3.6}$$

Equation (3.6) is the MODE for the steady convection-diffusion equation when the Cell-Centred Finite Difference method is used to discretize the ODE. This proves that the CCFD method is consistent, since equation (3.6) tends to the ODE as Δx tends to zero. However, in the implementation of the CCFD method, all the nodal values need to be updated by applying some interpolation schemes, such as the distance-weighted average defined by equation (3.2). For consistency analysis, the mesh is chosen to have uniform spacing Δx so that the equations used to update nodal values are only averaging the cell-centre values, i.e.,

$$\begin{aligned} u_{i+1} &= \frac{u_{c_i}+u_{c_{i+1}}}{2} \\ u_i &= \frac{u_{c_{i-1}}+u_{c_i}}{2} \ . \end{aligned} \tag{3.7}$$

Substituting equations (3.7) into (3.4) gives

$$\{-4-R\Delta x(1+\beta)\}u_{c_{i-1}} + \{8+2\beta R\Delta x\}u_{c_i} + \{-4+R\Delta x(1-\beta)\}u_{c_{i+1}} = 2\Delta x^2 S_{c_i}. \tag{3.8}$$

The MODE obtained from the FDE (3.8) is

$$-u'' + \frac{R}{2}u' = \frac{1}{2}S + \frac{\Delta x}{4}\beta R u'' + O(\Delta x)^2 \ . \tag{3.9}$$

Equation (3.9) shows an essential problem, the ODE is not the one which we are attempting to solve. There is an extra ½ factor in the coefficient of the convection term and the source term *S*, which means that the CCFD method with interpolation (3.7) does not satisfy the consistency requirement. Therefore, distance-weighted average applied to update the nodal values is not an acceptable method. Note, however, that consistency is satisfied for the homogeneous Laplace equation (R = 0, S = 0) considered in the above examples. Some alternative methods for the convection-diffusion equation are tested in the following section.

3.4 Other Methods to Update Nodal Values

The consistency analysis in section 3.3 confirms that the basic CCFD formulation is consistent, but the method will lose consistency if the nodal interpolation scheme is not properly chosen. Two alternative schemes to determine the nodal values are proposed in this section.

3.4.1 Shifting Method

In this method, cell-centres and nodes shift back and forth to calculate nodal values. Figure 3.7 shows the mesh with nodes placed at the domain boundaries, and Fig.3.8 illustrates the mesh with cell-centres at the boundaries.

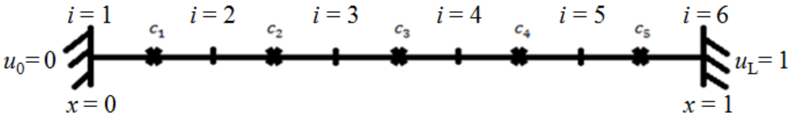


Figure 3.7 Mesh with nodes at domain boundaries

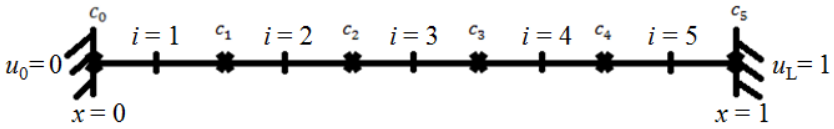


Figure 3.8 Mesh with cell-centres at domain boundaries

For the shifting method, the calculation begins by processing the data on the mesh in Fig. 3.7. First, all the interior nodal values need to be initialized. Then the discretisation equation (3.1) is applied on cell-centres to calculate the cell-centre values. In the next step, the calculations shift to the mesh shown in Fig. 3.8, where all the nodes in Fig. 3.7 become cell-centres and all the cell-centres change to nodes. Therefore, for the mesh in Fig. 3.8, all the nodal values are known. The same discretisation equation (3.1) is then applied on cell-centres to calculate the values on cell-centres in the mesh of Fig. 3.8. After this step, shift back to the mesh in Fig. 3.7, do the same calculations as the second step, and then shift back to Fig. 3.8. Continue the calculation until the differences for the nodal values in Fig. 3.7 between two iterations converge to within the specified tolerance.

To test this shifting scheme, consider the same steady convection-diffusion equation (2.1) with R = 10 and S = 0 on a uniform 40-cell mesh, on a domain of unit length with boundary conditions 0 and 1. Second order central differencing is applied on both the diffusion and convection terms. Convergence tolerance was set at 10^{-8}.

Figure 3.9 reveals the comparison of results from the shifting scheme and the exact results. The numerical results and the exact results are very close for most nodes. The maximum error occurs at node 45 with the value 0.000308.

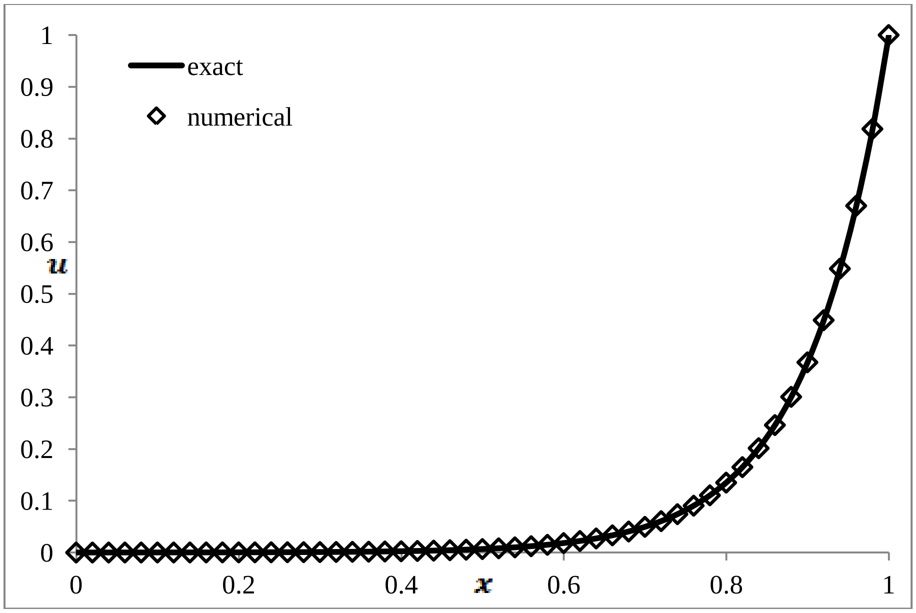


Figure 3.9 Comparison of exact and CCFD solutions of steady convection-diffusion equation (uniform mesh; N = 50; shifting method)

3.4.2 Differencing Method

The differencing method described in this subsection is similar to the shifting method, but is simpler to apply since it uses only one mesh. Also, the shifting method only applies on a uniform mesh and would likely be difficult to programme for higher dimensional problems. The differencing method does not have these restrictions.

In the differencing method, after initializing all the nodal values in the domain, the discretisation equation is applied at all the cell-centres to achieve all the values of cell-centres. Then, the same discretisation equation is applied at all the nodes to update the nodal values. The calculation continues, alternating between cell-centre and nodal values, until differences of the nodal values between two iterations converges to within the specified tolerance.

3.4.2.1 Uniform Mesh

Take the steady convection-diffusion equation (2.1) with backward differencing for the convective term (β = 1), no source term and the same boundary conditions as in previous examples. Discretisation equation (3.4), with uniform Δx, is applied on cell-centres to calculate cell-centre values. Then the equation

$$u_i = \frac{1}{8}\left[(4 + R\Delta x)u_{c_{i-1}} + (4 - R\Delta x)u_{c_i}\right] \tag{3.10}$$

is applied to update the nodal values. Figure 3.10 shows the comparison between numerical and exact results for R = 10 on a uniform mesh with 50 cells. The maximum absolute error occurs at node 45 with the value 0.000306.

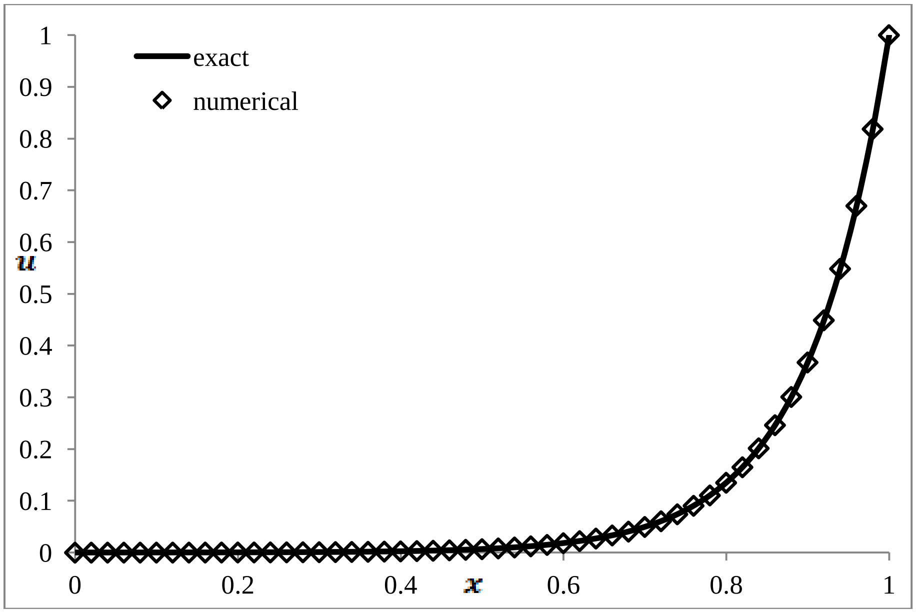


Figure 3.10 Comparison of exact and CCFD solutions of steady convection-diffusion equation (uniform mesh; N = 50; differencing method)

3.4.2.2 Clustered Mesh

Solve the same steady convection-diffusion equation (2.1) on a clustered mesh, with the same domain, boundary conditions and R as the above example. The formulation for a clustered mesh is not exactly the same as the uniform mesh. The discretisation equation (3.4) is still applied on cell-centres after initializing all the nodal values. However, since the mesh is not uniform, nodal values are updated using the following equation,

$$u_i = \frac{-a'' + Ra'}{b'' - Rb'} u_{c_{i-1}} + \frac{-c'' + Rc'}{b'' - Rb'} u_{c_i} \tag{3.11}$$

where $a'' = \frac{8}{\Delta x_{i-1}(\Delta x_{i-1} + \Delta x_i)}$, $b'' = \frac{8}{\Delta x_{i-1}\Delta x_i}$, $c'' = \frac{8}{\Delta x_i(\Delta x_{i-1} + \Delta x_i)}$,

$$a' = \frac{-2\Delta x_i}{\Delta x_{i-1}(\Delta x_{i-1} + \Delta x_i)},\ b' = \frac{2(\Delta x_i - \Delta x_{i-1})}{\Delta x_{i-1}\Delta x_i},\ c' = \frac{2\Delta x_{i-1}}{\Delta x_i(\Delta x_{i-1} + \Delta x_i)}.$$

Figure 3.11 illustrates the results comparison. For mesh 01, the largest absolute error is 0.00262 at node 30. For mesh 02, the largest absolute error is 0.000561 at node 41.

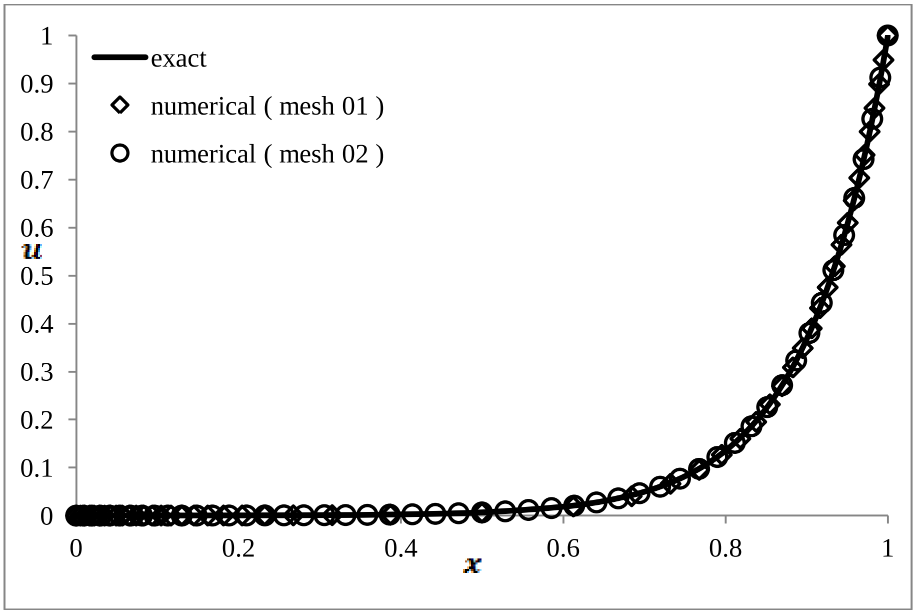


Figure 3.11 Comparison of exact and CCFD solutions of steady convection-diffusion equation (clustered mesh; N = 50; differencing method)

From Figs. 3.10 and 3.11, it is obvious that the numerical results are accurate either on a uniform mesh or clustered mesh, which means the differencing method, works well for solving the steady convection-diffusion equation. However, this method is actually another iterative version of the traditional finite difference method. But, unlike the TFDM, it can be implemented on a non-uniform mesh without mapping.

3.5 Unsteady Problems

In this section, the CCFD method is applied to unsteady equations to test if the averaging procedure can work for time-dependent problems. The unsteady convection-diffusion equation (1.17) is

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = S(x,t) \ . \tag{3.12}$$

If $a = 0$, equation (3.12) is a parabolic equation. If $\nu = 0$, the equation is hyperbolic. In the following examples, the source term *S* is taken to be zero. Averaging is applied to update nodal values.

3.5.1 Parabolic Equation

Consider the parabolic equation (a = 0) applied on a uniform mesh. The explicit Euler scheme, or forward differencing, is applied for the time derivative, and central differencing is used for the space derivative. This formulation produces the FDE

$$u_{c_i}^{n+1} = u_{c_i}^{n} + \frac{4\nu\Delta t}{(\Delta x)^2}\left(u_i^n - 2u_{c_i}^n + u_{i+1}^n\right) \tag{3.13}$$

where Δt is the time step.

3.5.1.1 Stability Analysis

Explicit time-marching schemes are known to become unstable if the time step is too large. The procedure to determine the time step restriction is based on von Neumann stability analysis as presented in Hoffmann and Chiang[1].

Define the diffusion number

$$d = \frac{\nu\Delta t}{(\Delta x)^2} \tag{3.14}$$

and assume the solution can be represented by a Fourier series, so that terms in the FDE (3.13) can be expressed as

$$u_{c_i}^{n} = U^n e^{I\theta c_i} \tag{3.15}$$

$$u_{c_i}^{n+1} = U^{n+1} e^{I\theta c_i} \tag{3.16}$$

$$u_{i}^{n} = U^n e^{I\theta\left(c_i - \frac{1}{2}\right)} \tag{3.17}$$

$$u_{i+1}^{n} = U^n e^{I\theta\left(c_i + \frac{1}{2}\right)} \tag{3.18}$$

where U is the amplitude, θ is the phase and $I = \sqrt{-1}$. The condition for numerical stability is that the amplification factor U_{n+1}/U_n satisfies the inequality $-1 \leq U_{n+1}/U_n \leq 1$.

Substituting (3.15), (3.16), (3.17) and (3.18) into (3.13) and re-arranging the equation yields the stability condition

$$d \leq \frac{1}{4}\ . \tag{3.19}$$

In other words, the CCFD method can be stable only when the condition (3.19) is satisfied. In terms of time and space increments, stability requires that $\Delta t \leq (\Delta x)^2/4\nu$.

3.5.1.2 Example

Equation (3.13) models the following fluid problem from Hoffmann and Chiang[1]. A fluid is bounded by two parallel plates extending to infinity such that no end effects are encountered. The plates and the fluid are initially at rest. Then the lower plate is suddenly accelerated in the *x*-direction. A spacial coordinate system is selected such that the lower plate includes the *xz* plane to which the *y*-axis is perpendicular. The spacing between the two plates is h = 0.04 m. The fluid is oil with a kinematic viscosity of 0.000217 m^2/s. The velocity of the lower wall is specified as U_0 = 40 m/s. Computing the velocity distribution within the plates is required. The analytical solution of this problem, subject to the imposed initial and boundary conditions, is

$$u = U_o\{\sum_{n=0}^{\infty} erfc[2n\eta_1 + \eta] - \sum_{n=0}^{\infty} erfc[2(n+1)\eta_1 - \eta]\} \tag{3.20}$$

where $\eta_1 = \frac{h}{2\sqrt{\nu t}}$, $\eta = \frac{y}{2\sqrt{\nu t}}$.

For this example, to satisfy the stability of the explicit Euler scheme, the time step should be $\Delta t \leq 0.0011503$ s.

Using the time steps Δt = 0.00115 s and 0.0009 s, which both satisfy the stability condition, the solution based on equation (3.13) was marched to $t_{\max}$ = 0.18 s. The difference between the numerical solution and analytical solution is obvious in Fig. 3.12. The reason for this discrepancy is explained below in subsection 3.5.1.3.

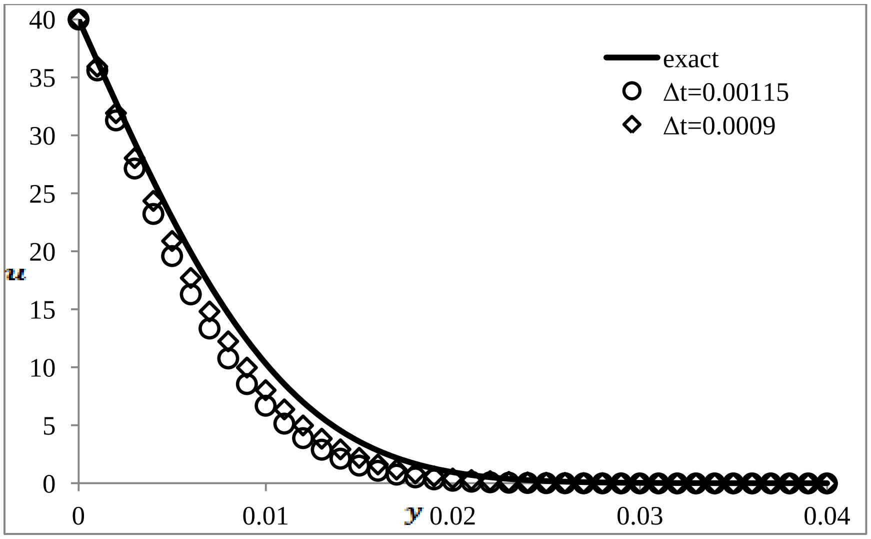


Figure 3.12 Solution of the model parabolic equation (uniform mesh; N = 40; averaging method; FTCS)

3.5.1.3 Analysis of Results

From the above figure it is clear that if CCFD is applied to the parabolic equation, the results are not accurate enough even if the stability condition is satisfied. In order to find the reasons for this, consistency should be checked. The same procedure used to determine the consistency for the steady convection-diffusion equation is followed.

The consistency is first checked before introducing interpolations for nodal values $u_i^n$and u_{i+1}^n. Expanding u_i^n and u_{i+1}^n in Taylor series about x_{c_i}, equation (3.13) becomes (with $\Delta x \equiv \Delta x_i$)

$$\begin{aligned}\left\{u_{c_i}^n + \Delta t\frac{\partial u}{\partial t} + \frac{(\Delta t)^2}{2!}\frac{\partial^2 u}{\partial t^2} + \frac{(\Delta t)^3}{3!}\frac{\partial^3 u}{\partial t^3} + O(\Delta t)^4\right\} = u_{c_i}^n \\ + \frac{4\nu\Delta t}{(\Delta x)^2}\{[u_{c_i}^n - \frac{\Delta x}{2}\frac{\partial u}{\partial x} + \frac{(\Delta x)^2}{(4)2!}\frac{\partial^2 u}{\partial x^2} - \frac{(\Delta x)^3}{(8)3!}\frac{\partial^3 u}{\partial x^3} + O(\Delta t)^4] - 2u_{c_i}^n \\ +[u_{c_i}^n + \frac{\Delta x}{2}\frac{\partial u}{\partial x} + \frac{(\Delta x)^2}{(4)2!}\frac{\partial^2 u}{\partial x^2} + \frac{(\Delta x)^3}{(8)3!}\frac{\partial^3 u}{\partial x^3} + O(\Delta t)^4]\}\end{aligned} \tag{3.21}$$

which simplifies to

$$\frac{\partial u}{\partial t} = \nu\frac{\partial^2 u}{\partial x^2} + O[(\Delta x)^2, (\Delta t)]. \tag{3.22}$$

Equation (3.22) is the modified PDE for the parabolic equation. It confirms that the scheme is consistent before interpolation. Applying the average to update the nodal valuesby substituting equations (3.7) into (3.13) yields

$$u_{c_i}^{n+1} = u_{c_i}^n + \frac{2\nu\Delta t}{(\Delta x)^2}(u_{c_{i-1}}^n - 2u_{c_i}^n + u_{c_{i+1}}^n)\ . \tag{3.23}$$

The modified PDE for equation (3.23) is

$$\frac{\partial u}{\partial t} = 2\nu\frac{\partial^2 u}{\partial x^2} + O[(\Delta x)^2, (\Delta t)]\ . \tag{3.24}$$

Equation (3.24) shows the reason why these results are not accurate, the PDE is not the one we are attempting to solve, which is similar as the problem for the steady convection-diffusion equation. This means that the CCFD method with the distance-weighted interpolation cannot satisfy the consistency requirement for the unsteady diffusion equation.

3.5.2 Hyperbolic Equation

Consider the hyperbolic equation (3.12) with $\nu = 0$ and $S = 0$ applied on a uniform mesh. The explicit Euler scheme, or forward differencing, is applied for the time derivative, and backward differencing is used for the space derivative. This formulation produces the FDE

$$u_{c_i}^{n+1} = u_{c_i}^{n} - \frac{2a\Delta t}{\Delta x}\left(u_{c_i}^{n} - u_{i-1}^{n}\right) \tag{3.25}$$

where Δt is the time step.

3.5.2.1 Stability Analysis

Explicit time-marching and backward differencing for space scheme is known to be conditionally stable. Von Neumann stability analysis is applied to determine the time step restriction using the same procedures as for the parabolic equation.

Define the Courant number

$$C = \frac{a\Delta t}{\Delta x}. \tag{3.26}$$

The analysis leads to a condition on the Courant number,

$$C \leq \frac{1}{2}\ . \tag{3.27}$$

Thus, the CCFD method will only be stable when $\Delta t \leq \frac{\Delta x}{2a}$.

3.5.2.2 Example

In this example, *a* is the speed of sound with a value of 250 m/s. A disturbance of a half sinusoidal shape is generated at time *t* = 0 [1]. Initial condition is specified as

$$\left.\begin{aligned} u(x,0) &= 0 && 0 \leq x \leq 50 \\ u(x,0) &= 100\left\{sin\left[\pi\left(\frac{x-50}{60}\right)\right]\right\} && 50 \leq x \leq 110 \\ u(x,0) &= 0 && 110 \leq x \leq 400 \end{aligned}\right\}$$

The number of cells is 80 and the domain is from 0 to 400 m.

The analytical solution of this problem, subject to the imposed initial and boundary conditions, is

$$u(x) = 100\left\{sin\left[\frac{\pi(x-250t-50)}{60}\right]\right\}. \tag{3.28}$$

To satisfy the stability condition of explicit Euler time-marching with backward differencing for space discretisation, the time step for this problem should be less than 0.01 s. Using the time steps Δt = 0.01 s and 0.005 s, which satisfy the stable condition, the solution based on equation (3.25) was marched to $t_{\max}$= 0.5 s and the results are shown in Fig. 3.13.

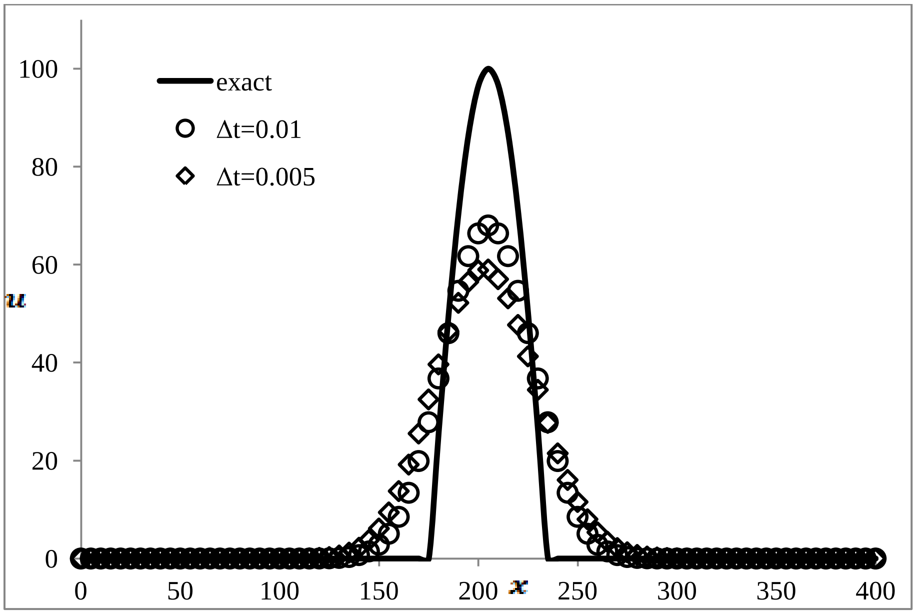


Figure 3.13 Solution of model hyperbolic equation (uniform mesh; N = 80; averaging method; FTCS)

3.5.2.3 Analysis of Results

From Fig. 3.13, it is seen that there is an obvious gap between the numerical solution and analytical solution. Since stability has been satisfied, consistency is another possible reason for this inaccuracy which needs to be checked. The same procedure is followed as the consistency check for the steady convection-diffusion equation and parabolic equation. Before applying interpolation, the CCFD method for the hyperbolic equation can be shown to satisfy the consistency condition. After introducing interpolation, the discretisation equation becomes

$$u_{c_i}^{n+1} = \left(1 - \frac{a\Delta t}{2\Delta x}\right) u_{c_i}^{n} + \frac{a\Delta t}{2\Delta x} u_{c_{i-1}}^{n} \quad . \tag{3.29}$$

The modified PDE for equation (3.29) is

$$\frac{\partial \mathrm{u}}{\partial \mathrm{t}} = \frac{\mathrm{a}}{2}\frac{\partial u}{\partial x} + O[(\Delta x),(\Delta t)] \ . \tag{3.30}$$

As seen in equation (3.30), there is an extra ½ factor. Therefore, the CCFD method, after introducing interpolation, is inconsistent. This explains the obvious error in Fig. 3.13.

3.6 Summary

For CCFD, using averaging to update nodal values is reasonable when solving the Laplace equation. However, if it is used to solve the steady convection-diffusion equation, the results are not correct because of inconsistency. Although this issue can be resolved by applying a differencing method to update nodal values, this method is actually a version of the traditional finite differencing method. The same inconsistency problem exists for solving unsteady equations, when apply averaging to update the nodal values, even if the stability requirement is satisfied. Due to these reasons, the CCFD method is not a satisfactory method for general implementation in a CFD code. However, another method can be devised which takes advantage of the main strength of the CCFD method. This proposed stencil mapping method is explained in next chapter.

CHAPTER 4

A NEW STENCIL MAPPING METHOD

The classical implementation of a finite difference formulation on a non-uniform mesh uses a coordinate mapping function to map the non-uniform mesh in the physical domain to a uniform mesh in the computational domain. The same mapping function is used to transform the differential equation to be solved from the physical coordinate system to the computational coordinate system. This transformed differential equation contains the metrics of the transformation and additional derivative terms. Although this approach has been very successful, it requires a high level of user expertise, both in grid generation techniques and computer programming. Furthermore, as shown in Chapter 2 and by other researchers, the choice of clustering function can have a significant effect on the accuracy of the numerical solution [11, 22,32].

In 1D applications, a prescribed clustering or stretching function is used to map the unequally spaced nodes with physical x-coordinates to equally spaced computational ξ-coordinates. These 1D mapping functions can also be used for Cartesian meshes in 2D and 3D. The main motivation for the development of the Cell-Centred Finite Difference scheme was to obtain a finite difference formulation that could handle an arbitrary distribution of nodes in a mesh without having to introduce coordinate transformations. Despite its success for unstructured meshes, simple interpolation schemes may cause the numerical scheme to lose consistency. In order to overcome this problem, more complicated interpolations are needed, making the method less attractive for higher dimensional problems.

The concept of a stencil lies at the core of all finite difference formulations. In this chapter, one of the key elements of the cell-centred scheme, namely that the value at a node should be calculated only by using the neighbouring values on the 3-point stencil (in 1D), is exploited to derive a generalized finite difference method for arbitrary nodal distribution in the mesh.

4.1 The Stencil Mapping

The proposed Stencil Mapping method can be applied on an arbitrary mesh. Therefore, to explain the concept of this method, an arbitrary mesh is selected. Figure 4.1 shows a 1D mesh, in which N+1 nodes are distributed randomly along the *x*-axis.

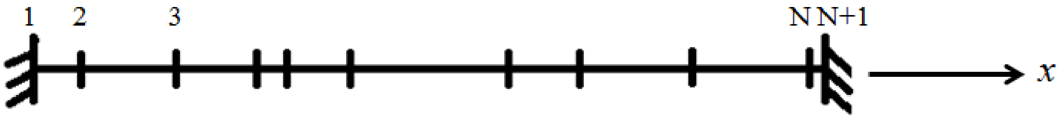


Figure 4.1 Arbitrary mesh along *x*-axis

In the stencil mapping method, for each individual set of three consecutive nodes (i.e., the stencil), a unique map is constructed that transforms the 3-point non-uniform stencil comprised of these three nodes to a 3-point uniform stencil [18,19]. Figure 4.2 shows how a representative non-uniform stencil (a) in the physical *x*-coordinates transforms to a uniform stencil (b) in the computational ξ-coordinates.

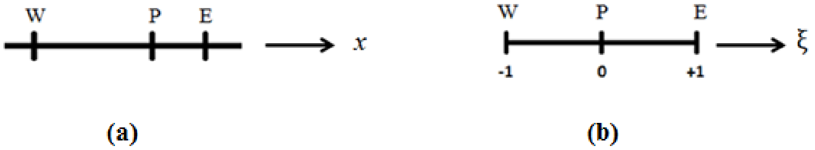


Figure 4.2 Stencil map

The transformation is based on a mapping function $x = x(\xi)$. This mapping function is a quadratic function which satisfies the three conditions $x(-1) = x_W$, $x(0) = x_P$ and $x(1) = x_E$, i.e.,

$$x = a_2\xi^2 + a_1\xi + a_0 \tag{4.1}$$

where $a_0 = x_P$, $a_1 = (x_E - x_W)/2$, and $a_2 = (x_E - 2x_P + x_W)/2$. Each stencil has its own unique quadratic mapping function and metrics.

4.2 Transformed Equation and Boundary Conditions

Under the mapping described above, the PDE (1.12) transforms to

$$\left.\frac{\partial u}{\partial t}\right|_P - \frac{1}{D}\left[\frac{1}{x_P'^{\,2}}\left.\frac{\partial^2 u}{\partial \xi^2}\right|_P - \frac{x_P''}{x_P'^{\,3}}\left.\frac{\partial u}{\partial \xi}\right|_P\right] + \frac{a}{x_P'}\left.\frac{\partial u}{\partial \xi}\right|_P = S_P \tag{4.2}$$

where x' and x'' are the derivatives of x with respect to ξ obtained from equation (4.1) and all quantities are evaluated at node P. It is interesting to note that $x'' = x_W - 2x_P + x_E$ is a measure of the non-uniformity of the mesh, and $x''= 0$ if the mesh is uniform.

Two types of boundary conditions are considered, Dirichlet and Neumann.

(a) Dirichlet Boundary Condition

For Dirichlet boundary conditions the boundary values are known. These values can be used in the calculation directly.

(b) Neumann Boundary Condition

The Neumann boundary condition occurs when the normal derivative of the solution is prescribed on the boundary of the domain. For example, consider a Neumann condition on the west boundary shown in Fig. 4.3.

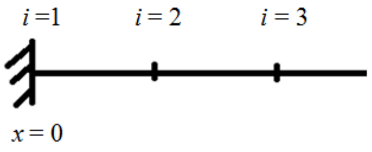


Figure 4.3 West boundary for a 1D mesh

Applying a Neumann boundary condition on the west side of the domain means that $\frac{\partial u}{\partial n} = g(t)$ at the west boundary, where $g(t)$ is a known function and $\frac{\partial u}{\partial n} = \vec{\nabla} u \cdot \hat{n}$ is the derivative in the direction of the outward unit normal vector $\hat{n}$. Since the normal vector at the west boundary points to the outside direction of the domain,

$$\left.\frac{\partial u}{\partial n}\right|_1 = g(t) \Rightarrow -\left.\frac{\partial u}{\partial x}\right|_1 = g(t) \Rightarrow -\frac{1}{x'}\left.\frac{\partial u}{\partial \xi}\right|_1 = g(t) \Rightarrow \left.\frac{\partial u}{\partial \xi}\right|_1 = -g(t)x' \tag{4.3}$$

A similar condition can be derived for a Neumann condition on the east boundary.

4.3 Discretisation and Implementation Issues

4.3.1 FDE – Second Order Scheme

The 1D unsteady convection-diffusion equation (4.2) is used to explain the algorithm. Using the 3-point central differencing formula for the diffusion term and 2-point backward differencing formula for the convection term, we can obtain the discretisation equation

$$\left.\frac{du}{dt}\right|_P + a_P u_P + a_W u_W + a_E u_E = S_P \tag{4.4}$$

where $a_P = \frac{2}{Dx_P'^{\,2}} + \frac{a}{x_P'}$,$a_W = -\frac{1}{Dx_P'^{\,2}} - \frac{x_P''}{2Dx_P'^{\,3}} - \frac{a}{x_P'}$, and $a_E = -\frac{1}{Dx_P'^{\,2}} + \frac{x_P''}{2Dx_P'^{\,3}}$.

Boundary conditions are applied at node 1 and node N+1, as illustrated in Fig. 4.1. For the unsteady case, the solution is marched in time from the prescribed initial condition. For the steady case, after initializing all the interior nodal values, all the interior nodal values can be updated through equation (4.4) with the time derivative term equal to zero. The iteration continues until the differences between two iterations at every node converge to within the specified tolerance.

A second order one-sided differencing formula is applied to equation (4.3) to approximate the boundary value u_1 if the west boundary is Neumann. Therefore, for a west Neumann boundary condition, $u_1 = \left(-\frac{1}{3}\right)\left(-2g(t)x'\big|_1 - 4u_2 + u_3\right)$. Similarly, for an east Neumann boundary condition, $u_{N+1} = \left(\frac{1}{3}\right)\left(2g(t)x'\big|_{N+1} - 4u_N + u_{N-1}\right)$.

4.3.1.1 Local Truncation Error

The local truncation error (L.T.E.) is an effective parameter used to evaluate the accuracy of the differencing method. In the stencil mapping method, the local truncation error is introduced to assess the accuracy of the scheme and to identify the nodes in the mesh which experience the largest errors. If the locations of the largest L.T.E. are known, the mesh can be refined in this region to further reduce the errors. This is referred to as adaptive meshing. To determine the local truncation error, the modified differential equation needs to be derived. Substituting Taylor series expressions back into the finite difference equation, the final expression after some algebraic manipulation is the modified equation [1]. In the modified differential equation, the dominant error can be clarified, namely the local truncation error.

To illustrate the derivation of the local truncation error consider the steady convection-diffusion equation in the form (2.1)

$$-\frac{d^2u}{dx^2}+R\frac{du}{dx}=S\ . \tag{4.5}$$

The discretisation equation, after transforming to the ξ-coordinate, becomes

$$a_Pu_P+a_Wu_W+a_Eu_E=S_P \tag{4.6}$$

where $a_P=\frac{2}{x_P^{'\,2}}+\frac{R}{x_P^{'}}$, $a_W=-\frac{1}{x_P^{'\,2}}-\frac{x_P^{''}}{2x_P^{'\,3}}-\frac{R}{x_P^{'}}$, $a_E=-\frac{1}{x_P^{'\,2}}+\frac{x_P^{''}}{2x_P^{'\,3}}$.

To get the modified ODE and the local truncation error, the Taylor series expansion at node P is substituted into equation (4.6):

$$\begin{aligned}
&a_Pu_P+a_Wu_W+a_Eu_E\\
&\qquad=a_Pu_P\\
&\qquad+a_W\left\{u_P-\Delta\xi\frac{du_P}{d\xi}+\frac{(\Delta\xi)^2}{2!}\frac{d^2u_P}{d\xi^2}-\frac{(\Delta\xi)^3}{3!}\frac{d^3u_P}{d\xi^3}+\frac{(\Delta\xi)^4}{4!}\frac{d^4u_P}{d\xi^4}+\cdots\right\}\\
&\qquad+a_E\left\{u_P+\Delta\xi\frac{du_P}{d\xi}+\frac{(\Delta\xi)^2}{2!}\frac{d^2u_P}{d\xi^2}+\frac{(\Delta\xi)^3}{3!}\frac{d^3u_P}{d\xi^3}+\frac{(\Delta\xi)^4}{4!}\frac{d^4u_P}{d\xi^4}+\cdots\right\}\\
&\qquad=\ S_P
\end{aligned}$$

Re-arranging this equation, we get the modified differential equation

$$\begin{aligned}
&-\frac{1}{x_P^{'\,2}}\frac{d^2u_P}{d\xi^2}+\frac{x_P^{''}}{x_P^{'\,3}}\frac{du_P}{d\xi}+\frac{R}{x_P^{'}}\frac{du_P}{d\xi}\\
&\qquad=S_P+\frac{R}{2x_P^{'}}\frac{d^2u_P}{d\xi^2}-\frac{x_P^{''}}{6x_P^{'\,3}}\frac{d^3u_P}{d\xi^3}-\frac{R}{6x_P^{'}}\frac{d^3u_P}{d\xi^3}+\frac{1}{12x_P^{'\,2}}\frac{d^4u_P}{d\xi^4}+\frac{R}{24x_P^{'}}\frac{d^4u_P}{d\xi^4}.
\end{aligned}$$

Noting the terms that appear in the original ODE, the remaining terms are the dominant terms in the local truncation error, i.e.,

$$L.T.E. = \frac{R}{2x'_P}\frac{d^2u_P}{d\xi^2} - \frac{x''_P}{6{x'_P}^3}\frac{d^3u_P}{d\xi^3} - \frac{R}{6x'_P}\frac{d^3u_P}{d\xi^3} + \frac{1}{12{x'_P}^2}\frac{d^4u_P}{d\xi^4}. \qquad (4.7)$$

The first and third terms in the L.T.E. are due to the first order upwind approximation of the convective term and the first term is usually referred to as artificial diffusion. The second and fourth terms in (4.7) are the result of the second order differencing of the diffusion term in equation (4.5), and the second term includes the effect of non-uniform grid spacing.

4.3.1.2 Calculation of Local Truncation Error

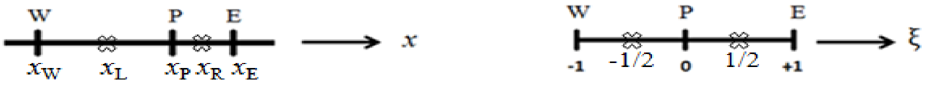
As shown in equation (4.7), there are 2nd, 3rd and 4th order derivatives in the L.T.E. Instead of using more than three nodes to approximate the higher derivatives, we want to retain the 3-node stencil in the local truncation error calculation to avoid any problem when dealing with the nodes near boundaries. The procedures are as follows:

After completing the calculation for all the nodal values in the domain, use these nodal values to approximate the 2nd derivative at every node including those at the boundary. For the interior nodes, apply 3-point central differencing and, for the boundary nodes, apply 3-point forward and backward differencing for the west and east boundary nodes, respectively. Then apply central differencing formulae on the values of the 2nd derivative to approximate the values of the 3rd and 4th derivatives at every interior node in the domain. Following this procedure, the local truncation error at every interior node can be calculated.

4.3.2 FDE – Fourth Order Scheme

Many CFD researchers are interested in extending their codes to higher-order differencing schemes [22,23,24,25,26]. Besides the obvious benefit of producing a more accurate numerical solution on a given mesh, higher-order schemes are of particular interest in Direct Numerical Simulations where a very fine mesh is required to resolve the small scales of turbulence. The main difficulty with higher-order schemes arises from the fact that higher-order difference formulae require more nodes, e.g., the 4th–order approximation of the 2nd derivative uses a 5-point stencil.

It is easy to extend the proposed stencil mapping method to higher-order schemes, while confining the stencil to three nodes. The same 3-node stencil shown in Fig. 4.2 is applied, with the same quadratic mapping function defined in equation (4.1). If a 4th-order differencing scheme is used for the diffusion term, five points along the 3-point stencil are needed to implement the scheme, positioned at x_W, x_L, x_P, x_R and x_E, as shown in Fig. 4.4a.



(a) (b)

Figure 4.4 Stencil with five discretisation points for the 4th-order scheme

After mapping, if the physical stencil is not uniform, the midpoints on the arms of the computational stencil may not be the midpoints on the physical stencil. Generally, the location of x_L and x_R are

$$x_L = \frac{3}{4}x_P + \frac{3}{8}x_W - \frac{1}{8}x_E \tag{4.8}$$

$$x_R = \frac{3}{4}x_P - \frac{1}{8}x_W + \frac{3}{8}x_E. \tag{4.9}$$

In this section, a 4th-order central differencing formula is applied for the diffusion term and a 2nd-order backward differencing formula is applied for the convection term in the steady convection-diffusion equation (4.5). The discretisation equation can be written as

$$a_P u_P + a_W u_W + a_E u_E = S_P - a_L u_L - a_R u_R \tag{4.10}$$

where $a_P = \frac{10}{x_P^{'\,2}} + \frac{3R}{x_P^{'}}$, $a_W = \frac{1}{3x_P^{'\,2}} + \frac{x_P^{''}}{6x_P^{'\,3}} + \frac{R}{x_P^{'}}$, $a_E = \frac{1}{3x_P^{'\,2}} - \frac{x_P^{''}}{6x_P^{'\,3}}$,

$$a_L = -\frac{16}{3x_P^{'\,2}} - \frac{4x_P^{''}}{3x_P^{'\,3}} - \frac{4R}{x_P^{'}}, \quad a_R = -\frac{16}{3x_P^{'\,2}} + \frac{4x_P^{''}}{3x_P^{'\,3}}.$$

The 4th-order scheme still uses the same 3-point stencil as the 2nd-order scheme. Thus, when calculating the values for nodes adjacent to the boundaries, no special formulae are required.

The next step is to use appropriate interpolations to estimate the values of u_L and u_R. Apply the 2nd-order central differencing formula for the diffusion term and 1st-order backward differencing formula for the convection term at the left (ξ = -1/2) and right (ξ = 1/2) intermediate nodes. Then re-arrange the equation to get the values of u_L and u_R. Taking the steady convection-diffusion equation (4.5), the equations for u_L and u_R are

$$u_L = \frac{4x_L' + x_L'' + 2Rx_L'^{\,2}}{2\left(4x_L' + Rx_L'^{\,2}\right)} u_W + \frac{4x_L' - x_L''}{2\left(4x_L' + Rx_L'^{\,2}\right)} u_P + \frac{x_L'^{\,3}}{2\left(4x_L' + Rx_L'^{\,2}\right)} S_L \tag{4.11}$$

$$u_R = \frac{4x_R' + x_R'' + 2Rx_R'^{\,2}}{2\left(4x_R' + Rx_R'^{\,2}\right)} u_P + \frac{4x_R' - x_R''}{2\left(4x_R' + Rx_R'^{\,2}\right)} u_E + \frac{x_R'^{\,3}}{2\left(4x_R' + Rx_R'^{\,2}\right)} S_R \ . \tag{4.12}$$

4.3.2.1 Local Truncation Error

Using the steady equation to calculate the local truncation error for the 4th-order scheme, the discretisation equation (4.10) is written as

$$a_P u_P + a_W u_W + a_E u_E + a_L u_L + a_R u_R = S_P \tag{4.13}$$

where the coefficients are defined above.

Then u_W, u_E, u_L, and u_R can be expanded in a Taylor series about node P. The left-hand side of equation (4.13) becomes:

$$a_P u_P + a_W u_W + a_E u_E + a_L u_L + a_R u_R =$$

$$a_P u_P + a_W \left\{ u_P - \Delta\xi \frac{du_P}{d\xi} + \frac{(\Delta\xi)^2}{2!}\frac{d^2u_P}{d\xi^2} - \frac{(\Delta\xi)^3}{3!}\frac{d^3u_P}{d\xi^3} + \frac{(\Delta\xi)^4}{4!}\frac{d^4u_P}{d\xi^4} - \frac{(\Delta\xi)^5}{5!}\frac{d^5u_P}{d\xi^5} \right.$$
$$\left. + \frac{(\Delta\xi)^6}{6!}\frac{d^6u_P}{d\xi^6} + \cdots \right\}$$

$$
\begin{aligned}
&+a_E\left\{u_P+\Delta\xi\frac{du_P}{d\xi}+\frac{(\Delta\xi)^2}{2!}\frac{d^2u_P}{d\xi^2}+\frac{(\Delta\xi)^3}{3!}\frac{d^3u_P}{d\xi^3}+\frac{(\Delta\xi)^4}{4!}\frac{d^4u_P}{d\xi^4}+\frac{(\Delta\xi)^5}{5!}\frac{d^5u_P}{d\xi^5}\right.\\
&\qquad\qquad\left.+\frac{(\Delta\xi)^6}{6!}\frac{d^6u_P}{d\xi^6}+\cdots\right\}\\
&+a_L\left\{u_P-\frac{\Delta\xi}{2}\frac{du_P}{d\xi}+\frac{(\Delta\xi)^2}{(4)2!}\frac{d^2u_P}{d\xi^2}-\frac{(\Delta\xi)^3}{(8)3!}\frac{d^3u_P}{d\xi^3}+\frac{(\Delta\xi)^4}{(16)4!}\frac{d^4u_P}{d\xi^4}-\frac{(\Delta\xi)^5}{(32)5!}\frac{d^5u_P}{d\xi^5}\right.\\
&\qquad\qquad\left.+\frac{(\Delta\xi)^6}{(64)6!}\frac{d^6u_P}{d\xi^6}+\cdots\right\}\\
&+a_R\left\{u_P+\frac{\Delta\xi}{2}\frac{du_P}{d\xi}+\frac{(\Delta\xi)^2}{(4)2!}\frac{d^2u_P}{d\xi^2}+\frac{(\Delta\xi)^3}{(8)3!}\frac{d^3u_P}{d\xi^3}+\frac{(\Delta\xi)^4}{(16)4!}\frac{d^4u_P}{d\xi^4}+\frac{(\Delta\xi)^5}{(32)5!}\frac{d^5u_P}{d\xi^5}\right.\\
&\qquad\qquad\left.+\frac{(\Delta\xi)^6}{(64)6!}\frac{d^6u_P}{d\xi^6}+\cdots\right\}
\end{aligned}
\tag{4.14}
$$

Re-arranging equation (4.14), setting $\Delta\xi = 1$ and simplifying coefficients, we get

$$
\begin{aligned}
-\frac{1}{{x_P'}^2}\frac{d^2u_P}{d\xi^2}+\frac{x_P''}{{x_P'}^3}\frac{du_P}{d\xi}+\frac{R}{x_P'}\frac{du_P}{d\xi}+[\frac{1}{1440{x_P'}^2}\frac{d^6u_P}{d\xi^6}+\frac{R}{768x_P'}\frac{d^6u_P}{d\xi^6}-\frac{x_P''}{480{x_P'}^3}\frac{d^5u_P}{d\xi^5}\\
-\frac{R}{12x_P'}\frac{d^3u_P}{d\xi^3}+\frac{R}{32x_P'}\frac{d^4u_P}{d\xi^4}-\frac{7R}{960x_P'}\frac{d^5u_P}{d\xi^5}+\cdots]
\end{aligned}
$$

Therefore, the local truncation error for the 4th-order accurate scheme is

$$
L.T.E.=-\frac{1}{1440{x_P'}^2}\frac{d^6u_P}{d\xi^6}-\frac{R}{768x_P'}\frac{d^6u_P}{d\xi^6}+\frac{x_P''}{480{x_P'}^3}\frac{d^5u_P}{d\xi^5}+\frac{7R}{960x_P'}\frac{d^5u_P}{d\xi^5}-\frac{R}{32x_P'}\frac{d^4u_P}{d\xi^4}+\frac{R}{12x_P'}\frac{d^3u_P}{d\xi^3}
\tag{4.15}
$$

4.3.2.2Calculation of the Local Truncation Error

Follow the same idea as the implementation of the local truncation error for the 2nd-order differencing scheme. Using the procedure in section 4.3.1.2, the values for 3rd and 4th derivatives can be evaluated. For the local truncation error of the 4th-order scheme, there are also 5th and 6th derivative terms in equation (4.15). Values of 4th-order derivatives can be used to approximate the 5th and 6th derivatives in a manner similar to the above. Then the local truncation error at all the interior nodes can be calculated.

4.4 Results

4.4.1 Poisson Equation

Consider the solution of the Poisson equation

$$\frac{d^2u}{dx^2} = 12x^2 + 6x \tag{4.16}$$

on either a uniform mesh or a clustered mesh with 50 cells.

There are two clustered mesh files tested for this stencil mapping method. The first clustered mesh is generated by the function *arcsinh* defined by equation (1.17) and the second clustered mesh is generated through the *log* function given by equation (1.18). As in previous chapters, these two meshes are denoted as mesh 01 and mesh 02, respectively. Both meshes are clustered at the two ends of the domain, with 50 cells in the mesh.

Both Dirichlet and Neumann boundary conditions are applied on this Poisson equation. For Dirichlet boundaries, the boundary conditions are 0 and 1 for west and east, respectively. If the west boundary is set to be a Neumann boundary, the value for g is 1. If a Neumann condition is applied at the east boundary, the value for g is 6. These values are obtained from the exact solution of (4.16). The Neumann boundary condition is applied at only one side in the following cases, and the other side is a Dirichlet boundary condition. The domain is from 0 to 1. Results from the 2nd-order accurate scheme and 4th-order scheme are compared for each mesh, along with the results from the exact solution $u(x) = x^4 + x^3 - x$.

4.4.1.1 Uniform Mesh

(a) Dirichlet Boundary Conditions

Figure 4.5 shows the comparison of the numerical solution with the exact solution on a uniform mesh. Local truncation errors for the 2nd-order and 4th-order schemes on the uniform mesh are illustrated in Fig. 4.6. Both schemes provide an accurate solution of the boundary value problem (BVP), but Fig. 4.6 shows that the 4th-order scheme has local truncation error which is four orders of magnitude smaller than the 2nd-order scheme.

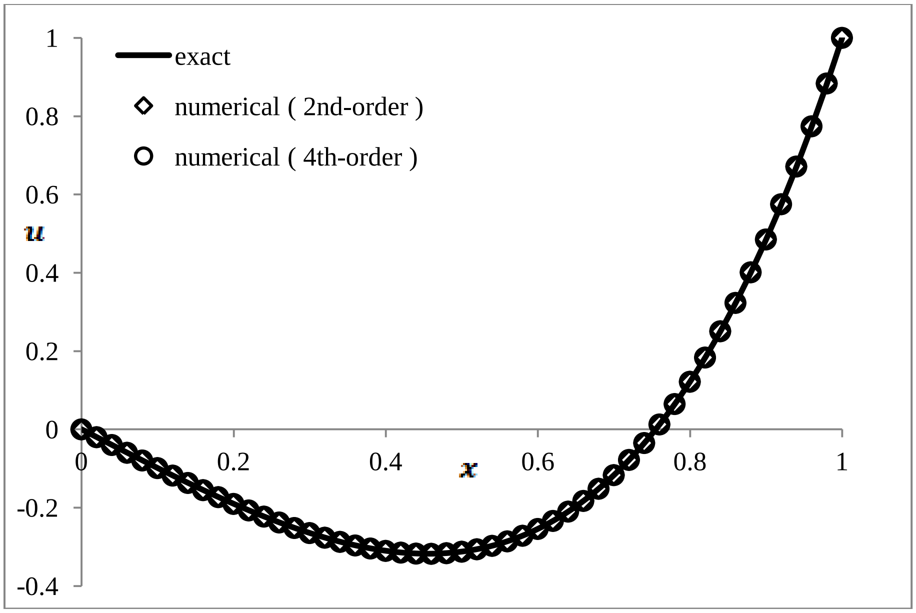


Figure 4.5 Numerical and exact results for Poisson equation (uniform mesh; N = 50; Dirichlet boundaries)

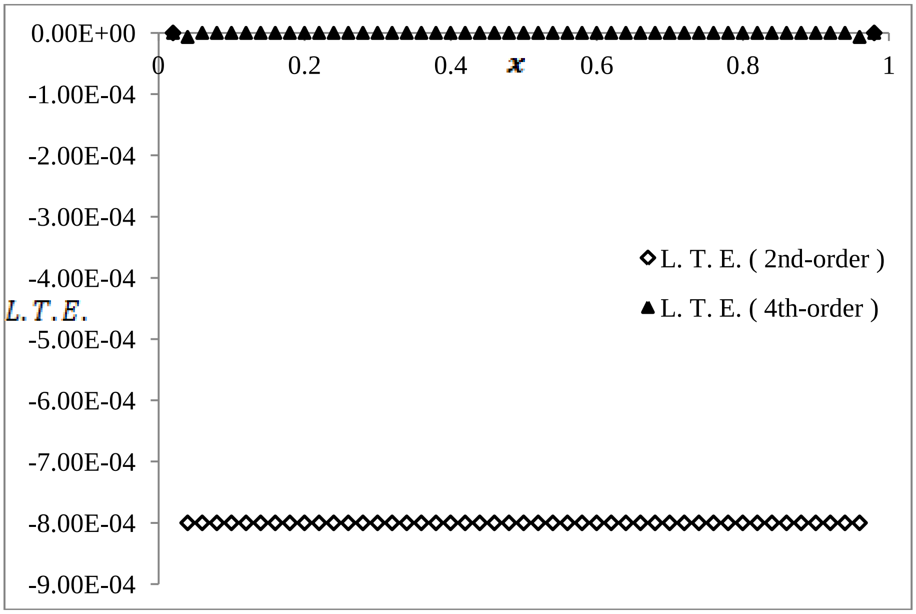


Figure 4.6 Local truncation error for Poisson equation (uniform mesh; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme)

(b) West Neumann Boundary Condition

Figure 4.7 shows the results for the numerical solution and exact solution when the Neumann condition is applied at the west boundary. Figure 4.8 provides the details of local truncation error for the 2nd-order and 4th-order schemes. Comparing these figures with Figs. 4.5 and 4.6, it is clear that the stencil mapping method can solve the Neumann BVP with the same accuracy as the Dirichlet problem.

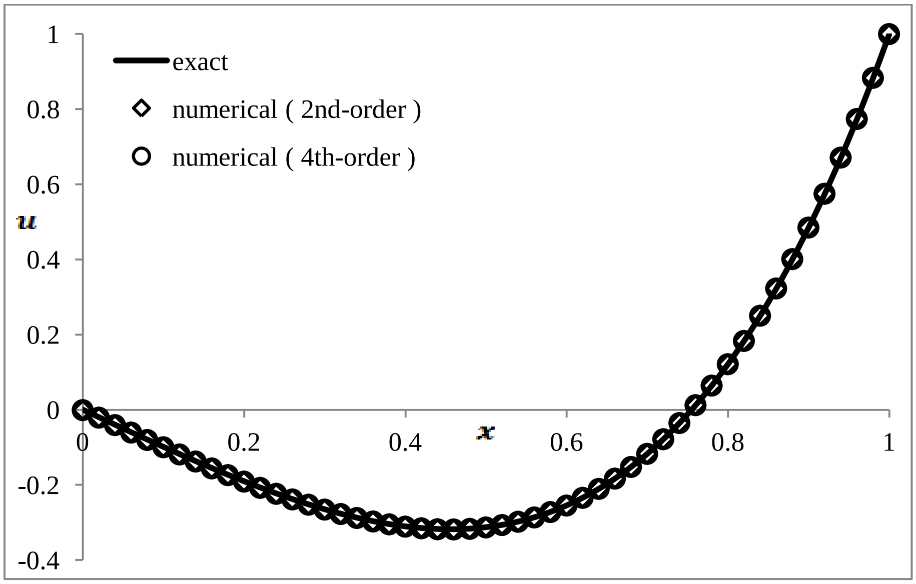


Figure 4.7 Numerical and exact results for Poisson equation (uniform mesh; N = 50; west Neumann boundary)

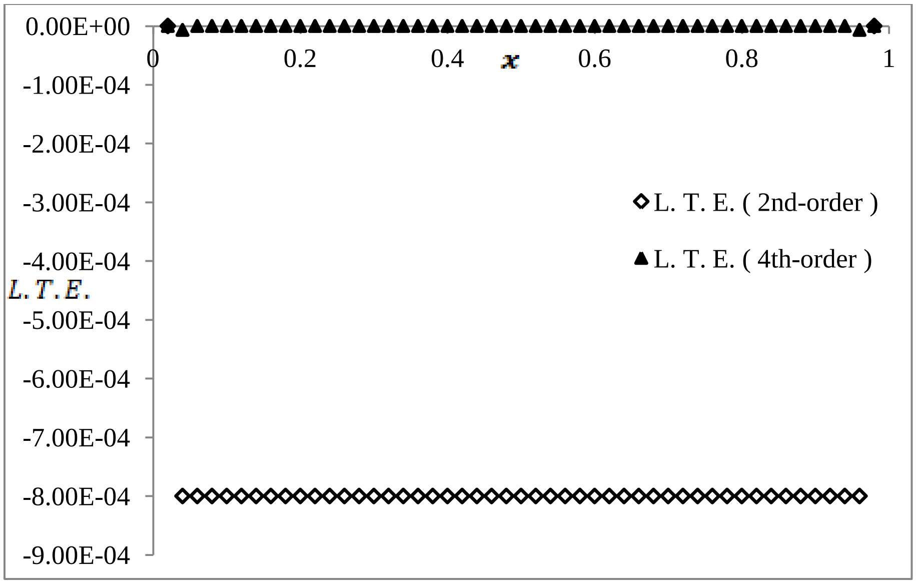


Figure 4.8 Local truncation error for Poisson equation (uniform mesh; N = 50; west Neumann boundary; 2nd-order vs. 4th-order scheme)

(c) East Neumann Boundary Condition

Figure 4.9 shows the results from the numerical solution and exact solution. Local truncation errors for the 2nd-order and 4th-order schemes are compared in Fig. 4.10. There is no loss in solution accuracy when Neumann boundary conditions are imposed.

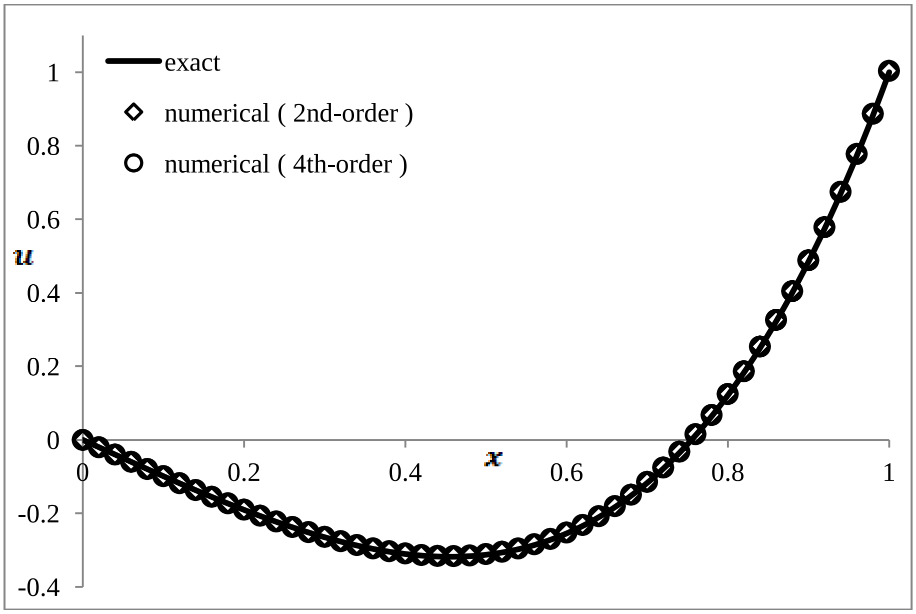


Figure 4.9 Numerical and exact results for Poisson equation (uniform mesh; N = 50; east Neumann boundary)

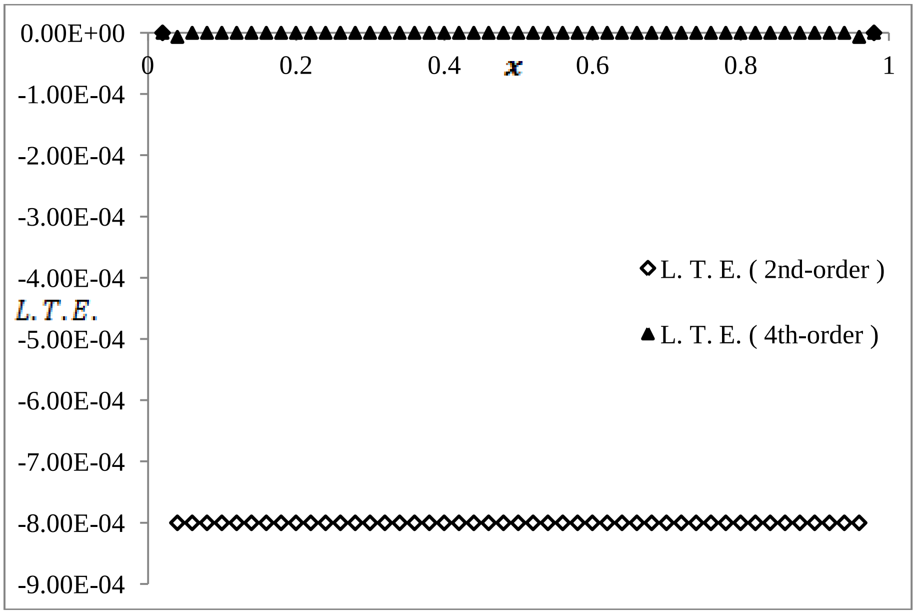


Figure 4.10 Local truncation error for Poisson equation (uniform mesh; N = 50; east Neumann boundary; 2nd-order vs. 4th-order scheme)

4.4.1.2 Clustered Mesh

(a) Dirichlet Boundary Condition

Figures 4.11 and 4.12 compare the solution results and the L.T.E. for clustered mesh 01. Figures 4.13 and 4.14 show the corresponding results for mesh 02. Figures 4.12 and 4.14 demonstrate that the local truncation error for the 4th-order scheme is much smaller and more uniformly distributed across the mesh than the 2nd-order L.T.E.

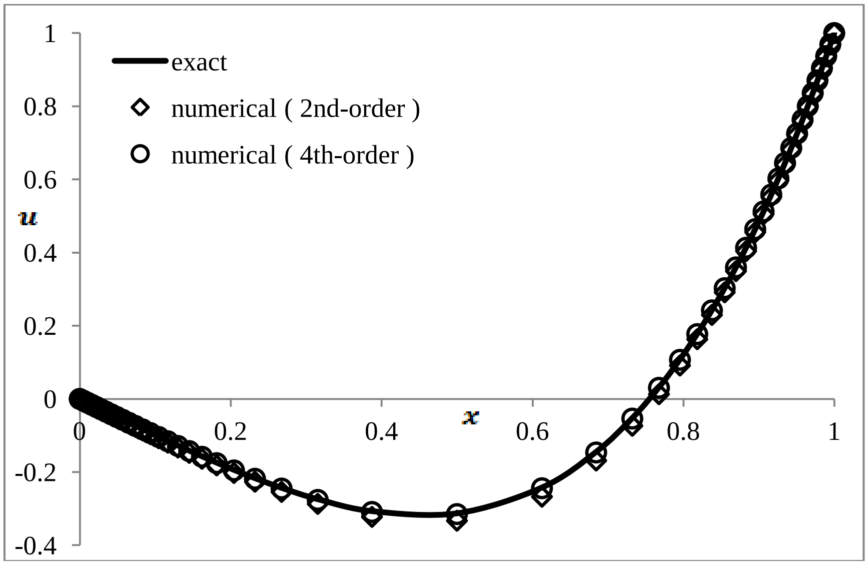


Figure 4.11 Numerical and exact results for Poisson equation (clustered mesh 01; N = 50; Dirichlet boundaries)

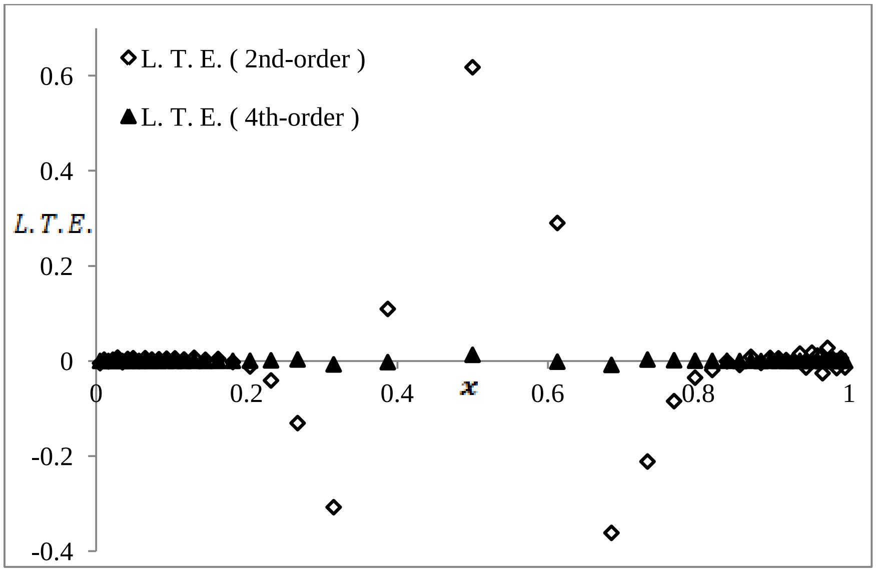


Figure 4.12 Local truncation error for Poisson equation (clustered mesh 01; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme)

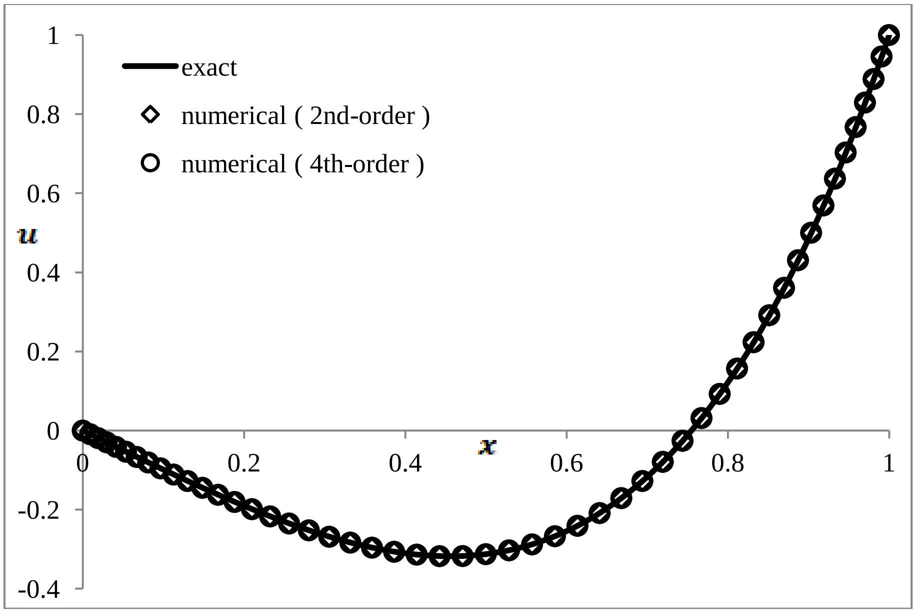


Figure 4.13 Numerical and exact results for Poisson equation (clustered mesh 02; N = 50; Dirichlet boundaries)

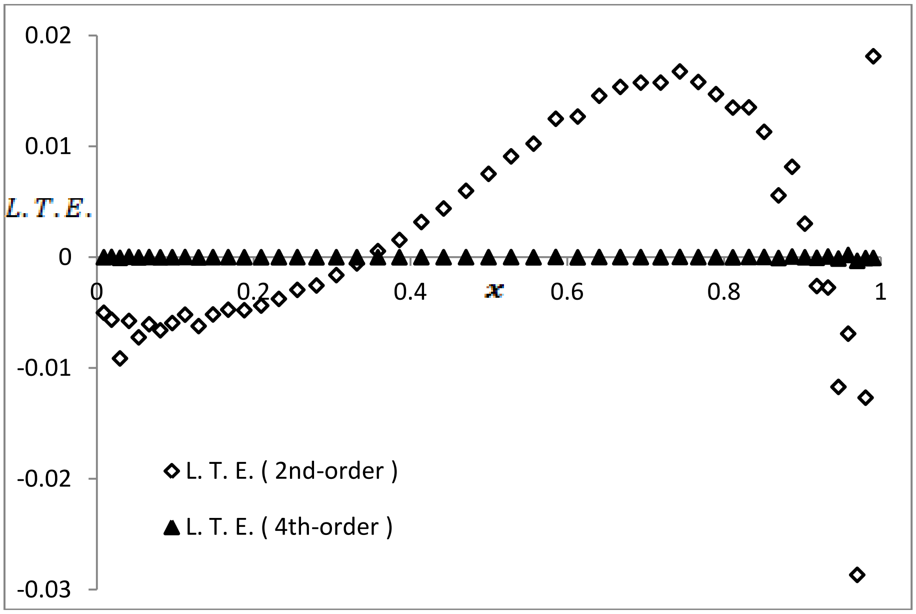


Figure 4.14 Local truncation error for Poisson equation (clustered mesh 02; N = 50; Dirichlet boundary; 2^{nd}-order vs. 4^{th}-order scheme)

(b) West Neumann Boundary Condition

Figure 4.15 shows the results from the numerical solution compared with the exact solution for mesh 01, and the same comparison for mesh 02 is demonstrated in Fig. 4.17.

Figures 4.16 and 4.18 provide the information for local truncation error. The improved accuracy obtained from the 4th-order scheme is again obvious.

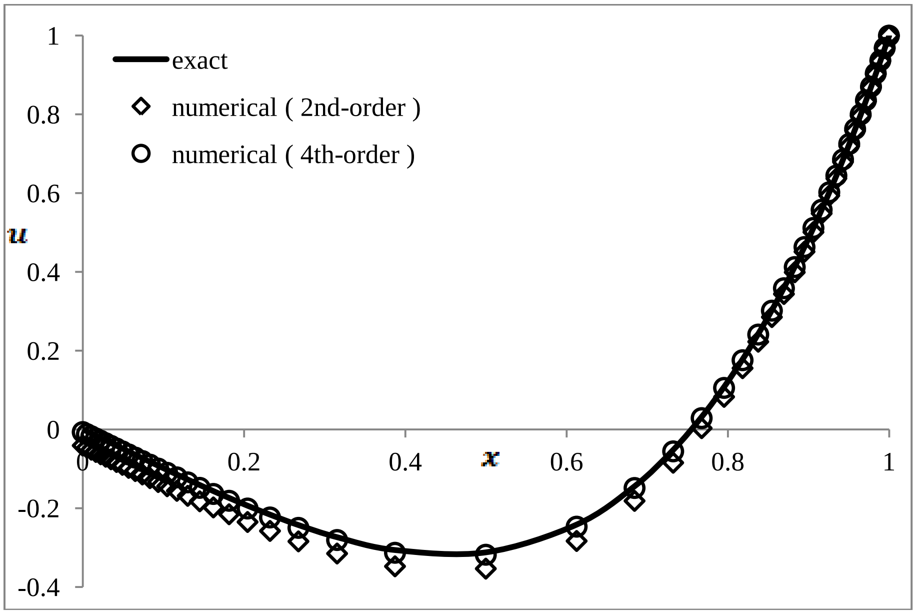


Figure 4.15 Numerical and exact results for Poisson equation (clustered mesh 01; N = 50; west Neumann boundary)

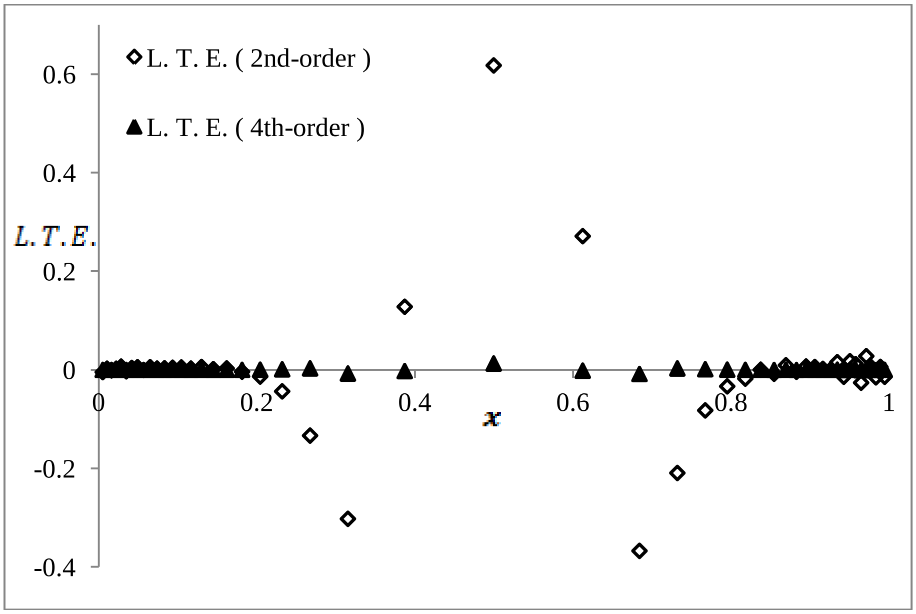


Figure 4.16 Local truncation error for Poisson equation (clustered mesh 01; N = 50; west Neumann boundary; 2nd-order vs. 4th-order scheme)

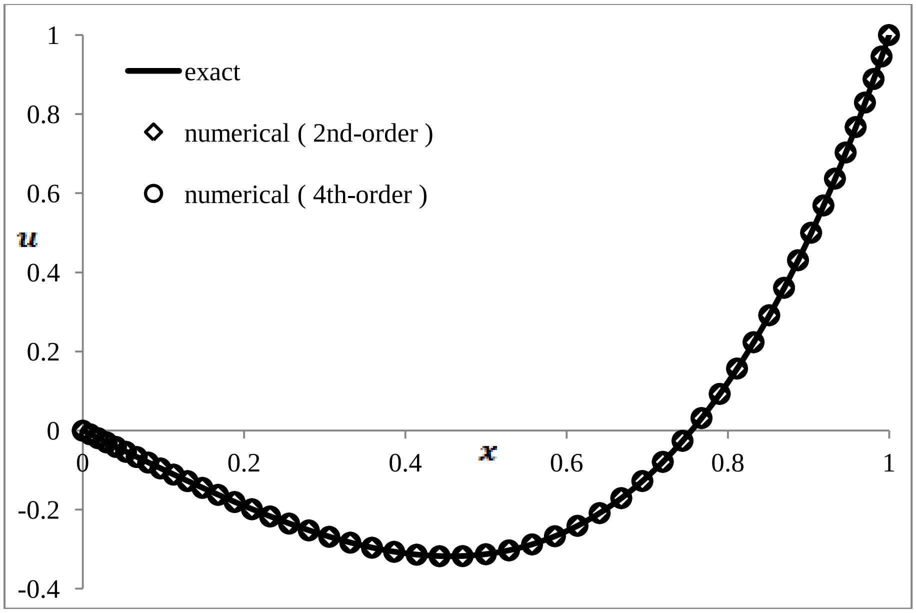


Figure 4.17 Numerical and exact results for Poisson equation (clustered mesh 02; N = 50; west Neumann boundary)

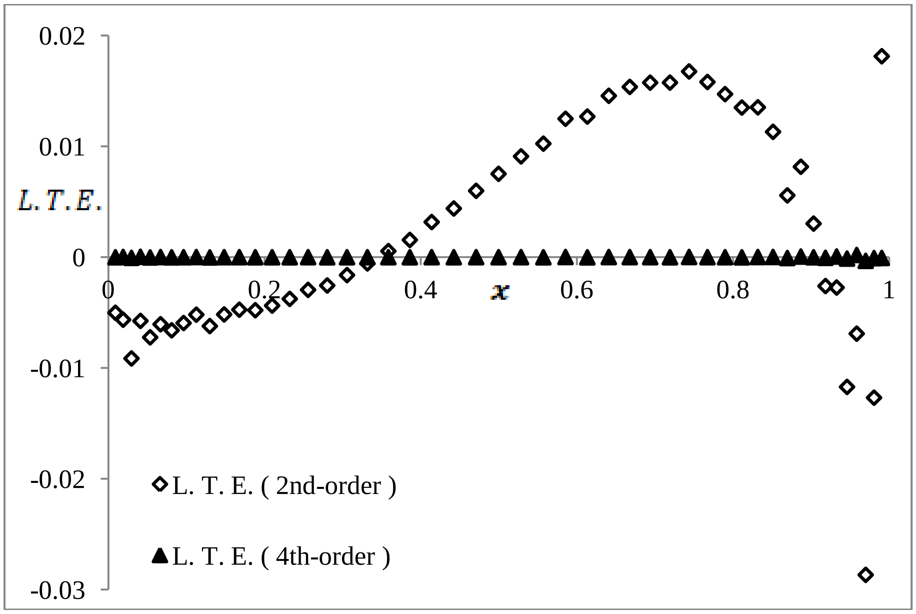


Figure 4.18 Local truncation error for Poisson equation (clustered mesh 02; N = 50; west Neumann boundary; 2nd-order vs. 4th-order scheme)

(c) East Neumann Boundary Condition

Figures 4.19 and 4.21 show the results from the numerical solution compared with the exact solution for a Neumann condition at the east boundary. Figures 4.20 and 4.22 illustrate the comparison of local truncation error for the 2nd-order and 4th-order scheme for mesh 01 and mesh 02, respectively. The superior performance of the 4th-order scheme is obvious. It is important to note that the discretisation at the boundary is of the same order as the interior nodes, so the solution does not lose accuracy at the Neumann boundary. This can be seen in both the solution and the L.T.E.

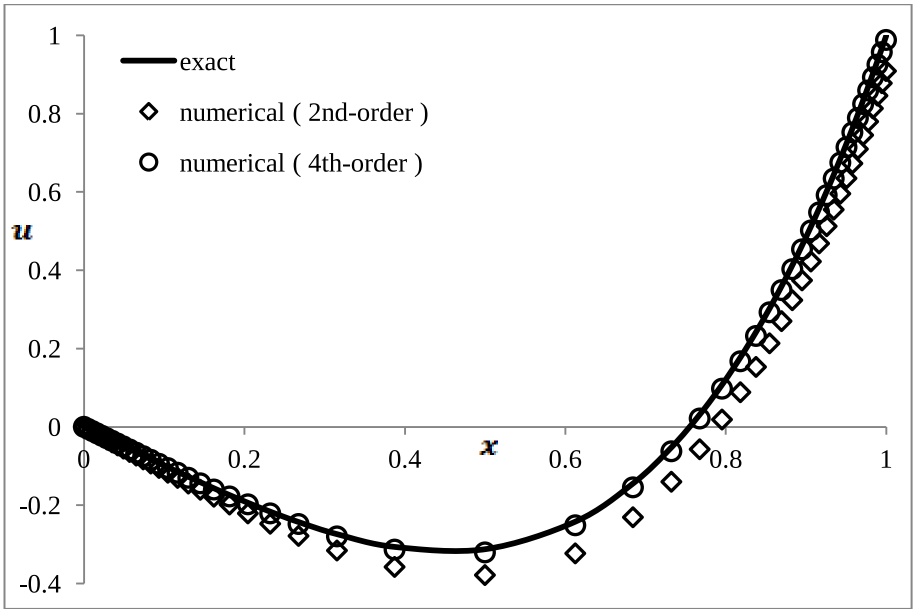


Figure 4.19 Numerical and exact results for Poisson equation (clustered mesh 01; N = 50; east Neumann boundary)

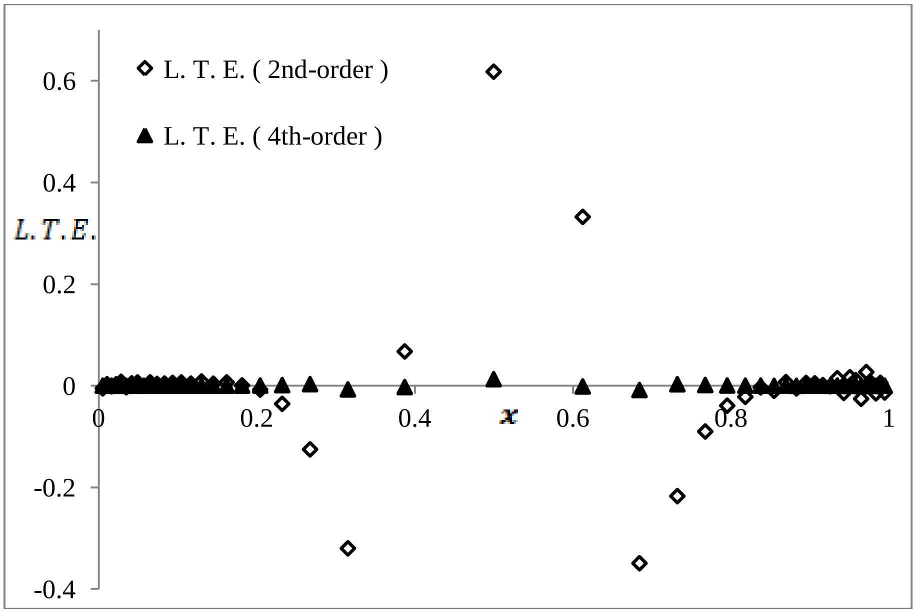


Figure 4.20 Local truncation error for Poisson equation (clustered mesh 01; N =50; east Neumann boundary; 2nd-order vs. 4th-order scheme)

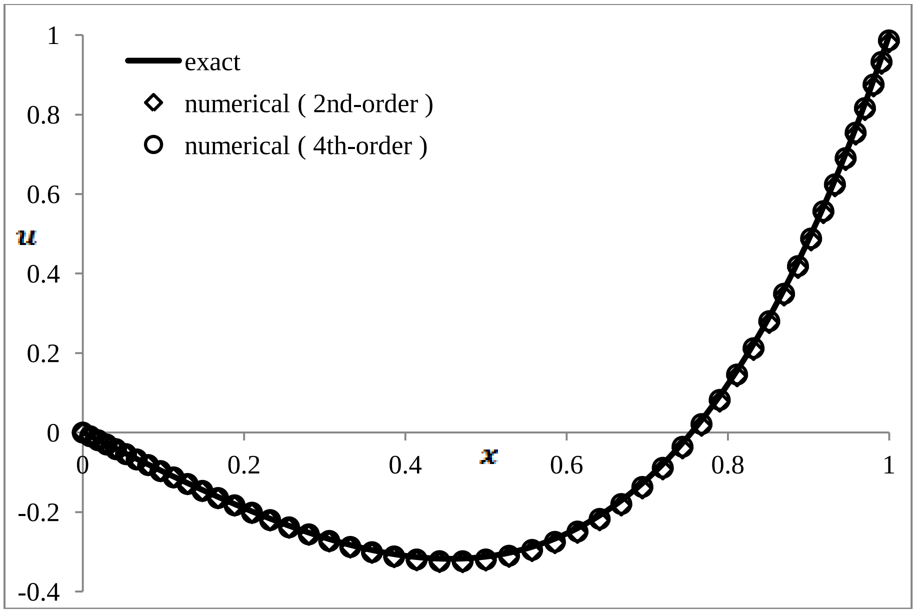


Figure 4.21 Numerical and exact results for Poisson equation (clustered mesh 02; N = 50; east Neumann boundary)

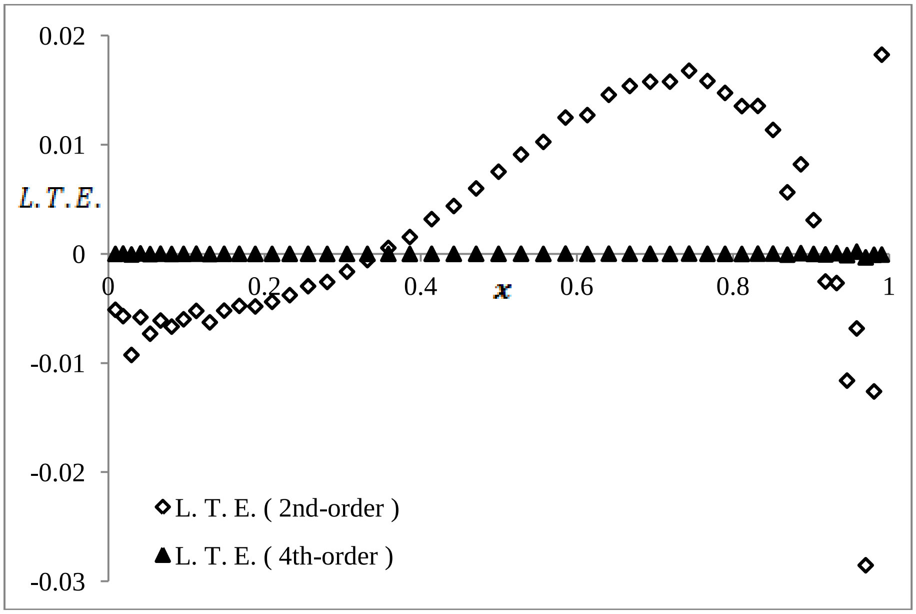


Figure 4.22 Local truncation error for Poisson equation (clustered mesh 02; N = 50; east Neumann boundary; 2nd-order vs. 4th-order scheme)

4.4.1.3 Largest Absolute Error

Table 4.1 provides details on the largest absolute error in each case considered in the previous two sections. It shows that, for all the cases, the largest absolute errors from the 4th-order scheme are much smaller than the largest absolute errors from the 2nd-order scheme, except for one case. For the uniform mesh with west Neumann boundary condition, the largest absolute error from the 4th-order scheme is about two times the largest absolute error from the 2nd-order scheme. However, if we analyze the local truncation errors for this case, the 4th-order scheme produces lower local truncation errors compared with the 2nd-order scheme. Since, in most cases, there are no exact results to compare with, the local truncation error can precisely reflect the accuracy of the schemes.

Table 4.1 Largest absolute error for each case for Poisson equation

	Boundary conditions	2nd-order scheme		4th-order scheme	
		Largest absolute error	node	Largest absolute error	node
Uniform mesh	Dirichlet	0.0001	26	0.000014	26
	west Neumann	0.0004	1	0.000791	1
	east Neumann	0.0044	51	0.004009	51
Clustered mesh 01	Dirichlet	0.0255	27	0.002423	27
	west Neumann	0.0413	27	0.006576	1
	east Neumann	0.0912	51	0.011817	51
Clustered mesh 02	Dirichlet	0.00049	37	0.000013	42
	west Neumann	0.00046	37	0.000273	1
	east Neumann	0.0185	51	0.013484	51

4.4.1.4 2nd-order scheme vs. 4th-order scheme

The Poisson equation (4.16) has been solved on several uniform meshes with Dirichlet boundary conditions. Table 4.2 shows maximum local truncation error from the 2nd-order scheme compared with 4th-order scheme. The 4th-order scheme dramatically improves the accuracy and significantly reduces the number of cells needed to achieve a specified level of accuracy. For example, the 2nd-order scheme requires 400 cells to produce a maximum L.T.E. of -1.25×10^{-5}, which is comparable to using 40 cells with the 4th-order scheme.

Table 4.2 Maximum local truncation error for Poisson equation

2nd-order		4th-order	
# of cells	max L.T.E.	# of cells	max L.T.E.
400	-1.250E-05	80	-2.604E-06
200	-5.000E-05	40	-1.042E-05
160	-7.813E-05	20	-4.167E-05
80	-3.125E-04		
40	-1.250E-03		
20	-5.000E-03		

4.4.2 Convection-Diffusion Equation

Consider the solution of the convection-diffusion equation (4.5) on a uniform mesh or clustered mesh with 50 cells and $S = 0$,

$$\frac{d^2u}{dx^2} - R\frac{du}{dx} = 0 \ . \tag{4.17}$$

Only Dirichlet boundary conditions are considered, but the stencil mapping method works equally well for Neumann conditions. The west boundary condition is 0 and east boundary condition is 1. Both the 2nd-order accurate scheme and the 4th-order accurate scheme are applied. The exact solution of this BVP is $u(x)= \frac{e^{Rx}-1}{e^{R}-1}$. The uniform mesh and clustered mesh files are identical to those applied for the Poisson equation in section 4.4.1.

4.4.2.1 Uniform Mesh

Figure 4.23 shows the results comparison between the numerical solution and exact solution on a uniform mesh. Figure 4.24 reveals the comparison of the local truncation error between 2nd-order and 4th-order schemes. These figures illustrate the improvement achieved with the 4th-order scheme, particularly in terms of the L.T.E. near the east boundary.

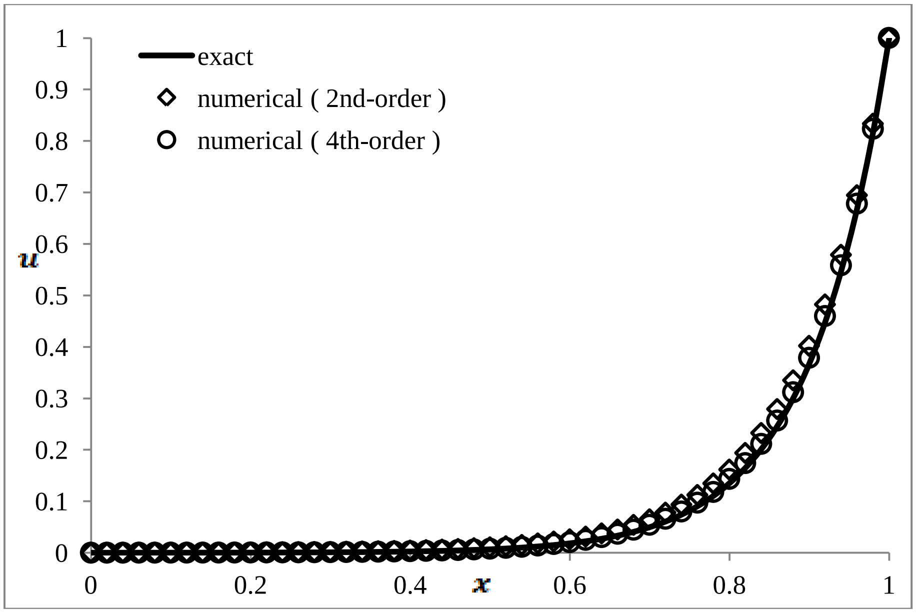


Figure 4.23 Numerical and exact results for steady convection-diffusion equation (uniform mesh; N = 50; Dirichlet boundaries)

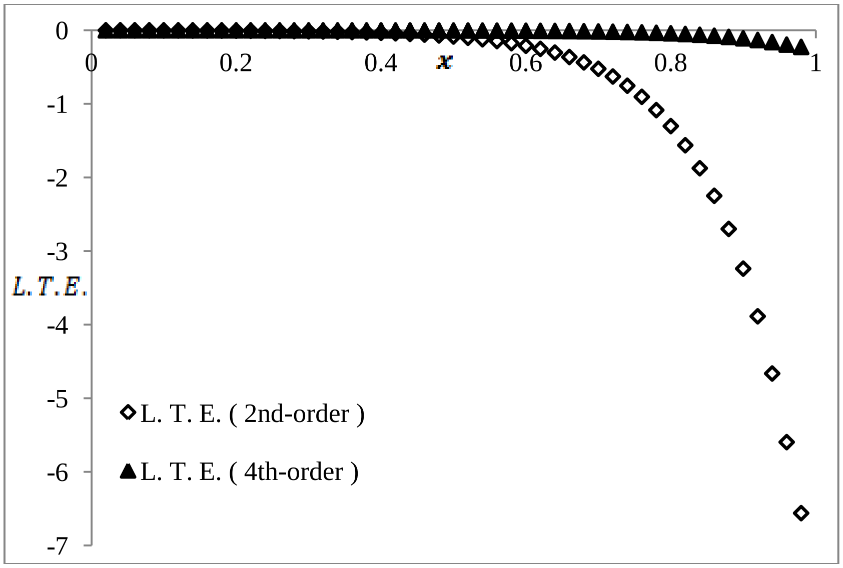


Figure 4.24 Local truncation error for steady convection-diffusion equation (uniform mesh; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme)

4.4.2.2 Clustered mesh

Details of the results and local truncation error comparison for mesh 01 and mesh 02 are shown in Figs.4.25, 4.26, 4.27 and 4.28. The L.T.E. plots clearly indicate the advantage of the 4th-order scheme and show that the 4th-order scheme L.T.E. only has small variations over all the nodes. The 2nd-order scheme, on the other hand, produces a wide range of L.T.E.s, especially near the east boundary where the solution gradients are high.

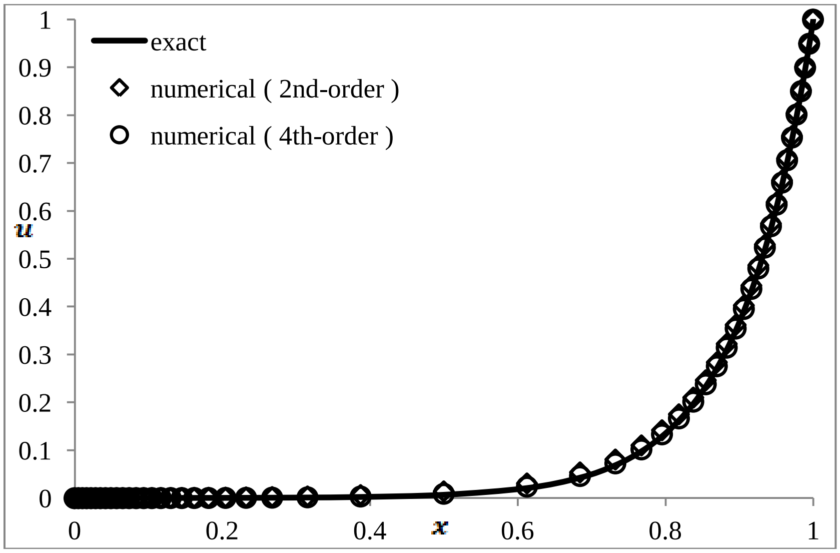


Figure 4.25 Numerical and exact results for steady convection-diffusion equation (clustered mesh 01; N = 50; Dirichlet boundaries)

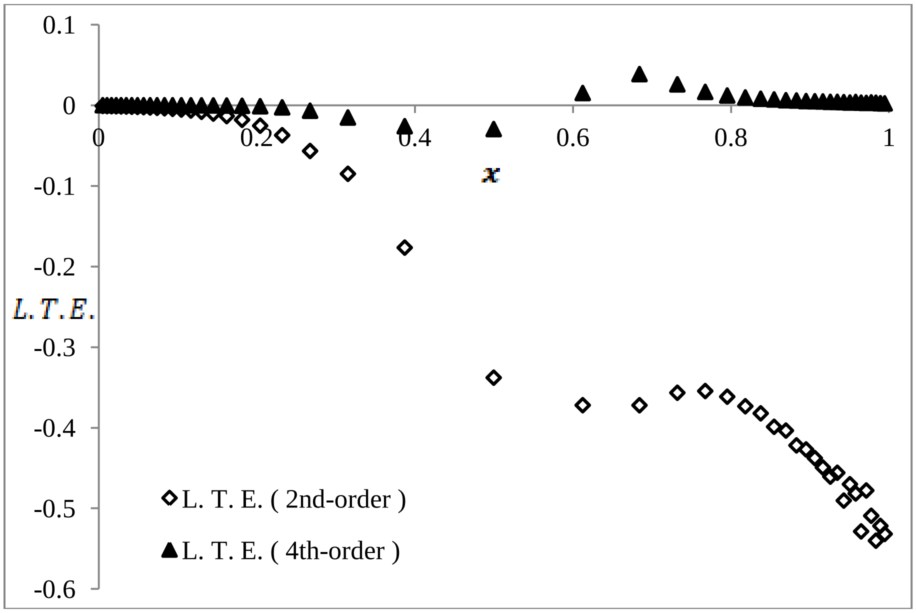


Figure 4.26 Local truncation error for steady convection-diffusion equation (clustered mesh 01; N = 50; Dirichlet boundaries; 2nd-order vs. 4th-order scheme)

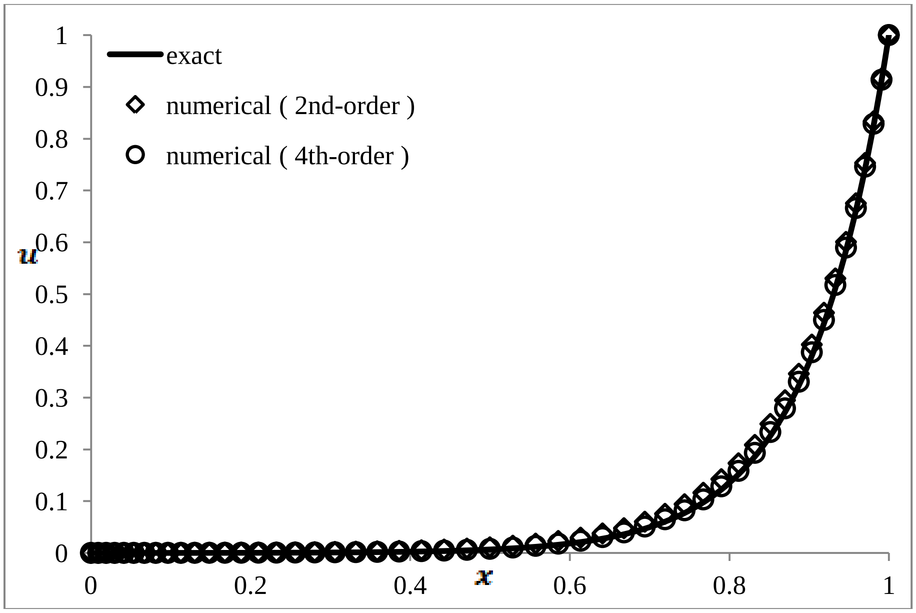


Figure 4.27 Numerical and exact results for steady convection-diffusion equation (clustered mesh 02; N = 50; Dirichlet boundaries)

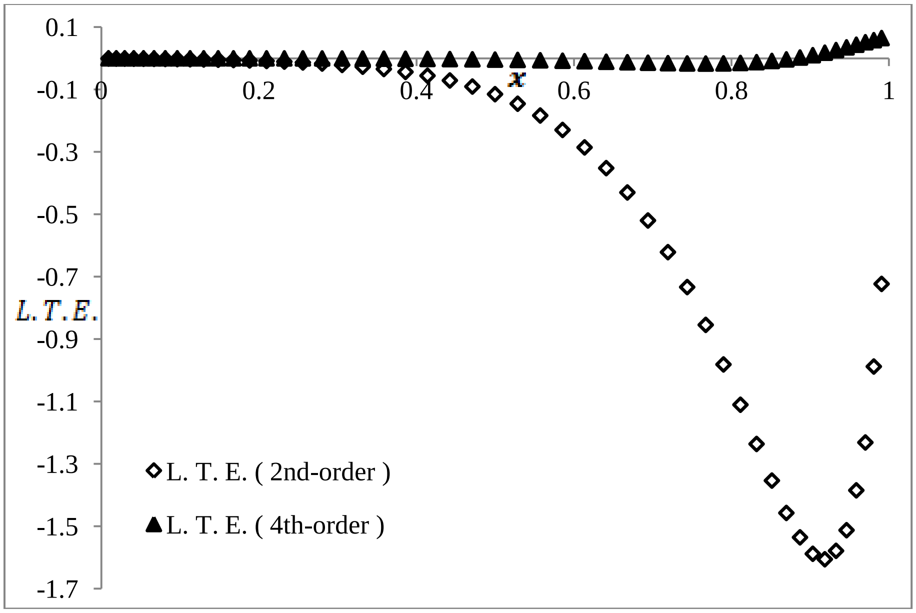


Figure 4.28 Local truncation error for steady convection-diffusion equation (clustered mesh 01; N = 50; Dirichlet boundaries; 2^nd-order vs. 4^th-order scheme)

4.4.2.3 Largest Absolute Error

Table 4.3 shows the largest absolute error and the corresponding node number for the cases of the convection-diffusion equation. For solving the steady convection-diffusion equation on the same mesh, the largest absolute error from the 4^th-order scheme is about one-third of the largest absolute error from the 2^nd-order scheme.

Table 4.3 Largest absolute error for each case for convection-diffusion equation

	Boundary conditions	2^nd-order scheme		4^th-order scheme	
		Largest absolute error	node	Largest absolute error	node
Uniform mesh	Dirichlet	0.0339611	46	0.0107388	46
Clustered mesh 01	Dirichlet	0.0125857	32	0.0040037	31
Clustered mesh 02	Dirichlet	0.0225428	41	0.0068516	41

4.5 Adaptive Meshing

When a coarse mesh produces results that are not accurate enough, one of the priority choices to resolve this problem is mesh refinement. In this section, a method is tested to adaptively refine the mesh, using the location of the largest L.T.E. as the indicator of where refinement is needed. Consider an initial uniform mesh with N = 20 cells as an example and solve the steady convection-diffusion equation using the 2nd-order scheme described above. Dirichlet boundary conditions 0 and 1 are imposed at each end of the unit domain. The adaptive meshing procedure is as follows:

Step1: Run the code to solve the differential equation on N cells.

Step 2: Locate the node (#M) where the largest local truncation error occurs among the interior nodes. Add two nodes adjacent to node M, each of which is located at the centre of the cell connected with node M. Then re-calculate the numerical solution on this refined mesh.

Step 3: If the results are not satisfactory, follow the same procedure outlined in step 2 to refine the mesh again and re-calculate the solution. Repeat this procedure until the results are acceptable.

Figure 4.29 shows the absolute error comparison for the adaptive mesh solutions. It is obvious that, overall, the absolute errors keep reducing during the three refinements, although there is some bounce back for the third adaptive mesh (N = 26). Figure 4.30 illustrates that the adaptive meshes produce accurate results compared with the exact solution. However, further refinement creates larger error near the east boundary. This is likely due to the fact that as the mesh is refined by the above procedure there may be a significant difference in the size of adjacent cells. It is expected that this problem can be resolved by employing a smoothing technique so that there is always a smooth transition between a cell and its neighbours.

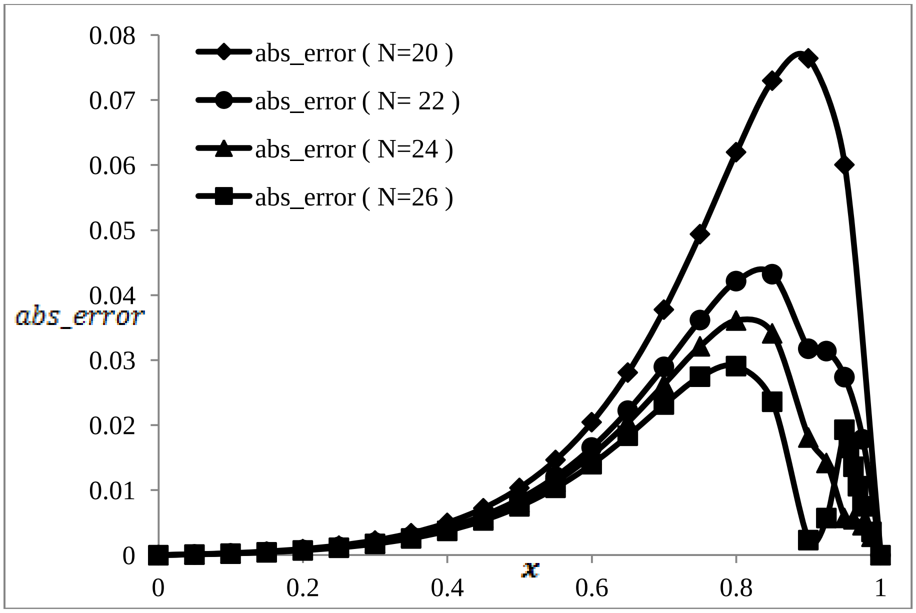


Figure 4.29 Absolute error comparison for adaptive meshing (initial uniform mesh with N = 20; convection-diffusion equation)

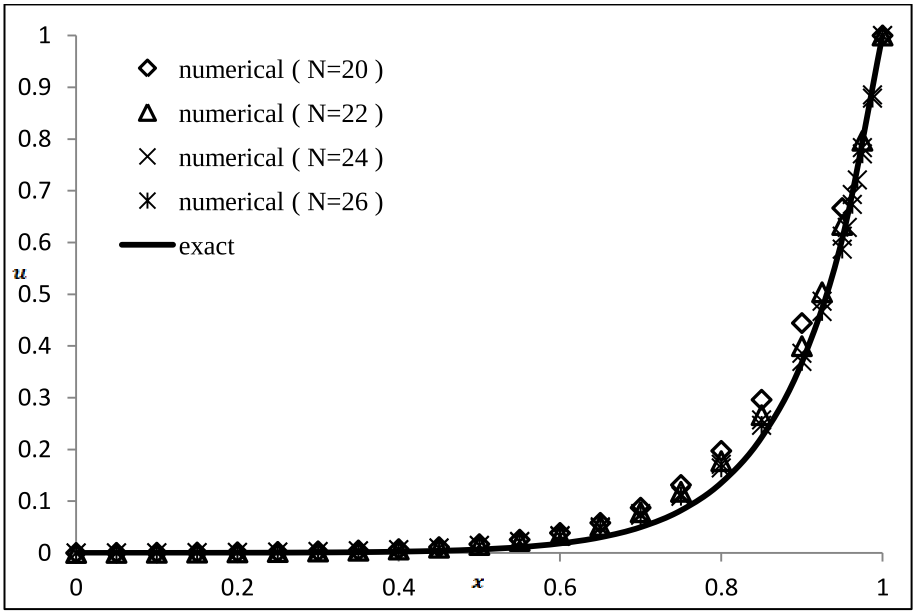


Figure 4.30 Numerical and exact results for adaptive meshes (initial uniform mesh with N = 20; convection-diffusion equation)

4.6 Summary

The new Stencil Mapping method is based on the three-point local mapping. This scheme is formulated not only for 2^{nd}-order, but also extended to higher-order accuracy (4^{th}-order). Local truncation error is introduced in order to measure the accuracy of the numerical solution. When the scheme is applied to solve practical physical problems, exact results may not be available for comparison, in which case the local truncation error is a convenient parameter which can be used to check the accuracy of the solution. The numerical schemes have been applied for the Poisson equation and the steady convection-diffusion equation on either a uniform mesh or clustered mesh. For the Poisson equation, Dirichlet and Neumann boundary conditions are applied. Both 2nd-order and 4th-order scheme works well for all the cases. Local truncation error shows that the 4th-order scheme is more accurate than the 2nd-order scheme, following the same trend as the absolute error. For the steady convection-diffusion equation, only Dirichlet boundary conditions are imposed. The numerical results are acceptable even for the clustered meshes, which could not produce reasonable results with the traditional finite difference scheme. It is clearly shown that the 4th-order scheme produces more accurate results. An adaptive meshing procedure is illustrated for the steady convection-diffusion equation, and the results show the effectiveness of the method in reducing the error.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

Based on the discussions in the previous chapters, the following conclusions can be drawn:

1. For the Traditional Finite Difference method, multi-block implementation is complicated due to the inter-block communication. The method to deal with the interface problem needs to be applied cautiously; otherwise, it may cause trouble in the programming and the accuracy. Applying the TFDM on a clustered mesh must also be done with care, since the results depend significantly on the functions used to generate the clustered mesh.
2. For the CCFD method, using averaging interpolation to update nodal values is a problem when solving the convection-diffusion equation, the results are not correct because of the inconsistency. The CCFD method is not a satisfactory method for general implementation in a CFD code. However, another method can be devised which takes advantage of the main strength of the CCFD method.
3. A new generalized finite difference method, referred to as the Stencil Mapping method, has been proposed. It has been formulated for both 2^{nd}-order and 4^{th}-order schemes, and can easily be extended to higher order. Treatment of the near-boundary nodes is facilitated by confining the differencing stencil to three adjacent nodes. Local truncation error is introduced as an essential parameter to measure the accuracy of the scheme. Both 2^{nd}-order and 4^{th}-order schemes work well on uniform, multi-block and clustered meshes. Numerical solution for the 4^{th}-order scheme is more accurate and has smoother local truncation error than the 2^{nd}-order scheme. An adaptive meshing procedure is proposed for the steady convection-diffusion equation, and the results show the effectiveness of the method in reducing the error.

5.2 Recommendations

This research has demonstrated that the Stencil Mapping method works well for steady convection-diffusion equations. Some directions for further investigations with the Stencil Mapping method are:

- A complete analysis for the 1D unsteady convection-diffusion equation. This research should include a study of higher-order time marching schemes and application of the method to moving boundary problems
- Extension to 2D and 3D
- Development of higher-order schemes using the 3-point stencil
- Implementation of a smoothing algorithm to improve the adaptive mesh procedure
- Development of the method for the Navier-Stokes equations.

REFERENCES

[1] K.A. Hoffmann and S.T. Chiang. *Computational Fluid Dynamics*: Vols. 1, 2 & 3. 4th Edition, EES, Wichita, KS, 2000.

[2] J.D. Ramshaw. *Elements of Computational Fluid Dynamics*. Imperial College Press, World Scientific Publishing Co., Hackensack, NJ, USA, 2011.

[3] H. Lomax, T.H. Pulliam and D.W. Zingg. *Fundamentals of Computational Fluid Dynamics*. Springer, 2003.

[4] D.A. Anderson, J.C. Tannehill and R.H. Pletcher. *Computational Fluid Mechanics and Heat Transfer*. Hemisphere Publishing Corp., Taylor & Francis Group, New York, 1984.

[5] P.J. Roache. *Computational Fluid Dynamics*. Hermosa Publishers, Albuquerque, NM, 1976.

[6] D.A. Anderson. *Computational Fluid Dynamics: The Basics with Applications*. McGraw-Hill, USA, 1995.

[7] H.K. Versteeg and M. Malalasekera. *An Introduction to CFD: The Finite Volume Method*. 3rd Edition, UK, 2007.

[8] J.H. Ferziger and M. Peric. *Computational Methods for Fluid Dynamics*. 3rd Edition, Springer, USA, 2002.

[9] T.J. Chung. *Computational Fluid Dynamics*. Cambridge University Press, UK, 2002.

[10] S.V. Patankar. *Numerical Heat Transfer and Fluid Flow*. Hemisphere Publishing Corp., Taylor & Francis Group, New York, 1980.

[11] J.C. Kalita. Effects of clustering on the simulation of incompressible viscous flows. *Engineering Applications of Computational Fluid Mechanics*, 1(1):36-48, 2007.

[12] P.M. Gresho and R.L. Lee. Don’t suppress the wiggles - they’re telling you something. *Computers & Fluids*, 9:223-253, 1981.

[13] G.D. Thiart. Finite difference scheme for the numerical solution of fluid flow and heat transfer problems on nonstaggered grids. *Numerical Heat Transfer*, 17:43-62, 1990.

[14] A.W. Date. Solution of transport equations on unstructured meshes with cell-centered collocated variables. Part I: Discretization. *Int. J. Heat and Mass Transfer*, 48:1117-1127, 2005.

[15] J.C. Chai and Y.F. Yap. A distance-function-based Cartesian (DIFCA) grid method for irregular geometries. *Int. J. Heat and Mass Transfer*, 51:1691-1706, 2008.

[16] B. Seibold. Minimal positive stencils in mesh free finite difference methods for the Poisson equation. *Comput. Methods Appl. Mech. Engineering*, 198:592-601, 2008.

[17] R.B. Banks. *Towing Icebergs, Falling Dominoes, and Other Adventures in Mathematics*. Princeton University Press, Princeton, 1998.

[18] A. Salih (2011). *A New Cell-Centred Finite Difference Scheme for CFD Simulations*. Master Thesis, Department of Mechanical, Automotive and Materials Engineering, University of Windsor, Windsor, ON, Canada.

[19] J.J. Situ (2011). *Development of a Cell-Centred Finite Difference Numerical Methodology on Triangulated Domains*. Master Thesis, Department of Mechanical, Automotive and Materials Engineering, University of Windsor, Windsor, ON, Canada.

[20] M. Lakner and I. Plazl. The finite differences method for solving systems on irregular shapes. *Computers and Chemical Engineering*, 32:2891-2896, 2008

[21] R.M. Barron and A. Balachandar. Private communication.

[22] W.F. Spotz (1995). *High-Order Compact Finite Difference Scheme for Computational Mechanics*. PhD Thesis, Dept. of Aerospace Engineering and Engineering Mechanics, the University of Texas at Austin, Austin, TX, USA.

[23] J.E. Castillo, J.M. Hyman, M.J. Shashkov and S. Steinberg. Fourth- and sixth-order conservative finite difference approximations of divergence and gradient. *Applied Numerical Mathematics*, 37:171-187, 2001.

[24] J.E. Castillo, J.M. Hyman, M.J. Shashkov and S. Steinberg. The sensitivity and accuracy of fourth order finite-difference schemes on nonuniform grids in one dimension. *Computers Math. Applic.*, 30:41-55, 1995.

[25] P.W. Hemker. Novel defect-correction high-order, in space and time, accurate schemes for parabolic singularity perturbed convection-diffusion problems. *Computational Methods in Appl. Math.*, 3:387-404, 2003.

[26] B.J. Noye. A new third-order finite difference method for transient one-dimensional advection-diffusion. *Commun. in Applied Numerical Methods*, 6:279-288, 1990.

[27] ANSYS Fluent. *ANSYS, Inc.*, Canonsburg, PA, USA.

[28] STAR-CCM+. *CD-adapco*, Melville, NY, USA.

[29] CONVERGE. *Convergent Science*, Inc., Middleton, WI, USA.

[30] FLOW-3D. *Flow Science, Inc.*, Santa Fe, NM, USA.

[31] COMSOL. *COMSOL, Inc.*, Palo Alto, CA, USA.

[32] S. Sundaresan (1995). *Clustered Grids and Mesh Independence in Numerical Simulation of 2D Lid-driven Cavity Flows*. PhD Thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

VITA AUCTORIS

NAME:	Shixin He
PLACE OF BIRTH:	Harbin, Heilongjiang, China
YEAR OF BIRTH:	1987
EDUCATION:	Harbin No. 6 High School, Harbin, Heilongjiang, China, 2006 Beijing Jiaotong University, B.Sc., Beijing, China, 2010 University of Windsor, M.Eng., Windsor, ON, 2012 University of Windsor, M.A.Sc., Windsor, ON, 2014